

NS486™ SXF Optimized 32-bit 486-class Controller With On-chip Peripherals for Embedded Systems

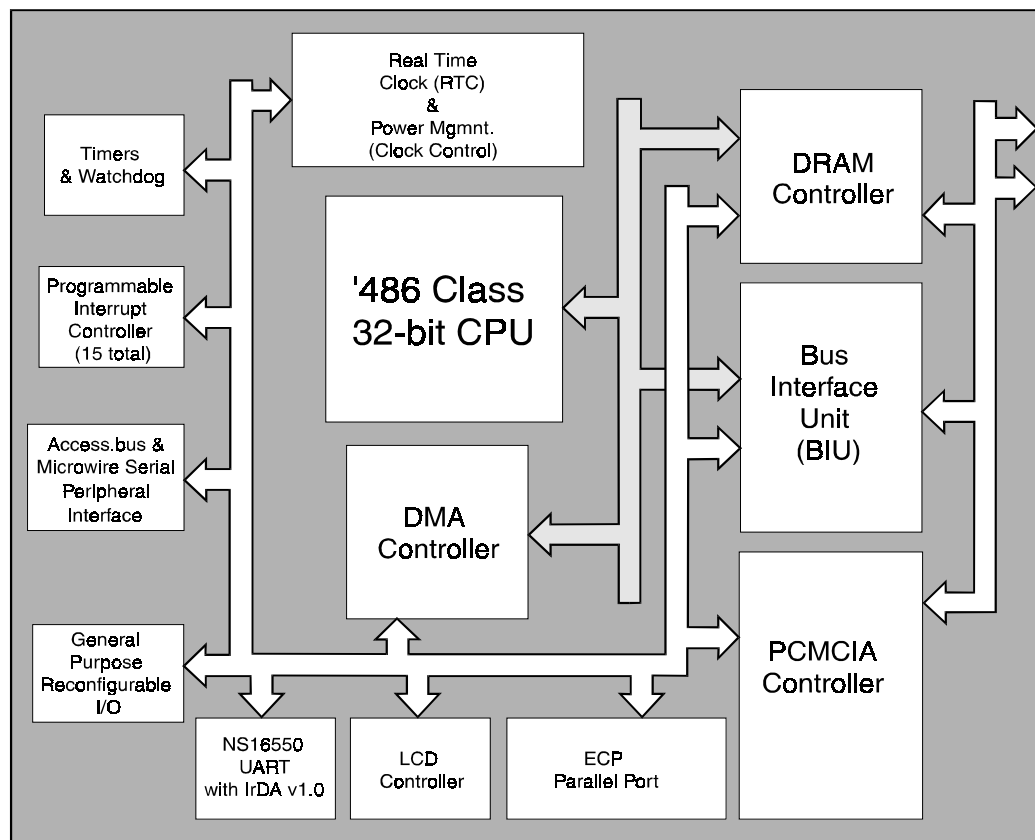
General Description

The NS486SXF is a highly integrated embedded system controller incorporating an Intel486™-class 32-bit processor, all of the necessary System Service Elements, and a set of peripheral I/O controllers tailored for embedded control systems. It is ideally suited for a wide variety of applications running in a segmented protect-mode environment.

Key Features

- 100% compatible with VxWorks®, VRTX®, QNX® Neutrino, pSOS+™, and other popular real-time executives and operating system kernels
- Intel486™ instruction set compatible (protected mode only) with optimized performance
- CPU includes a 1kByte Instruction Cache
- Operation at 25 MHz with 5 Volt supply
- Low cost 160-pin PQFP package
- Industry standard interrupt controller, timers, real time clock, UART with IrDA v1.0 (Infrared Data Association) port
- Intel 82365 compatible PCMCIA interface
- Protected WATCHDOG™ timer
- Optimized DRAM Controller (supports two banks, up to 8 Mbytes each)
- Up to nine versatile, programmable chip selects
- Glueless interface to ISA-type peripherals
- Arbitration support for auxiliary processor
- Four external DMA channels (max. transfer rate of 25MByte/sec @ 25MHz) support many transfer modes
- High performance IEEE 1284 (ECP) Bidirectional Parallel Port
- MICROWIRE™/Access.bus synchronous serial interfaces
- LCD Controller for monochrome supertwist Liquid Crystal Displays up to 480 X 320
- Reconfigurable I/O: Up to 29 I/O pins can be used as general purpose bidirectional I/O lines
- Flexible, programmable, multilevel power saving modes maximize power savings

NS486SXF Single-Chip Embedded Controller



Contents

1.0 Pin Description Tables	1
2.0 System Overview	11
2.1 NS486SXF System Overview	11
2.2 32-bit Processor Core	12
2.3 System Service Elements	13
2.3.1 DRAM Controller	13
2.3.2 DMA Controller	13
2.3.3 Programmable Interval Timer	13
2.3.4 WATCHDOG Timer	13
2.3.5 Interrupt Controller	14
2.3.6 Real Time Clock/Calendar	14
2.3.7 Power Management Features	14
2.4 NS486SXF System Bus	15
2.5 Other On-board Peripherals	16
2.5.1 Reconfigurable I/O Lines	16
2.5.2 IEEE 1284 Bidirectional Port	16
2.5.3 PCMCIA Interface	16
2.5.4 MICROWIRE/Access.bus Interface	16
2.5.5 UART Serial Port	16
2.5.6 LCD Controller	16
2.6 ICE Support	17
2.7 Other Issues	17
3.0 Programming the NS486SXF	19
3.1 Overview	19
3.2 I/O Address Map	20
3.3 System Service Elements	26
3.3.1 The DRAM Controller	26
Reset Condition	27
DRAM Parity Checking	27
Refresh	27
RAS Recovery (pre-charge) Rate	27
The DRAM Registers	28
3.3.2 The DMA Controller	37
DMA Transfer Modes	38
Autoinitialize	40
Transfer Types	40
Device Types	40
Maximum Performance	40
Software Initiated DMA Transfers	41
DMA Control Registers	41
DMA Configuration Registers	50
3.3.3 The Programmable Interval Timer (PIT)	57
Programming the PIT System	58
Control Word Register	58
Counter Write Operations	59
Counter Read Operations	59
Counter Latch Command	60
Read-Back Command	60
External Timer Control Signals	61
Timer I/O Control Register	62
Timer Clock Register	63
Mode Descriptions	63
Gate Input	65
3.3.4 The WATCHDOG Timer	66
WATCHDOG Timer Control Register	66
Retriggering the WATCHDOG Timer Count to Prevent Resets	67
Interrupt Request Supported	67
Programming Timer 2	67
3.3.5 The Interrupt Controller	69
External Cascading	70
Selecting Interrupt Source(s)	70
Interrupt Source Selection	71
Miscellaneous (PCMCIA and Extended Ca- pabilities Port (ECP)) Interrupt Selection Registers	78
Programming the PIC	80
General Operation	86
Interrupt Sequence	86
End-of-interrupt (EOI) Modes	87
Priority Nesting	87
Priority	88
POLL Command	88
3.3.6 The Real Time Clock/Calendar	89
Memory Map	89
Bus Interface	89
Time Generation	90
RAM	91
Power Management and System Lock-out Features	91
Interrupt Handling	92
Control Registers	93
3.3.7 Power Management Features	96
Power Management Modes	97
Enabling/Disabling Individual Peripheral Clocks	98
Global Enable/Disable of Peripheral Clocks	99
Power Supply Level Configuration	99
Power Management Configuration Registers	99
Crystal Oscillator Circuitry	101
3.3.8 NS486SXF System Bus	102
3.4 Other On-chip Peripherals	103
3.4.1 The Reconfigurable I/O Pins	103
Reconfigurable I/O Control Register	103
Data Direction Register	104
Data Port Out Register	105
Data Port In Register	105
3.4.2 The IEEE 1284 Bidirectional Parallel Port	107
ECP Pinouts	108
ECP Registers	111

ECP Power Management	112	Control Register (MACON)	139
Register Addressing	112	MICROWIRE Registers	140
Index Register (Base address + 403h) . .	113	MICROWIRE/Access.bus	
Setup Register (Base address + 404h) . .	113	Interface Programming Notes	144
Data Register (DATAR)		3.4.5 The Serial UART Port	145
(Base address + 000h)	117	Serial Port Registers	146
Device Status Register (DSR)		IrDA v1.0 Compatible Infrared Link. . .	156
(Base address + 001h)	118	3.4.6 The LCD Controller	157
Device Control Register (DCR)		LCD Display Basics	158
(Base address + 002h)	118	Direct Control Signals	161
Extended Control Register		Indirectly Applied Signals	165
(Base address + 402h)	120	LCD Configuration Registers	166
Interrupt Pending Register		4.0 The System Bus	171
(Base address + 405h)	121	4.1 Bus Interface Unit (BIU)	171
CNFG A Register		4.1.1 Internal Peripheral Accesses	172
(Base address + 400h, in Configuration		NS486SXF Peripheral Control Registers	172
mode)	122	Global Enable	173
CNFG B Register		Accessing The Internal UART	173
(Base address + 401h, in Configuration		Accessing The Internal	
Mode)	122	Extended Capabilities Port (ECP)	173
ECP Address FIFO (AFIFO)		Accessing Other Internal Peripherals . . .	173
(Base address + 000h, in ECP Mode) . . .	123	Accessing The Internal	
Parallel Port FIFO Register (CFIFO)		PCMCIA Controller	173
(Base address + 400h, in		4.1.2 PCMCIA Memory Accesses	174
Parallel Port FIFO Mode)	123	PCMCIA Access Timing	174
ECP Data FIFO Register (DFIFO)		4.1.3 Interrupt Acknowledge Cycles	175
(Base address + 400h, in ECP Mode) . . .	123	4.1.4 HALT Cycles	175
Test FIFO Register (TFIFO) (Base		4.1.5 SHUTDOWN Cycles	175
address + 400h, in FIFO Test Mode) . . .	124	4.2 External Bus Cycles	176
Software Controlled Data Transfer		4.2.1 ISA-like Bus Operation	176
(Host - Standard Parallel Port		4.2.2 ISA-like Bus Timing	177
and PS/2 modes)	124	4.3 Device Configuration at RESET	179
Automatic Data Transfer		4.3.1 Boot-up ROM BIOS Size	179
(Host - Parallel Port FIFO		4.3.2 Normal Mode vs. ICE Mode	179
and ECP modes)	124	4.4 Chip Select Logic	180
Software Controlled Data Transfer		4.4.1 Boot ROM Data Bus Size	180
(Slave - Standard Parallel Port		4.4.2 Other Chip Selects' Data Bus Size . .	180
and PS/2 modes)	126	4.4.3 Programmable Chip Select	
Automatic Data Transfer		Architecture	180
(Slave Parallel Port FIFO		4.4.4 Programming Logical Chip Selects . .	182
and ECP modes)	126	4.4.5 Combining Chip Selects	184
FIFO Test Access	127	4.4.6 Programmable Access Timing	184
Configuration Registers Access	127	4.5 Bus Interface Unit Registers	189
Interrupts	127	4.5.1 Bus Interface Unit Control	
3.4.3 The PCMCIA Interface	129	Register 1	189
PCMCIA Address Register	131	4.5.2 Bus Interface Unit Control	
PCMCIA Clock Selection Register	131	Register 2	190
PCMCIA Interrupt Selection Registers .	131	4.5.3 Chip Select Base Address	
3.4.4 The MICROWIRE or Access.bus		Registers	191
Interface	132	Chip Select Base Address Register 1 . .	191
MICROWIRE Interface	132	Chip Select Base Address Register 2 . .	191
MICROWIRE Transfers	133	Chip Select Base Address Register 3 . .	191
The Access.Bus Interface	135	Chip Select Base Address Register 4 . .	191
Access.bus Data Transfers	136	Chip Select Base Address Register 5 . .	191
Addressing Transfer Formats	138		
MICROWIRE/Access.bus			

Chip Select Base Address Register 6 . . .191	4.7 Device ID and Revision Registers 203
Chip Select Base Address Register 7 . .191	4.7.1 Device ID Register 203
Chip Select Base Address Register 8 . .191	4.7.2 Device Revision Register 203
4.5.4 Chip Select Address Mask	5.0 NS486 CPU Overview 205
Registers192	5.1 Architectural Overview 205
Chip Select Address Mask Register 1 .192	5.2 Key Differences From The 486 206
Chip Select Address Mask Register 2 .192	5.3 Register Set 207
Chip Select Address Mask Register 3 .192	5.3.1 Application Registers 207
Chip Select Address Mask Register 4 .192	5.3.2 Memory Management Registers 212
Chip Select Address Mask Register 5 .192	GDTR: Global Descriptor
Chip Select Address Mask Register 6 .192	Table Register 216
Chip Select Address Mask Register 7 .193	IDTR: Interrupt Descriptor
Chip Select Address Mask Register 8 .193	Table Register 217
4.5.5 Chip Select Enable Register193	LDTR: Local Descriptor
4.5.6 Chip Select Type Register193	Table Register 217
4.5.7 Chip Select Access Time	TR: Task Register 218
Registers193	5.3.3 Debug Registers 220
Chip Select Access Time Register 1 . .193	5.3.4 Control Registers 222
Chip Select Access Time Register 2 . .194	5.4 General Operands and Memory 222
Chip Select Access Time Register 3 . .194	5.4.1 Immediate Operands 223
Chip Select Access Time Register 4 . .195	5.4.2 Addressing Modes: Offsets 223
Boot ROM Access Time Register195	5.5 Initial Reset State of Core 227
4.5.8 Cacheable Chip Selects	5.6 Instruction Set Summary 228
Register196	5.6.1 Integer Instructions 228
4.5.9 External Chip Selection	5.6.2 Multiple-Segment Instructions 229
Registers196	5.6.3 Operating System Instructions 230
Boot ROM Selection Register196	5.7 I/O Addressing 230
External Chip Select	5.8 Memory Addressing 231
Selection Register 1196	6.0 Test and Development Support 233
External Chip Select	6.1 Modes 233
Selection Register 2197	6.1.1 Chip Test Modes 233
External Chip Select	6.1.2 Board Test Modes 233
Selection Register 3197	AND All Inputs (Testmode 58) 233
External Chip Select	Drive All Outputs High (Testmode 59) . 233
Selection Register 4197	Drive All Outputs Low (Testmode 60) . . 233
External Chip Select	Tri-State All Outputs (Testmode 61) . . 233
Selection Register 5197	Toggle All Outputs (Testmode 63) 233
External Chip Select	6.1.3 Setting up for Test Mode 233
Selection Register 6198	6.2 ICE Considerations 238
External Chip Select	7.0 Device Specifications 239
Selection Register 7198	7.1 DC Electrical Specifications 5V ± 5% 239
External Chip Select	7.1.1 Recommended Operating
Selection Register 8198	Conditions 239
4.5.10 Programmable 16-bit Logical	7.1.2 Absolute Maximum Ratings (Notes 2 and
Chip Select Register199	3) 239
4.5.11 PCMCIA Memory Base	7.1.3 Capacitance: TA = 25°C, f = 1 MHz. 239
Address Register199	7.1.4 DC Characteristics 240
4.5.12 PCMCIA Clock Selection Register .200	External Bus 240
4.6 Auxiliary Processor, Shared Memory	DMA Control Unit 240
Interface200	DRAM Control Unit 240
4.6.1 Auxiliary Processor	Auxiliary Processor Interface 241
Communications200	HP-SIR/UART 241
4.6.2 CPU Controlled Transfers201	
4.6.3 DMA Controlled Transfers203	
4.6.4 Auxiliary Processor Chip Select	
Selection Register203	

External Bus Control	241
Oscillator (CPUX1/CLK)	241
LCD Interface	241
Real Time Clock (RTCX1/CLK)	242
PCMCIA (RIO8-15)	242
IEEE-1284 Port (ECP Mode) & (RIO16-31)	242
Timer	242
General Purpose Chip Selects	243
Interrupt Controller	243
3-Wire I/O (& Access.bus)	243
7.2 General AC Specifications	244
7.2.1 Power Ramp Times	246
7.2.2 PWRGOOD and Power Rampdown Timing	246
7.3 AC Switching Specifications	247
7.3.1 DRAM Interface Timing Specification	248
7.3.2 ISA-like Bus Cycles Timing Specifica- tion	250
7.3.3 Ready Feedback Timing Specifications	252
7.3.4 OSCX1 AC Specification	252
7.3.5 Peripheral Timing Specifications	253
DMA Controller	253
PIC AC Specs	254
Parallel Port	255
PCMCIA Controller	256
MICROWIRE (3-Wire) & Access.bus	259
FIFO UART	260
LCD Controller	263
Supported Testmodes	264
7.4 Physical Description	265
Appendix A: Key Processor Differences from the '486	267
Appendix B: Dealing with Unused Signals on the NS486SXF	269
Appendix C: Getting Started	273
Appendix D: External Interrupt Controller	277
Appendix E: Instruction Set	279
Appendix F: Glossary	305

Pin Description Tables

NS486SXF Package Pinout Diagram	1
Bus Interface Unit Pins	2
DMA Control Pins	3
DRAM Control Pins	3
Power Pins	4
Reset Logic Pins	4
Auxiliary Processor Interface Pins	5
Test Pins	5
Interrupt Control Pins	6
Real Time Clock Pins	6
LCD Interface Pins	6
Oscillator Pins	6
HP-SIR/UART Pins	7
PCMCIA pins	7
IEEE-1284 Port (ECP Mode)	8
Timer Pins	8
3-Wire Serial I/O Pins	9
General Purpose Chip Select Pins	10
Summary of Reconfigurable I/O Pins	10

System Overview

NS486SXF System Overview

NS486SXF Internal Resource to Pins Map	11
--	----

32-bit Processor Core

System Service Elements

NS486SXF System Bus

NS486SXF Internal Busses	15
------------------------------------	----

Other On-board Peripherals

ICE Support

Other Issues

Programming the NS486SXF

Overview

NS486SXF I/O Utilization	19
------------------------------------	----

I/O Address Map

I/O Address Map	20
---------------------------	----

System Service Elements

Memory Addressing	26
Valid DRAM Address Chip Select Generation	31
DRAM Read Cycle (4 cycle page miss)	34
DRAM Write Cycle (4 cycle page miss)	34
DRAM Read Followed by DRAM Write (4 cycle page miss)	34
CAS Before RAS Refresh Cycle (4 cycle page miss)	35
DRAM Read Cycle (3 cycle page miss)	35
DRAM Write Cycle (3 cycle page miss)	35
DRAM Read Followed by DRAM Write (3 cycle page miss)	36
CAS Before RAS Refresh Cycle	36
DMA Controller Internal Block Diagram	37
DMA Controller Address Registers	42

Buffer Chaining Example	47
I/O Mapped DMA Requester, No Wait State Write Cycle (2 clocks per transfer)	52
I/O Mapped DMA Requester, One Wait State Write Cycle (4 clocks per transfer)	52
I/O Mapped DMA Requester, No Wait State Write Cycle	53
I/O Mapped DMA Requester, No Wait State Write Cycle	53
I/O Mapped DMA Requester, No Wait State Read Cycle (3 clocks per transfer)	54
I/O Mapped DMA Requester, One Wait State Read Cycle (4 clocks per transfer)	54
I/O Mapped DMA Requester, Three Wait State Read Cycle (8 clocks per transfer)	55
Memory Mapped DMA Requester, No Wait State Read Cycle (4 clocks per transfer)	55
Memory Mapped DMA Requester, No Wait State Write Cycle (4 clocks per transfer)	56
I/O Mapped DMA Requester, No Wait State Write Cycle (2 clocks per transfer)	56
Programmable Interval Timer.	57
Programmable Interrupt Controller Block Diagram.	69
Interrupt Request Selection.	70
Accessing the PIC Registers.	85
Interrupt Sequence	86
RTC Memory Map	89
Oscillator External Circuitry.	90
Typical Battery Configuration	91
Typical Battery Current During Battery Backed Mode	91
Interrupt Status Timing.	92
Power Management Options.	96
Oscillator External Circuitry.	101
Other On-chip Peripherals	
Reconfigurable I/O Pins and Register bits	104
ECP Block Diagram	107
Parallel Port Pin Out	108
Phase Transition States.	110
ECP Registers	111
Host - Forward Write Cycle	125
Host - Reverse Read Cycle.	125
PCMCIA Interface	129
NS486SXF PCMCIA Registers	130
A MICROWIRE System Example	132
MICROWIRE Normal SCLK Mode Timing Diagram	133
MICROWIRE Alternate SCLK Mode Timing Diagram	134
A Sample Access.bus System.	135
Multiple Resources	135
Bit Transfer.	136
Start and Stop Conditions	136
Access.bus Data Transfer	137

Access.bus Acknowledge Cycle.	137
A Complete Access.bus Data Transfer.	138
Summary of NS486SXF UART Registers.	147
NS486SXF Composite Serial Data	148
UART Reset Configuration	149
NS486SXF UART Divisors, Baud	
Rates and Clock Frequencies	150
NS486SXF UART Interrupt Control Functions.	155
LCD Controller Diagram	157
LCD Power Application.	158
LCD Row/Pixel Organization	159
A Complete Frame	160
LCD Controller Timing	161
LCD Memory Space.	162
DRAM Video Data to LCD Pixel Mapping for 2 and 4 Gray Levels	163
GLUT Word Mapping	164
Generating the WF Signal	165
Typical LCD and Touch Screen Interface	166
The System Bus	
Bus Interface Unit (BIU)	
NS486SXF Internal Buses	171
External Bus Cycles	
ISA-like Bus Cycle Timing	178
Device Configuration at RESET	
Chip Select Logic	
Logical Chip Selection	181
External Chip Selection	181
Address Masking	182
No Wait States, No Command Delays, External Interface Bus Cycles (Internal access timing is similar)	185
No Wait States, One Command Delay, External Bus Cycles	185
One Wait State, No Command Delays, External Interface Bus Cycles.	185
One Wait State, One Command Delay, External Interface Bus Cycles.	186
Two Wait States, One Command Delay, External Interface Bus Cycles.	186
One Command Delay, Ext. Device Inserts Wait State, External Interface Bus Cycles.	186
One Command Delay, Ext. Device Inserts Wait State, One Clock Period of RDY Extension, External Interface Bus Cycles	187
One Command Delay, Ext. Device Inserts Wait State, Two Clock Periods RDY Extension, External Interface Bus Cycles.	187
16-bit to 8-bit Cycle Translations.	188
Bus Interface Unit Registers	
Auxiliary Processor, Shared Memory	
Interface	
Auxiliary Processor Shared Memory Block Diagram	202
Shared Memory Single Access (CPU Controlled).	202
Device ID and Revision Registers	

NS486 CPU Overview

Architectural Overview	
Processor Core Block Diagram	205
Key Differences From The 486	
Register Set	
General Purpose, Segment and Flag Registers	207
Segment Selector	208
EFLAGS Register	210
Functions of EFLAG Register Bits	211
Memory Management	
Registers	212
Memory Segment Descriptor Format	213
System Segment Descriptor Format	215
Gate Descriptor Format	216
Exceptions and Interrupts	217
Task State Segment (TSS)	
Format	218
286-Compatible Task State Segment (TSS) Format	219
Debug Registers	220
Functions of Debug Status	
Register Bits (DR6)	220
Functions of Debug Ctrl. Reg. Bits (DR7)	221
Control Register CR0	222
General Operands and Memory	
One-Byte Format	223
16-Bit Addressing Modes by mod and r/m Fields	224
One-Byte Addressing Modes by mod and r/m Fields	225
Two-Byte Format	226
32-bit Two-Byte Addressing Modes by mod and base Fields	226
Indexing Expressions and	
Encodings	227
Initial Reset State of Core	
Instruction Set Summary	
I/O Addressing	
Memory Addressing	
Memory Addressing	231
Test and Development Support	
Modes	
Pin Functions in Test Mode	234
ICE Considerations	
160-pin Test Clip (Pomona 5645)	238
Device Specifications	
DC Electrical Specifications 5V \pm 5%	
General AC Specifications	
Switching Characteristic Measurement Waveforms	244
More Switching Specifications	245
Power Supply Rise and Fall	246
Vdd Rise and Fall Times	246

Vdd Rampdown vs. PWRGOOD	246
PWGOOD in relation to Vdd	246
AC Switching Specifications	
DRAM Timing Diagram	248
4 Cycle Page Miss Preliminary Specifications	248
3 Cycle Miss Preliminary Specifications	249
ISA-like Bus Timing Diagram	250
No Command Delay ISA-like Bus Specifications	250
One Programmed Command Delay ISA-like Bus Specifications	251
Ready Feedback Timing Diagram	252
Ready Signal Timing Specifications	252
TTL Clock Input Timing Diagram	252
TTL Clock Input Specification	252
DMA Controller Read Timing Diagram	253
DMA Controller Write Timing Diagram	253
DMA Controller Specifications	253
PIC Timing Diagram	254
PIC Timing Specifications	254
Parallel Port Compatibility Mode Handshake Timing Values	255
Parallel Port IEEE 1284 Mode Handshake Timing Values	255
Memory Read Timing	256
PCMCIA Memory Read Timing Specifications	256
Memory Write Timing Diagram	256
Memory Write Timing	256
I/O Read Timing	257
PCMCIA I/O Read Specifications	257
I/O Write Timing Diagram	258
PCMCIA I/O Write Specifications	258
Access.bus Timing Diagram	259
Access.Bus Timing Specifications	259
UART Baud Rate and Infrared Clocks	260
UART IRQ Timing	261
UART Modem Control Timing	262
LCD Controller Timing Diagram	263
LCD Controller Timing Specifications	263
Testmode Timing Diagram	264
Physical Description	
Plastic Package Specifications	265

1.0 Pin Description Tables

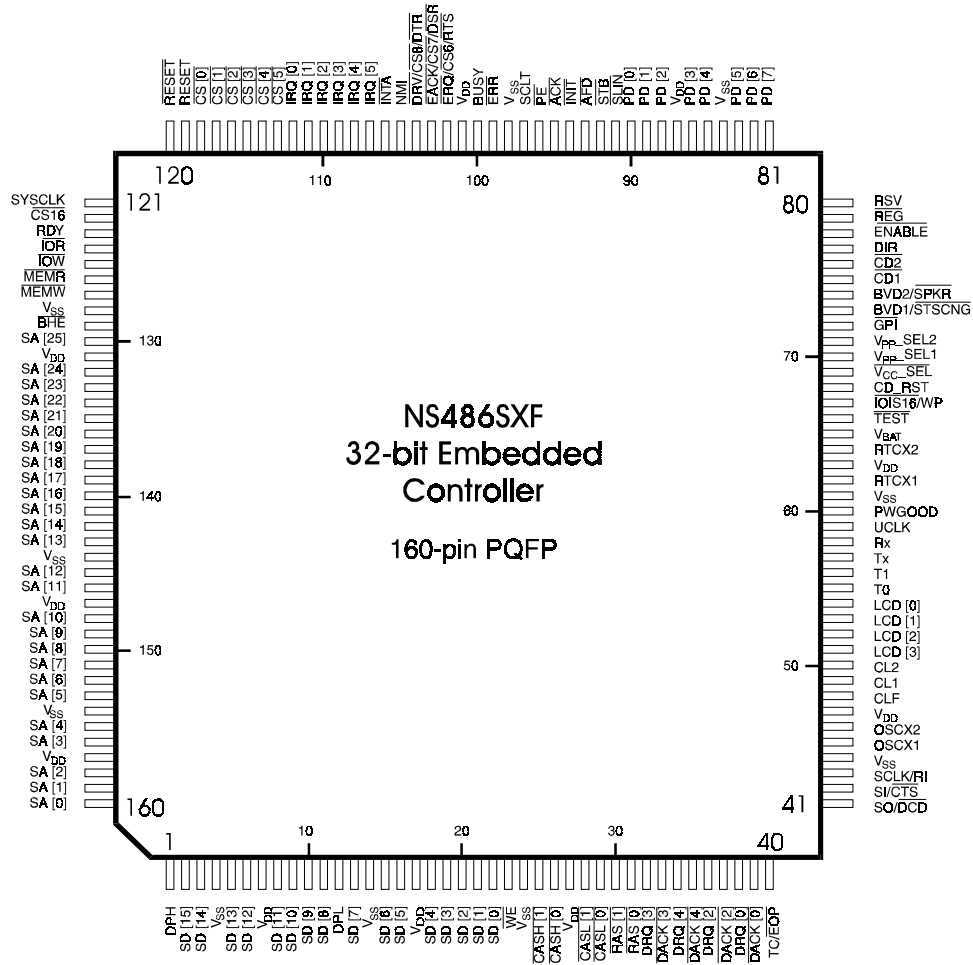


Figure 1-1 NS486SXF Package Pinout Diagram

The **NS486SXF** single chip controller is provided in a compact 160-pin, industry standard JEDEC PQFP package. The following tables detail the Symbol, Type, and Description of each pin. The tables divide the pins into functional groups as follows: Bus Interface Unit Pins, DMA Control Pins, DRAM Control Pins, Power Pins, Reset Logic Pins, Auxiliary Processor Interface Pins, Test Pins, Interrupt Control Pins, Real Time Clock Pins, LCD Interface Pins, Oscillator Pins, UART/IrDA Pins, PCMCIA Pins, IEEE-1284 Port (ECP Mode) Pins, Timer Pins, 3-Wire Serial I/O Pins, General Purpose Chip Select Pins, and Reconfigurable I/O Pins. Twenty-nine I/O pins are multipurpose. In their standard modes, they perform specific I/O controller functions. When those particular I/O functions are not required in the system, however, those pins can be reprogrammed to become general purpose, bidirectional I/O lines.

Note: In the above figure and in the following tables, all active low signals are shown with an overbar.

Table 1-1: Bus Interface Unit Pins

Symbol	Pins	Type	Function
SA[25:0]	130, 132, 133, 134, 135, 136, 137, 138, 139, 140, 141, 142, 143, 145, 146, 148, 149, 150, 151, 152, 153, 155, 156, 158, 159, 160	O	System Address bus. These output-only signals carry the latched address for the current access. DRAM accesses multiplex the row and column addresses for the DRAMs on the SA[12:1] pins. During Interrupt Acknowledge cycles, the internal master interrupt controller's cascade line signals, CAS[2:0], are driven onto SA[25:23], respectively. SA[0] is sampled at the end of reset to determine if the part will run normally or enter ICE TRISTATE mode.
SD[15:0]	2, 3, 5, 6, 8, 9, 10, 11, 13, 15, 16, 18, 19, 20, 21, 22	I/O	System Data bus: This bi-directional data bus provides the data path for all memory and I/O accesses. During transfers with 8-bit devices, the upper data byte is not used (SD[15:8]).
$\overline{\text{SBHE}}$	129	O	Byte High Enable. This active-low signal indicates that the high byte (odd address byte) is being transferred. External 16-bit devices should use this signal to help them determine that a data byte is to be transferred on the upper byte of the System Data bus (SD[15:8]). Eight-bit devices should ignore this signal. $\overline{\text{SBHE}}$ is sampled at the end of power good reset to determine if the boot ROM is 8 or 16-bit wide.
$\overline{\text{IOR}}$	124	O	IO Read command. This active-low signal instructs an I/O device to place data onto the system data bus.
$\overline{\text{IOW}}$	125	O	IO Write command. This active-low signal indicates to an I/O device that a write operation is in process on the system bus.
$\overline{\text{MEMR}}$	126	O	MEMory Read command. This active-low signal instructs a memory mapped device to place data onto the system data bus.
$\overline{\text{MEMW}}$	127	O	MEMory Write command. This active-low signal indicates to a memory mapped device that a write operation is in process on the system bus.
$\overline{\text{CS16}}$	122	I/O	Chip Select 16-bit. This active-low feedback signal indicates that the device being accessed is a 16-bit device. This signal should be driven by external devices with an open collector driver. If a chip select is programmed to force 16-bit accesses, this signal will be asserted (low) during the access.
RDY	123	I	ReaDY. An external device may drive this signal inactive low to insert wait states and extend the external bus cycle. This signal should be driven with an open collector or be TRI-STATE driven.

Table 1-2: DMA Control Pins

Symbol	Pins	Type	Function
DRQ[4], DRQ[3], DRQ[2], DRQ[0]	34, 32, 36, 38	I	DMA ReQuest. A DRQn signal requests the internal DMA Controller to transfer data between the Requesting Device and memory.
$\overline{\text{DACK}}[4]$, $\overline{\text{DACK}}[3]$, $\overline{\text{DACK}}[2]$, $\overline{\text{DACK}}[0]$	35, 33, 37, 39	O	DMA ACKnowledge: When the CPU has relinquished control of the bus to a requesting DMA channel, the appropriate active-low $\overline{\text{DACK}}n$ signal acknowledges the winning DRQn.
TC/EOP	40	I/O	Terminal Count/End Of Process: This signal may operate either as a terminal count output or an active-low End of Process input. As TC, an active-high pulse occurs on this signal when the terminal count for any DMA channel has been reached. As EOP, an external device may terminate the DMA transfer by driving this signal active-low.

Table 1-3: DRAM Control Pins

Symbol	Pins	Type	Function
$\overline{\text{RAS}}[1:0]$	30, 31	O	Row Address Strobe. On the falling edge of these active-low signals, Bank 1 and Bank 0 respectively, should latch in the row address off of SA[12:1]. If only one bank of DRAMs are supported, $\overline{\text{RAS}}0$ will support that bank and $\overline{\text{RAS}}1$ will be unused.
$\overline{\text{CASH}}[1:0]$	25, 26	O	Column Address Strobe (High Byte). These active-low signals indicate when the column access is being made to the high byte of DRAM Bank 1 and DRAM Bank 0 respectively. If only one bank of DRAMs are supported, $\overline{\text{CASH}}0$ will support the high byte of that bank and $\overline{\text{CASH}}1$ will be unused.
$\overline{\text{CASL}}[1:0]$	28, 29	O	Column Address Strobe (Low Byte). These active-low signals indicate when the column access is being made to the low byte of DRAM Bank 1 and DRAM Bank 0, respectively. If only one bank of DRAMs are supported, $\overline{\text{CASL}}0$ will support the low byte of that bank and $\overline{\text{CASL}}1$ will be unused.
WE	23	O	Write Enable. Active low signal for writing the data into the DRAM bank.
DPH, DPL	1, 12	I/O	DRAM Data Parity. DRAM data parity may be enabled or disabled; if disabled these two pins will be unused. Otherwise, for DRAM writes the NS486SXF 's DRAM Controller will generate odd parity and drive the odd parity onto these two pins. For DRAM reads the NS486SXF 's DRAM Controller will read the values driven on these two pins and check it for odd parity in association with the appropriate data byte.

Table 1-4: Power Pins

Symbol	Pins	Type	Function
V _{DD}	7, 17, 27, 47, 63, 87, 101, 131, 147, 157	I	+5V power to core and I/O.
V _{SS}	4, 14, 24, 44, 61, 84, 98, 128, 144, 154	I	Ground to core and I/O.

Table 1-5: Reset Logic Pins

RESET	119	O	RESET system output driver: This active high signal resets or initializes system peripheral logic during power up or during a low line voltage outage.
RESET	120	O	Inverse of RESET for peripherals requiring active low reset.
PWGOOD	60	I	PoWer GOOD . This active-high (schmitt trigger) input will cause a hardware reset to the NS486SXF whenever this input goes low. This pin will typically be driven by the power supply and PWGOOD will remain low until the power supply determines that stable and valid voltage levels have been achieved.

Table 1-6: Auxiliary Processor Interface Pins

Symbol	Pins	Type	Function
$\overline{\text{EREQ/CS6/RTS}}$	102	O	<p>This pin has three programmable options controlled by the Modem Signal Control Register (refer to the UART section):</p> <ol style="list-style-type: none"> 1) External bus REQuesT (active-low) to an auxiliary processor. 2) Chip Select 6 (active-low) pin. 3) Request To Send. When low, this signal informs the MODEM or data set that the UART is ready to exchange data. The $\overline{\text{RTS}}$ output signal can be set to an active low by programming bit 2 (RTS) of the MODEM Control Register. A Master Reset operation sets this signal to its inactive (high) state. Loop mode operation holds this signal in its inactive state.
$\overline{\text{EACK/CS7/DSR}}$	103	I/O I O I	<p>This pin has three possible programmable options controlled by the Modem Signal Control Register (refer to the UART section):</p> <ol style="list-style-type: none"> 1) External bus ACKnowledge (active-low) from an auxiliary processor. 2) Chip Select 7 (active-low) pin. 3) Data Set Ready. When low, it indicates that the MODEM or data set is ready to link with the UART. The $\overline{\text{DSR}}$ signal is a MODEM status input whose condition can be tested by the CPU reading bit 5 (DSR) of the MODEM Status Register. Bit 5 is the complement of the $\overline{\text{DSR}}$ signal. Bit 1 (DDSR) of the MODEM Status Register indicates whether the $\overline{\text{DSR}}$ input has changed state since the previous reading of the MODEM Status Register. <p>Note: Whenever the DSR bit of the MODEM Status Register changes state, an interrupt is generated if the MODEM Status Interrupt is enabled.</p>
$\overline{\text{DRV/CS8/DTR}}$	104	O	<p>This pin has three possible programmable options controlled by the Modem Signal Control Register (refer to the UART section):</p> <ol style="list-style-type: none"> 1) DSP shared memory DRiVe control signal. 2) Chip Select 8 (active-low) pin. 3) Data Terminal Ready. When low, this signal informs the MODEM or data set that the UART is ready to establish a communications link. The $\overline{\text{DTR}}$ output signal can be set to an active low by programming bit 0 (DTR) of the MODEM Control Register to a high level. A Master Reset operation sets this signal to its inactive (high) state. Loop mode operation holds this signal in its inactive state.

Table 1-7: Test Pins

Symbol	Pins	Type	Function
TEST	66	I/O	Reserved for testing and development system support.

Table 1-8: Interrupt Control Pins

Symbol	Pins	Type	Function
NMI	105	I	Non-Maskable Interrupt. This active-high signal will generate a non-maskable interrupt to the CPU when it is active high. Normally this signal is used to indicate a serious system error.
$\overline{\text{INTA}}$	106	O	INTerrupt Acknowledge. During each interrupt acknowledge cycle this signal will strobe low; it should be used by external cascaded interrupt controllers.
IRQ[5:0]	107, 108, 109, 110, 111, 112	I	Interrupt ReQuests. These inputs are either rising edge or low-level sensitive interrupt requests, depending on the configuration of the internal interrupt controllers. These interrupt requests may also be programmed to support externally cascaded interrupt controller(s). The IRQ pins are also used to select a particular test in test mode. If the PCMCIA controller is enabled, IRQ[5] becomes the $\overline{\text{IREQ}}$ signal.

Table 1-9: Real Time Clock Pins

Symbol	Pins	Type	Function
RTCX1	62	I	Real Time Clock crystal oscillator input: 32kHz crystal.
RTCX2	64	O	Real Time Clock crystal oscillator output: 32kHz crystal.
Vbat	65	I	External + battery input for real time clock.

Table 1-10: LCD Interface Pins

Symbol	Pins	Type	Function
LCD[3:0]	51, 52, 53, 54	O	Data Output Word to LCD , 1 = White, 0 = Blue/black.
CL2	50	O	Word CLock to LCD .
CL1	49	O	Row CLock to LCD .
CLF	48	O	Frame CLock to LCD .

Table 1-11: Oscillator Pins

Symbol	Pins	Type	Function
SYSCLK	121	O	SYStem CLocK. This clock output pin will either be driven with a signal half the frequency of the OSCX1 input clock frequency or the CPU's clock frequency, which is determined in the Power Management Control Register 1. The source selection for this signal is determined by bit 1 of the Power Management Control Register 3.
OSCX1	45	I	OSCillator Crystal 1 input. This pin should either be driven by a TTL oscillator or be connected to an external crystal circuit. This signal is the fundamental clock source for all clocked elements in the NS486SXF , except the Real-Time Clock, which has its own crystal pins.
OSCX2	46	O	OSCillator Crystal 2 output. This is the output side of the NS486SXF on-chip circuitry provided to support an external crystal circuit. If a TTL oscillator drives OSCX1, this pin should be a no connect.

Table 1-12: HP-SIR/UART Pins

Symbol	Pins	Type	Function
Tx	57	O	UART Transmit data. In HP-SIR mode this pin is the UART output encoded for the serial infrared link. Otherwise it is the transmit output of the 16550 UART.
Rx	58	I	UART Receive data. In HP-SIR mode this pin is routed through the serial infrared decoder. Otherwise, it is the receive input to the 16550.
UCLK	59	O	Uart CLoCK. Output of programmable rate UART/MODEM clock. Typically used for the Infrared Modulator

Table 1-13: PCMCIA pins

Symbol	Pins	Type	Function
CD_RST	68	O	CarD ReSeT. This active high signal resets the PCMCIA card during a soft-reset.
IOIS16/WP	67	I	IO port IS 16 bits/ Write Protect : When a PCMCIA card is configured as an IO card, this signal is asserted to indicate the currently addressed IO port is 16-bits wide. When a PCMCIA card is configured as a memory card, an active high signal indicates the card is currently write protected.
BVD2/SPKR	74	I	Battery Voltage Detect bit 2/ SPeAKeR output. When a PCMCIA card is configured as a memory card, this input along with BVD[1] will provide status information about the card's on-board battery condition. When a PCMCIA card is configured as an IO card, this pin will act as the audio output of the card to the system.
BVD1/STSCNG	73	I	Battery Voltage Detect bit 1/ STatuS ChaNGe output. When a PCMCIA card is configured as a memory card, this input along with BVD[2] will provide status information about the card's on-board battery state. When a PCMCIA card is configured as an I/O card, the status change signal indicates one or more of the memory status signals (BVD[2:1], WP, RDY or $\overline{\text{BSY}}$) has changed states.
$\overline{\text{V}}_{\text{CC_SEL}}$	69	O	PCMCIA V_{CC} SElect . When this signal is low, the V_{CC} power to the PCMCIA card should be enabled.
Vpp_SEL1, Vpp_SEL2	70, 71	O	PCMCIA Vpp SElect 1 and 2. These signals indicate the voltage with which the Vpp power to the PCMCIA card should be driven.
$\overline{\text{GPI}}$	72	I	General Purpose Input . This signal is a general purpose input signal used with a PCMCIA card to indicate a valid VPP state, a pending card eject/insertion, or as an interrupt source.
$\overline{\text{CD}}_2, \overline{\text{CD}}_1$	76, 75	I	Card Detect . Both signals are low when the PCMCIA card is correctly inserted.
DIR	77	O	DIR ection. Used to control the direction of the data line buffers to the PCMCIA interface.
$\overline{\text{ENABLE}}$	78	O	ENABLE PCMCIA. Enables the buffer drivers to the PCMCIA interface. Low true signal.
REG	79	O	REG . PCMCIA card support.

Note: If PCMCIA is enabled, Chip Selects 1 and 2 become Card Enable 1 and 2. See Table 1-17, "General Purpose Chip Select Pins," on page 10. Also, IRQ[5] becomes the PCMCIA $\overline{\text{IREQ}}$ signal.

Table 1-14: IEEE-1284 Port (ECP Mode)

Symbol	Pins	Type	Function
PD[7:0]	81, 82, 83, 85, 86, 88, 89, 90	O/I	Parallel Data. Bi-directional data pins transfer data and address information to and from the parallel port.
SLIN	91	O/I	SeLect INput: Used in a closed-loop handshake with BUSY to transfer data or address information from the host to the peripheral. Host driven.
STB	92	O/I	data STroBe. Driven high by the host while in ECP Mode. Asserted low by host to terminate ECP Mode and return link to Compatibility Mode. Host driven.
$\overline{\text{AFD}}$	93	O/I	Automatic FeED. The host asserts this line low for flow control in the reverse direction. It is used in a interlocked handshake with $\overline{\text{ACK}}$. Provides command information in the forward direction. Host driven. Active low.
INIT	94	O/I	INITialize. When this signal is asserted low to place the data channel in the reverse direction, the peripheral is allowed to drive the data bus. Host driven. Active low.
$\overline{\text{ACK}}$	95	I/O	ACKnowledge. Used in closed-loop handshake with $\overline{\text{AFD}}$ to transfer data to the host. Peripheral device drive. Active low.
PE	96	I/O	Peripheral Error. Asserted low to acknowledge INIT , reverse request. Peripheral device drive.
SLCT	97	I/O	SeLeCT. Asserted high when selected or indicating an affirmative response for each respective extensibility byte. Peripheral device drive. Active high.
ERR	99	I/O	ERROR. This input is asserted low by the peripheral to request host communications. Valid only in the forward direction. Peripheral device drive. Active low.
BUSY	100	I/O	BUSY. This is asserted low by the peripheral for flow control in the forward direction, de-asserted to acknowledge transfer of data or address completion. Peripheral device drive. Active low.

Table 1-15: Timer Pins

Symbol	Pins	Type	Function
T0	55	I/O	Programmable Timer pin 0. This Bidirectional pin may be selected to control one of the following four functions via bits 1-0 of the Timer I/O Control Register: 1) The GATE input into Timer 0. 2) The GATE input into Timer 1. 3) The OUT output from Timer 0. 4) The CLK input into Timer 1.
T1	56	I/O	Programmable Timer pin 1. This Bidirectional pin may be selected to control one of the following four functions via bits 3-2 of the Timer I/O Control Register: 1) The GATE input into Timer 0. 2) The GATE input into Timer 1. 3) The OUT output from Timer 1. 4) The CLK input into Timer 0.

Table 1-16: 3-Wire Serial I/O Pins

Symbol	Pins	Type	Function
SO/ $\overline{\text{DCD}}$	41	I/O	<p>This pin has two possible programmable options controlled by the Modem Signal Control Register (refer to the UART section):</p> <ol style="list-style-type: none"> 1) The Serial data Output signal for MICROWIRE. 2) Data Carrier Detect. When low, this input signal indicates that the data carrier has been detected by the MODEM or data set. The $\overline{\text{DCD}}$ signal is a MODEM status input whose condition can be tested by the CPU reading bit 7 ($\overline{\text{DCD}}$) of the MODEM Status Register. Bit 7 is the complement of the $\overline{\text{DCD}}$ signal. Bit 3 ($\overline{\text{DDCD}}$) of the MODEM Status Register indicates whether the $\overline{\text{DCD}}$ input has changed state since the previous reading of the MODEM Status Register. $\overline{\text{DCD}}$ has no effect on the receiver. <p>Note: Whenever the DCD bit of the MODEM Status Register changes state, an interrupt is generated if the MODEM Status Interrupt is enabled.</p>
SI/ $\overline{\text{CTS}}$	42	I/O	<p>This pin has two possible programmable options controlled by the Modem Signal Control Register (refer to the UART section):</p> <ol style="list-style-type: none"> 1) The Serial data Input signal for MICROWIRE or the serial data I/O for Access.bus. 2) Clear To Send. When low, this input signal indicates that the MODEM or data set is ready to exchange data. The $\overline{\text{CTS}}$ signal is a MODEM status input whose conditions can be tested by the CPU reading bit 4 ($\overline{\text{CTS}}$) of the MODEM Status Register. Bit 4 is the complement of the $\overline{\text{CTS}}$ signal. Bit 0 ($\overline{\text{DCTS}}$) of the MODEM Status Register indicates whether the $\overline{\text{CTS}}$ input has changed state since the previous reading of the MODEM Status Register. $\overline{\text{CTS}}$ has no effect on the Transmitter. <p>Note: Whenever the CTS bit of the MODEM Status Register changes state, an interrupt is generated if the MODEM Status Interrupt is enabled.</p>
SCLK/ $\overline{\text{RI}}$	43	I/O O I	<p>This pin has two possible programmable options controlled by the Modem Signal Control Register (refer to the UART section):</p> <ol style="list-style-type: none"> 1) The Serial CLocK signal for MICROWIRE and Access.bus. 2) Ring Indicator. When low, this input signal indicates that a telephone ringing signal has been received by the MODEM or data set. The $\overline{\text{RI}}$ signal is a MODEM status input whose condition can be tested by the CPU reading bit 6 ($\overline{\text{RI}}$) of the MODEM Status Register. Bit 6 is the complement of the $\overline{\text{RI}}$ signal. Bit 2 ($\overline{\text{TERI}}$) of the MODEM Status Register indicates whether the $\overline{\text{RI}}$ input signal has changed from a low to high state since the previous reading of the MODEM Status Register. <p>Note: Whenever the CTS bit of the MODEM Status Register changes state, an interrupt is generated if the MODEM Status Interrupt is enabled.</p> <p>Note: Whenever the RI bit of the MODEM Status Register changes from a high to a low state, an interrupt is generated if the MODEM Status Interrupt is enabled.</p>

Note: For MICROWIRE Slave Mode, a pin must be selected to be the Chip Select Input.

Table 1-17: General Purpose Chip Select Pins

Symbol	Pins	Type	Function
$\overline{CS}[0]$	118	O	Chip Select 0: This output is used as the chip-select for the system boot ROM. It defaults to the upper 64k Bytes of memory.
$\overline{CS}[5:1]$	113, 114, 115, 116, 117	I/O	Chip Select 1-5. These pins can be programmed to be either memory or I/O mapped chip selects, which are used for glue-less connection to external peripherals. When the PCMCIA Controller is enabled $\overline{CS}[1]$ and $\overline{CS}[2]$ become PCMCIA Card Enable outputs 1 and 2 ($\overline{CE1}$ and $\overline{CE2}$, respectively).

Table 1-18: Summary of Reconfigurable I/O Pins

Symbol	Pins	Type	Pin #	Original Function	Power Up State
REG	1	I/O	79	PCMCIA	TRISTATE
ENABLE	1	I/O	78	PCMCIA	1
DIR	1	I/O	77	PCMCIA	0
GPI	1	I/O	72	PCMCIA	TRISTATE
V _{pp} _SEL2	1	I/O	71	PCMCIA	0
V _{pp} _SEL1	1	I/O	70	PCMCIA	0
V _{CC} _SEL	1	I/O	69	PCMCIA	1
CD_RST	1	I/O	68	PCMCIA	TRISTATE
CLF	1	I/O	48	LCD	0
CL2	1	I/O	50	LCD	0
CL1	1	I/O	49	LCD	0
LCD [3:0]	4	I/O	51, 52, 53, 54	LCD	0, 0, 0, 0
PD [7:0]	8	I/O	81, 82, 83, 85, 86, 88, 89, 90	ECP	TRISTATE
Rx	1	I/O	58	UART	TRISTATE
UCLK	1	I/O	59	UART	Oscillating
$\overline{CS}[4]$	1	I/O	114	CS4	1
$\overline{CS}[3]$	1	I/O	115	CS3	1
$\overline{CS}[2]$	1	I/O	116	CS2	1
$\overline{CS}[1]$	1	I/O	117	CS1	1

These 29 pins, typically used for various I/O peripheral purposes, as defined in the above tables, can be reconfigured for use as general purpose I/O pins if the normally defined I/O function is not required.

2.0 System Overview

2.1 NS486SXF System Overview

The NS486SXF is a highly integrated embedded system controller. It includes an Intel486-class 32-bit processor, all resources required for the System Service Elements of a Real-Time Executive, and a generous set of peripherals. This “system-on-a-chip” is ideal for implementing a wide variety of embedded applications. These include (but are not limited to) fax machines, multifunction peripherals (fax, scanners, printers) mobile companions (both organizer and communicator), television set-top boxes, and telephones (mobile and desktop).

The 32-bit processor core executes all of the Intel486 instructions with a similar number of clocks per instruction. An on-board 1Kbyte instruction cache pro-

vides for efficient execution from ROM. Intel486 debug features are supported. The processor has been optimized for operating system kernels such as VRTX, VxWorks, pSOS+ and QNX. These environments only need the '486 protected mode operation (no real mode or virtual 8086 support), flat or linear memory addressing (no virtual memory paging), and floating point execution in software only (no co-processor interface).

In fact, the NS486SXF includes all of the System Service Elements required by a typical kernel, including an efficient DRAM controller that supports page-mode DRAMs for data cache-like performance; a six-channel DMA controller with two channels support-

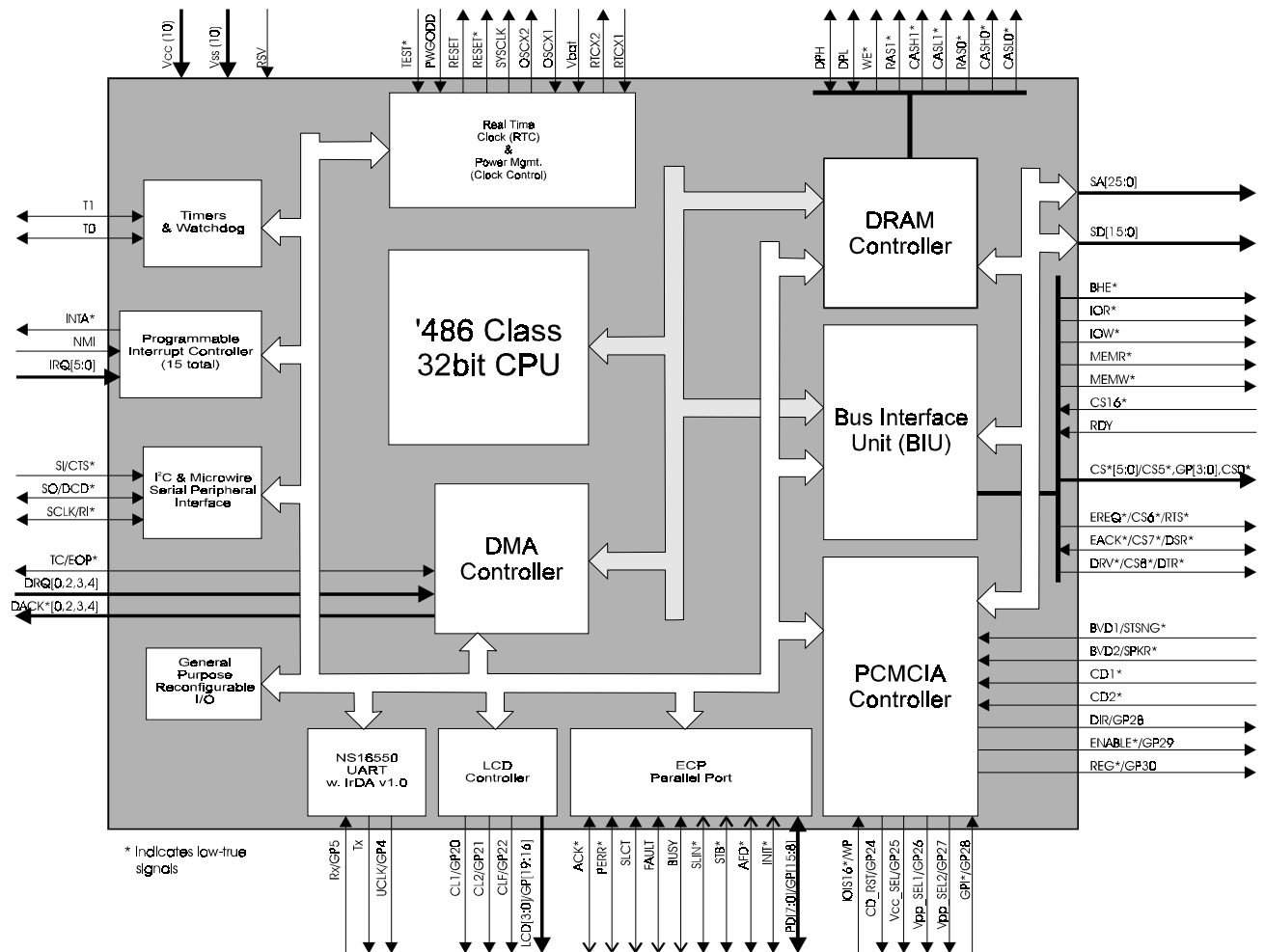


Figure 2-1 NS486SXF Internal Resource to Pins Map

ing data transfers from on-chip peripherals (the IEEE 1284 ECP or Extended Capabilities Port, and the LCD controller), and four channels supporting external devices such as scanners, and print engines; three timer channels (including one configured as a protected WATCHDOG Timer); two programmable 8259 interrupt controllers provide 15 on-chip interrupt sources; an industry standard real time clock and calendar (RTC) with battery backup; and support for comprehensive power management schemes.

In addition, the **NS486SXF** also incorporates the key I/O peripherals required for implementing a wide variety of embedded applications: an IEEE 1284 Bidirectional Parallel Port that includes both Host and Slave modes, an Intel 82365-compatible PCMCIA controller for one card slot, an industry standard high-performance NS16550-compatible UART with HP-SIR and IrDA v1.0 infrared option, an LCD panel interface with DMA supported refresh for many of the standard resolutions, an 8254 timer, and a general purpose 2- or 3-wire synchronous serial interface for easy interface to low-cost EEPROMs and other serial peripherals. System expansion is supported with nine programmable Chip Select (CS) signals and a generic ISA-type bus interface for external devices and memory.

Certain I/O lines not being used by disabled peripherals can be reconfigured for use as general purpose bidirectional I/O lines (up to 29 pins). This gives the designer maximum flexibility in designing various systems using the **NS486SXF** device. It is expected that an **NS486SXF** system will minimally include the **NS486SXF** system controller with on-board processor and I/O devices, boot ROM, and working RAM memory. Many applications will not require any additional I/O support.

Finally, the **NS486SXF** implements a very flexible power management scheme that permits selective control of individual I/O subsystems, with varying levels of power consumption.

NS486SXF provides a cost-effective hardware platform for the design and implementation of a wide range of office automation and communication systems. With its powerful embedded '486-class processor, comprehensive set of on-chip peripheral controllers, flexible power management structure and

reconfigurable I/O lines, **NS486SXF** makes possible a variety of end-user systems based on the same hardware. Because of its optimized design and on-board resources, a very cost effective system can be achieved.

2.2 32-bit Processor Core

The NS486SXF processor core is an implementation of the protected mode '486 instruction set architecture, optimized using a RISC-like design philosophy for embedded applications. Using this approach, the most frequently used instructions are optimized, and on an average execute in a lower number of clock cycles than a '486.

The NS486SXF features a three stage pipeline, efficient instruction prefetching mechanism, and single cycle instruction decoding for most instructions. Additionally, a 1K byte instruction cache and single cycle DRAM access provide higher memory performance than a larger unified cache implementation.

The NS486SXF processor provides the same programming model and register set as the standard '486 except that real mode, virtual memory, and floating point support have been eliminated. These features have little or no impact in embedded applications and save significant silicon real estate. At reset, unlike the standard '486, the NS486SXF starts up in protected mode instead of real mode. All '486 instructions appropriate to protected mode and our hardware configuration are supported, including debug instructions.

The NS486SXF is initially available to run 25 MHz at 5 Volts. The processor clock is obtained by dividing the crystal frequency by two. For example, a 25 MHz NS486SXF runs with a 50 MHz crystal oscillator as the master clock.

As a result of our innovative design, the NS486SXF achieves performance equivalent to a standard '486 with less circuitry. This translates into reduced power consumption and a lower overall system cost. It also makes the NS486SXF ideal for "green" systems and battery operated systems.

2.3 System Service Elements

The **NS486SXF** controller provides the basic hardware resources required for the O/S-defined System Service Elements. These include a DRAM controller, a DMA controller, programmable interval timer, a protected WATCHDOG timer, a programmable interrupt controller, a real-time clock and calendar, and comprehensive power management features.

2.3.1 DRAM Controller

The **NS486SXF** DRAM controller supports one or two adjustable-sized banks of dynamic RAM using a 16-bit data path. Support is provided for byte parity (if desired), requiring the DRAM banks to be 18-bits wide when parity is enabled. Banks can be up to 8 Mbytes in size. The DRAM controller supports page mode read and write operations and can also support both byte and word accesses. All access control signals for read, write and parity checking are generated as well as an automatic and programmable CAS-before-RAS refresh. If self-refresh DRAMs are used, refresh can be disabled, saving power.

NS486SXF provides flexible support for use of a number of different DRAM configurations, using popular DRAM devices. Access is optimized for fast page mode DRAMs, and they will provide the highest performance with contiguous data. When accessing data bytes or words in the same DRAM page, the data access is in one cycle. This performance provides fast data access times without the overhead of a separate data cache. Page sizes can be 512, 1024, 2048 or 4096 bytes. Flexibility for DRAM timing is provided through programming of the DRAM controller registers: 3 or 4 cycle page miss accesses and extended CAS cycles can be selected.

Memory bank 0 starts at address 0h; memory bank 1 can start at any address in the 128 Mbyte address map that is a multiple of its size.

2.3.2 DMA Controller

The **NS486SXF** Direct Memory Access (DMA) controller is a high speed 16-bit controller that improves system performance by off-loading from the processor the task of managing data transfers to and from memory and external devices. Data transfers are done independently from the processor at a maximum data rate of 2 bytes per 2 clock cycles. (A 25 MHz clock yields a 25 megabyte per second transfer rate.)

There are six independent DMA channels. Requestor and target addresses have a maximum addressable memory range of 64 Mbytes. Three standard transfer modes, single, block and demand, are provided giving the designer a wide range of DMA options. A special transfer type, cascade-master, allows an external master to access the **NS486SXF** ISA-like bus. Normal transfers can be from memory to memory, memory to I/O and I/O to memory. DMA transfers are controlled by DMA control registers in the **NS486SXF** control register I/O map.

2.3.3 Programmable Interval Timer

The **NS486SXF** programmable interval timer is compatible with the Intel 8254 programmable interval timer and contains three identical timers (CH0-CH2). CH0 and CH1 can be used to generate accurate timing delays under software control. CH2 may be configured to provide a WATCHDOG timer function.

2.3.4 WATCHDOG Timer

The **NS486SXF** WATCHDOG timer, CH2, is a protected 16-bit timer that can be used to prevent system “lockups or hangups.” It uses a 1 KHz clock generated by the on-chip real-time clock circuit. If the WATCHDOG timer is enabled and times out, a reset or interrupt will be generated allowing graceful recovery from an unexpected system lockup.

2.3.5 Interrupt Controller

The **NS486SXF** interrupt controller consists of two cascaded programmable interrupt controllers that are compatible with the Intel 8259A Programmable Interrupt Controller. They provide a total of 15 (out of 16) programmable interrupts. Three interrupts are reserved for a real time clock-tick interrupt, a real time clock interrupt request, and a cascade interrupt channel. The remaining 13 interrupts can be used by internal or external sources. Additional external interrupt controllers can be cascaded as well.

2.3.6 Real Time Clock/Calendar

The **NS486SXF** Real Time Clock/Calendar is a low power clock that provides a time-of-day clock and 100-year calendar with alarm features and battery operation. Time is kept in BCD or binary format. It includes 50 bytes of general purpose CMOS RAM and 3 maskable interrupt sources. It is compatible with the DS1287 and MC146818 RTC/Calendar devices, except for the general purpose memory size.

2.3.7 Power Management Features

The **NS486SXF** power management structure includes a number of power saving mechanisms that can be combined to achieve comprehensive power savings under a variety of system conditions. First of all, the core processor power consumption can be controlled by varying the processor/system clock frequency. The internal CPU clock can be divided by 4, 8, 16, 32 or 64. In addition, in idle mode, the internal processor clock will be disabled. Finally, if an external crystal oscillator circuit is being used, it can be disabled. For maximum power savings, all internal clocks can be disabled (except for the real-time clock oscillator).

The clocks of the on-board peripherals can be individually or globally controlled. By setting bits in the power management control registers, the internal clocks to the DMA controller, the ECP port, the three-wire interface, the timer, the LCD controller, the DRAM controller, the PCMCIA controller and the UART can be disabled.

In addition to these internal clocks, the external SY-SCLK can be disabled via a bit in the power management control registers.

Using various combinations of these power saving controls with the **NS486SXF** controller will result in excellent programmable power management for any application.

2.4 NS486SXF System Bus

The **NS486SXF** system bus provides the interface to off-chip peripherals and memory. It offers an ISA-compatible interface and is therefore capable of directly interfacing to many ISA peripheral control devices. The interface is accomplished through the Bus Interface Unit (BIU). The BIU generates all of the access signals for both internal and external peripherals and memory. Depending upon whether the access is to internal peripherals, external peripherals or external memory, the BIU generates the timing and control signals to access those resources. The BIU is designed to support a glueless interface to many ISA-type peripherals.

For debug purposes, the **NS486SXF** can be set to generate external bus cycles at the same time as an internal peripheral access takes place. This gives logic analyzers or other debug tools the ability to track and capture internal peripheral accesses.

Access to internal peripherals is accomplished in three CPU T-states (clock cycles). The fastest access to off-chip I/O is also three T-states. When accessing off-chip memory and I/O, wait state generation is accomplished through a combination of **NS486SXF** chip select logic and off-chip peripheral feedback signals.

When the CPU is in idle mode, the BIU is designed to mimic the CPU during DMA interchanges between memory and peripherals. By responding to DRQs and generating DACK, and HOLDA signals as required, the BIU eliminates the need to reactivate the CPU during such transfers as screen updates from memory to the LCD controller. This gives the designer added flexibility in conserving power while maintaining basic system functions.

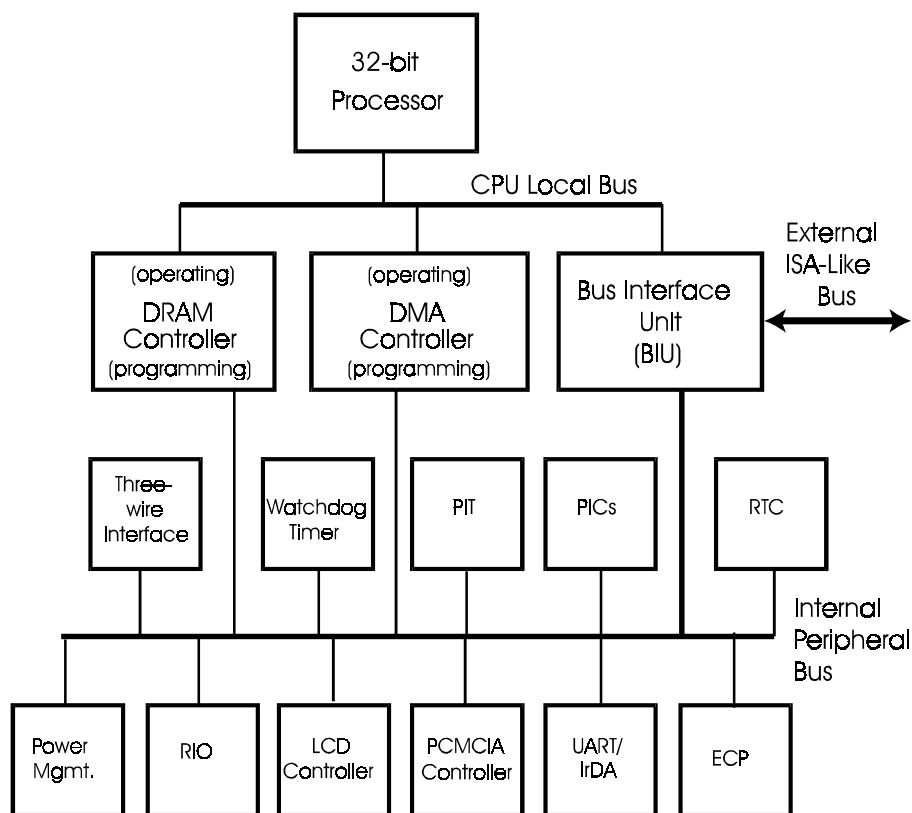


Figure 2-1 NS486SXF Internal Busses

2.5 Other On-board Peripherals

In addition to those peripherals and system control elements needed for System Service Elements, the **NS486SXF** also includes a number of I/O controllers and resources that make implementing a complete embedded system possible with just a single-chip **NS486SXF** controller. These include an IEEE 1284 Extended Capabilities Port, a serial UART port, a LCD controller, a PCMCIA interface and a MICROWIRE or Access.bus synchronous serial bus interface. In addition, unused I/O controllers free up their I/O pins for general purpose use.

2.5.1 Reconfigurable I/O Lines

The **NS486SXF** supports reconfigurable I/O. For example, if the UART, ECP Parallel Port, LCD or PCMCIA functions are not being used, the I/O pins associated with them can be reconfigured as general purpose bidirectional I/O pins. Up to 29 pins can be reconfigured for this purpose. This capability makes the **NS486SXF** extremely versatile and ideal for supporting different end product configurations with a single **NS486SXF** device.

2.5.2 IEEE 1284 Bidirectional Port

The **NS486SXF** parallel port is a multifunction 8-bit parallel port that is compatible with the IEEE 1284 bidirectional parallel port standard. The operation of the parallel port is set by the content of the **NS486SXF** parallel port I/O control registers. The port can operate in one of two modes: a standard parallel port mode (PC compatible), or a full Extended Capabilities Port (ECP) mode. The **NS486SXF** ECP port can support both Host and Slave ECP mode. In slave mode, the **NS486SXF** becomes a versatile microprocessor for parallel I/O peripheral devices.

2.5.3 PCMCIA Interface

The **NS486SXF** PCMCIA interface supports the direct connection of a single PCMCIA 2.0 IC card. Exchange Card Architecture (ExCA release 1.50) compatibility and eXecute In Place (XIP) capability is also provided.

Accessing the PCMCIA interface switches the external bus automatically into the PCMCIA mode and permits Memory Window Mapping and Address Offset to be handled inside the **NS486SXF** device. Power

management and “hot” card insertion/removal options can be implemented using external buffering, if required.

2.5.4 MICROWIRE/Access.bus Interface

The **NS486SXF** MICROWIRE/Access.bus interface provides for full support of either the three-wire MICROWIRE or the two-wire Access.bus serial interfaces. MICROWIRE has an alternate clock phasing option that supports the SPI bus protocol as well. These industry standard interfaces permit easy interfacing to a wide range of low-cost specialty memories and I/O devices. These include EEPROMs, SRAMs, timers, clock chips, A/D converters, D/A converters, and peripheral device drivers.

2.5.5 UART Serial Port

The **NS486SXF** UART provides complete NS16550 (PC standard) serial communications port compatibility including the performance enhancing 16-byte deep FIFO. It performs serial-to-parallel conversion from external devices to the **NS486SXF** and parallel-to-serial conversion from the **NS486SXF** to external peripherals. Full modem control can be supported.

A serial IrDA v1.0 and HP-SIR (infrared) mode is also supported, making possible low-cost wireless communications between an **NS486SXF**-based system and other wireless infrared systems.

2.5.6 LCD Controller

The **NS486SXF** LCD controller is capable of controlling a variety of monochrome supertwist LCD configurations including 320x240, 320x200 and 480x320 black & white or grayscale graphics LCD modules equipped with self-contained screen drivers. It uses a video frame buffer in system DRAM with either a 1- or 2-bit per pixel grayscale. A 60 to 90 Hz frame refresh rate is supported. Special controls permit the fine tuning of display characteristics to precisely optimize visual display quality.

2.6 ICE Support

National Semiconductor has worked closely with Microtek International to provide hardware in-circuit emulator support for the **NS486SXF**. The Microtek product (PowerPack[®] EA-NS486) uses a special bondout version of the **NS486SXF** to deliver a full-featured hardware emulator that is capable of tracing on chip activity, including peripheral interrupt and I/O activity. The emulator runs at full speed, and supports overlay memory and multiple triggers.

PowerPack is a registered trademark of Microtek International.

Also, there are many third party tool sets that will allow an executable application to be built to run directly on the target hardware without an O/S environment.

2.7 Other Issues

NS486SXF provides a comprehensive set of on-board peripherals. Also, it is designed to easily interface to external peripherals. In addition to this ISA-like bus which supports ISA-compatible peripherals, the **NS486SXF** provides an interface to an external master with a shared memory space. The external master or auxiliary processor interface allows low cost interfacing to shared external memory belonging to other external masters (including another **NS486SXF** controller).

To program the resources of the **NS486SXF**, a set of internal control registers exists. These registers provide precise control over all internal resources and the setup of external **NS486SXF** control signals. It is the designer's responsibility to ensure the proper initialization of the registers in this I/O map.

In addition, the **NS486SXF** core processor itself requires several descriptor tables and initialization parameters that must be set by user-written start-up software.

The **NS486SXF** is designed from the ground up for optimum price/performance in embedded systems. This makes the **NS486SXF** the logical choice as the base hardware platform for executing an embedded operating system kernel such as those available from Microtec International, Wind River, ISI, QNX, and many others. Any Operating System or Real-Time Executive that will operate in a segmented or flat memory model protect mode environment is a suitable complement to the **NS486SXF**.

Deliberately Blank Page

3.0 Programming the NS486SXF

3.1 Overview

NS486SXF resources are setup and controlled via a set of control registers located in the I/O map. It is the system designer's responsibility to configure the various I/O devices and other elements of the NS486SXF hardware through startup software routines.

Like other '486-type processors, the NS486SXF has a 64Kbyte I/O address space map. In the NS486SXF, the I/O address space addresses physical memory and I/O ports are accessed using the IN and OUT I/O instructions.

For more information on the NS486SXF core processor and its addressing modes see Section 5, CPU Overview.

As shown in the figure, the NS486SXF I/O map is logically divided into several segments, each containing various control registers for the hardware elements of the NS486SXF single-chip system. To provide some continuity for the programmer familiar with the standard personal computer DOS/MS-DOS/Windows environment, key address locations for the UART, interrupts, real time clock, etc., have been retained from that I/O model wherever possible. Where continuity is not possible, or new resources are available, new locations are used. In particular some DMA controller resources have been placed in discontinuous locations due to historical and space constraint reasons. This results in a fragmented map for the DMA controller.

The complete NS486SXF address map of the control registers is shown in the tables on the following pages.

* not to scale

Starting Address	Resource
FFFF	
EFE0	DMA Controller
EFC0	RIO
EFB1	Interrupt Controller
EFA0	LCD Controller
EF90	Power Management
EF80	DRAM Controller
EF00	Bus Interface Unit
07BC	ECP
0778	ECP
0678	ECP
03F8	UART
03E8	UART
03E2	PCMCIA
03E0	PCMCIA
03BC	ECP
0378	ECP
02F8	UART
02E8	UART
0278	ECP
00D0	DMA Controller
00C0	DMA Controller
00A0	Slave Interrupt Controller
0080	DMA Controller
0070	Real Time Clock
0050	3-wire Interface
0040	Programmable Interval Timer
0020	Master Interrupt Controller
0000	DMA Controller

* all numbers in hexadecimal

Figure 3-1 NS486SXF I/O Utilization

3.2 I/O Address Map

The following tables list all of the registers within the NS486SXF, their respective I/O address locations and where more information about them can be found.

Table 3-1: I/O Address Map

I/O Address	Register Name	Reference Section
0000h - 0001h	DMA Channel 0 Base and Current Address Register (DRAM)	DMA Controller
0002h - 0003h	DMA Channel 1 Base and Current Address Register (DRAM)	DMA Controller
0004h - 0005h	DMA Channel 2 Base and Current Address Register (DRAM)	DMA Controller
0006h - 0007h	DMA Channel 3 Base and Current Address Register (DRAM)	DMA Controller
0008h	DMA Command Register	DMA Controller
0009h	DMA Channels 3-0 Data Mode/Request Status Registers	DMA Controller
000Ah	DMA TC Status Register	DMA Controller
000Bh	DMA Channels 3-0 Channel Mode Register	DMA Controller
000Ch	Reserved	DMA Controller
000Dh	DMA Master Clear Command	DMA Controller
000Eh	DMA Clear All Mask Command	DMA Controller
000Fh	DMA Mask Register	DMA Controller
0010h - 0011h	DMA Channel 0 Base and Current Address Register (requestor)	DMA Controller
0012h - 0013h	DMA Channel 2 Base and Current Address Register (requestor)	DMA Controller
0014h - 0015h	DMA Channel 4 Base and Current Address Register (requestor)	DMA Controller
0016h - 0017h	Reserved	DMA Controller
0018h	DMA Channel 0 High page address register (target)	DMA Controller
0019h	DMA Channel 1 High page address register (target)	DMA Controller
001Ah	DMA Channel 2 High page address register (target)	DMA Controller
001Bh	DMA Channel 3 High page address register (target)	DMA Controller
001Ch	DMA Channel 4 High page address register (target)	DMA Controller
001Dh	DMA Channel 5 High page address register (target)	DMA Controller
001Eh	Reserved	DMA Controller
001Fh	Reserved	DMA Controller
0020h	Interrupt Controller (I/O Port 0)	Master Interrupt Controller (7-0)
0021h	Interrupt Controller (I/O Port 1)	Master Interrupt Controller (7-0)
0040h	Counter 0 Register	Prog. Interval Timer
0041h	Counter 1 Register	Prog. Interval Timer
0042h	Counter 2 Register	Prog. Interval Timer
0043h	Control Word Register	Prog. Interval Timer
0044h	Timer I/O Control Register	Prog. Interval Timer

Table 3-1: I/O Address Map

I/O Address	Register Name	Reference Section
0045h	Timer Clock Register	Prog. Interval Timer
0046h	WATCHDOG Retrigger	WATCHDOG Timer
0047h	WATCHDOG Timer Control Register	WATCHDOG Timer
0050h	System Control Register	3-Wire Interface
0051h	Serial Input/Output Register	3-Wire Interface
0052h	MICROWIRE Control Register	3-Wire Interface
0053h	Serial Input/Output DATA Register	3-Wire Interface
0054h	Access.bus Status Register	3-Wire Interface
0055h	Access.bus Control Register	3-Wire Interface
0056h	Own Address Register	3-Wire Interface
0057h	Access.Bus Control Register 2	3-Wire Interface
0070h	Real Time Clock Index Register	Real Time Clock
0071h	Real Time Clock Data Port	Real Time Clock
0080h	DMA Channel 0 Low Page Address Register (target)	DMA Controller
0081h	DMA Channel 1 Low Page Address Register (target)	DMA Controller
0082h	DMA Channel 2 Low Page Address Register (target)	DMA Controller
0083h	DMA Channel 3 Low Page Address Register (target)	DMA Controller
0084h	DMA Channel 4 Low Page Address Register (target)	DMA Controller
0085h	DMA Channel 5 Low Page Address Register (target)	DMA Controller
0086h	Reserved	DMA Controller
0087h	Reserved	DMA Controller
0088h - 0089h	DMA Channel 0 Page Address Register (requestor)	DMA Controller
008Ah - 008Bh	DMA Channel 2 Page Address Register (requestor)	DMA Controller
008Ch - 008Dh	DMA Channel 4 Page Address Register (requestor)	DMA Controller
008Eh - 008Fh	Reserved	DMA Controller
00A0h	Interrupt Controller (I/O Port 0)	Slave Interrupt Controller (15-8)
00A1h	Interrupt Controller (I/O Port 1)	Slave Interrupt Controller (15-8)
00C0h - 00C1h	DMA Channel 4 Base and Current Address Register (target)	DMA Controller
00C2h - 00C3h	DMA Channel 3 Base and Current Address Register (requestor)	DMA Controller
00C4h - 00C5h	DMA Channel 5 Base and Current Address Register (target)	DMA Controller
00C6h - 00C7h	DMA Channel 3 Page Address Register (requestor)	DMA Controller
00C8h - 00C9h	Reserved	DMA Controller
00CAh - 00CBh	Reserved	DMA Controller
00CCh - 00CDh	DMA Channel 0 Base and Current Byte Count Register	DMA Controller

Table 3-1: I/O Address Map

I/O Address	Register Name	Reference Section
00CEh - 00CFh	DMA Channel 1 Base and Current Byte Count Register	DMA Controller
00D0h - 00D1h	DMA Channel 2 Base and Current Byte Count Register	DMA Controller
00D2h - 00D3h	DMA Channel 3 Base and Current Byte Count Register	DMA Controller
00D4h - 00D5h	DMA Channel 4 Base and Current Byte Count Register	DMA Controller
00D6h - 00D7h	DMA Channel 5 Base and Current Byte Count Register	DMA Controller
00D8h - 00D9h	Reserved	DMA Controller
00DAh	DMA Channels 3-0 Chaining Mode Register	DMA Controller
00DBh	DMA Channels 5-4 Data Mode Register	DMA Controller
00DCh	DMA Channels 5-4 Channel Mode Register	DMA Controller
00DDh	DMA Channels 5-4 Chaining Mode Register	DMA Controller
00DEh	DMA Chaining Mode Channel Status Register	DMA Controller
00DFh	DMA Chaining Mode Base Empty Status Register	DMA Controller
0278h	DATAR or AFIFO (<i>option 1</i>)	ECP
0279h	DSR (<i>option 1</i>)	ECP
027Ah	DCR (<i>option 1</i>)	ECP
02E8h - 02EFh	COM4 (<i>option</i>)	UART
02F8h - 02FFh	COM2 (<i>option</i>)	UART
0378h	DATAR or AFIFO (<i>option 2</i>)	ECP
0379h	DSR (<i>option 2</i>)	ECP
037Ah	DCR (<i>option 2</i>)	ECP
03BCh	DATAR or AFIFO (<i>option 3</i>)	ECP
03BDh	DSR (<i>option 3</i>)	ECP
03BEh	DCR (<i>option 3</i>)	ECP
03E0h	PCMCIA Controller Index Register	PCMCIA
03E1h	PCMCIA Controller Data Register	PCMCIA
03E2h	PCMCIA Controller Index Register (<i>option</i>)	PCMCIA
03E3h	PCMCIA Controller Data Register (<i>option</i>)	PCMCIA
03E8h - 03EFh	COM3 (<i>option</i>)	UART
03F8h - 03FFh	COM1 (<i>default UART location</i>)	UART
0678h	CFIFO or DFIFO or TFIFO or CNFGA (<i>option 1</i>)	ECP
0679h	CNFGB (<i>option 1</i>)	ECP
067Ah	ECR (<i>option 1</i>)	ECP
067Bh	INDEX (<i>option 1</i>)	ECP
067Ch	SETUP (<i>option 1</i>)	ECP
067Dh	IPR (<i>option 1</i>)	ECP

Table 3-1: I/O Address Map

I/O Address	Register Name	Reference Section
0778h	CFIFO or DFIFO or TFIFO or CNFGA (<i>option 2</i>)	ECP
0779h	CNFGB (<i>option 2</i>)	ECP
077Ah	ECR (<i>option 2</i>)	ECP
077Bh	INDEX (<i>option 2</i>)	ECP
077Ch	SETUP (<i>option 2</i>)	ECP
077Dh	IPR (<i>option 2</i>)	ECP
07BCh	CFIFO or DFIFO or TFIFO or CNFGA (<i>option 3</i>)	ECP
07BDh	CNFGB (<i>option 3</i>)	ECP
07BEh	ECR (<i>option 3</i>)	ECP
07BFh	INDEX (<i>option 3</i>)	ECP
07C0h	SETUP (<i>option 3</i>)	ECP
07C1h	IPR (<i>option 3</i>)	ECP
EF00h	Bus Interface Control Register 1	Bus Interface Unit
EF01h	Bus Interface Control Register 2	Bus Interface Unit
EF02h	Chip Select Enable Register	Bus Interface Unit
EF03h	Chip Select Type Register	Bus Interface Unit
EF04h-EF07h	Chip Select Base Address Register 1	Bus Interface Unit
EF08h-EF0Bh	Chip Select Base Address Register 2	Bus Interface Unit
EF0Ch-EF0Fh	Chip Select Base Address Register 3	Bus Interface Unit
EF10h-EF13h	Chip Select Base Address Register 4	Bus Interface Unit
EF14h-EF17h	Chip Select Base Address Register 5	Bus Interface Unit
EF18h-EF1Bh	Chip Select Base Address Register 6	Bus Interface Unit
EF1Ch-EF1Fh	Chip Select Base Address Register 7	Bus Interface Unit
EF20h-EF23h	Chip Select Base Address Register 8	Bus Interface Unit
EF24h-EF27h	Chip Select Address Mask Register 1	Bus Interface Unit
EF28h-EF2Bh	Chip Select Address Mask Register 2	Bus Interface Unit
EF2Ch-EF2Fh	Chip Select Address Mask Register 3	Bus Interface Unit
EF30h-EF33h	Chip Select Address Mask Register 4	Bus Interface Unit
EF34h-EF37h	Chip Select Address Mask Register 5	Bus Interface Unit
EF38h-EF3Bh	Chip Select Address Mask Register 6	Bus Interface Unit
EF3Ch-EF3Fh	Chip Select Address Mask Register 7	Bus Interface Unit
EF40h-EF43h	Chip Select Address Mask Register 8	Bus Interface Unit
EF44h	ROM BIOS Selection Register	Bus Interface Unit
EF45h	External Chip Select Selection Register 1	Bus Interface Unit
EF46h	External Chip Select Selection Register 2	Bus Interface Unit

Table 3-1: I/O Address Map

I/O Address	Register Name	Reference Section
EF47h	External Chip Select Selection Register 3	Bus Interface Unit
EF48h	External Chip Select Selection Register 4	Bus Interface Unit
EF49h	External Chip Select Selection Register 5	Bus Interface Unit
EF4Ah	External Chip Select Selection Register 6	Bus Interface Unit
EF4Bh	External Chip Select Selection Register 7	Bus Interface Unit
EF4Ch	External Chip Select Selection Register 8	Bus Interface Unit
EF4Eh	PCMCIA Base Memory Address Register	Bus Interface Unit
EF4Fh	PCMCIA Clock Selection Register	Bus Interface Unit
EF50h	Chip Select Access Time Register 1	Bus Interface Unit
EF51h	Chip Select Access Time Register 2	Bus Interface Unit
EF52h	Chip Select Access Time Register 3	Bus Interface Unit
EF53h	Chip Select Access Time Register 4	Bus Interface Unit
EF54h	ROM BIOS Access Time Register	Bus Interface Unit
EF55h	Catchable Chip Selects Register	Bus Interface Unit
EF56h	Auxiliary Processor Chip Select Selection Register	Bus Interface Unit
EF57h	Programmed 16-bit Logical Chip Select Register	Bus Interface Unit
EF70h	UART Clock Divisor Register	UART
EF71h	Modem Signal Control Register	UART
EF80h - EF81h	DRAM Control Register	DRAM Controller
EF82h - EF83h	Refresh Rate Register	DRAM Controller
EF84h - EF85h	$\overline{\text{RAS}}$ Timeout Register	DRAM Controller
EF86h - EF87h	DRAM Status Register	DRAM Controller
EF88h	DRAM Bank Size Register	DRAM Controller
EF89h	Bank 1 Address Register	DRAM Controller
EF8Ah	Bank 0 Mask Register	DRAM Controller
EF8Bh	Bank 1 Mask Register	DRAM Controller
EF90h	Power Management Register 1	Power Mgmt.
EF91h	Power Management Register 2	Power Mgmt.
EF92h	Power Management Register 3	Power Mgmt.
EF93h	Power Management Register 4	Power Mgmt.
EFA0h	LCD Configuration Register 1	LCD Controller
EFA1h	LCD Configuration Register 2	LCD Controller
EFA2h	LCD Configuration Register 3	LCD Controller
EFA3h	LCD Configuration Register 4	LCD Controller
EFB0h	Misc. Interrupt Selection Register 1	Interrupt Controller

Table 3-1: I/O Address Map

I/O Address	Register Name	Reference Section
EFB1h	Internal IRQ1 Source Selection Register	Interrupt Controller
EFB2h	Misc. Interrupt Selection Register 2	Interrupt Controller
EFB3h	Internal IRQ3 Source Selection Register	Interrupt Controller
EFB4h	Internal IRQ4 Source Selection Register	Interrupt Controller
EFB5h	Internal IRQ5 Source Selection Register	Interrupt Controller
EFB6h	Internal IRQ6 Source Selection Register	Interrupt Controller
EFB7h	Internal IRQ7 Source Selection Register	Interrupt Controller
EFB8h	Misc. Interrupt Selection Register 3	Interrupt Controller
EFB9h	Internal IRQ9 Source Selection Register	Interrupt Controller
EFBAh	Internal IRQ10 Source Selection Register	Interrupt Controller
EFBBh	Internal IRQ11 Source Selection Register	Interrupt Controller
EFBCh	Internal IRQ12 Source Selection Register	Interrupt Controller
EFBDh	Internal IRQ13 Source Selection Register	Interrupt Controller
EFBEh	Internal IRQ14 Source Selection Register	Interrupt Controller
EFBFh	Internal IRQ15 Source Selection Register	Interrupt Controller
EFC0h	Reconfigurable I/O Control Register	RIO
EFC1h	MICROWIRE Slave Mode Chip Select Register	RIO
EFC2h	Device ID Register	Bus Interface Unit
EFC3h	Device Revision Register	Bus Interface Unit
EFC4h - EFC7h	Data Direction Register	RIO
EFC8h - EFCBh	Data Port In Register	RIO
EFCCh - EFCFh	Data Port Out Register	RIO
EFE0h	Internal DMA Requester Selection Register	DMA Controller
EFE1h	CPU DMA Request Register	DMA Controller

3.3 System Service Elements

NS486SXF on-chip System Service Elements include the DRAM controller, a DMA controller, programmable interval timers, a protected watchdog timer, programmable interrupt controllers, a real-time clock and calendar, and power management logic.

3.3.1 The DRAM Controller

NS486SXF's DRAM controller is designed specifically for fast-page mode DRAMs with CAS-before-RAS refresh capability. Configuration and operation is controlled by the settings in eight DRAM Controller registers in the NS486SXF I/O map. (See Table 3-1 on page 20). The user must set the page size of the DRAMs, the size of the memory banks and where bank 1 will be located, whether parity will be used, and if so, whether a parity error will generate an NMI interrupt, and finally, whether the DRAMs use 3 or 4 clock cycles on a page miss access. (A typical configuration with a CPU clock of 25 MHz and a 3 cycle page miss would require 60 ns DRAMs.)

One or two banks of up to 8 Mbytes each of Dynamic RAM are supported. Bank 0 must be placed at the beginning of the flat memory address space of the NS486SXF processor. The banks can have different page sizes and memory sizes. The NS486SXF DRAM controller supports page mode read and write operations and both byte and word accesses.

All access control signals for read, write and parity checking are generated, as well as an automatic and programmable CAS-before-RAS refresh. If self-refresh DRAMs are used, refresh is not generated, saving power.

A number of different DRAM configurations using popular DRAM devices are supported. Page sizes can be 512, 1024, 2048, or 4096 bytes.

Memory bank 0 must start at address 0h; memory bank 1 must start at an address that is a multiple of that bank's size. For example, a 4 Mbyte bank must start at 400000h, 800000h, C00000h, etc. Memory bank address ranges cannot overlap.

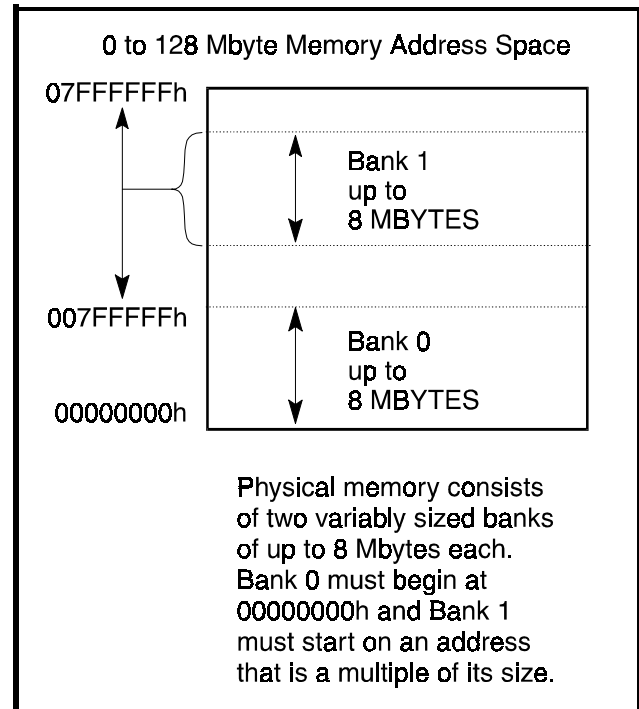


Figure 3-2 Memory Addressing

The NS486SXF DRAM controller supports five different DRAM Bank Sizes (0.5, 1, 2, 4, and 8 MByte) and five different DRAM Device Page Sizes (8, 9, 10, 11, and 12 Address Bits) independent of each other to provide the maximum flexibility for a wide range of new and old DRAM devices and memory modules.

Self-refresh mode for low-power operation can be used if the specific DRAM type supports it. Exiting the self-refresh mode is accomplished automatically when a hardware interrupt is generated, or by writing the appropriate control register bit (DRAM Control Register, bit 3=0).

The DRAM cannot be a DMA Requestor device. The DRAM must be the Target device for memory to memory and I/O to memory transfers.

Hardware Interconnection

NS486SXF address signals SA1... SA11 connect to Logical Address signals A0... A10 on the DRAM devices. NS486SXF control signals CASH0/CASL0 and CASH1/CASL1 are used to select upper and lower byte access. The NS486SXF generates separate CAS signals for each byte. If 16-bit wide DRAM chips are used, they should have two CAS signals.

The DRAM Controller can drive up to a maximum load of 100 pF on the SA12-1 signal pins. (**Note:** A system that uses the external bus for more than just DRAM will require signal buffering.) The DRAM Controller can drive a maximum of 30 pF on the $\overline{\text{CAS}}$ signals, 60 pF on the $\overline{\text{RAS}}$ signals and 120 pF on the $\overline{\text{WE}}$ signal. Typically, a system can support a maximum of 12 DRAM devices at a time, 6 per bank. In this case, the DRAMs must have a 4-bit or wider data bus (except for the parity bit if implemented separately).

When running the CPU at 25 MHz, the DRAM Controller will operate with most 60ns DRAMs using 3 CPU cycles for a page miss and one cycle for a page hit (**Note:** Some 60ns DRAMs have a slow CAS access time). Slower DRAMs may be used by using a slower clock or by using Four Cycle Miss Mode (DRAM Control Register, bit 7 = 1).

3.3.1.1 Reset Condition

A system reset will place the DRAM Controller into an idle state. Following reset, the DRAM controller will wait for at least 100 μsec or until the DRAM Enable bit (DRAM Control Register, I/O address EF80-81h, bit 0) is programmed to a 1, whichever occurs later. After the 100 μsec delay, and with the DRAM Enable bit set, the DRAM Controller will generate eight CAS-before-RAS refresh cycles. At the end of the eighth refresh cycle, the DRAM Initialization Complete bit (DRAM Status Register, address EF86-87, bit 0) is set to a 1, indicating that the DRAM may now be read and written. The user program can check this bit to see when the DRAM is ready. Note that if during operation, the DRAM Enable bit (DRAM Control Register, I/O address EF80-81h, bit 0) is set to 0, the DRAM controller will not stop immediately. It will wait until the end of the next refresh cycle. However, subsequent re-enabling of the DRAM con-

troller will occur immediately when a 1 is written to the DRAM Enable bit.

3.3.1.2 DRAM Parity Checking

The Odd-Parity generation and checking of the DRAM memory may be enabled or disabled by the Parity Enable bit. Internally, the Parity Error interrupt is connected to the CPU NMI logic for system intervention and error handling. When the Parity Error NMI Enable bit (DRAM Control Register, bit 14) is set to one, a parity error will generate an NMI to the CPU.

When the CPU gets an NMI, it can check to see if it came from the DRAM by checking the bits in the DRAM Status Register at location EF86-87h. Note that even if NMI is not enabled, the CPU can poll the Status Register to detect a parity error.

3.3.1.3 Refresh

Standard CAS-before-RAS refresh mode is supported. The refresh rate is programmable via the Refresh Rate Register (I/O address EF82h).

By programming the Self-Refresh Mode bit, (DRAM Control Register, I/O address EF80h, bit 3), to a 1, the controller will put the DRAMs into Self-Refresh mode by initiating a $\overline{\text{CAS}}$ -before- $\overline{\text{RAS}}$ refresh cycle and holding both $\overline{\text{CAS}}$ and $\overline{\text{RAS}}$ low for the duration of the power-down mode. To exit Self-Refresh mode, either a 0 can be written to the Self-Refresh Mode bit, or a system interrupt can force the DRAM Controller out of Self-Refresh Mode. Exiting Self-Refresh mode will only provide one $\overline{\text{CAS}}$ -before- $\overline{\text{RAS}}$ cycle. The DRAMs must be $\overline{\text{CAS}}$ -before- $\overline{\text{RAS}}$ type, they must not require burst-refresh. Self-refresh DRAMs must be able to come out of refresh with a single $\overline{\text{CAS}}$ before $\overline{\text{RAS}}$ only. (**CAUTION:** Any DRAM access attempt while self-refresh is enabled will not complete and as a result will lock up the NS486SXF CPU. In this event, the WATCHDOG timeout is the only means of recovering.)

3.3.1.4 RAS Recovery (pre-charge) Rate

The DRAM Controller will force $\overline{\text{RAS}}$ high if it has remained low more than a certain period of time. The amount of time is programmed via the RAS Timeout Register. If the value written to this register is greater than the value written to the Refresh Rate Register,

then it is assumed that the programmed refresh period is less than or equal to the maximum RAS low period; therefore, the RAS Timeout Register will have no effect on the operation of the DRAM Controller.

3.3.1.5 The DRAM Registers

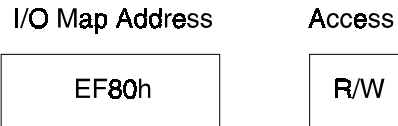
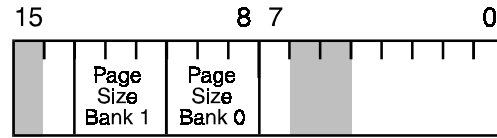
The DRAM controller is setup and its operation controlled by eight DRAM Controller registers in the NS486SXF I/O map. Four registers are 16-bit and four are 8-bit. They are:

Name	I/O Address *	Size
DRAM Control Register	EF80-81	16 bits
Refresh Rate Register	EF82-83	16 bits
RAS Timeout Register	EF84-85	16 bits
DRAM Status Register	EF86-87	16 bits
DRAM Bank Size Register	EF88	8 bits
Bank 0 Mask Register	EF8A	8 bits
Bank 1 Mask Register	EF8B	8 bits
Bank 1 Address Register	EF89	8 bits
* addresses in hex		

It is important that the DRAM Controller be disabled (Bit 0 set to 0 in the DRAM Control Register, I/O address EF80h) before programming in order to avoid unpredictable or race conditions.

3.3.1.5.1 DRAM Control Register

This 16-bit register is used to control the operation of the DRAM Controller. The DRAM Control Register is written through I/O address EF80h.



- Bit 15: Reserved
- Bit 14: Parity Error NMI Enable
0 = Parity Error NMI Disabled (default)
1 = Parity Error NMI Enabled
Note: Bit 0 of the Miscellaneous Interrupt Selection Register 1 must be set in order for the NMI to reach the CPU. See Section 3.3.5.4.1 on page 78
- Bits 13-11: DRAM Page Size, Bank 1.

Bit 13	Bit 12	Bit 11	DRAM Page Size (Address Bits)
0	0	0	8
0	0	1	9
0	1	0	10
0	1	1	11 (default)
1	0	0	12
1	0	1	Reserved
1	1	0	Reserved
1	1	1	Reserved

Bits 10-8: DRAM Page Size, Bank 0.

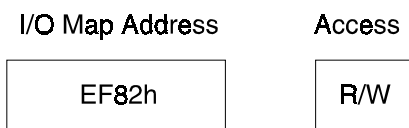
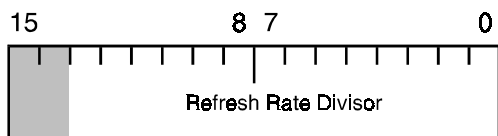
Bit 10	Bit 9	Bit 8	DRAM Page Size (Address Bits)
0	0	0	8
0	0	1	9
0	1	0	10
0	1	1	11 (default)

Bit 10	Bit 9	Bit 8	DRAM Page Size (Address Bits)
1	0	0	12
1	0	1	Reserved
1	1	0	Reserved
1	1	1	Reserved

- Bit 7: Four Cycle Page Miss Mode
0 = Three Cycle Page Miss (default)
1 = Four Cycle Page Miss
- Bits 6-5: Reserved
- Bit 4: Extended CAS
0 = Normal CAS with 0.5 CPU clock low and high time (default)
1 = Extended CAS Low with 1.0 CPU clocks low and high time
- Bit 3: Self-Refresh Mode
0 = Normal Refresh Mode (default). A system interrupt will also exit Self-Refresh Mode.
1 = Enter Self-Refresh Mode
- Bit 2: Parity Enable
0 = Parity Disabled (default)
1 = Parity Enabled
- Bit 1: Bank 1 Present
0 = Bank 1 Empty (default)
1 = Bank 1 Present
- Bit 0: DRAM Enable
0 = DRAM Disabled (default)
1 = DRAM Enabled

3.3.1.5.2 Refresh Rate Register

This 16-bit register is used to establish the frequency of refresh cycles. At reset, it is initialized to: 02EEh. The Refresh Rate Register is written through I/O address EF82h.



- Bits 15-14: Reserved
- Bits 13-0: Refresh Rate Divisor — This value is the number of oscillator clocks to count be-

tween each refresh. For example, with a 50MHz oscillator, programming the value to 02EEh will result in a 15µsec refresh rate. Programming the register to 1770h with the same oscillator will result in a 120µsec refresh rate.

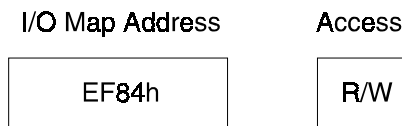
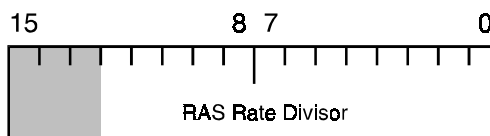
$$\text{Refresh period} = (\text{Oscillator period}) \times (\text{Refresh Rate Register Value})$$

If the oscillator clock runs at 50 MHz and therefore has a 20 nsec period, and the Refresh Rate Register is set to 02EEh (750):

$$(20\text{nsec}) \times (750) = 15 \mu\text{sec}.$$

3.3.1.5.3 RAS Timeout Register

This 16-bit register is used to set how long the $\overline{\text{RAS}}$ pins may be kept low. If the $\overline{\text{RAS}}$ low time exceeds the value programmed into this register, the DRAM Controller will force that $\overline{\text{RAS}}$ signal high, resulting in a page miss on the next access. If the $\overline{\text{RAS}}$ low time is greater than the refresh period, then this register value can be programmed to a value greater than the Refresh Rate Register, and the counter will have no effect. At reset, this register is initialized to: 1388h. The RAS Timeout Register is written through I/O address EF84h.



- Bits 15-13: Reserved
- Bits 12-0: RAS Rate Divisor — This value is the maximum number of oscillator clocks $\overline{\text{RAS}}$ may remain low. If this count is reached, the DRAM controller will force $\overline{\text{RAS}}$ high, resulting in a page miss. The count is reset when $\overline{\text{RAS}}$ goes high due to a refresh or a page miss.

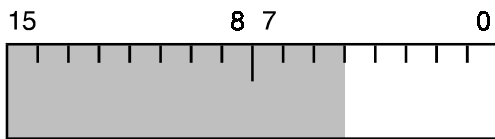
$$\text{RAS Timeout Period} = (\text{Oscillator period}) \times (\text{RAS Timeout Register value} + 1)$$

If the oscillator clock runs at 50 MHz and therefore has a 20 nsec period, and the RAS Timeout Register is loaded with

01770h (6000 decimal):
 (20 nsec) X (6000) = 120 μsec.
 Note: If the RAS Timeout value is changed during operation, the new value will not take affect until after the next re-fresh period.

3.3.1.5.4 DRAM Status Register

This 16-bit register shows the status of the DRAM Controller. The DRAM Status Register is written through I/O address EF86h.



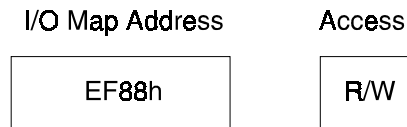
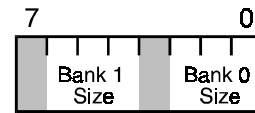
- Bits 15-5: Reserved
- Bit 4: Bank 1 Parity Error (read only)
 0 = No error in Bank 1 (default)
 1 = Parity error in Bank 1
- Bit 3: Bank 0 Parity Error (read only)
 0 = No error in Bank 0 (default)
 1 = Parity error in Bank 0
- Bit 2: Clear Parity Error (write only, self-clearing)
 0 = No effect
 1 = Clear Parity Error. Will clear itself back to 0.
- Bit 1: Parity Error Detected (read only)
 0 = No Parity error (default)
 1 = Parity error detected
- Bit 0: Initialization Complete (read only)
 0 = DRAM Disabled/Initialization incomplete (default)
 1 = Initialization complete

During initialization, Bit 0 can be checked to determine when the DRAM controller is ready for operation.

When a parity error causes an NMI, the interrupt service routine can quickly check if it was caused by a DRAM parity error by reading Bit 1. If it is set to one, Bits 3 and Bits 4 can be checked to determine which bank had the parity error. Finally, writing a 1 to Bit 2 clears the parity error bits, Bits 1, 3, and 4.

3.3.1.5.5 DRAM Bank Size Register

This 8-bit register is used to select the size of each bank of DRAM. The DRAM Bank Size Register is written through I/O location EF88h.



- Bit 7: Reserved
- Bits 6-4: DRAM Bank 1 Size —Each bank of memory is 16-bits wide, so, for example, a 1Mbyte bank size would probably consist of two 512Kx8/9 DRAMs.

Bit 6	Bit 5	Bit 4	DRAM Bank 1 Size (Mbytes)
0	0	0	0.5
0	0	1	1
0	1	0	2
0	1	1	4
1	0	0	8 (default)
1	0	1	Reserved
1	1	0	Reserved
1	1	1	Reserved

Bit 3: Reserved
 Bits 2-0: DRAM Bank 0 Size

Bit 2	Bit 1	Bit 0	DRAM Bank 0 Size (Mbytes)
0	0	0	0.5
0	0	1	1
0	1	0	2
0	1	1	4
1	0	0	8 (default)
1	0	1	Reserved
1	1	0	Reserved
1	1	1	Reserved

3.3.1.5.6 Bank Addressing and Masks

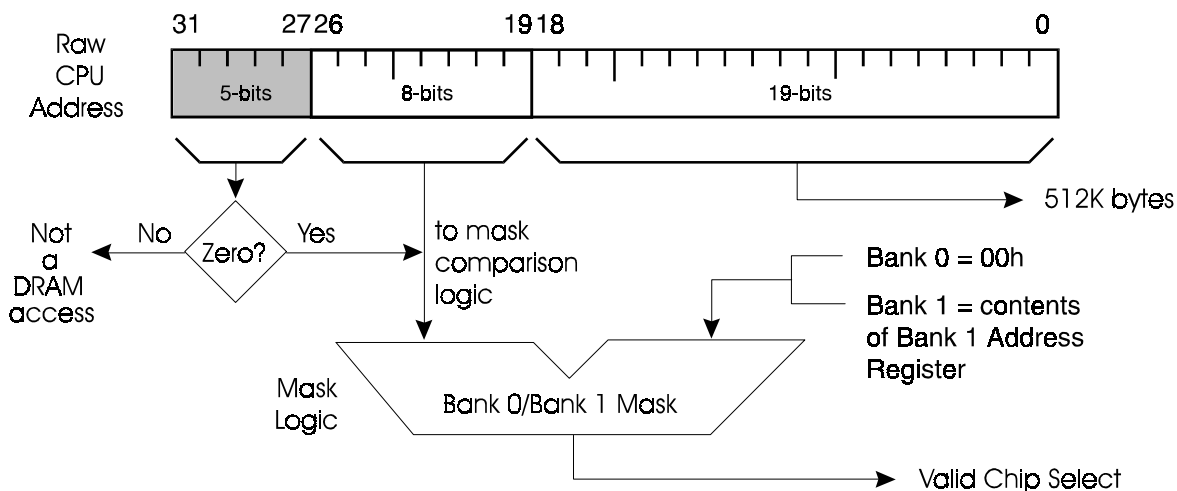
To determine if a CPU local bus cycle is to access DRAM, the DRAM controller decodes the address as well as the cycle type. The DRAM controller’s address comparison logic compares the CPU local bus address with the programmed address range(s) of the two memory banks. The user sets a valid range through a Bank 1 Address Register and two bank “mask” registers. The mask register value is compared to CPU local bus address bits A26-A19, and if there is match, the address is determined to be valid.

The address coming from the CPU can be broken into three groups of bits: 31-27, 26-19, and 18-0. For both banks, address bits 31-27 will always be zero when accessing the DRAM. For both banks, 18-0 determine the decoding for the lower 512K bytes. Bits 26-19 are compared with the values in the bank 1 address register and the two mask registers and a “true” comparison enables a chip select signal and the DRAM access.

When the DRAM controller is enabled, a CPU local bus access must meet the following three requirements to access DRAM Bank 0:

- 1 -- It must be a memory cycle.
- 2 -- The CPU local bus A31-A27 signals must all be zeroes.
- 3 -- Each of the CPU local bus A26-A19 signals must be a zero, or the corresponding bit in the Bank 0 Mask Register must be a zero.

Figure 3-3 Valid DRAM Address Chip Select Generation



When the DRAM controller is enabled and DRAM Bank 1 is also active, a CPU local bus access must meet the following three requirements to access DRAM Bank 1:

- 1 -- It must be a memory cycle.
- 2 -- The CPU local bus A31-A27 signals must all be zeroes.
- 3 -- Each of the CPU local bus A26-A19 signals must match the corresponding bit in the Bank 1 Address Register or the corresponding bit in the Bank 1 Mask Register must be a zero.

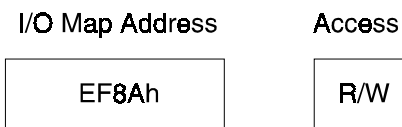
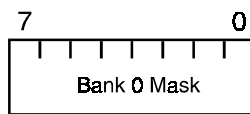
Typical addresses and masks are shown below:

Bank# (Start Address)	Bank 0 (0M)	Bank 1 (1M)	Bank 1 (8M)
A26-A19	00h	02h	10h
Mask Value	FFh	FEh	F0h

If a large DRAM device is used, the lower bits of the Mask Register should be set to zero. For example, a 1Mbyte DRAM device will use A19 as part of the direct address. Therefore, the Mask Register bit 0 should be set to 0 to allow this bit to be either 1 or 0. This allows a valid Chip Select throughout the entire 1 Mbyte address range.

3.3.1.5.7 Bank 0 Mask Register

This 8-bit register is used to mask address bits 26-19 in the internal DRAM Controller chip select logic for Bank 0.

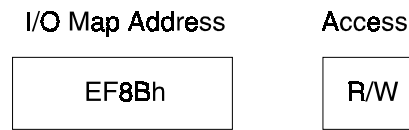
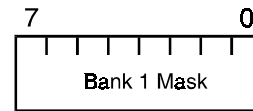


Bits 7-0: Bank 0 Mask — These eight bits mask the DRAM Controller Bank 0 chip select address comparison bits 26-19 which are always 00h in the case of Bank 0. Every

bit set to a one in these registers indicates the associated chip select address (always 0) must match the CPU address for an active internal chip select to be decoded. If a register bit is programmed to a zero, then the associated chip select address comparison bit does not need to match. The upper 5 address bits (bits 31-27) must always be equal to 0.

3.3.1.5.8 Bank 1 Mask Register

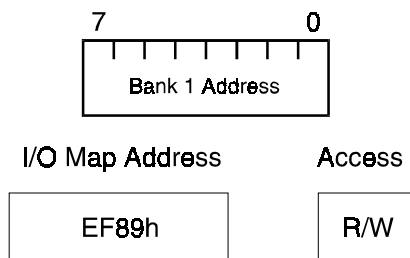
This 8-bit register is used to mask address bits 26-19 in the internal DRAM Controller chip select logic for Bank 1. It sets the high order location of Bank 1 memory in the memory map.



Bits 7-0: Bank 1 Mask — These eight bits mask the DRAM Controller Bank 1 chip select address comparison bits 26-19 as set in the Bank 1 Address Register. Every bit set to a one in these registers indicates the associated Bank 1 Address Register bit must match the CPU address for an active internal chip select to be decoded. If a register bit is programmed to a zero, then the associated chip select address comparison bit does not need to match.

3.3.1.5.9 Bank 1 Address Register

This 8-bit register is compared with address bits 26-19 in the internal DRAM Controller chip select logic for Bank 1.



Bits 7-0: Bank 1 Address — These eight bits must exactly match address bits 26-19 (unless masked in the Bank 1 Mask Register) for an active internal DRAM chip select for Bank 1 to be generated. The upper 5 address bits (bits 31-27) must always be equal to 0.

3.3.1.5.10 DRAM Timing Diagrams

Below are some typical read and write cycle timing diagrams. They show the relationship between clock cycles and valid data writes and reads.

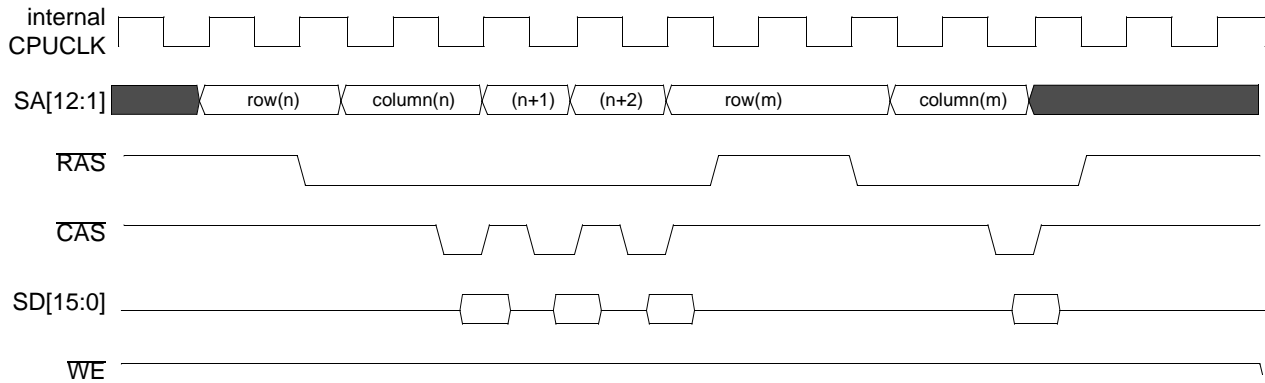


Figure 3-4 DRAM Read Cycle (4 cycle page miss)

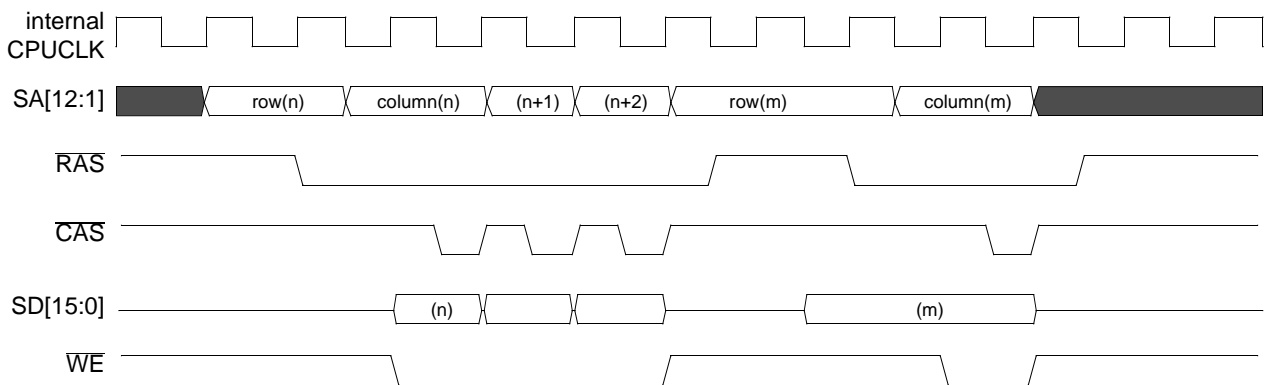


Figure 3-5 DRAM Write Cycle (4 cycle page miss)

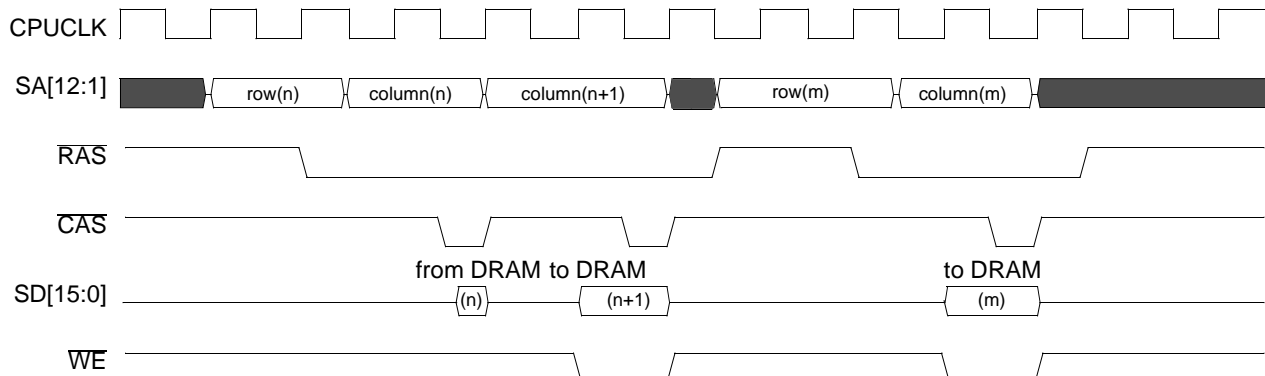


Figure 3-6 DRAM Read Followed by DRAM Write (4 cycle page miss)

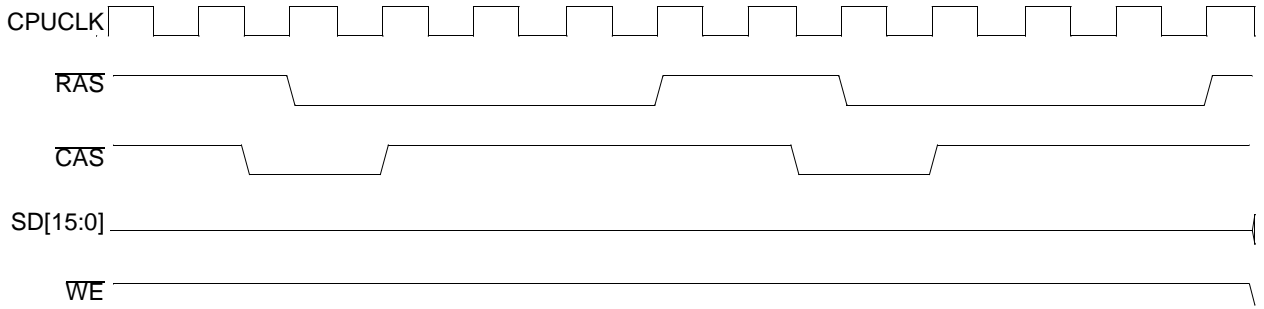


Figure 3-7 $\overline{\text{CAS}}$ Before $\overline{\text{RAS}}$ Refresh Cycle (4 cycle page miss)

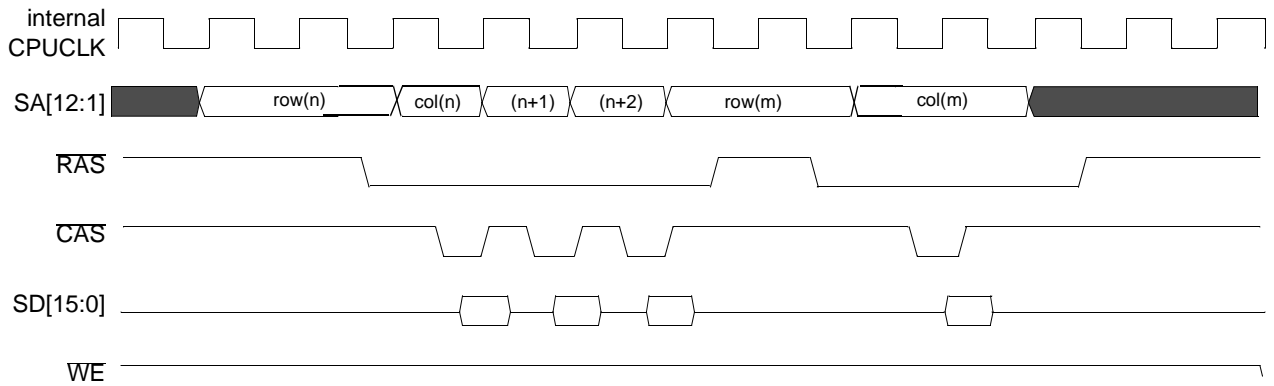


Figure 3-8 DRAM Read Cycle (3 cycle page miss)

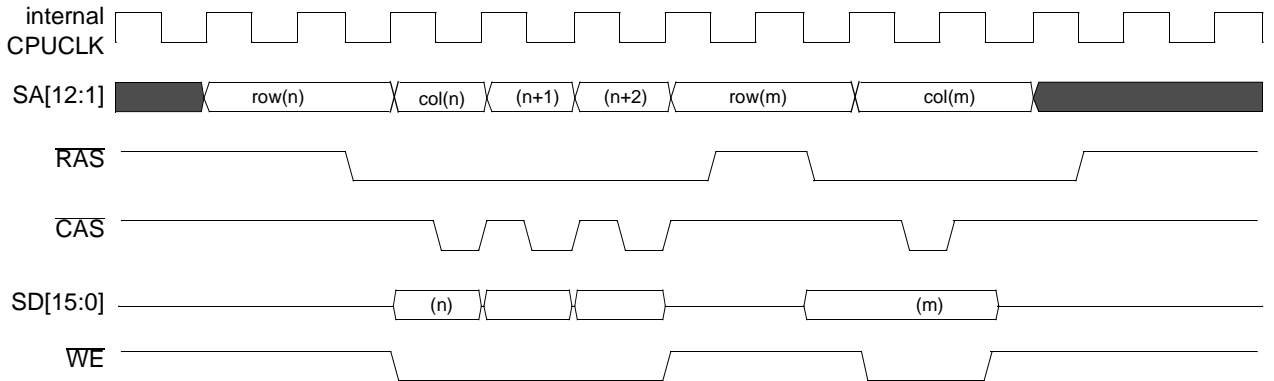


Figure 3-9 DRAM Write Cycle (3 cycle page miss)

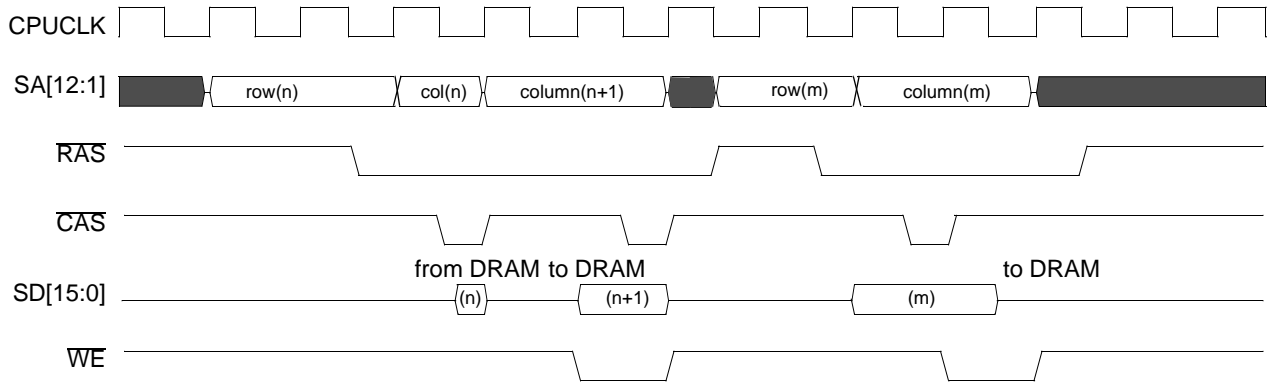


Figure 3-10 DRAM Read Followed by DRAM Write (3 cycle page miss)

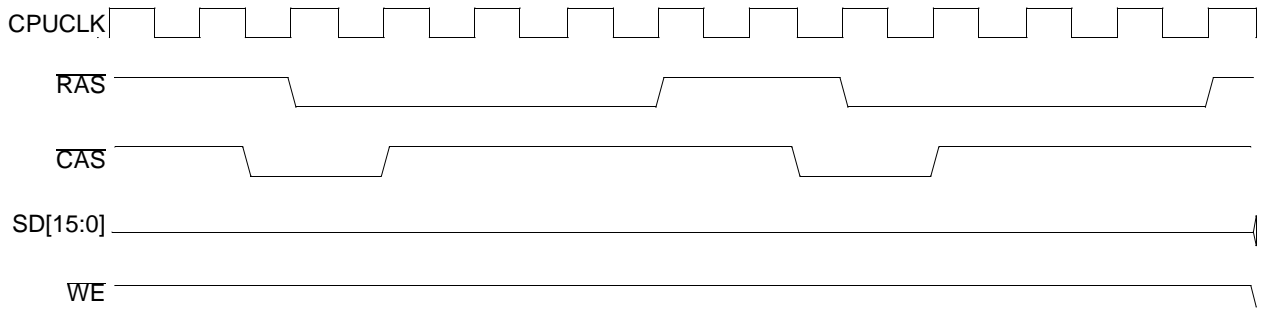


Figure 3-11 $\overline{\text{CAS}}$ Before $\overline{\text{RAS}}$ Refresh Cycle

3.3.2 The DMA Controller

The NS486SXF Direct Memory Access (DMA) controller is a high speed 16-bit controller with six independent channels. Its resources are controlled by programmable registers located throughout the NS486SXF I/O map. Processor independent data transfers are performed at a maximum data rate of 2 bytes per 2 clock cycles. (A 25 MHz clock yields a 25 megabyte per second transfer rate.)

There are six independent channels, four for external resources and two for internal peripherals. The channel priority can be set to be fixed or rotated. requester and target addresses can be independently incremented and have a maximum addressable memory range of 64 Mbytes. Three basic transfer modes, single, block, and demand are provided giving the user a wide range of DMA options. A cascade-master mode allows external masters to access the NS486SXF ISA-like bus. DMA transfers can be from memory to memory, memory to I/O and I/O to memory. All DMA transfers are controlled by the status of DMA

control registers in the NS486SXF control register I/O map.

Immediately after system reset, the DMA controller is not accessible via its I/O mapped registers until access is enabled to it via the Bus Interface Unit (BIU) control registers 1 and 2. To enable accesses to the DMA controller, a “1” must be written to bit 5 of BIU control register 1 (I/O address EF00h) and a “1” must also be written to bit 2 of BIU control register 2 (I/O address EF01h) and “1” must also be written to the DMA command register bit 0. Refer to the BIU section of this document for more information.

Each channel can be individually programmed to autoinitialize to its original condition upon receiving an End of Process (\overline{EOP}) signal or by reaching a terminal count (TC). In addition to the three basic transfer modes, the controller also provides a buffer chaining mode allowing transfer of data from a peripheral to several different areas of memory within one transfer operation (or vice versa).

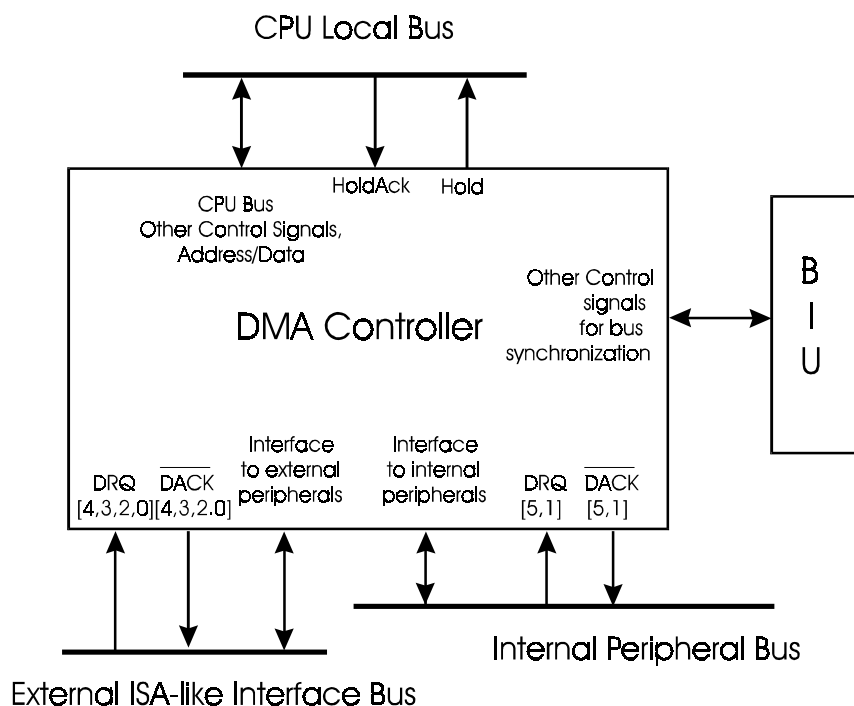


Figure 3-12 DMA Controller Internal Block Diagram

As shown in the figure, the DMA controller interfaces to four buses: the CPU local bus, the Bus Interface Unit (BIU), the internal peripherals bus, and the external ISA-like bus. The DMA controller is designed to request bus control from the CPU, synchronize its accesses to memory and I/O with the BIU and to transfer data to and from memory addresses and I/O locations.

The typical DMA transfer sequence goes as follows:

- 1) One or more peripherals request a DMA transfer using the DRQ[n] signals. This starts an arbitration sequence where the highest priority requester (according to a preset priority) will get (win) the first DMA service.
- 2) The DMA Controller requests control of the NS486SXF internal CPU local bus using the internal HOLD signal to the CPU.
- 3) When the CPU finishes its current cycle, it relinquishes its address, data and control buses and signals the DMA controller via the internal HLDA signal. (Note: in a LOCK cycle, the CPU will finish the LOCK cycle completely first.) The DMA Controller takes control of the bus, and sends a $\overline{\text{DACK}}[n]$ signal to the requester that won the arbitration.
- 4) The DMA controller reads data from the DMA device (I/O or memory mapped) and writes to system memory, or vice versa. It then automatically increments or decrements the system memory or DMA device addresses, and decrements the Byte Count Register.
- 5) The DMA transfer ends in one of several ways. In Single Mode, it ends after a single byte or word is transferred. In Demand Mode, it can end when the winning DMA channel's DRQ signal goes low (inactive). Or, in Block and Demand modes, it ends when a terminal count (TC) is recorded (when the byte count equals the number of bytes to be transferred), or an $\overline{\text{EOP}}$ (end of process) signal is received.

Priority between the channels can be programmed to be fixed, with any given channel having the highest priority and the next lowest numbered channel having the next highest, and so on, or priority can be transferred in a rotated or "round-robin" manner. In the latter case, a channel is designated "first" and it gets the highest priority first. The next time, the next highest

numbered channel gets the highest priority. The next time, the next highest numbered channel gets the highest priority and so on.

Channels 0, 2, 3, and 4 are for use with external peripherals or memory. They support memory to I/O, I/O to memory, and memory to memory modes. Channels 1 and 5 are for the internal peripherals (LCD controller and ECP port). Channels 1 and 5 can be used for either the LCD controller or the ECP port. Each channel has associated DRQ[n] and $\overline{\text{DACK}}[n]$ handshake signals.

The DMA Controller keeps track of start and stop addresses through the following registers:

- 1) The Base Target Address Register - this register holds a pointer to the **original starting** address of the target or system memory data location.
- 2) The Current Target Address Register - this register holds a pointer to the **current or updated** address of the target data location (after each transfer, it is updated).
- 3) The Base requester Address Register - this register holds a pointer to the **starting** address of the data in the DMA device (memory mapped only).
- 4) The Current requester Address Register - this register holds a pointer to the **current** address of the data in the DMA device (memory mapped only).
- 5) The Base Byte Count Register - this register holds the **starting** number of bytes to be transferred.
- 6) The Current Byte Count Register - this register holds the **current or updated** number of bytes left to be transferred.

Target addresses are required for all transfers while requester addresses are only required for memory to memory transfers.

DMA devices can use the RDY signal to extend DMA cycles by deasserting RDY. Since the DMA controller internally synchronizes RDY to its own clock, you must program at least one wait state (with 0 wait states, RDY may be ignored).

3.3.2.1 DMA Transfer Modes

There are three data transfer modes: single, block and demand. A fourth mode, cascade-master, is for relinquishing control of the bus to an external master. This allows an external master to access the NS486SXF

ISA-like bus. (Note: It does not allow access to NS486SXF on-board peripherals or the DRAM.)

In all cases, DMA request/grant is performed using the DRQ[n]/ $\overline{\text{DACK}}[n]$ signals and the type of DMA transfer mode is set up by setting the appropriate bits in the DMA channel mode registers.

3.3.2.1.1 Single Transfer Mode

A DMA channel programmed for Single Transfer Mode performs one transfer for each arbitration cycle. The DMA software programs the channel's Base Byte Count register with the appropriate number of bytes to transfer. The DMA controller decrements (by 1 for byte transfers; by 2 for word transfers) the channel's Current Byte Count register and increments/decrements (by 1 for byte transfers; by 2 for word transfers) its Current Address register after each transfer. The transfer completes when the Current Byte Count register reaches terminal count or when an external End of Process ($\overline{\text{EOP}}$) is received. Terminal Count or $\overline{\text{EOP}}$ causes the current register to be reloaded from the base registers if the channel is programmed for autoinitialize. Otherwise, the channel will be masked off.

A DMA device requests a Single Transfer Mode DMA transfer by asserting DRQ[n] high and holding it until seeing $\overline{\text{DACK}}[n]$ asserted low. The DMA device may or may not hold DRQ[n] asserted throughout the single transfer. The system logic negates $\overline{\text{DACK}}[n]$ and the DMA channel releases the internal CPU bus after a single transfer. If DRQ[n] remains asserted, the DMA controller immediately requests the internal CPU bus again. The DMA arbitration controller performs the arbitration, and asserts the winning channel's $\overline{\text{DACK}}[n]$ to signal the bus grant after receiving hold acknowledge (HLDA) from the CPU. The DMA channel then performs another single transfer. The current registers hold the intermediate address and byte count value between arbitration cycles.

3.3.2.1.2 Block Transfer Mode

A DMA channel programmed for Block Transfer Mode performs a block transfer for each arbitration cycle. The DMA software programs the channel's Base Byte Count register with the appropriate number of bytes to transfer. The DMA controller decrements

(by 1 for byte transfers; by 2 for word transfers) the channel's Current Byte Count register and increments/decrements (by 1 for byte transfers; by 2 for word transfers) its Current Address register after each transfer. The transfer completes when the Current Byte Count register reaches terminal count or an end of process ($\overline{\text{EOP}}$) is received. Terminal Count or $\overline{\text{EOP}}$ causes the current registers to be reloaded from the base registers if the channel is programmed for autoinitialize. Otherwise, the channel will be masked until reprogrammed.

A DMA device requests a Block Mode DMA transfer by asserting DRQ[n] high and holding it until seeing $\overline{\text{DACK}}[n]$ asserted low. The DMA device may or may not hold DRQ[n] asserted throughout the block transfer or may release DRQ[n] after sampling $\overline{\text{DACK}}[n]$ asserted low.

3.3.2.1.3 Demand Transfer Mode

A DMA channel programmed for Demand Transfer Mode performs a group of transfers for each arbitration cycle. The user's DMA software programs the channel's Base Byte Count register for the appropriate number of bytes to transfer. The DMA controller decrements (by 1 for byte transfers; by 2 for word transfers) the channel's Current Byte Count register and increments/decrements (by 1 for byte transfers; by 2 for word transfers) its Current Address register after each transfer. The transfer continues until the device negates DRQ[n], the Current Byte Count register reaches terminal count or an End of Process ($\overline{\text{EOP}}$) is received. Terminal Count or $\overline{\text{EOP}}$ causes the current registers to be reloaded from the base registers if the channel is programmed for autoinitialize.

A DMA device requests a Demand Mode DMA transfer by asserting DRQ[n] high. The DMA device holds DRQ[n] asserted until it runs out of data or until the transfer terminates. DRQ[n] is sampled one CPU clock period before the end of the write cycle (either IOW, or MEMW). Thus to stop the next DMA cycle, DRQ[n] must be deasserted ($\text{DRQ}[n] = 0$) at least one CPU clock period before the end of the last DMA write cycle.

The negation of DRQ[n] interrupts the transfer. When the next DRQ[n] is asserted (and wins the arbitration), the transfer continues from that same point.

3.3.2.1.4 Cascade-Master Mode

An external DMA channel programmed for Cascade-Master Mode will not perform any transfers for each arbitration cycle. Instead, the Cascade-Master Mode allows an external Bus Master to arbitrate for control of the NS486SXF's external ISA-like interface bus. When a Cascaded Master has been granted control (i.e. when its $\overline{\text{DACK}}[n]$ is asserted low) the following signals will TRI-STATE until $\text{DRQ}[n]$ goes low again:

$\overline{\text{IOR}}$, $\overline{\text{IOW}}$, $\overline{\text{MEMR}}$, $\overline{\text{MEMW}}$
 $\text{SA}[25:0]$, $\overline{\text{SBHE}}$, $\text{SD}[15:0]$

DRAM refresh cycles will continue during cascade master mode. Since we use CAS-before-RAS refresh, the use of the address lines is not necessary for refreshing the DRAM (the Refresh counter is internal to the DRAM devices).

The Cascade Master will not be able to access the DRAM supported by the DRAM controller, or any of the internal NS486SXF peripherals.

3.3.2.2 Autoinitialize

An autoinitialize channel automatically loads the Current Target Address, Current requester Address and Current Byte Count registers from the Base Target Address, Base requester Address, and Base Byte Count registers each time the DMA controller reaches terminal count or an End of Process ($\overline{\text{EOP}}$) is received. By programming a bit in the Mode register, a channel can be set up for Autoinitialization. The mask bit is not set at the end of a transfer when the channel is in Autoinitialize mode. Following autoinitialize, the channel is ready to perform another DMA service without CPU intervention as soon as the DMA device requests and wins the bus again.

3.3.2.3 Transfer Types

Single, Block and Demand Mode transfers may have two Transfer Types: Write Transfer Type and Read Transfer Type.

3.3.2.3.1 Write Transfer Type

A DMA Write Transfer reads from the NS486SXF's memory (external DRAM or SRAM) and writes that data to the DMA requesting device.

3.3.2.3.2 Read Transfer Type

A DMA Read Transfer, reads from the DMA requesting device and writes to the NS486SXF's memory (external DRAM or SRAM).

3.3.2.4 Device Types

Single, Block and Demand Mode transfers support two DMA requester Device Types: I/O Device Type and Memory Device Type.

3.3.2.4.1 I/O Device Type

For a DMA requester that is programmed as an I/O Device Type, the NS486SXF's DMA Controller will produce the appropriate I/O access strobe for accesses to the DMA requester ($\overline{\text{IOR}}$ for Read Transfer Types and $\overline{\text{IOW}}$ for Write Transfer Types).

3.3.2.4.2 Memory Device Type

For a DMA requester that is programmed as a Memory Device Type, the NS486SXF's DMA Controller will produce the appropriate memory address as well as the appropriate memory access strobe for accesses to the DMA requester ($\overline{\text{MEMR}}$ for Read Transfer Types and $\overline{\text{MEMW}}$ for Write Transfer Types). Note that the DRAM cannot be a DMA Requestor.

3.3.2.5 Maximum Performance

The following table shows the maximum number of bytes that can be transferred by the various combinations of the DMA types. For maximum throughput performance, a single clock cycle DRAM memory hit and RDY high is assumed for every access to the NS486SXF's memory and no DMA arbitration time is considered.

Device Type	Transfer Type	Maximum Transfer Rate (bytes per clock)
I/O	Read	0.667 bytes per clock
I/O	Write	1.0 byte per clock
Memory	Read	0.5 bytes per clock
Memory	Write	0.5 bytes per clock

Formula:

$$\text{Time} = (\# \text{ of bytes to transfer}) \times (8\text{-bits/Requester Bus Size})$$

* (Clocks per Transfer)

* (Clock period)

Transfer Rate = (# of bytes to transfer) / Time

3.3.2.6 Software Initiated DMA Transfers

If the external DRQ[n] pins are left floating, they can generate DMA requests. Therefore, when using a DMA channel to perform a transfer under CPU control, a DMA transfer will be initiated when the DMA channel is unmasked. This will happen regardless of the state of the “CPU DMA Request Register.” If this behaviour is not desired, a pull-down resistor could be added to the external DRQ pins.

3.3.2.7 DMA Control Registers

There are many programmable registers for controlling the operation of the DMA controller.

The DMA Address Registers for example, consist of a set of base and current registers for each channel. The Base Target Address Register holds the location (in NS486SXF system memory) from which or to which the data is to be transferred. The Current Target Address Register holds the updated destination/source memory address (it is updated after each byte or word transfer). All six DMA channels require a Base and Current Target Address Register set.

NOTE: The Base registers are write only and are written once by the DMA controller software before the start of the DMA process. During each byte or word transfer, the Current registers are updated automatically by the DMA controller logic. The Current registers can be read by user software. The Base and Current registers share the same NS486SXF I/O map locations - a write enters data into the Base registers, a read returns the value of the Current registers. For example, a write to I/O location 0000-0001h writes a 16-bit value into the lower 16-bits of the Base Target Address Register. A read from 0000-0001h, obtains the value of the Current Target Address Register. Because of historical reasons, the addresses for Channels 0-3 are located in the lower portion of the I/O map. Because of convenience, the addresses for Channels 4-5 are located in a non-contiguous area in the I/O map.

All six DMA channels require a Base and Current Target Address Register set. Only the four external

channels using memory to memory mode require the Base and Current requester Address Registers. These channels are Channels 0, 2, 3, and 4. These registers define the source/destination addresses of external memory for transfers for memory mapped DMA devices.

In all types of transfers and for all six channels, the Base and Current Byte Count registers are required. The Base Byte Count Register holds the starting number of bytes to be transferred. As transfers are made, the DMA controller logic automatically updates the Current Byte Count Register. Again, the Base register is write only, and the Current register is read only. They share a single I/O map address (two byte addresses per channel).

The Command Register holds bits controlling priority modes and priorities as well as the global DMA controller enable/disable bit.

Channel Mode Registers control the characteristics of each channel. Two registers are required to support all six channels. Data transfer type, TC vs. EOP termination, requester device type and transfer type are set in these write-only registers.

The two write-only Data Mode Registers set up the data timing parameters for each channel.

A Mask Register enables or disables individual channels. (The Command Register holds the global enable/disable bit.)

The TC Status Register flags when a channel has reached the terminal count or received EOP.

The Request Status Register flags which channels have pending DMA requests.

The Buffer Chaining Registers control the chaining mode of DMA transfer. Setting up the chain registers allows a single DMA arbitration cycle to support a series of reads and writes.

A write-only Master Clear Register acts as a master disable location. A write-only Clear Mask Register acts as a quick channel enable.

Finally, two configuration registers, Internal DMA Request Selection and CPU DMA Request Register configure generation of DRQ signals.

3.3.2.7.1 Address Registers

There are Address Registers for each channel. These are shown below. The first four registers consist of two 8-bit page registers (high and low) and a lower 16-bit address. This combination yields a 4 Gigabyte address range.

- Base Target Address (32-bit register)
- Current Target Address (32-bit register)
- Base requester Address Register (26-bit register)

- Current requester Address Register (26-bit register)
 - Base Byte Count (16-bit, providing transfers up to 64Kbytes)
 - Current Byte Count (16-bit, supporting transfers up to 64Kbytes)
- Note: The Base registers are write only and the Current registers are read only, which allow them to occupy the same I/O locations without conflict..

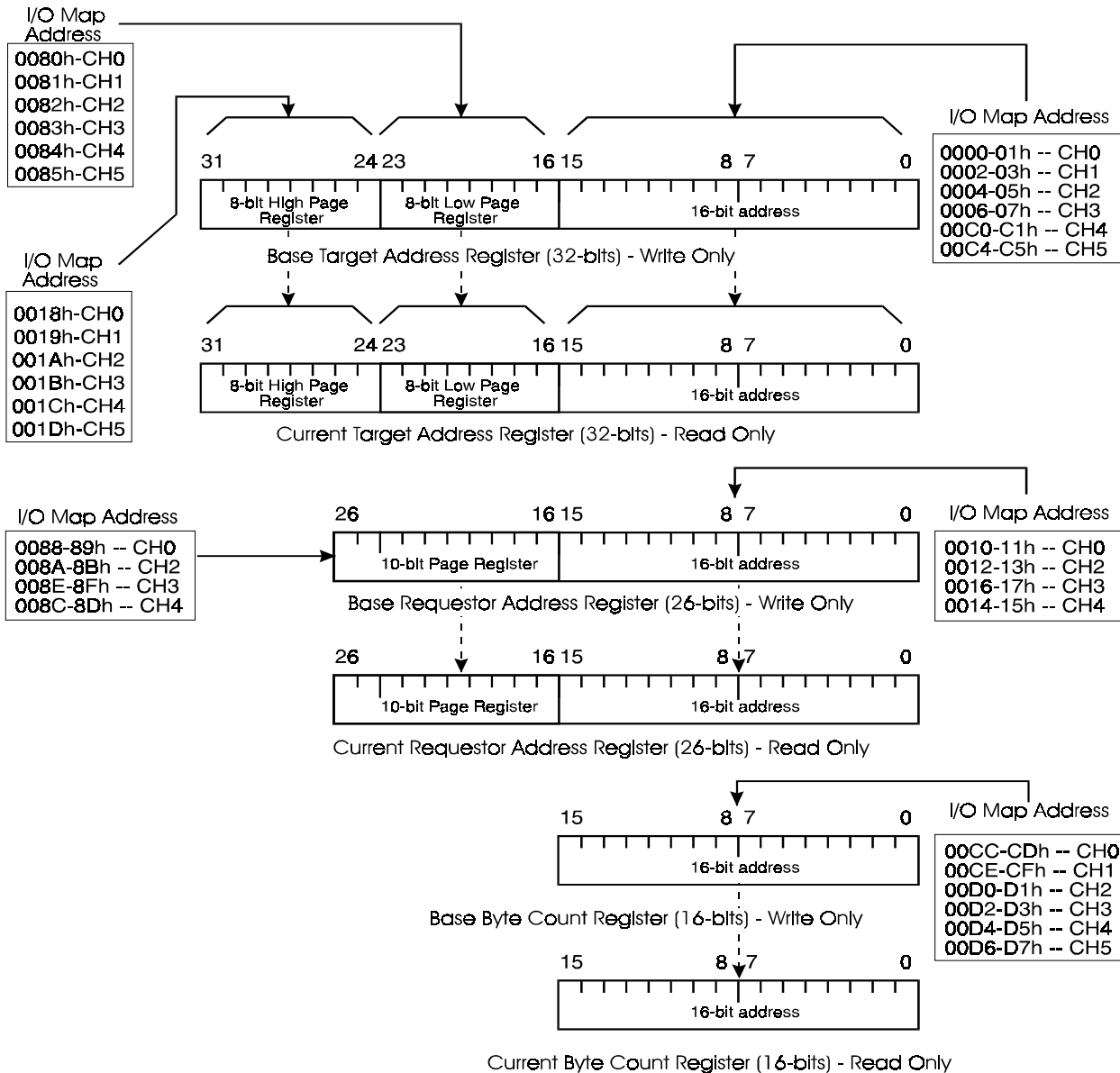
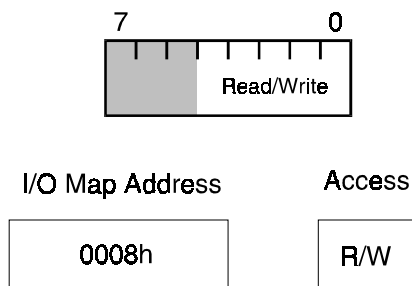


Figure 3-13 DMA Controller Address Registers

3.3.2.7.2 Command Register

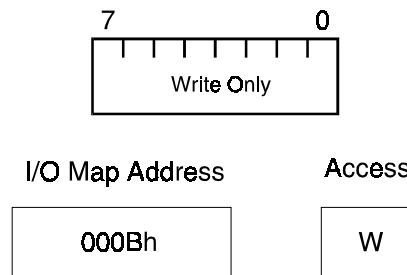
This 8-bit write only register controls the operation of the DMA controller. It is programmed by the NS486SXF CPU and is cleared by Reset or a Master Clear instruction. The reset value of this register is 18h.



- Bits 7-5: Reserved
- Bits 4-2: Programmable Priority Bits — The value loaded in these bits determine the lowest priority channel. Whatever value (from 0-5) is placed in these bits sets that DMA channel as the lowest priority. The highest priority is next highest channel (in rotating order, i.e., 0, 1, 2, 3, 4, 5, 0, 1, 2, 3, 4, 5, 0, 1, 2, 3, 4, 5, and so on). The reset default value of these three bits is 110₆, an invalid setting. Be sure to program a value of 0-5 into these bits. If a value of 2 or 010₆ is loaded into these bits, then Channel 2 will be the lowest priority (and Channel 3 the highest). If a value of 3 is loaded, Channel 3 will be the lowest and Channel 4 will be the highest, and so on. **Note:** Do not program a non-existent channel number into these bits, or the chip can hang-up.
- Bit 1: Priority Mode
 0 = Fixed priority
 1 = Rotating Priority
- Bit 0: Controller Enable/Disable
 0 = Controller disabled
 1 = Controller enabled

3.3.2.7.3 Channel Mode Register CH0-CH3

The Channel Mode Register is write-only and is used to set up the particular characteristics of the DMA channel. Channel programming is accomplished on a channel by channel basis, with the channel to be programmed selected by bits 0 and 1. Following a reset, all the bits which may be programmed via this register will be zeros.



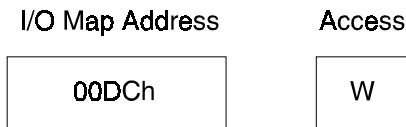
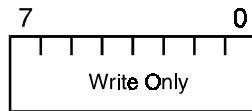
- Bits 7, 6: Data Transfer Mode
 00 demand mode
 01 single mode
 10 block mode
 11 cascade-master mode
- Bit 5: TC vs. $\overline{\text{EOP}}$
 0 = TC/ $\overline{\text{EOP}}$ pin will operate as terminal count (TC).
 1 = TC/ $\overline{\text{EOP}}$ pin will operate as End Of Process EOP input.
- Bit 4: requester Device Type
 0 = I/O
 1 = Memory
- Bit 3: Transfer Type
 0 = Read
 1 = Write
- Bit 2: Autoinitialization
 0 = Autoinitialization disable
 1 = Autoinitialization enable
- Bits 1, 0: Channel Select

Bits	Channel
00	channel 0
01	channel 1
10	channel 2
11	channel 3

Note: Channel 1 does not support memory to memory.

3.3.2.7.4 Channel Mode Register CH4-CH5

The Channel Mode Register is write only and is used to set up the particular characteristics of the DMA channel. Channel programming is accomplished on a channel by channel basis, with the channel to be programmed selected by bits 0 and 1. Following a reset all the bits which may be programmed via this register will be zeros.



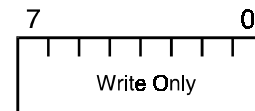
- Bits 7, 6: Data Transfer Mode
 - 00 demand mode
 - 01 single mode
 - 10 block mode
 - 11 cascade-master mode
- Bit 5: TC vs. \overline{EOP}
 - 0 = TC/ \overline{EOP} pin will operate as terminal count (TC).
 - 1 = TC/ \overline{EOP} pin will operate as End Of Process EOP input.
- Bit 4: requester Device Type
 - 0 = I/O
 - 1 = Memory
- Bit 3: Transfer Type
 - 0 = Read
 - 1 = Write
- Bit 2: Autoinitialization
 - 0 = Autoinitialization disable
 - 1 = Autoinitialization enable
- Bits 1, 0: Channel Select

Bits	Channel
00	channel 4
01	channel 5
10	reserved
11	reserved

Note: Channel 5 does not support memory to memory.

3.3.2.7.5 Data Mode Register CH0-CH3

The Data Mode Register is a write-only register which sets up the data timing and width parameters for DMA channels 0-3. Programming is accomplished the same as with the Channel Mode Register. Following a reset, all of the bits which may be programmed via this register will be zeros. This register shares I/O map address 0009h with the Request Status Register (read-only).



- Bit 7: Requester Address Increment/Decrement
 - 0 = decrement
 - 1 = increment
- Bit 6: Target Address Increment/Decrement
 - 0 = decrement
 - 1 = increment
- Bits 5-3: Clocks per Transfer

Bits 5 4 3	I/O Write	I/O Read	Memory (Read or Write)
000	2 clocks	3 clocks	4 clocks
001	4 clocks	4 clocks	4 clocks
010	6 clocks	6 clocks	6 clocks
011	8 clocks	8 clocks	8 clocks
100	10 clocks	10 clocks	10 clocks
101	12 clocks	12 clocks	12 clocks
110	14 clocks	14 clocks	14 clocks
111	16 clocks	16 clocks	16 clocks

- Bit 2: Requester Bus Size
 - 0 = 8-bit data transfer
 - 1 = 16-bit data transfer

Bits 1, 0: Channel Select

Bits	Channel
00	channel 0
01	channel 1
10	channel 2
11	channel 3

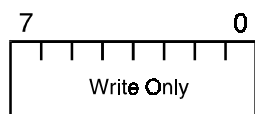
Bit 2: Requester Bus Size
 0 = 8-bit data transfer
 1 = 16-bit data transfer

Bits 1, 0: Channel Select

Bits	Channel
00	channel 4
01	channel 5
10	reserved
11	reserved

3.3.2.7.6 Data Mode Register CH4-CH5

The Data Mode Register is a write only register which sets up the data timing and width parameters for DMA channels 4-5. Programming is accomplished the same as with the Channel Mode Register. Following a reset, all of the bits which may be programmed via this register will be zeros.



I/O Map Address	Access
00DBh	W

3.3.2.7.7 Mask Register

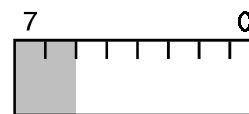
Each channel has an associated mask bit that can be set to disable the incoming DRQ[n]. If the channel is not programmed for Autoinitialize, each mask bit is set when its associated channel produces a Terminal Count (TC) or received an End of Process (EOP).

Each bit of the Mask register may be set or cleared separately under software control. The entire register is set to ones by a Reset or a Master Clear. This disables all DMA requests until a clear Mask register instruction allows them to occur. This register may be read or written.

Bit 7: Requester Address Increment/
 Decrement
 0 = decrement
 1 = increment

Bit 6: Target Address Increment/
 Decrement
 0 = decrement
 1 = increment

Bits 5-3: Clocks per Transfer



I/O Map Address	Access
000Fh	R/W

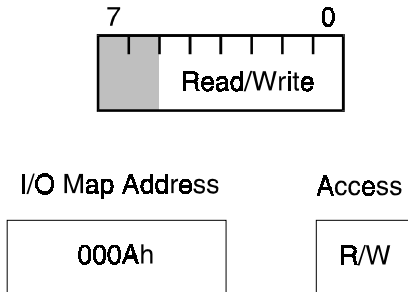
Bits 5 4 3	I/O Write	I/O Read	Memory (Read or Write)
000	2 clocks	3 clocks	4 clocks
001	4 clocks	4 clocks	4 clocks
010	6 clocks	6 clocks	6 clocks
011	8 clocks	8 clocks	8 clocks
100	10 clocks	10 clocks	10 clocks
101	12 clocks	12 clocks	12 clocks
110	14 clocks	14 clocks	14 clocks
111	16 clocks	16 clocks	16 clocks

Bit 7: Reserved
 Bit 6: Reserved
 Bit 5: Channel 5 Mask bit
 0 = Channel enable
 1 = Channel disable
 Bit 4: Channel 4 Mask bit
 0 = Channel enable
 1 = Channel disable
 Bit 3: Channel 3 Mask Bit
 0 = Channel enable
 1 = Channel disable
 Bit 2: Channel 2 Mask Bit

- 0 = Channel enable
- 1 = Channel disable
- Bit 1: Channel 1 Mask Bit
- 0 = Channel enable
- 1 = Channel disable
- Bit 0: Channel 0 Mask Bit
- 0 = Channel enable
- 1 = Channel disable

3.3.2.7.8 TC Status Register

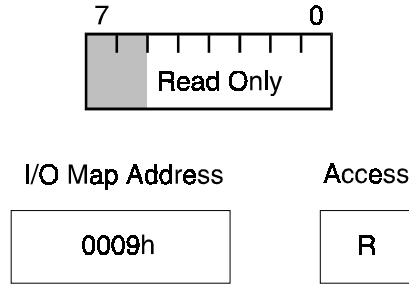
The Terminal Count (TC) Status Register indicates which channels have reached a terminal count. When a channel reaches its terminal count, or an \overline{EOP} associated with that channel is produced, that channel's terminal count bit is set to a one. These bits are cleared to zeros upon Reset and upon a master-clear. Write a zero to a specific bit to clear the bit.



- Bit 7: Reserved
- Bit 6: Reserved
- Bit 5: 1 = Channel 5 at Terminal Count
- Bit 4: 1 = Channel 4 at Terminal Count
- Bit 3: 1 = Channel 3 at Terminal Count
- Bit 2: 1 = Channel 2 at Terminal Count
- Bit 1: 1 = Channel 1 at Terminal Count
- Bit 0: 1 = Channel 0 at Terminal Count

3.3.2.7.9 Request Status Register

The Request Status Register is read-only. This register contains information about which channels have pending DMA requests. These bits are set to one when their corresponding channel is requesting service. These bits are cleared to zeros upon Reset and upon Master-Clear.



- Bit 7: Reserved
- Bit 6: Reserved
- Bit 5: 1 = Channel 5 Request
- Bit 4: 1 = Channel 4 Request
- Bit 3: 1 = Channel 3 Request
- Bit 2: 1 = Channel 2 Request
- Bit 1: 1 = Channel 1 Request
- Bit 0: 1 = Channel 0 Request

3.3.2.7.10 Buffer Chaining

Buffer Chaining is a method of efficiently transferring data by enabling source/destination register updating simultaneously with data transfers. In this mode, the DMA controller allows the CPU to write new base register information to the DMA controller while the old base/current target address and byte count registers are being used to move data.

This is accomplished by programming the DMA controller to interrupt the CPU for more programming information while the previously programmed transfer is still in progress. In Buffer Chaining mode, the DMA controller loads the new transfer information automatically when the first transfer completes. In this way, the entire transfer can be completed without interrupting the operation of the DMA device. This mode is most useful for single cycle or demand modes of the controller where the transfer process allows

time for the CPU to execute the DMA interrupt routine.

The example below shows the sequence of a Buffer Chaining operation.

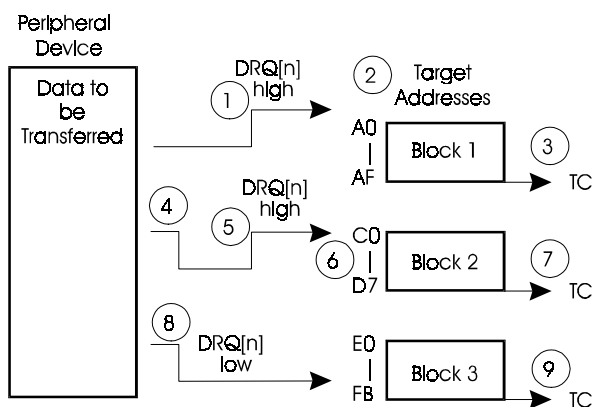


Figure 3-14 Buffer Chaining Example

In this example, a block of data is to be transferred to three discrete locations starting at A0h, C0h and E0h. The number of bytes transferred will be 10h, 18h and 1Ch, respectively. Before starting, it is understood that user software has programmed the DMA controller to be in buffer chaining mode, that it is enabled, and that CPU interrupt routines exist to handle the requirements of the buffer chaining operation.

INITIAL STATE: The DRQ[n] signal is low, indicating no request is being made.

- 1) **START:** The peripheral asserts its DRQ[n] DMA request signal high. This starts a DMA arbitration sequence. If the DMA channel wins the arbitration, the following sequence begins.
 - 2) The CPU has programmed the DMA Controller with the initial values of Base Target Address Register (A0h), Base requester Address Register (needed in memory-to-memory transfers only), and the Base Byte Count Register (10h). These values are automatically and immediately transferred from the base registers to the current registers. The user software then sets the Chaining Mode bits, enabling buffer chaining mode operation. Then the user software loads the DMA Controller Base registers with the values for the second transfer (C0h and 18h). Because the
- Chaining Mode bits are set, these values will not be immediately transferred into the Current Registers. The DMA Controller begins the transfer of data using the address pointed to in the current address registers.
- 3) Only after the terminal count is reached for the first transfer, will the DMA Controller automatically transfer the second set of base register values (C0h and 18h) into the current registers. It will then set the appropriate bits in the Chaining Base Empty Status Register to indicate that the base registers are now empty and ready to be updated. The DMA Controller will then also assert an internal interrupt request to the CPU for DMA Chaining mode. This pending interrupt request indicates to the CPU that the DMA Controller is in Buffer Chaining mode and that the Base registers are ready to be refilled with the next target address. While this CPU activity is occurring, the DMA Controller is continuing with the second data transfer.
 - 4) **Note for Demand Mode:** In order for the CPU to regain control of the bus and thus be able to recognize the pending interrupt, DRQ[n] must be deasserted before the second TC occurs.
 - 5) To continue the DMA transfer in Demand Mode, the DRQ[n] must be reasserted.
 - 6) In the interrupt routine, the CPU loads new values into the Base registers (E0h and 1Ch), sets the Chain Mode Enabled bits, and clears the Chaining Base Empty Status bits. This routine must be completed BEFORE the second TC is reached.
 - 7) After the second TC is reached, the DMA Controller again automatically transfers the third set of base register values (E0h and 1Ch) into the Current registers.
 - 8) **Note:** In the interrupt routine triggered by the second TC, the user software realizes that the Chaining Mode transfer is almost finished (no further chaining action is required) and so the user's software disables Chaining Mode by writing the appropriate values to the Chaining Mode Enable bits. Again, this routine must be completed BEFORE the third TC is reached. Because the third (and final) set of addresses are already in place in the DMA controller, there is no further need to assert any additional interrupt requests. If software does not disable the chaining mode after

the second TC, the DMA controller will mask off the channel.

- 9) The DMA transfer terminates when the third TC is reached because Chaining Mode is no longer enabled.

A channel is initialized for buffer chaining by programming the DMA base register with the appropriate initial values, then programming the Chaining Mode register to enable chaining mode (Chaining register, bits 4, 3 = 01). The base registers must then be programmed with the appropriate address/byte count values for the next DMA transfer. When the Current Byte (or Word) Count register reaches terminal count, the DMA controller loads the Current registers from the Base register, sets the appropriate bit in the Chaining Base Empty Status Register and asserts an internal interrupt request for DMA Chaining mode service. This pending interrupt request indicates that the Base register are empty and chaining mode is enabled.

The Base register must be updated and the Chaining register must be set to “base register update complete (Chaining register bits 3, 2 = 11)” before the Current Byte Count register reaches zero, or the DMA controller will terminate the data transfer by setting the channel’s bit in the Channel Base Empty Register, and setting the channel’s mask bit in the Mask Register. Note that setting bits 3,2 = 11 in the Chaining Register also automatically clears the bits in the Chaining Base Empty Status Register.

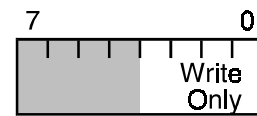
The DMA controller asserts its internal interrupt request only after reaching a terminal count or receiving an EOP (with chaining mode enabled). It does not assert the internal interrupt request during the initial programming sequence that loads the DMA base register twice. The interrupt handler updates the channel’s base registers, then programs the Chaining Mode register with “base register update complete (Chaining register bits 3, 2 = 11)”.

When chaining mode is enabled, only the Base registers are loaded by the user software. The Current registers load automatically after the Current Byte Count register reaches terminate count or EOP is received. The processor can read the Current registers, but not load them.

The base empty bit for each DMA channel (refer to the Chaining Base Empty Status Register) is cleared whenever its “base register update complete” is performed (Writing 11 into bits 3,2 of the Chaining Register). On the other hand, a DMA channel’s base empty bit is set to a one whenever it reaches a Terminal Count (TC) or receives an End of Process (EOP) signal.

3.3.2.7.11 Chaining Register CH0-CH3

The Chaining Register is a write-only register which allows the enabling or disabling of chaining mode on a per channel (CH0-CH3) basis. This register also provides a means to indicate that the base registers have been updated. Following a reset all bits that are programmable via this register will be zeros.

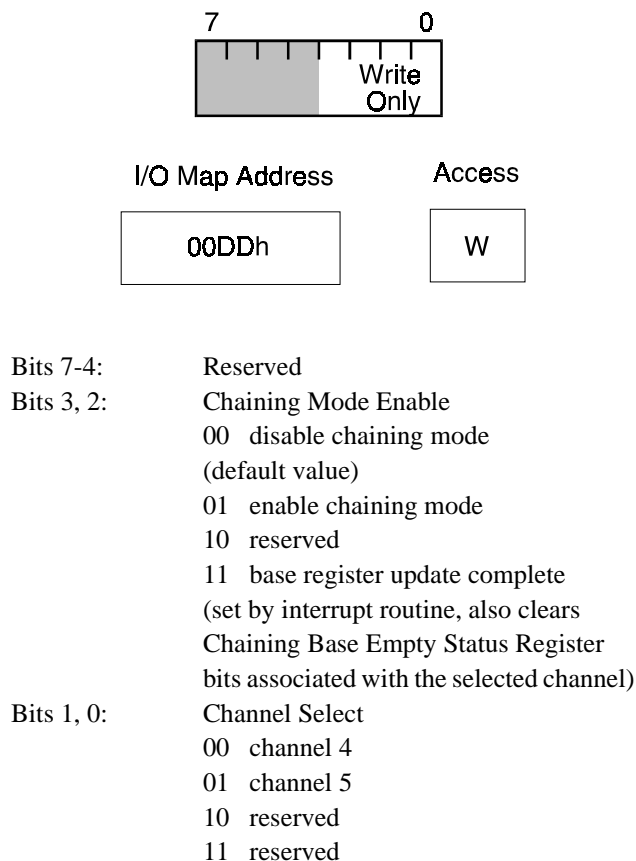


I/O Map Address	Access
00DAh	W

Bits 7-4:	Reserved
Bits 3, 2:	Chaining Mode Enable 00 disable chaining mode (default value) 01 enable chaining mode 10 reserved 11 base register update complete (set by interrupt routine, also clears Chaining Base Empty Status Register bits associated with the selected channel)
Bits 1, 0:	Channel Select 00 channel 0 01 channel 1 10 channel 2 11 channel 3

3.3.2.7.12 Chaining Register CH4-CH5

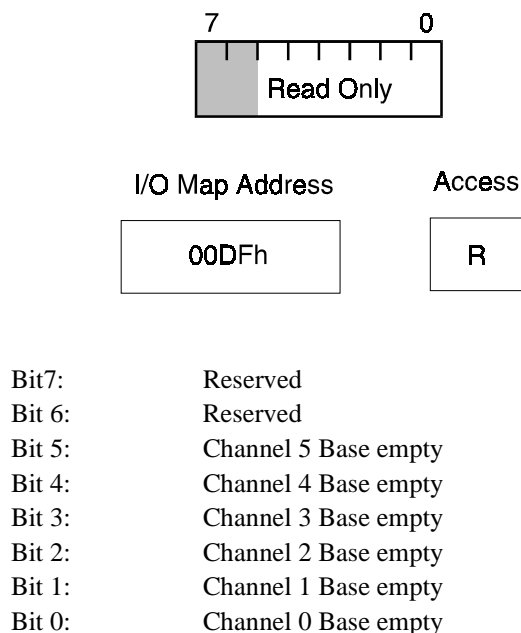
The Chaining Register is a write-only register which allows the enabling or disabling of chaining mode on a per channel (CH4-CH5) basis. This register also provides a means to indicate that the base registers have been updated. Following a reset all bits that are programmable via this register will be zeros.



3.3.2.7.13 Chaining Base Empty Status Register

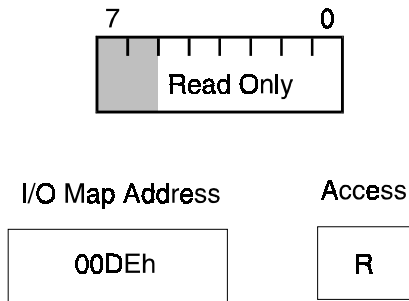
This read-only status register indicates which (if any) of the channel's base registers are empty. This register only has meaning for channels which are programmed for chaining mode transfers. Following a Reset or Master Clear all bits in this register are zeros. The bits are set to one everytime the Chaining Mode it is enabled and the base register value is loaded into the current address register. It indicates that the base is available to be reprogrammed with the next value. These bits are automatically cleared when the values 11 are written into bits 3 and 2 of the Chaining Register (indicating that the base values have been successfully rewritten by the CPU interrupt program).

When a Chaining Base Empty Status bit becomes set, the DMA controller generates the chaining interrupt. Setting the base register update complete clears the chaining interrupt.



3.3.2.7.14 Chaining Mode Channel Status Register

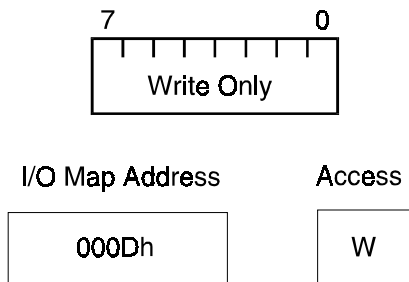
This read-only status register indicates which (if any) of the channels are programmed to be in chaining mode. Following a Reset or Master Clear all bits in this register are zeros.



- Bit 7: Reserved
- Bit 6: Reserved
- Bit 5: Channel 5 enabled (for chaining)
- Bit 4: Channel 4 enabled (for chaining)
- Bit 3: Channel 3 enabled (for chaining)
- Bit 2: Channel 2 enabled (for chaining)
- Bit 1: Channel 1 enabled (for chaining)
- Bit 0: Channel 0 enabled (for chaining)

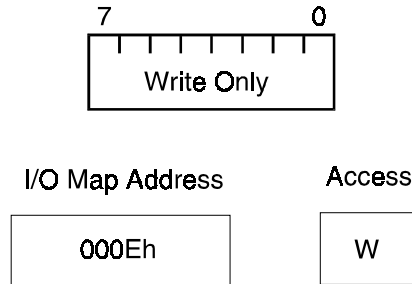
3.3.2.7.15 Master Clear — Write Only

The Master Clear instruction clears Command and Status registers and sets the Mask register to disable DMA requests it also clears the chaining mode registers and chaining base empty status register. Any operation in progress is aborted. When the DMA controller is enabled, any write access to I/O location 000Dh will generate a Master Clear.



3.3.2.7.16 Clear Mask Register — Write Only

The Clear Mask register command enables all DMA channels by clearing the mask bits. Any write access to this register performs the clear.

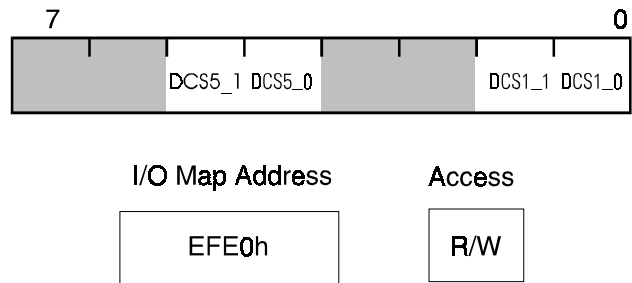


3.3.2.8 DMA Configuration Registers

3.3.2.8.1 Internal DMA Request Selection

During a system reset the bits in this register are set to 00h. This register is located at I/O location EFE0h.

This register determines what internal function if any is connected to the internal DMA channels 1 and 5.



- Bit 7-6: Reserved.
- Bit 5-4: DCS5_1-DCS5_0 — Internal DMA Channel Selection for Channel 5, bits 1-0. These two bits determine the internal device connected to internal channel 5 of the DMA Controller as follows:

DCS5_1	DCS5_0	Function
0	X	None
1	0	ECP
1	1	LCD

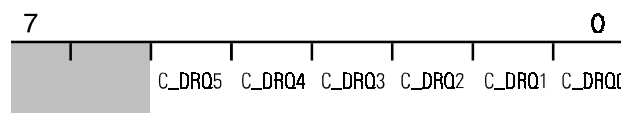
Bit 3-2: Reserved
 Bit 1-0: DCS1_1-DCS1_0 — Internal DMA Channel Selection for Channel 1, bits 1-0. These two bits determine the internal device connected to internal channel 1 of the DMA Controller as follows:

DCS1_1	DCS1_0	Function
0	X	None
1	0	ECP
1	1	LCD

3.3.2.8.2 CPU DMA Request Register

During a system reset the bits in this register are set to 00h. This register is located at I/O location EFE1h.

Writing a 1 into a bit in this register will result in generating a DMA request to the associated internal DMA Channel. In this manner the NS486SXF may make a DMA request via software without any additional external hardware. Transfers with auxiliary processors and shared memory are also supported via this requesting technique. **Note:** The DMA channel must be unmasked for this register to cause a DMA request to occur.



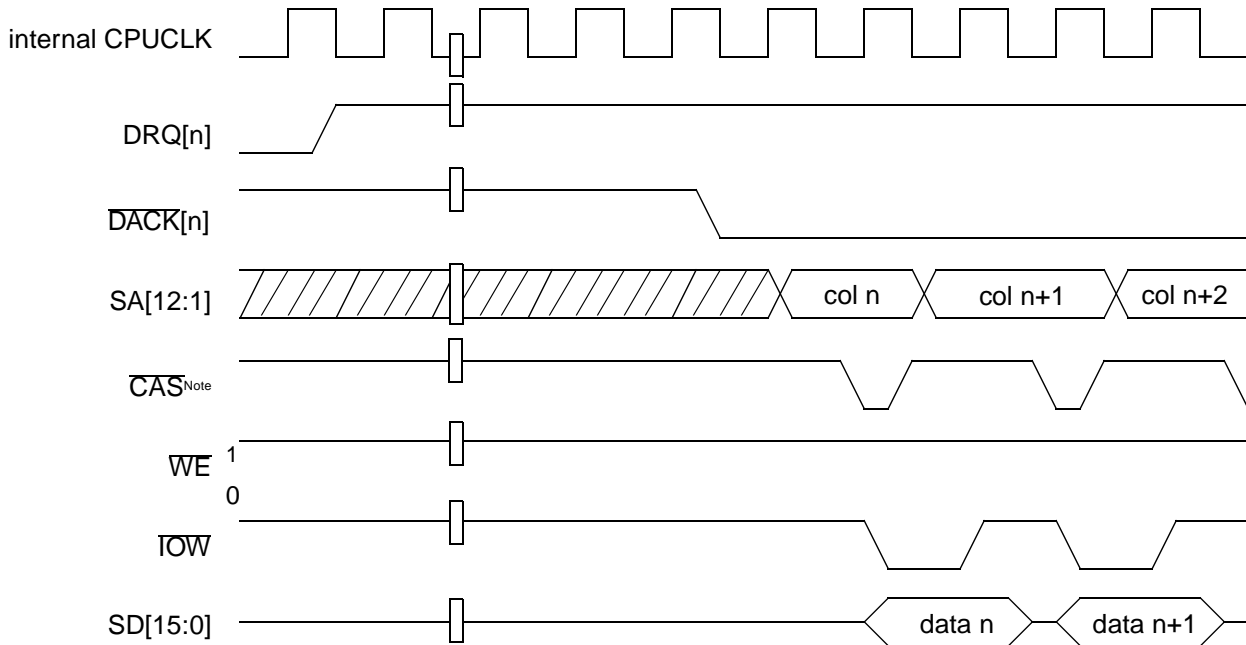
I/O Map Address

Access

EFE1h

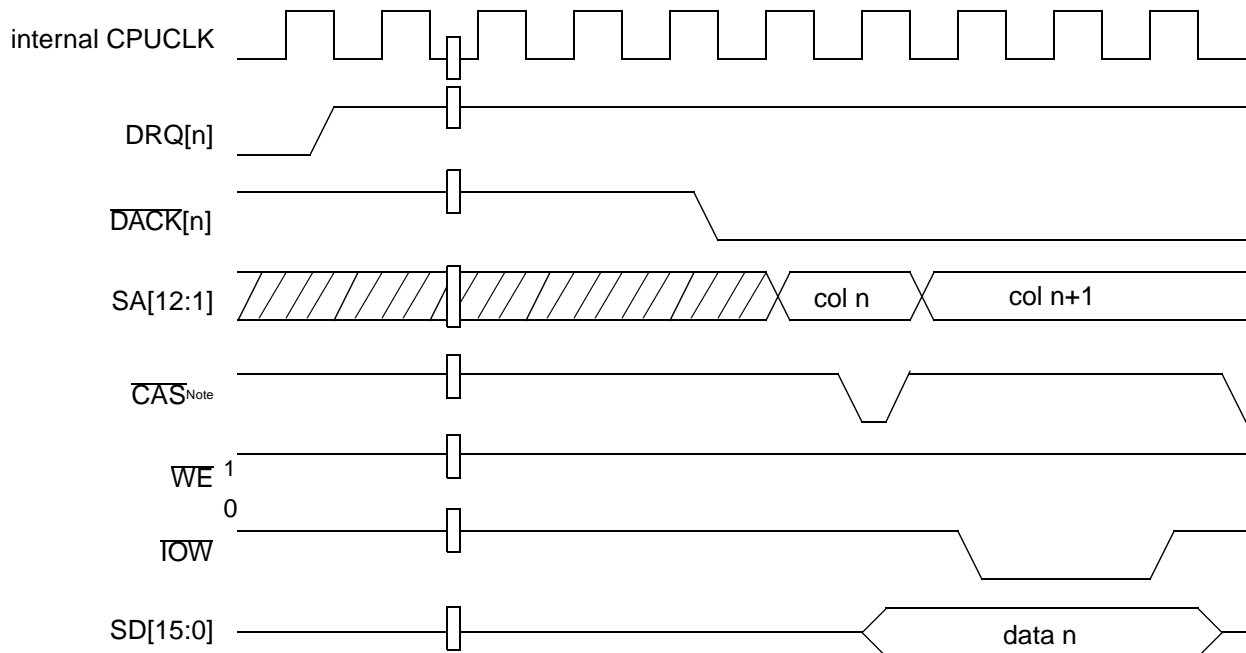
R/W

Bit 7-6: Reserved.
 Bit 5-0: C_DRQ5 - C_DRQ0. CPU DMA Requests 5-0. When a one is written to these bits, a DMA request is generated to the corresponding DMA channel(s). The DMA request(s) will remain active until zeroes are written into this register.



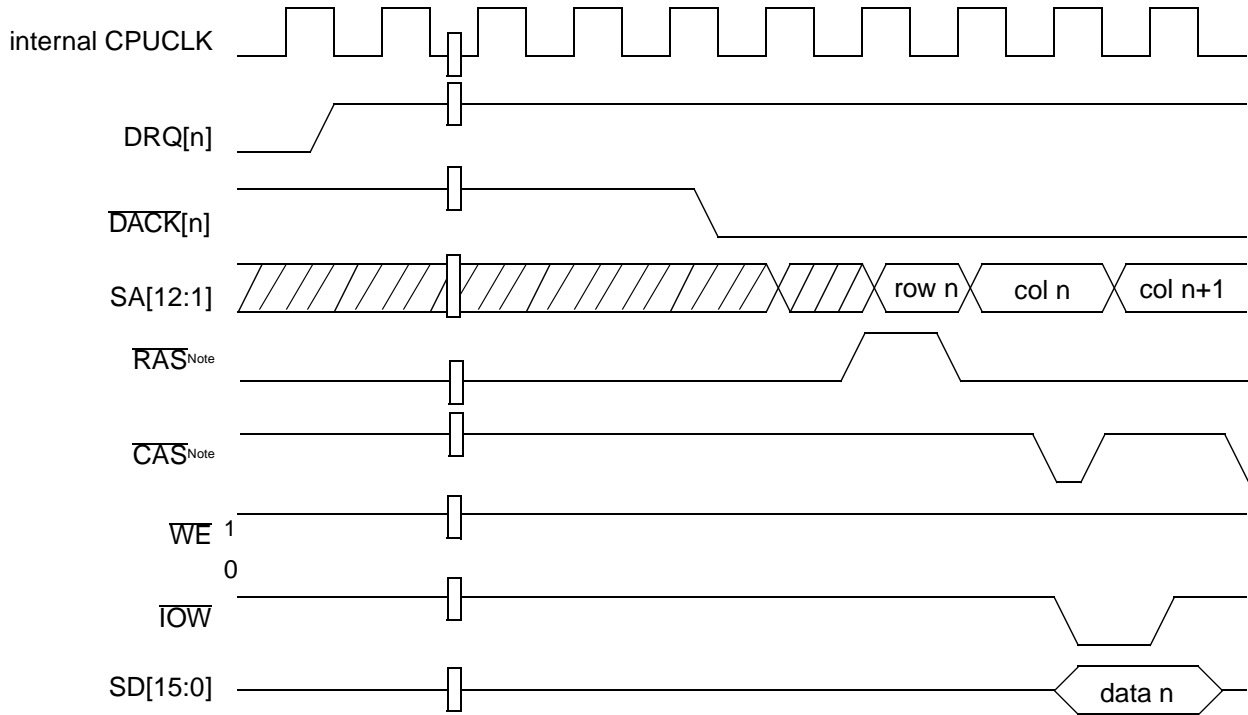
Note: A DRAM Page Hit is assumed for this timing diagram.

**Figure 3-15 I/O Mapped DMA Requester, No Wait State Write Cycle
(2 clocks per transfer)**



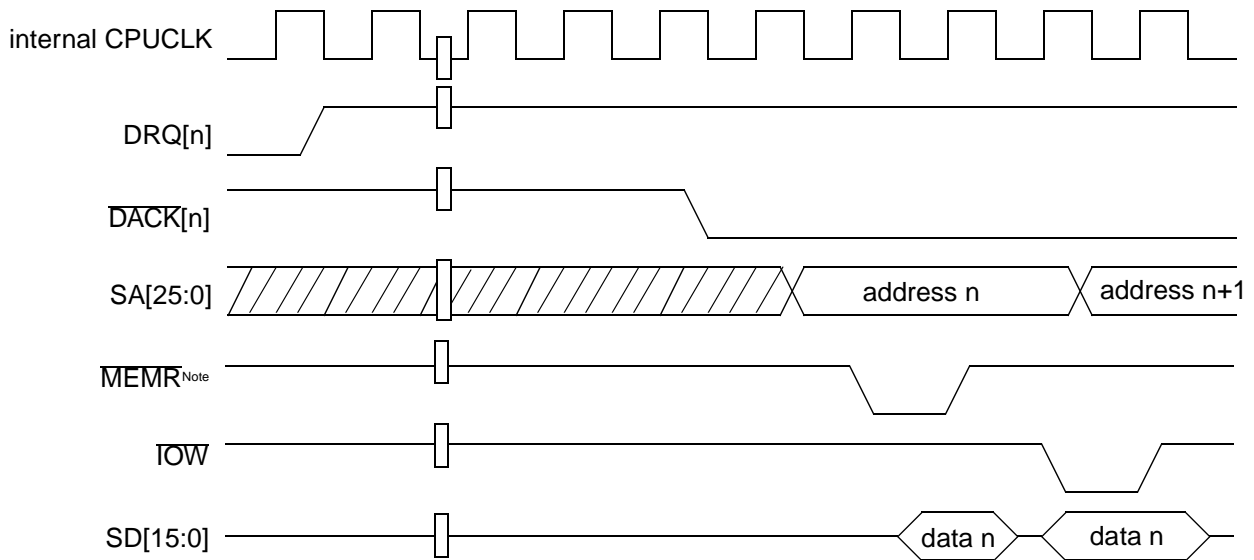
Note: A DRAM Page Hit is assumed for this timing diagram.

**Figure 3-16 I/O Mapped DMA Requester, One Wait State Write Cycle
(4 clocks per transfer)**



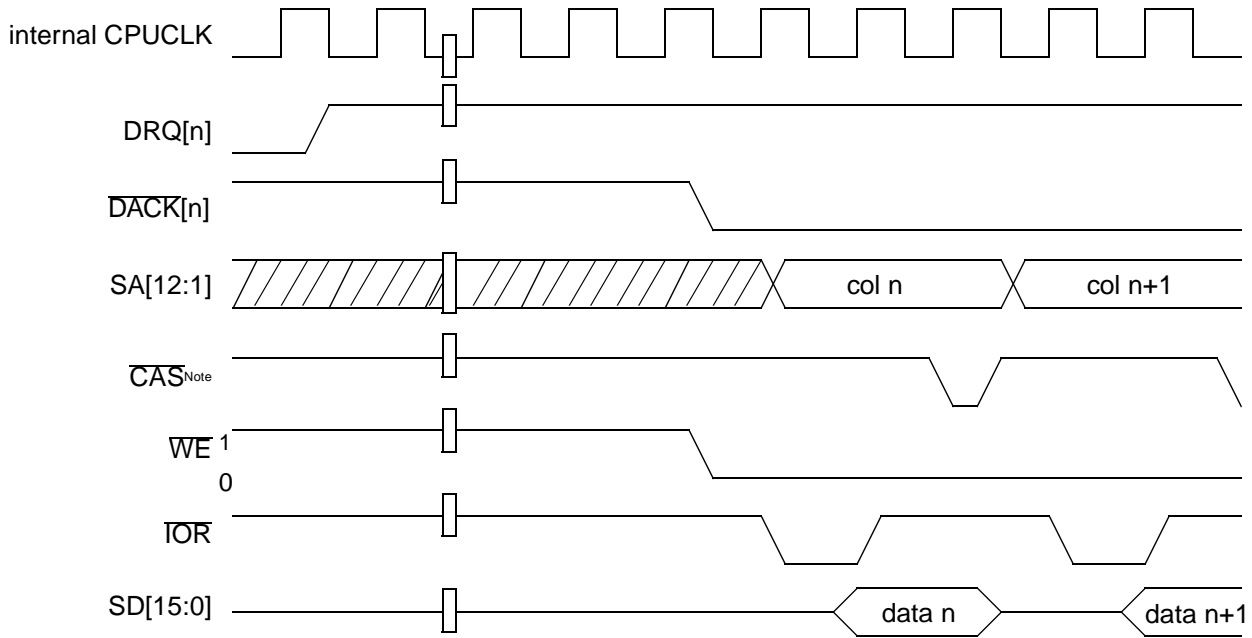
Note: A 3 Clock DRAM Page Miss is assumed for this timing diagram.

Figure 3-17 I/O Mapped DMA Requester, No Wait State Write Cycle



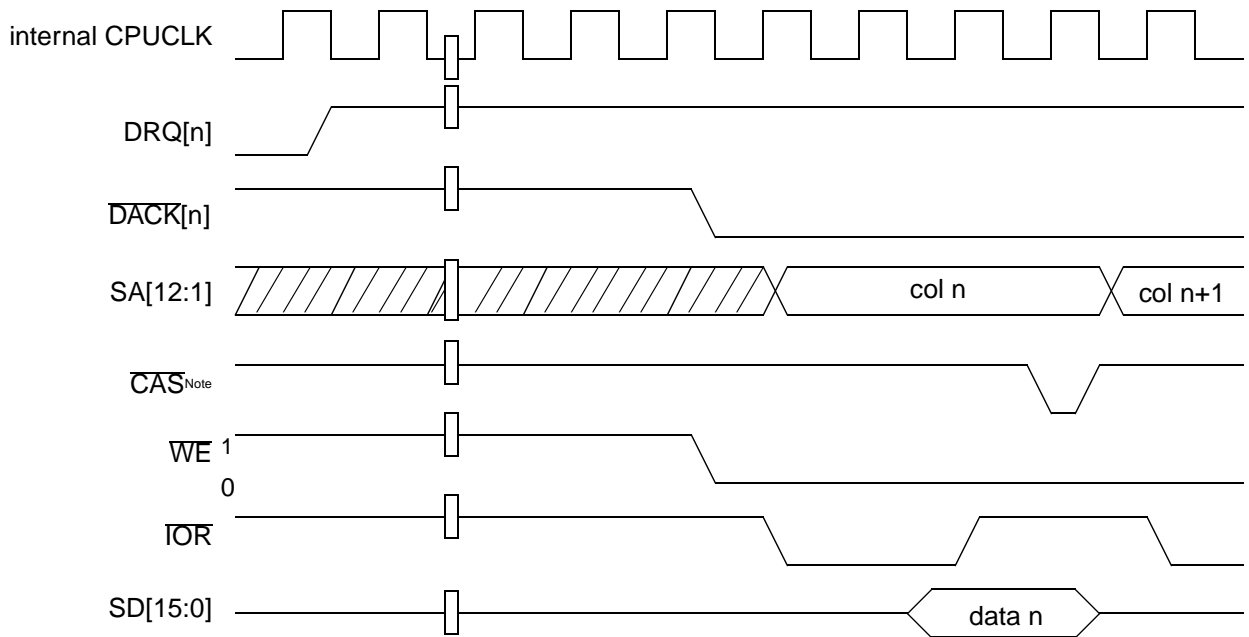
Note: An external no wait state SRAM cycle is assumed for this timing diagram.

Figure 3-18 I/O Mapped DMA Requester, No Wait State Write Cycle



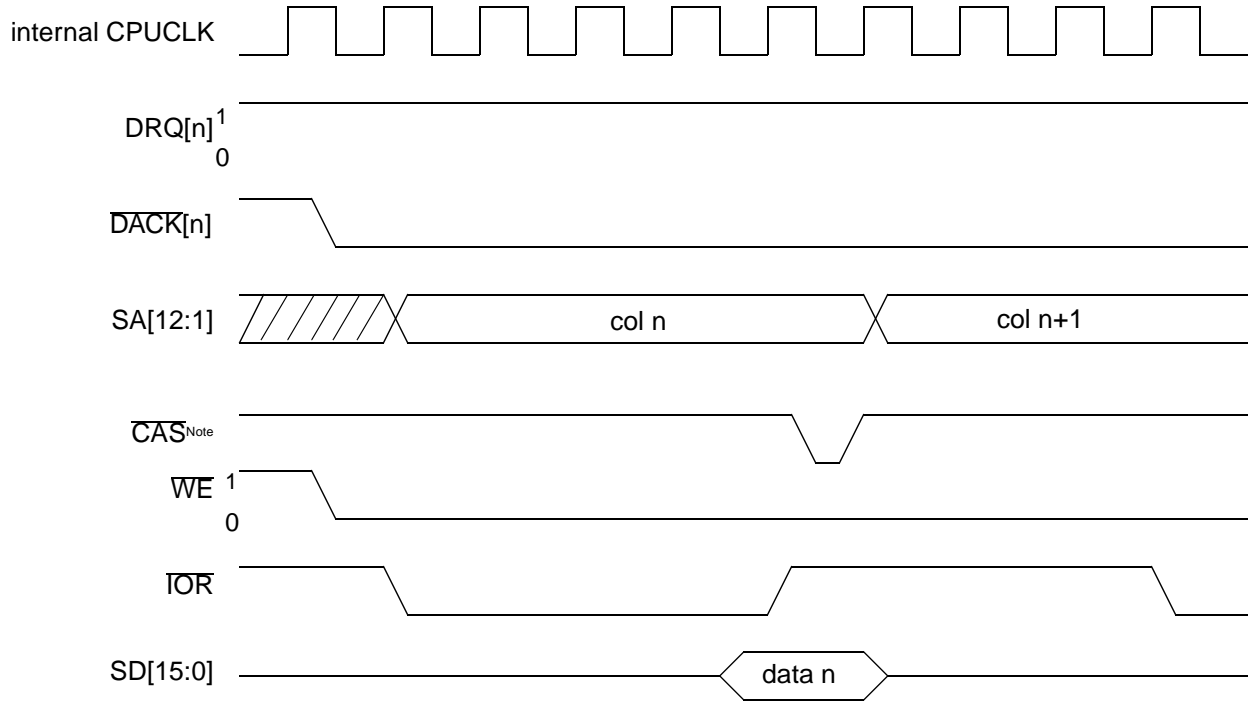
Note: A DRAM Page Hit is assumed for this timing diagram.

Figure 3-19 I/O Mapped DMA Requester, No Wait State Read Cycle (3 clocks per transfer)



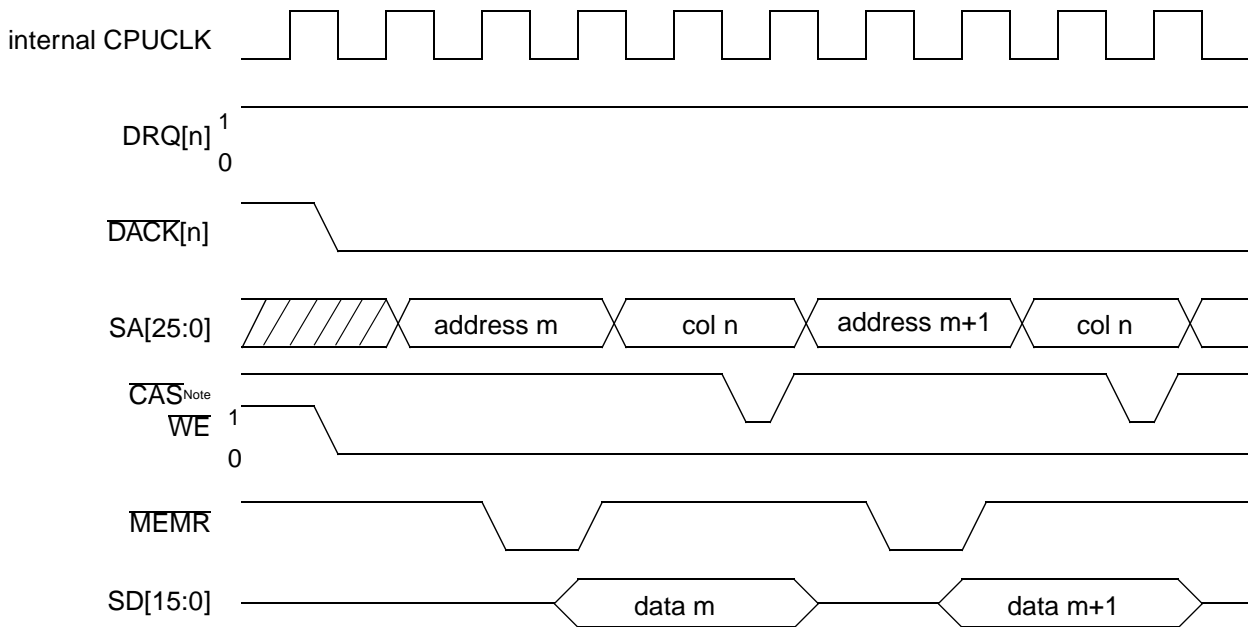
Note: A DRAM Page Hit is assumed for this timing diagram.

Figure 3-20 I/O Mapped DMA Requester, One Wait State Read Cycle (4 clocks per transfer)



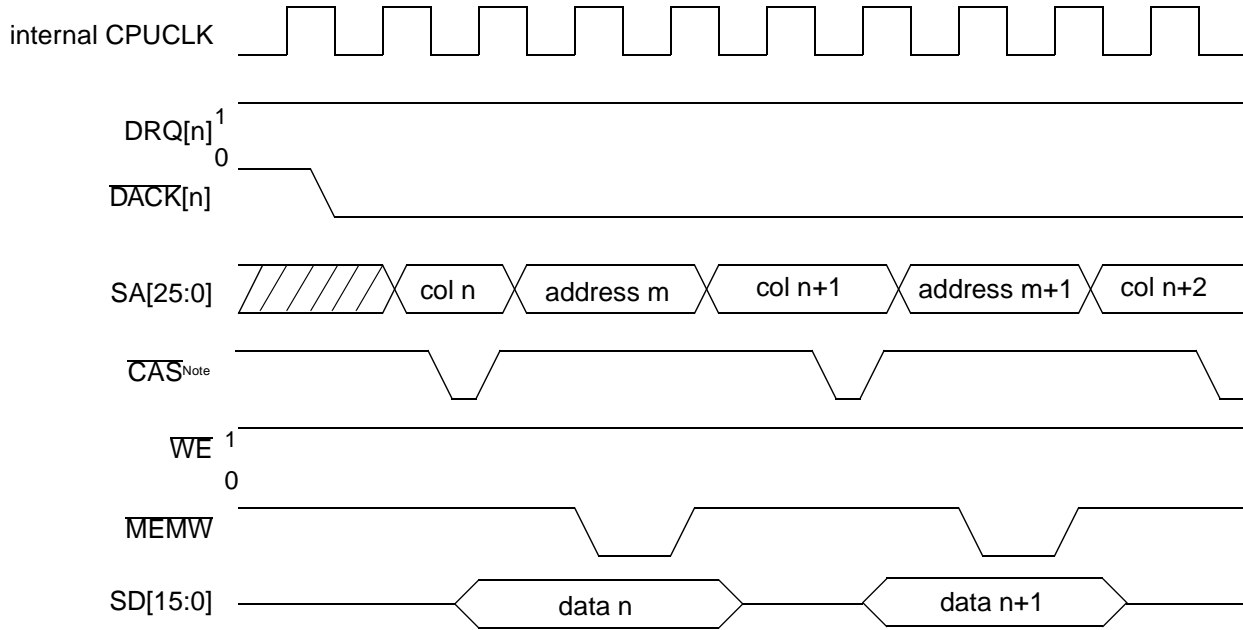
Note: A DRAM Page Hit is assumed for this timing diagram.

Figure 3-21 I/O Mapped DMA Requester, Three Wait State Read Cycle (8 clocks per transfer)



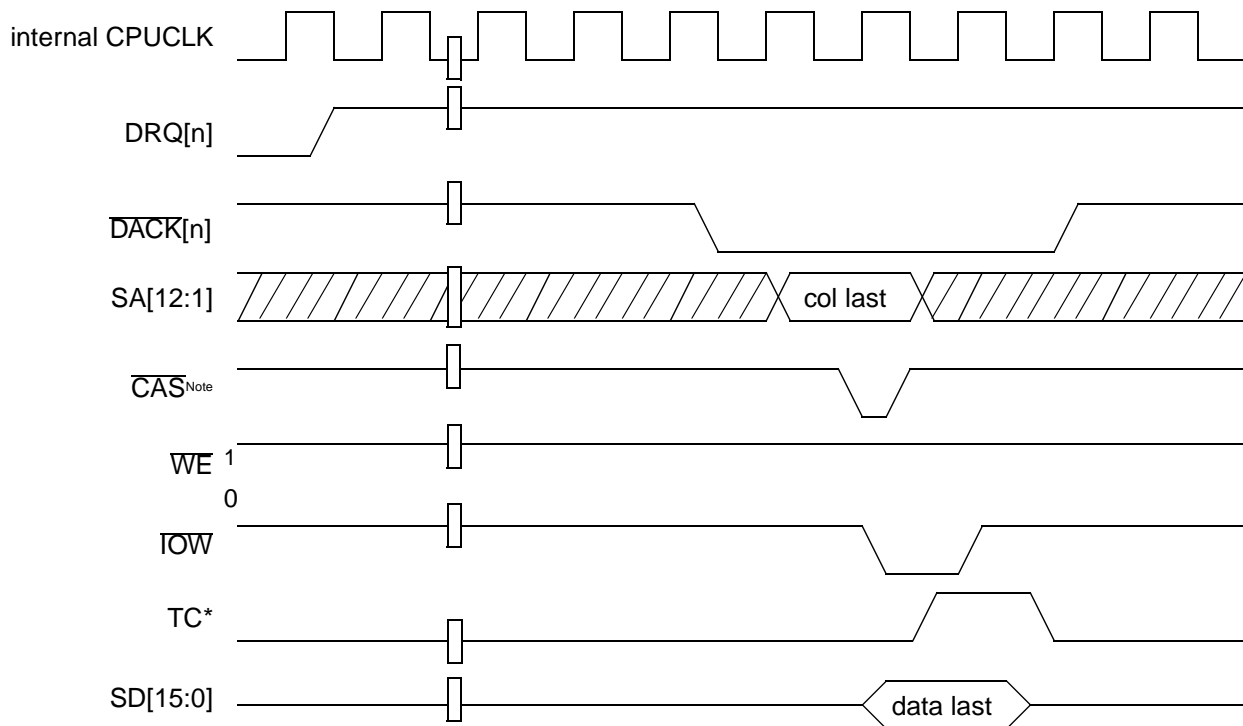
Note: A DRAM Page Hit is assumed for this timing diagram.

Figure 3-22 Memory Mapped DMA Requester, No Wait State Read Cycle (4 clocks per transfer)



Note: A DRAM Page Hit is assumed for this timing diagram.

**Figure 3-23 Memory Mapped DMA Requester, No Wait State Write Cycle
(4 clocks per transfer)**



Note: A DRAM Page Hit is assumed for this timing diagram.

*EOP has the same timing as TC, except it is the responsibility of the DMA Requester to drive EOP active low.

**Figure 3-24 I/O Mapped DMA Requester, No Wait State Write Cycle
(2 clocks per transfer)**

3.3.3 The Programmable Interval Timer (PIT)

The NS486SXF programmable interval timer is compatible with the Intel 8254 programmable interval timer and contains three identical timers, CH0, CH1, and CH2. CH0 and CH1 can be used to generate accurate timing delays under software control. CH2 can be used to provide a WATCHDOG timer function.

The PIT is programmed through I/O ports 0040h-0043h. Three timer channels, CH0, CH1, and CH2, are outputs of three 16-bit presetable down counters that can be programmed to count in binary or binary coded decimal (BCD). The counters are completely independent and can operate either as timers or counters. Both can be programmed to operate in various modes. CH0 and CH1 are driven from either an external source or an internally generated clock signal divided by 4, 8, 16 or 32. The internally generated clock can be selected to be the CPU operating frequency or the raw input (oscillator) clock divided by two (See Section 3.3.7.) CH2 is driven by a 1 kHz clock provided by the Real Time Clock.

Note: To use the WATCHDOG timer function (CH2), the Real Time Clock's crystal circuit must be installed and operating.

Logic common to all of the counters reads and writes data to and from the 8-bit data port to the appropriate counter.

Each counter is identical and contains a Control Word Register, a Status Register, a 16-bit down counting element, two input holding registers, two output holding registers, a clock input, a gate input (GATE) and an out signal (OUT).

The PIT contains one register, the Control Word Register, which determines how the counters operate. When latched, the Status Register contains the current contents of the Control Word Register and the status of the output and null count flag. The 16-bit down counting element is presetable and synchronous.

The Input Holding Registers are two 8-bit latches used to store the LSB and MSB of the 16-bit count written to the counter until they are transferred to the counting element on the next falling edge of the appropriate clock input. The Output Holding Registers are two 8-bit latches used to latch the LSB and MSB of the present 16-bit count, thus enabling the count to be read.

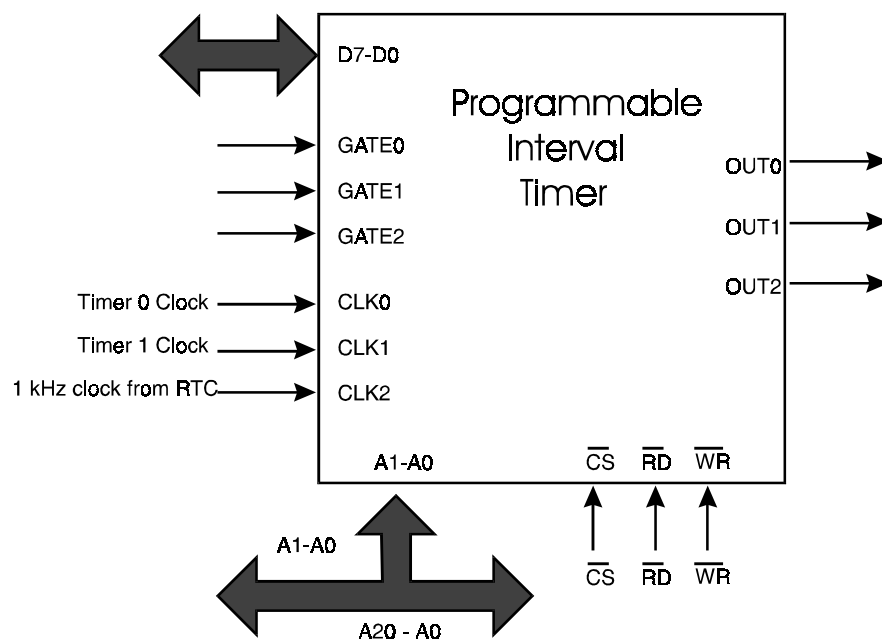


Figure 3-25 Programmable Interval Timer

The internally divided Oscillator clock provides the clock signal for loading and decrementing the counting element. The functionality of CH0's and CH1's GATE and OUT signals are defined by the counter mode and the status of the counting element.

3.3.3.1 Programming the PIT System

Immediately after a system reset, the PIT is not accessible via its I/O mapped registers until access is enabled via the Bus Interface Unit (BIU) Control Register 1 and 2. To enable accesses to the PIT, a "1" must be written to bit 3 of BIU Control Register 1 (I/O address EF00h). Refer to the BIU section for more information.

The PIT's registers and counters power up with random contents. Therefore, each counter must be programmed before it can be used. Counters are programmed by first writing to the Control Word Register, followed by writing an initial count to the appropriate counter. The table below shows the PIT register's I/O addresses.

I/O Address	PIT Register	Type of Access Permitted
0040h	Counter 0	R/W
0041h	Counter 1	R/W
0042h	Counter 2	R/W
0043h	Control Word Register	W only

3.3.3.2 Control Word Register

The Control Word Register is at I/O address 0043h and is a write only register. The Control Word Register determines the counter to be programmed, the counter's mode of operation, the method by which the counter will be read and written, as well as whether the counter is binary or BCD. Note that the Control Word Register will need to be written once for each counter. When a Control Word is written for a counter, all of that counter's control logic is immediately reset and its OUT signal goes to a known initial state.

Bit 7: Select Counter Bit 1 (SC1)
 Bit 6: Select Counter Bit 0 (SC0)

These two bits select the counter to be programmed or the Read Back Command as follows:

Bit 7	Bit 6	Counter/Command Selected
0	0	Counter 0
0	1	Counter 1
1	0	Counter 2
1	1	Read Back Command

Bit 5: Read/Write Bit1 (RW1)

Bit 4: Read/Write Bit0 (RW0)

These two bits select the method that the counter selected with SC1-SC0 will be read or written, or the Counter Latch Command for the selected counter.

Bit 5	Bit 4	Read/Write Method or Command
0	0	Counter Latch Command
0	1	R/W LSB of counter only
1	0	R/W MSB of counter only
1	1	R/W LSB first followed by MSB of counter

The following table summarizes the command and counter accesses as programmed with the Control Word Register Bits 7-4.

Bit 7	Bit 6	Bit 5	Bit 4	Command/Format
0	0	0	0	Counter Latch Command for Counter 0
0	0	0	1	Counter 0 LSB R/W only
0	0	1	0	Counter 0 MSB R/W only
0	0	1	1	Counter 0 LSB and MSB R/W
0	1	0	0	Counter Latch Command for Counter 1
0	1	0	1	Counter 1 LSB R/W only
0	1	1	0	Counter 1 MSB R/W only
0	1	1	1	Counter 1 LSB and MSB R/W
1	0	0	0	Counter Latch Command for Counter 2
1	0	0	1	Counter 2 LSB R/W only
1	0	1	0	Counter 2 MSB R/W only

Bit 7	Bit 6	Bit 5	Bit 4	Command/Format
1	0	1	1	Counter 2 LSB and MSB R/W
1	1	X	X	Read Back Command

Bit 3: Mode Bit 2 (M2)
 Bit 2: Mode Bit 1 (M1)
 Bit 1: Mode Bit 0 (M0)

These three bits determine the mode of operation for the selected counter as follows:

Bit 3	Bit 2	Bit 1	Counting Mode	Mode Description
0	0	0	Mode 0	Interrupt on Terminal Count
0	0	1	Mode 1	Hardware Retriggerable One Shot
X	1	0	Mode 2	Rate Generator
X	1	1	Mode 3	Square Wave Mode
1	0	0	Mode 4	Software Triggered Strobe
1	0	1	Mode 5	Hardware Triggered Strobe

Bit 0: Binary Coded Decimal Bit (BCD)
 0 = Selected Counter is a 16-bit Binary Counter
 1 = Selected Counter is a 4 Decade BCD Counter

3.3.3.4 Counter Read Operations

It is usually desirable to read the value of a counter without disturbing the present count in process. The NS486SXF PIT is able to provide this function in two ways: the Counter Latch and the Read-Back Command.

3.3.3.3 Counter Write Operations

For each counter, the Control Word Register must be written before the initial count. The initial count must follow the counter read/write format as programmed in the Control Word Register.

A count value may also be directly read from a counter. When this method is used, the CLK input for the selected counter must not be enabled, to ensure that a stable count value is read. If the count value is changing at the same instant that the count is read, an invalid count value will be read. When a counter is read (Counter 0 - 0040h, Counter 1 - 0041h and Counter 2 - 0042h), the 16-bit count value must be read in accordance with the read/write sequence programmed through the Control Word Register.

New initial counts may be written into a counter any time without re-issuing a control word, as long as the existing format is observed. This will not affect the counter's programmed mode. The counters are 16-bit and are accessed through an 8-bit port. This means that writes to a counter can be performed in one of three ways:

1. LSB only
2. MSB only
3. LSB followed by MSB.

Current Count Registers

Bits 15-0: Count Bits -- These bits define the count value currently in the counter.

The method of access is defined by the Control Word format. When a Control Word is written, the Input Holding Registers are automatically set to zeros.

Input Holding Registers

Bits 15-0: Count Bits -- These bits define the count value loaded in the counter.

3.3.3.5 Counter Latch Command

This command is written to the Control Word Register (I/O address 0043H). The bits SC1 and SC0 select one of the three counters. Bits RW1 and RW0 distinguish this command from a Control Word. When the Counter Latch Command is received by the Control Word Register, the PIT's Output Holding Registers latch the count of the selected counter. This count is held until read by the CPU or the counter is reprogrammed.

If a counter's value is latched, and the Counter Latch Command is issued again before the CPU has read it, the second counter latch command will be ignored and the value in the Output Holding Register will be that of the first count latched. Similarly, issuing a Read Back command before the CPU has read the latched value will not update the latched count either. The latched count value must be read according to the previously programmed format in the Control Word Register.

Once the counter latch command has been issued for a counter, the value latched is read through the counter I/O port (Counter 0- 0040h, Counter 1 - 0041h, and Counter 2 - 0042h), according to the read/write format programmed in the control word.

Counter Latch Command (0043h) Bits:

Bit 7: Select Counter Bit1 (SC1)
 Bit 6: Select Counter Bit0 (SC0)
 These two bits select the counter to be latched as follows:

Bit 7	Bit 6	Counter Selected
0	0	Counter 0 (I/O Port 0040h)
0	1	Counter 1 (I/O Port 0041h)
1	0	Counter 2 (I/O Port 0042h)
1	1	Reserved

Bit 5: Read/Write Bit1 (RW1)
 Bit 4: Read/Write Bit0 (RW0)
 These two bits must equal 0 to issue the Counter Latch Command.
 Bits 3-0: Don't Care for the Counter Latch Command

Output Holding Register

Bits 15-0: Count Bits
 These bits define the count value latched with the Counter Latch Command.

3.3.3.6 Read-Back Command

This command is written to the Control Word Register (address 0043H). The bits SC1 and SC0 distinguish this command from a Control Word. When these bits are 1,1 the Control Word becomes the Read-Back command and provides the following alternative bit definitions:

Bit 7: Select Counter Bit1 (SC1)
 Bit 6: Select Counter Bit0 (SC0)
 These two bits must be equal to 1 to issue the Read-Back Command.
 Bit 5: Count Bit (COUNT)
 0 = Latch Count of the Selected Counters
 1 = Do Not Latch Count of the Selected Counters
 Bit 4: Status Bit (STATUS)
 0 = Latch Status of the Selected Counters
 1 = Do Not Latch Status of the Selected Counters
 Bit 3: Count 2 Bit (CNT2)
 0 = Do not select Counter 2
 1 = Select Counter 2
 Bit 2: Count 1 Bit (CNT1)
 0 = Do not select Counter 1
 1 = Select Counter 1
 Bit 1: Count 0 Bit (CNT0)
 0 = Do not select Counter 0
 1 = Select Counter 0
 Bit 0: Reserved: 0

The Read-Back Command allows the user to check the value of a selected counter, determine its programmed mode and monitor the status of the OUT signal and Null Count Flag.

This command may be used to latch multiple Output Holding Registers for the counters in the PIT. By setting Bit 5 to zero and selecting the desired counters to be read, the counters' Output Holding Registers can be latched at the same instant, thus enabling all counters to be latched at the same instant. When reading the counters, the pre-programmed format should be observed. The specific counter is automatically un-

latched when read or when the counter is reprogrammed.

(Section 3.3.3.10 on page 63 for a complete description of each mode)

Output Holding Registers:

Bits 15-0: Count Bits — These bits define the count value latched with the Read-Back Command.

The Read-Back Command can also latch status information of selected counters by setting Bit 4 to zero. The Status Register must be latched to be read. The status of a counter is obtained by a read from the selected counter when the Status Register is latched.

This is the Status Byte:

- Bit 7: Output Bit
0 = signal is 0
1 = signal is 1
- Bit 6: Null Count Bit
0 = Count can be read (Count has been loaded)
1 = Null Count (Count has not been loaded)
This bit indicates if the count has been loaded into the counter. The instant this happens is Mode dependent.
- Bit 5: Read/Write Bit1 (RW1)
- Bit 4: Read/Write Bit0 (RW0)
These bits reflect the counter's programmed read/write format as follows:

Bit 5	Bit 4	Programmed Read/Write Method
0	0	Counter Latch Command
0	1	R/W LSB of counter only
1	0	R/W MSB of counter only
1	1	R/W LSB first followed by MSB of counter

- Bit 3: Mode Bit2 (M2)
- Bit 2: Mode Bit1 (M1)
- Bit 1: Mode Bit 0 (M0)
These three bits reflect the counter's programmed mode as follows:

Bit 3	Bit 2	Bit 1	Counting Mode	Mode Description
0	0	0	Mode 0	Interrupt on Terminal Count
0	0	1	Mode 1	Hardware Retriggerable One Shot
X	1	0	Mode 2	Rate Generator
X	1	1	Mode 3	Square Wave Mode
1	0	0	Mode 4	Software Triggered Strobe
1	0	1	Mode 5	Hardware Triggered Strobe

- Bit 0: Binary Coded Decimal Bit (BCD)
0 = Counter is programmed to be a 16-bit Binary Counter
1 = Counter is programmed to be a 4 Decade BCD Counter

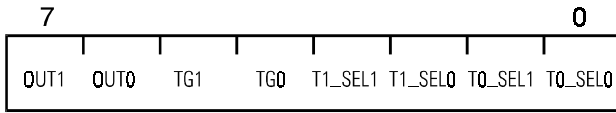
Both the Status and the Count of the selected counters can be latched simultaneously by setting bits 4 and 5 to zeros. The first read after this will always return the status byte. The following one or two reads return the latched count depending on the programmed read/write mode for the counter. Subsequent reads return unlatched counts.

3.3.3.7 External Timer Control Signals

The function of the T1 and T0 pins of the NS486SXF are determined by bits 3-0 of this register. The combinations supported provide the user with access to two function signals on either Timer 0 or Timer 1; or access to one function signal on Timer 0 and one on Timer 1.

This register is located at I/O address 0044h and has a reset value of 00h.

3.3.3.8 Timer I/O Control Register



I/O Map Address

0044h

Access

R/W

- Bit 7: OUT1 — Timer 1 OUT signal. When read, this bit is the state of Timer 1’s OUT signal. This bit is not writable.
- Bit 6: OUT0 — Timer 0 OUT signal. When read, the state of Timer 0’s OUT signal will be read. This bit is not writable.
- Bit 5: TG1 — Timer 1 GATE. When neither the T1 pin nor the T0 pin is selected to drive the GATE input to Timer 1, this bit determines the state of the GATE signal associated with Timer 1.
- Bit 4: TG0 — Timer 0 GATE. When neither the T1 pin nor the T0 pin is selected to drive the GATE input to Timer 0, this bit determines the state of the GATE signal associated with Timer 0.
- Bits 3-2: T1_SEL1, T1_SEL0 — T1 pin function selection bits 1 and 0. These two bits determine the function of the T1 pin of the NS486SXF. If these bits and bits 1-0 of this register result in both T1 and T0 driving either GATE0 or GATE1, then these two signals will be logically ORed to make a single GATE# signal.

T1_SEL1	T1_SEL0	Function of T1 pin
0	0	Timer 0’s GATE input signal (GATE0).
0	1	Timer 1’s GATE input signal (GATE1).
1	0	Timer 1’s OUT output signal (OUT1).
1	1	Timer 0’s CLK input signal (CLK0)

Note: When the T1 pin does not drive Timer 0’s CLK input, the programmable timer clock controlled by the Timer Clock Register will drive the CLK0 input.

When switching between these two possible CLK0 sources, CLK0 may glitch, so the user should not use the timer until after first selecting the appropriate clock source.

Bits 1-0: T0_SEL1, T0_SEL0 — T0 pin function selection bits 1 and 0. These two bits determine the function of the T0 pin of the NS486SXF. If these bits and bits 3-2 of this register result in both T1 and T0 driving either GATE0 or GATE1, then these two signals will be logically ORed together to make a single GATE# signal.

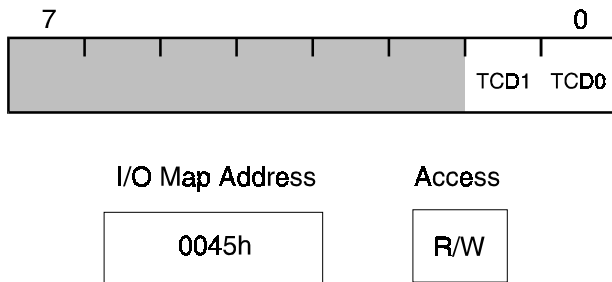
T0_SEL1	T0_SEL0	Function of T0 pin
0	0	Timer 0’s GATE input signal (GATE0).
0	1	Timer 1’s GATE input signal (GATE1).
1	0	Timer 0’s OUT output signal (OUT0).
1	1	Timer 1’s CLK input signal (CLK1).

Note: When the T0 pin does not drive Timer 1’s CLK input, the programmable timer clock controlled by the Timer Clock Register will drive the CLK1 input.

When switching between these two possible CLK1 sources, CLK1 may glitch, so the user should not use the timer until after first selecting the appropriate clock source.

3.3.3.9 Timer Clock Register

This register produces a clock signal which may be used by Timer 0 or Timer 1. This programmable clock divides the frequency of a selected internal clock by 4, 8, 16 or 32. The clock selected may be either the CPU operating clock (affected by Power Management Power Save Modes) or the “raw” oscillator clock divided by two. This register should be programmed before the clock it produces is used. This register is located at IO address 0045h and has a reset value of 00h. The input clock frequency is selected by programming Power Management Register 3. See Section 3.3.7.5.3.



Bits 7-2: Reserved.
 Bits 1-0: TCD1, TCD0 — Timer Clock Divisor bits 1 and 0. These two bits determine the divisor selected to generate a clock which may be used by Timer 1 and/or Timer 0.

TCD1	TCD0	Resulting Clock
0	0	Selected clock divided by 4.
0	1	Selected clock divided by 8.
1	0	Selected clock divided by 16.
1	1	Selected clock divided by 32.

Whenever, the T1 pin is not selected to drive Timer 0’s input CLK signal (see bits 3-2 of the Timer I/O Control Register), this clock source will provide the clock for Timer 0.

Whenever, the T0 pin is not selected to drive Timer 1’s input CLK signal (see bits 1-0 of the Timer I/O Control Register), this clock source will provide the clock for Timer 1.

3.3.3.10 Mode Descriptions

In the following Mode Descriptions section the terms GATE and OUT are used generically to apply as each timers GATE input signal and OUT output signal.

3.3.3.10.1 Mode 0: Interrupt on Terminal Count

After the Control Word is written, OUT is low and remains low until the count reaches zero. Then OUT goes high and remains high until a new count or a new Control Word is written. GATE = 1 enables counting, GATE = 0 disables counting.

The first clock pulse after a Control Word and initial count has been written loads initial count. This clock pulse does not decrement the count. If a two byte count is written, the first byte disables counting and OUT is set low. The second byte allows the new count to be loaded on the next clock pulse. An initial count can be written when GATE = 0 and will still be loaded on the next clock pulse. No clock pulse is needed to load the initial count when GATE goes to a 1.

3.3.3.10.2 Mode 1: Hardware Retriggerable One-Shot

Initially high OUT will go low on the clock pulse after a trigger. This begins the one-shot pulse. OUT remains low until the count reaches zero. OUT then goes high and remains high until the next trigger, then goes low on the next clock pulse. After writing a Control Word and initial count the counter is armed. A trigger loads the initial count and sets OUT low on the next clock pulse. The one-shot is Re-triggerable and can be repeated without writing the same count or Control Word into a counter. GATE has no effect on OUT.

3.3.3.10.3 Mode 2: Rate Generator

OUT is initially high. When the initial count is decremented to 1, OUT goes low for one clock pulse, then goes high again. The counter reloads the initial count and the process is repeated. This same sequence is repeated indefinitely. GATE = 1 enables counting, GATE = 0 disables counting. If GATE goes low during an output pulse, OUT goes high immediately. A trigger loads the counter with the initial count on the next clock pulse. After the trigger, OUT goes low when the count has expired. After writing a Control Word and initial count, the counter will be loaded on the next clock pulse. After the initial count is written, OUT goes low when the count has expired, thus the synchronizing of the counter by software can be achieved. Writing a new count while counting does not affect the current counting sequence. If a trigger is received after writing a new count, but before the end of the current period, the counter will be loaded with the new count on the next clock pulse. Counting will continue from the new count. A count of 1 is illegal in Mode 2.

3.3.3.10.4 Mode 3: Square Wave

Typically used for Baudrate generation, OUT will be initially high, and goes low when half of the initial count has expired and remains low for the remainder of the count. This is periodic and the sequence is continued indefinitely. This is similar to Mode 2. GATE = 1 enables counting, GATE = 0 disables counting. If GATE goes low during an output pulse, OUT goes high immediately. A trigger reloads the counter with the initial count on the next clock pulse. After the trigger, OUT goes low when the count has expired. After writing a Control Word and initial count, the counter will be loaded on the next clock pulse. After the initial count is written, OUT goes low when the count has expired, thus the synchronizing of the counter by software can be achieved. Writing a new count while counting does not affect the current counting sequence. If a trigger is received after writing a new count but before the end of the current half-cycle of the square wave, the counter will be loaded with the new count on the next clock pulse. Counting will continue from the new count. Otherwise, the new count will be loaded at the end of the half-cycle.

For even counts, OUT is initially high. The initial count is loaded on the next clock pulse and is decremented by two on succeeding clock pulses. When the count expires, OUT goes low. The count is reloaded and the process repeats indefinitely.

For odd counts, OUT is initially high. The initial count minus one is loaded on the next clock pulse and is decremented by two on succeeding clock pulses. One pulse after the count expires, OUT goes low. The count is reloaded with the initial count minus one and the process repeats indefinitely.

3.3.3.10.5 Mode 4: Software Triggered Strobe

With OUT initially high, OUT goes low for one clock pulse and then goes high when the initial count expires. Writing the initial count triggers the counter. GATE = 1 enables counting, GATE = 0 disables counting. GATE has no effect on OUT. The counter will be loaded on the next clock pulse after writing the Control Word and initial count. This clock pulse does not decrement the count. A new count, written during counting, will be loaded on the next clock pulse and counting will continue from the new count. If a two byte count is written, the first byte will have no effect on counting, and the second byte will allow the new count to be loaded on the next clock pulse.

3.3.3.10.6 Mode 5: Hardware Triggered Strobe

OUT is initially high. Counting is triggered by a rising edge on the GATE input. OUT will go low for one clock pulse when the count has expired. After writing the initial count and Control Word, the counter will not be loaded until the clock pulse after the trigger. This pulse does not decrement the count. The counter is loaded on the next clock pulse after a trigger. GATE has no effect on OUT. If a new count is written during counting, the current count will not be affected. If a trigger is issued before a count expires, but after a new count is written, the counter will be loaded with the new count on the next clock pulse and counting will continue from there.

3.3.3.11 Gate Input

Sampling of the GATE input occurs on the rising edge of the CLK input to the counter. The GATE input is level sensitive and the logic level is sampled on the rising edge of the clock in modes 0, 2, 3 and 4. In modes 1, 2, 3 and 5 the GATE input is rising edge sensitive. In these latter modes, a rising edge of the GATE sets an edge sensitive flip-flop in the counter. This flip-flop is sampled on the next rising edge of the clock. Immediately after the flip-flop is sampled, it is reset. This ensures a trigger (rising edge of the GATE input) will always be detected.

3.3.4 The WATCHDOG Timer

The NS486SXF WATCHDOG timer, CH2, is a protected 16-bit timer that can be used to prevent system “lockups or hangups”. It uses a 1 KHz clock generated by the on-chip Real Time Clock circuit.

When the WATCHDOG timer is enabled it must be reset by the CPU before it times out. If the WATCHDOG Timer is enabled and times out, either a reset or a non-maskable interrupt (NMI) will be generated, based upon the values in bits 1 and 2 of the WATCHDOG Timer Control Register.

3.3.4.1 WATCHDOG Timer Control Register

The WATCHDOG Timer Control Register may be used to enable the WATCHDOG Timer and select the action to take when the WATCHDOG Timer times out. This register is located at I/O address 0047h and all of its bits will be reset to zero by a hardware reset.



I/O Map Address	Access
0047h	R/W

- Bits 7-3: Reserved
- Bit 2: WT_RST — WATCHDOG Timer Re-Set. When the enabled WATCHDOG Timer times out and this bit is a one, the entire NS486SXF will be reset except for this bit. A hardware reset (i.e. PWGOOD going low) will reset this bit to a zero. This bit may only be written when the WATCHDOG Timer is disabled (i.e. WTE = 0). This bit may be written at the same time the WTE bit is written to a one.
- Bit 1: WT_NMI — WATCHDOG Timer Non-Maskable Interrupt. The first time the enabled WATCHDOG Timer times out, WT_RST is zero and this bit is a one, a NMI will be generated to the CPU. If the WATCHDOG Timer times out a second time under the above conditions without having been reset after the first CPU

NMI, the entire NS486SXF will be reset except for this bit. A hardware reset or setting WT_RST to a one will reset this bit to a zero.

This bit may only be written when the WATCHDOG Timer is disabled (i.e. WTE = 0). This bit may be written at the same time the WTE bit is written to a one.

- Bit 0: WTE — WATCHDOG Timer Enable. This bit enables/disables the WATCHDOG Timer function. When this bit is set to a one, all accesses to the Timer 2 registers in the programmable interval timer will be ignored

When this bit has been set, writes attempting to modify this register will also be ignored. The only ways to reset this bit back to a zero are:

- 1) Hardware Reset (PWGOOD goes low).
- 2) A WATCHDOG Timer generated reset.
- 3) An I/O write to address 0047h with data 69h, followed by an I/O write to address 0046h with data 5Ah.

3.3.4.2 Retriggering the WATCHDOG Timer Count to Prevent Resets

To reset the WATCHDOG Timer's count, the 16-bit value 8421h must be written to IO address 0046h. This will restart the WATCHDOG Timer and prevent either a reset or an NMI.

3.3.4.3 Interrupt Request Supported

The WATCHDOG Timer may also be configured via the Internal Interrupt Steering logic to drive a number of possible interrupt request lines when it times out. In this manner, the user may write zeros to both bits 1 and 2 of the WATCHDOG Timer Control Register and use a standard interrupt request to request servicing. Similar to the NMI functionality, if the WATCHDOG Timer times out twice before being reset, then the entire NS486SXF will be reset.

The user should be warned that using interrupt requests to service the WATCHDOG Timer is an unprotected method of operation. A run-away program may corrupt the interrupt request selection registers and effectively prevent the interrupt request from reaching the CPU. However, in such a case, ultimately the WATCHDOG Timer will timeout twice and reset the entire NS486SXF.

3.3.4.4 Programming Timer 2

To use Timer 2 as a WATCHDOG Timer, the user should program Timer 2 to operate in Mode 5: Hardware Triggered Strobe. The 16-bit count value may be programmed per the user's needs. The user should also note that Timer 2 may only be programmed when the WATCHDOG Timer is disabled (i.e. WTE = 0).

Deliberately Blank Page

3.3.5 The Interrupt Controller

The internal NS486SXF interrupt controller consists of two cascadable programmable interrupt controllers that are compatible with the Intel 8259-2 Programmable Interrupt Controller. They provide a total of 16 programmable interrupts. Three interrupts are reserved for a real time clock tick interrupt, a real time clock interrupt request, and as a cascaded interrupt channel. The remaining 13 interrupts can be used by internal or external sources. **Note:** Both interrupt controllers must be initialized for correct operation.

The following interrupts are defined as specific interrupts request:

- 1) internal_IRQ0, Timer OUT0 (real time tick interrupt)
- 2) Internal_IRQ2 is the cascaded channel
- 3) internal_IRQ8, Real Time Clock (RTC) interrupt request

Interrupt requests internal_IRQ1, internal_IRQ3, internal_IRQ4, internal_IRQ5, internal_IRQ6, internal_IRQ7, internal_IRQ9, internal_IRQ10, internal_IRQ11, internal_IRQ12, internal_IRQ13, internal_IRQ14 and internal_IRQ15 are all considered general purpose interrupt request sources.

External pin IRQ5 becomes the PCMCIA IREQ input when PCMCIA is enabled, otherwise, it can be used for general purpose interrupt requests.

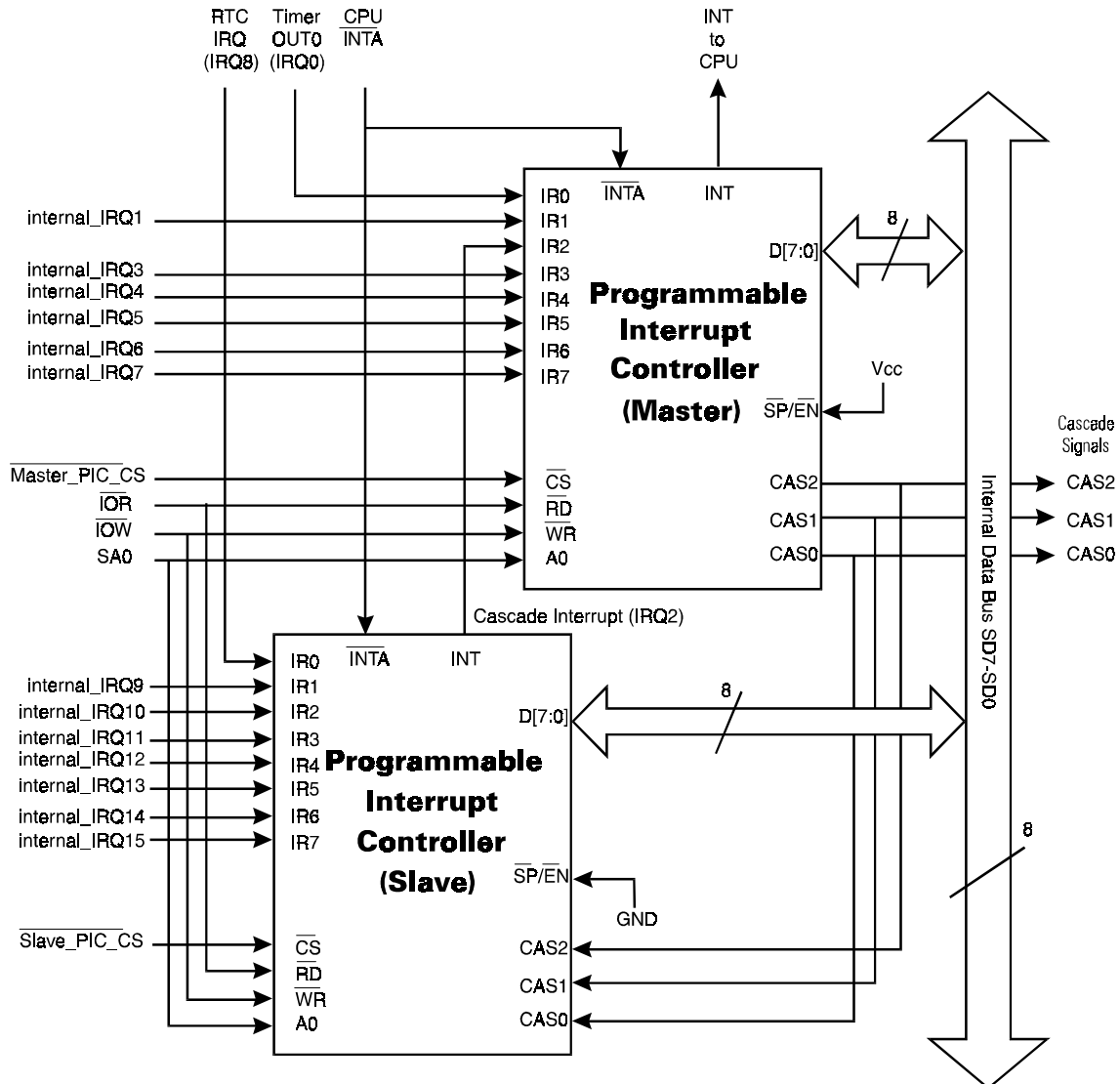


Figure 3-26 Programmable Interrupt Controller Block Diagram

Via steering logic, the six externally defined interrupts (IRQ5-IRQ0) may be steered to the internal interrupt request lines. Also, via programmable steering logic interrupt requests from the internal peripherals may be steered to the available general purpose internal interrupt requests. User software must prevent any interrupt sharing conflicts. These eight interrupt request sources:

- 1) WATCHDOG Timer
- 2) Timer OUT1
- 3) Three-wire Interface
- 4) UART
- 5) LCD Controller
- 6) PCMCIA (internal interrupt)
- 7) ECP Parallel Port
- 8) DMA Controller Chaining

may require up to eight unique internal interrupt requests, which will leave a minimum of five interrupt requests to be connected to the external interrupt requests (IRQ5-IRQ0).

3.3.5.1 External Cascading

Only internal interrupt requests connected to the master interrupt controller can be used to cascade an additional external interrupt controller. This means only internal_IRQ1, internal_IRQ3, internal_IRQ4, internal_IRQ5, internal_IRQ6 and internal_IRQ7 may be connected to a cascaded interrupt controller via the interrupt request steering logic. See Appendix D: External Interrupt Controller.

3.3.5.2 Selecting Interrupt Source(s)

The six external interrupt request pins (IRQ5 - IRQ0) may drive any of the thirteen general purpose internal interrupt request signals. Refer to the Interrupt Steering Registers in the following section for information on how to program which internal interrupt is driven by IRQ5 - IRQ0.

The following table indicates which internal interrupt requests may be selected for the various internal devices (an X means that the associated internal IRQ number may be selected as an interrupt request).

Table 3-2: Interrupt Request Selection

Internal Device	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMA Chaining	X					X			X							
ECP (See ECP Section)		X	X		X	X	X		X		X					
LCD			X	X			X		X			X				
PCMCIA (See PCMCIA Section)	X	X		X	X	X	X		X		X	X	X			
3-wire		X			X		X				X				X	
Timer OUT1		X		X						X			X			
WATCHDOG Timer			X		X					X	X				X	
UART	X					X						X	X			

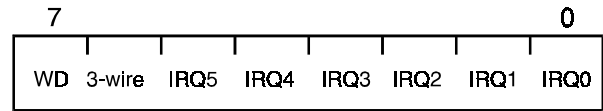
Refer to the following section for information on how to program which internal interrupt is driven by the DMA Controller Chaining, LCD Controller, the 3-wire Interface, the Timer OUT1, WATCHDOG Timer and the UART.

3.3.5.3 Interrupt Source Selection

The following thirteen registers determine what source drives the internal interrupt request lines.

3.3.5.3.1 Internal Interrupt 1 Selection Register

During a system reset the bits in this register are set to 00h. This 8-bit register is located at I/O address EFB1h.



I/O Map Address



Writing a 1 into a bit in this register will result in the associated source driving internal interrupt request 1. If the interrupt controller is programmed to be level sensitive, more than one bit may be programmed to share internal interrupt request 1. If the interrupt controller is programmed to be edge sensitive, then only one bit should be written as a 1; otherwise conflicts may result (i.e., edge sensitive interrupts cannot be shared). It is the responsibility of the software to avoid any conflicts.

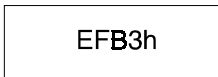
- Bit 7: WD — WATCHDOG. When this bit is a 1, the WATCHDOG timer’s interrupt request signal will drive the internal interrupt request 1.
- Bit 6: 3-wire — When this bit is a 1, the 3-wire interface’s interrupt request signal will drive internal interrupt request 1.
- Bits 5-0: IRQ5-IRQ0 — When the corresponding register bit is a 1, the external interrupt request pin (IRQ5-0) will drive internal interrupt request 1.

3.3.5.3.2 Internal Interrupt 3 Selection Register

During a system reset the bits in this register are set to 00h. This 8-bit register is located at I/O address EFB3h.



I/O Map Address



Writing a 1 into a bit in this register will result in the associated source driving internal interrupt request 3. If the interrupt controller is programmed to be level sensitive, more than one bit may be programmed to share internal interrupt request 3. If the interrupt controller is programmed to be edge sensitive, then only one bit should be written as a 1; otherwise conflicts may result. It is the responsibility of the software to avoid any conflicts.

The PCMCIA interrupt signal may also drive internal interrupt request 3. That configuration is detailed in the PCMCIA section.

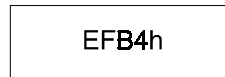
- Bit 7: UART — When this bit is a 1, the UART's interrupt request signal will drive the internal interrupt request 3.
- Bit 6: TM1 — Timer OUT1. When this bit is a 1, the Timer OUT1 signal will drive internal interrupt request 3.
- Bits 5-0: IRQ5-IRQ0 — When the corresponding register bit is a 1, the external interrupt request pin (IRQ5-0) will drive internal interrupt request 3.

3.3.5.3.3 Internal Interrupt 4 Selection Register

During a system reset the bits in this register are set to 00h. This 8-bit register is located at I/O address EFB4h.



I/O Map Address



Writing a 1 into a bit in this register will result in the associated source driving internal interrupt request 4. If the interrupt controller is programmed to be level sensitive, more than one bit may be programmed to share internal interrupt request 4. If the interrupt controller is programmed to be edge sensitive, then only one bit should be written as a 1; otherwise conflicts may result. It is the responsibility of the software to avoid any conflicts.

- Bit 7: UART — When this bit is a 1, the UART's interrupt request signal will drive the internal interrupt request 4.
- Bit 6: LCD — When this bit is a 1, the LCD's interrupt request signal will drive the internal interrupt request 4.
- Bits 5-0: IRQ5-IRQ0 — When the corresponding register bit is a 1, the external interrupt request pin (IRQ5-0) will drive internal interrupt request 4.

3.3.5.3.4 Internal Interrupt 5 Selection Register

During a system reset the bits in this register are set to 00h. This 8-bit register is located at I/O address EFB5h.



I/O Map Address

EFB5h

Writing a 1 into a bit in this register will result in the associated source driving internal interrupt request 5. If the interrupt controller is programmed to be level sensitive, more than one bit may be programmed to share internal interrupt request 5. If the interrupt controller is programmed to be edge sensitive, then only one bit should be written as a 1; otherwise conflicts may result. It is the responsibility of the software to avoid any conflicts.

The PCMCIA and ECP interrupt signals may also drive internal interrupt request 5. That configuration is detailed in the PCMCIA and ECP sections.

- Bit 7: WD — WATCHDOG. When this bit is a 1, the WATCHDOG timer's interrupt request signal will drive the internal interrupt request 5.
- Bit 6: 3-wire — When this bit is a 1, the 3-wire interface's interrupt request signal will drive internal interrupt request 5.
- Bits 5-0: IRQ5-IRQ0 — When the corresponding register bit is a 1, the external interrupt request pin (IRQ5-0) will drive internal interrupt request 5.

3.3.5.3.5 Internal Interrupt 6 Selection Register

During a system reset the bits in this register are set to 00h. This 8-bit register is located at I/O address EFB6h.



I/O Map Address

EFB6h

Writing a 1 into a bit in this register will result in the associated source driving internal interrupt request 6. If the interrupt controller is programmed to be level sensitive, more than one bit may be programmed to share internal interrupt request 6. If the interrupt controller is programmed to be edge sensitive, then only one bit should be written as a 1; otherwise conflicts may result. It is the responsibility of the software to avoid any conflicts.

The PCMCIA card status change interrupt signal may also drive internal interrupt request 6. That configuration is detailed in the PCMCIA section.

- Bit 7: WD — WATCHDOG. When this bit is a 1, the WATCHDOG timer's interrupt request signal will drive the internal interrupt request 6.
- Bit 6: TM1 — Timer OUT1. When this bit is a 1, the Timer OUT1 signal will drive internal interrupt request 6.
- Bits 5-0: IRQ5-IRQ0 — When the corresponding register bit is a 1, the external interrupt request pin (IRQ5-0) will drive internal interrupt request 6.

3.3.5.3.6 Internal Interrupt 7 Selection Register

During a system reset the bits in this register are set to 00h. This 8-bit register is located at I/O address EFB7h.



I/O Map Address

EFB7h

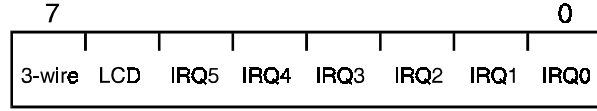
Writing a 1 into a bit in this register will result in the associated source driving internal interrupt request 7. If the interrupt controller is programmed to be level sensitive, more than one bit may be programmed to share internal interrupt request 7. If the interrupt controller is programmed to be edge sensitive, then only one bit should be written as a 1; otherwise conflicts may result. It is the responsibility of the software to avoid any conflicts.

The PCMCIA and ECP interrupt signals may also drive internal interrupt request 7. That configuration is detailed in the PCMCIA and ECP sections.

- Bit 7: DMA — DMA Chaining Interrupt Request. When this bit is a 1, the DMA Controller chaining interrupt request signal will drive internal interrupt request 7.
- Bit 6: LCD — When this bit is a 1, the LCD's interrupt request signal will drive the internal interrupt request 7.
- Bits 5-0: IRQ5-IRQ0 — When the corresponding register bit is a 1, the external interrupt request pin (IRQ5-0) will drive internal interrupt request 7.

3.3.5.3.7 Internal Interrupt 9 Selection Register

During a system reset the bits in this register are set to 00h. This 8-bit register is located at I/O address EFB9h.



I/O Map Address

EFB9h

Writing a 1 into a bit in this register will result in the associated source driving internal interrupt request 9. If the interrupt controller is programmed to be level sensitive, more than one bit may be programmed to share internal interrupt request 9. If the interrupt controller is programmed to be edge sensitive, then only one bit should be written as a 1; otherwise conflicts may result. It is the responsibility of the software to avoid any conflicts.

The PCMCIA and ECP interrupt signals may also drive internal interrupt request 9. That configuration is detailed in the PCMCIA and ECP sections.

- Bit 7: 3-wire — When this bit is a 1, the 3-wire interface's interrupt request signal will drive internal interrupt request 9.
- Bit 6: LCD — When this bit is a 1, the LCD's interrupt request signal will drive the internal interrupt request 9.
- Bits 5-0: IRQ5-IRQ0 — When the corresponding register bit is a 1, the external interrupt request pin (IRQ5-0) will drive internal interrupt request 9.

3.3.5.3.8 Internal Interrupt 10 Selection Register

During a system reset the bits in this register are set to 00h. This 8-bit register is located at I/O address EFBAh.



I/O Map Address



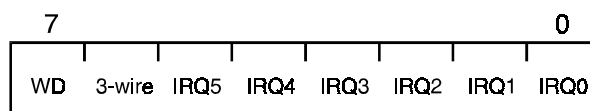
Writing a 1 into a bit in this register will result in the associated source driving internal interrupt request 10. If the interrupt controller is programmed to be level sensitive, more than one bit may be programmed to share internal interrupt request 10. If the interrupt controller is programmed to be edge sensitive, then only one bit should be written as a 1; otherwise conflicts may result. It is the responsibility of the software to avoid any conflicts.

The PCMCIA and ECP interrupt signals may also drive internal interrupt request 10. That configuration is detailed in the PCMCIA and ECP sections.

- Bit 7: UART — When this bit is a 1, the UART's interrupt request signal will drive the internal interrupt request 10.
- Bit 6: DMA — DMA Chaining Interrupt Request. When this bit is a 1, the DMA Controller's chaining interrupt request signal will drive internal interrupt request 10.
- Bits 5-0: IRQ5 - IRQ0 — When the corresponding register bit is a 1, the external interrupt request pin (IRQ5-0) will drive internal interrupt request 10.

3.3.5.3.9 Internal Interrupt 11 Selection Register

During a system reset the bits in this register are set to 00h. This 8-bit register is located at I/O address EFBh.



I/O Map Address



Writing a 1 into a bit in this register will result in the associated source driving internal interrupt request 11. If the interrupt controller is programmed to be level sensitive, more than one bit may be programmed to share internal interrupt request 11. If the interrupt controller is programmed to be edge sensitive, then only one bit should be written as a 1; otherwise conflicts may result. It is the responsibility of the software to avoid any conflicts.

The PCMCIA and ECP interrupt signals may also drive internal interrupt request 11. That configuration is detailed in the PCMCIA and ECP sections.

- Bit 7: WD — WATCHDOG. When this bit is a 1, the WATCHDOG timer's interrupt request signal will drive the internal interrupt request 11.
- Bit 6: 3-wire — When this bit is a 1, the 3-wire interface's interrupt request signal will drive internal interrupt request 11.
- Bits 5-0: IRQ5-IRQ0 — When the corresponding register bit is a 1, the external interrupt request pin (IRQ5-0) will drive internal interrupt request 11.

3.3.5.3.10 Internal Interrupt 12 Selection Register

During a system reset the bits in this register are set to 00h. This 8-bit register is located at I/O address EF-BCh.



I/O Map Address



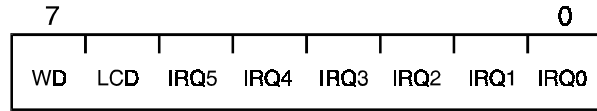
Writing a 1 into a bit in this register will result in the associated source driving internal interrupt request 12. If the interrupt controller is programmed to be level sensitive, more than one bit may be programmed to share internal interrupt request 12. If the interrupt controller is programmed to be edge sensitive, then only one bit should be written as a 1; otherwise conflicts may result. It is the responsibility of the software to avoid any conflicts.

The PCMCIA and ECP interrupt signals may also drive internal interrupt request 12. That configuration is detailed in the PCMCIA and ECP sections.

- Bit 7: TM1 — Timer OUT1. When this bit is a 1, the Timer OUT1 signal will drive internal interrupt request 12.
- Bit 6: LCD — When this bit is a 1, the LCD's interrupt request signal will drive the internal interrupt request 12.
- Bits 5-0: IRQ5-IRQ0 — When the corresponding register bit is a 1, the external interrupt request pin (IRQ5-0) will drive internal interrupt request 12.

3.3.5.3.11 Internal Interrupt 13 Selection Register

During a system reset the bits in this register are set to 00h. This 8-bit register is located at I/O address EFB-Dh.



I/O Map Address



Writing a 1 into a bit in this register will result in the associated source driving internal interrupt request 13. If the interrupt controller is programmed to be level sensitive, more than one bit may be programmed to share internal interrupt request 13. If the interrupt controller is programmed to be edge sensitive, then only one bit should be written as a 1; otherwise conflicts may result. It is the responsibility of the software to avoid any conflicts.

- Bit 7: WD — WATCHDOG. When this bit is a 1, the WATCHDOG timer's interrupt request signal will drive the internal interrupt request 13.
- Bit 6: LCD — When this bit is a 1, the LCD's interrupt request signal will drive the internal interrupt request 13.
- Bits 5-0: IRQ5-IRQ0 — When the corresponding register bit is a 1, the external interrupt request pin (IRQ5-0) will drive internal interrupt request 13.

3.3.5.3.12 Internal Interrupt 14 Selection Register

During a system reset the bits in this register are set to 00h. This 8-bit register is located at I/O address EFBEh.



I/O Map Address

EFBEh

Writing a 1 into a bit in this register will result in the associated source driving internal interrupt request 14. If the interrupt controller is programmed to be level sensitive, more than one bit may be programmed to share internal interrupt request 14. If the interrupt controller is programmed to be edge sensitive, then only one bit should be written as a 1; otherwise conflicts may result. It is the responsibility of the software to avoid any conflicts.

The PCMCIA and ECP interrupt signals may also drive internal interrupt request 14. That configuration is detailed in the PCMCIA and ECP sections.

- Bit 7: TM1 — Timer OUT1. When this bit is a 1, the Timer OUT1 signal will drive internal interrupt request 14.
- Bit 6: 3-wire — When this bit is a 1, the 3-wire interface's interrupt request signal will drive internal interrupt request 14.
- Bits 5-0: IRQ5-IRQ0 — When the corresponding register bit is a 1, the external interrupt request pin (IRQ5-0) will drive internal interrupt request 14.

3.3.5.3.13 Internal Interrupt 15 Selection Register

During a system reset the bits in this register are set to 00h. This 8-bit register is located at I/O address EFBFh.



I/O Map Address

EFBFh

Writing a 1 into a bit in this register will result in the associated source driving internal interrupt request 15. If the interrupt controller is programmed to be level sensitive, more than one bit may be programmed to share internal interrupt request 15. If the interrupt controller is programmed to be edge sensitive, then only one bit should be written as a 1; otherwise conflicts may result. It is the responsibility of the software to avoid any conflicts.

The PCMCIA interrupt signal may also drive internal interrupt request 15. That configuration is detailed in the PCMCIA section.

- Bit 7: DMA — DMA Chaining Interrupt Request. When this bit is set to a 1 the DMA Controller's chaining interrupt request signal will drive the internal interrupt request 15.
- Bit 6: UART — When this bit is a 1, the UART's interrupt request signal will drive the internal interrupt request 15.
- Bits 5-0: IRQ5-IRQ0 — When the corresponding register bit is a 1, the external interrupt request pin (IRQ5-0) will drive internal interrupt request 15.

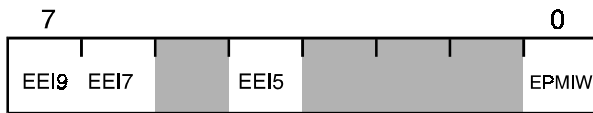
3.3.5.4 Miscellaneous (PCMCIA and Extended Capabilities Port (ECP)) Interrupt Selection Registers

Both the PCMCIA Controller and the ECP peripherals have logic to select which interrupt request they drive. Refer to the PCMCIA and ECP sections of the datasheet for more information.

To prevent conflicts between rising edge-triggered interrupt requests and active-low level-triggered interrupt requests, additional logic external to the PCMCIA Controller and ECP is used to enable or disable the interrupt signals generated by these peripherals. The following registers define how to enable each interrupt request from the PCMCIA controller and/or the ECP.

3.3.5.4.1 Miscellaneous Interrupt Selection Register 1

During a system reset the bits in this register are set to 00h. This 8-bit register is located at I/O address EFB0h.



I/O Map Address



Bit 7: EEI9 -- Enable ECP Interrupt request 9. When this bit is set to a 1 it will enable the ECP to generate an interrupt on the internal interrupt request 9.

Note: Software will also have to configure the ECP logic to drive its interrupt request onto the internal interrupt request 9.

Bit 6: EEI7 -- Enable ECP Interrupt request 7. When this bit is set to a 1 it will enable the ECP to generate an interrupt on the internal interrupt request 7.

Note: Software will also have to configure the ECP logic to drive its interrupt request onto the internal interrupt request 7.

Bit 5: Reserved.

Bit 4: EEI5 -- Enable ECP Interrupt request 5. When this bit is set to a 1 it will enable the ECP to generate an interrupt on the internal interrupt request 5.

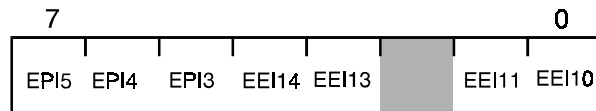
Note: Software will also have to configure the ECP logic to drive its interrupt request onto the internal interrupt request 5.

Bits 3-1: Reserved.

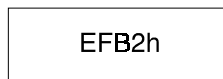
Bit 0: EPMIW -- Enable Power Management Interrupt Wakeup. When this bit is set to a 1, the Power Management logic will exit IDLE mode upon any CPU interrupt request. When this bit is a zero, a CPU interrupt request will have no effect on the Power Management logic. This bit also controls the DRAM parity error NMI interrupt enable.

3.3.5.4.2 Miscellaneous Interrupt Selection Register 2

During a system reset the bits in this register are set to 00h. This 8-bit register is located at I/O address EFB2h.



I/O Map Address



Bit 7: EPI5 -- Enable PCMCIA Interrupt request 5. When this bit is set to a 1 it will enable the PCMCIA controller to generate an interrupt on the internal interrupt request 5.

Note: Software will also have to configure the PCMCIA controller logic to drive its interrupt request onto the internal interrupt request 5.

Bit 6: EPI4 -- Enable PCMCIA Interrupt request 4. When this bit is set to a 1 it will enable the PCMCIA controller to generate an interrupt on the internal interrupt request 4.

Note: Software will also have to configure the PCMCIA controller logic to drive

its interrupt request onto the internal interrupt request 4.

Bit 5: EPI3 -- Enable PCMCIA Interrupt request 3. When this bit is set to a 1 it will enable the PCMCIA controller to generate an interrupt on the internal interrupt request 3.

Note: Software will also have to configure the PCMCIA controller logic to drive its interrupt request onto the internal interrupt request 3.

Bit 4: EEI14 -- Enable ECP Interrupt request 14. When this bit is set to a 1 it will enable the ECP to generate an interrupt on the internal interrupt request 14.

Note: Software will also have to configure the ECP logic to drive its interrupt request onto the internal interrupt request 14.

Bit 3: EEI13 -- Enable ECP Interrupt request 13. When this bit is set to a 1 it will enable the ECP to generate an interrupt on the internal interrupt request 13.

Note: Software will also have to configure the ECP logic to drive its interrupt request onto the internal interrupt request 13.

Bit 2: Reserved.

Bit 1: EEI11 -- Enable ECP Interrupt request 11. When this bit is set to a 1 it will enable the ECP to generate an interrupt on the internal interrupt request 11.

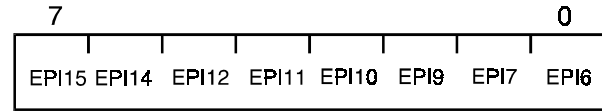
Note: Software will also have to configure the ECP logic to drive its interrupt request onto the internal interrupt request 11.

Bit 0: EEI10 -- Enable ECP Interrupt request 10. When this bit is set to a 1 it will enable the ECP to generate an interrupt on the internal interrupt request 10.

Note: Software will also have to configure the ECP logic to drive its interrupt request onto the internal interrupt request 10.

3.3.5.4.3 Miscellaneous Interrupt Selection Register 3

During a system reset the bits in this register are set to 00h. This 8-bit register is located at I/O address EFB8h.



I/O Map Address



Bit 7: EPI15 -- Enable PCMCIA Interrupt request 15. When this bit is set to a 1 it will enable the PCMCIA controller to generate an interrupt on the internal interrupt request 15.

Note: Software will also have to configure the PCMCIA controller logic to drive its interrupt request onto the internal interrupt request 15.

Bit 6: EPI14 -- Enable PCMCIA Interrupt request 14. When this bit is set to a 1 it will enable the PCMCIA controller to generate an interrupt on the internal interrupt request 14.

Note: Software will also have to configure the PCMCIA controller logic to drive its interrupt request onto the internal interrupt request 14.

Bit 5: EPI12 -- Enable PCMCIA Interrupt request 12. When this bit is set to a 1 it will enable the PCMCIA controller to generate an interrupt on the internal interrupt request 12.

Note: Software will also have to configure the PCMCIA controller logic to drive its interrupt request onto the internal interrupt request 12.

Bit 4: EPI11 -- Enable PCMCIA Interrupt request 11. When this bit is set to a 1 it will enable the PCMCIA controller to generate an interrupt on the internal interrupt request 11.

Note: Software will also have to configure the PCMCIA controller logic to drive

- its interrupt request onto the internal interrupt request 11.
- Bit 3: EPI10 -- Enable PCMCIA Interrupt request 10. When this bit is set to a 1 it will enable the PCMCIA controller to generate an interrupt on the internal interrupt request 10.
Note: Software will also have to configure the PCMCIA controller logic to drive its interrupt request onto the internal interrupt request 10.
- Bit 2: EPI9 -- Enable PCMCIA Interrupt request 9. When this bit is set to a 1 it will enable the PCMCIA controller to generate an interrupt on the internal interrupt request 9.
Note: Software will also have to configure the PCMCIA controller logic to drive its interrupt request onto the internal interrupt request 9.
- Bit 1: EPI7 -- Enable PCMCIA Interrupt request 7. When this bit is set to a 1 it will enable the PCMCIA controller to generate an interrupt on the internal interrupt request 7.
Note: Software will also have to configure the PCMCIA controller logic to drive its interrupt request onto the internal interrupt request 7.
- Bit 0: EPI6 -- Enable PCMCIA Interrupt request 6. When this bit is set to a 1 it will enable the PCMCIA controller to generate an interrupt on the internal interrupt request 6.
Note: Software will also have to configure the PCMCIA controller logic to drive its interrupt request onto the internal interrupt request 6.

3.3.5.5 Programming the PIC

Immediately after system reset, the PIC is not accessible via its I/O mapped registers until access is enabled to it via the Bus Interface Unit (BIU) Control Registers 1 and 2. To enable accesses to the PIC, a “1” must be written to bit 4 of BIU Control Register 1 (I/O address EF00h) and a “1” must also be written to bit 2 of BIU Control Register 2 (I/O address EF01h). Refer to the BIU section for more information.

Before operation, the PIC must be initialized by a sequence of four initialization Command Words (ICW1-ICW4) that establish the PIC's basic mode of operation. After initialization, the PIC's operation may be modified by the Operation Command Words (OCW1-OCW3). The OCWs may be written at any time during normal operation.

Note that the PICs internal to the NS486SXF operate somewhat differently than an external 8259A. The interrupt lines on the internal PICs are automatically inverted in level-trigger mode to create low-true level sensitive interrupts. This allows sharing of the interrupts externally by pulling up the IRQ pin with a resistor. In a standard 8259A the level triggered interrupts are high-true.

3.3.5.5.1 Initialization Programming

The initialization sequence begins by writing ICW1 to the PIC. For the Master PIC, this is accomplished by writing to I/O address 0020h with data bit 4 set to 1. For the Slave PIC, a write to I/O address 00A0h with data bit 4 set to 1, will be seen as an ICW1 write cycle. The writing of ICW1 starts the PIC initialization sequence and automatically establishes the following conditions:

- 1) The Interrupt Mask is reset for all levels. This will enable all interrupts.
- 2) Interrupt level 7 is assigned the lowest priority (7).
- 3) The slave mode address is set to 7.
- 4) Special Mask mode is reset.
- 5) Status Read pointer is set to IRR.
- 6) If bit 0 (IC4) = 0, all ICW4 functions are set to zero. **Note: You must program the ICW4 functions in the NS486SXF (IC4 = 1)!**

Initialization Command Word 1 (ICW1)

Master PIC (Write only, I/O address 0020h,
Bit 4 = 1)

Slave PIC (Write only, I/O address 00A0h,
Bit 4 = 1)

Bits 7-5: Reserved

Bit 4: Must be 1, otherwise the PIC will not recognize the write as a ICW1.

Bit 3: Level Trigger Interrupt Mode (LTIM) — When this bit is set to 1, the PIC will operate in level trigger mode. When this bit is 0, the PIC will operate in edge triggered mode.
 0 = the interrupt request inputs (IRQs) are edge-sensitive.
 1 = the interrupt request inputs (IRQs) are level-sensitive; i.e. requests are generated as long as IRQ remains low.

Bit 2: Must be 0.

Bit 1: Must be a 0. Otherwise the PIC may not operate properly.

Bit 0: Initialization Command 4 (IC4) — When set to 1, ICW4 must be written. When set to 0, ICW4 is not written and all ICW4 functions are set to zero.
 0 = ICW4 not written, all ICW4 functions are set to zero
 1 = ICW4 must be written.

ICW2 must be written during the write cycle that follows the ICW1 write cycle. ICW2 will be written by a write command to I/O address 0021h for the Master PIC and I/O address 00A1h for the Slave PIC immediately following the ICW1 write cycle.

Initialization Command Word 2 (ICW2)

Master PIC (Write only, I/O address 0021h, following ICW1 write cycle to PIC)

Slave PIC (Write only, I/O address 00A1h, following ICW 1 write to PIC)

Bits 7-3: Interrupt Vector Address (T7-T3) — These five bits are programmed with the vector address bits 7-3, which the PIC allows the processor to read during the interrupt acknowledge sequence.

Bits 2-0: Not Programmed — These three bits are the decoded address of the active interrupt level during an interrupt acknowledge sequence.

Initialization Command Word 3 (ICW3)

ICW3 is used to support cascading Interrupt Controllers.

Master PIC (Write only, I/O address 0021h, following ICW2 write to PIC)

Slave PIC (Write only, I/O address 00A1h, following ICW2 write to PIC)

For the Master PIC, ICW3 defines which internal interrupt request inputs have slaves attached to them:

Bits 7-0: 0 = No slave on this internal interrupt request input.
 1 = Slave present on this internal interrupt request input.

For the Slave PIC, ICW3 identifies the slave's address:

Bits 7-3: Reserved.
 Bits 2-0: 000 = Slave ID = 0.
 001 = Slave ID = 1.
 010 = Slave ID = 2.
 011 = Slave ID = 3.
 100 = Slave ID = 4.
 101 = Slave ID = 5.
 110 = Slave ID = 6.
 111 = Slave ID = 7

Initialization Command Word 4 (ICW4)

ICW4 is written only when ICW1 bit 0 (IC4) = 1 (**required in NS486SXF**). Otherwise, all of the ICW4 functions are set to 0. ICW4 is addressed by the write cycle following ICW3 if IC4 = 1. Writing ICW4 always completes the initialization sequence.

Master PIC (Write only, I/O address 0021h following ICW3 write cycle to PIC)

Slave PIC (Write only, I/O address 00A1h following ICW3 write cycle to PIC)

Bits 7-5: Must be 0.

Bit 4: Special Fully Nested Mode — (**Unsupported** see Section 3.3.5.9 on page 87)
 0 = Enable Normal Nested Mode
 1 = Special Fully Nested Mode

Bits 3-2: Buffer Mode — **Unsupported** mode, you should always program this bit to a 0. When bit 3 is set to 0, Non-Buffered Mode will be used and the Master-Slave bit (Bit 2) is a don't care. When bit 3 is set to 1, buffered mode will be used and bit 2 determines whether the PIC is operating as a master or a slave.

Bit 3	Bit 2	Function
0	X	Non-Buffered Mode
1	0	Buffered Mode (Slave)
1	1	Buffered Mode (Master)

Bit 1: Automatic End of Interrupt (AEOI)
 0 = Normal EOI
 1 = Automatic EOI

Bit 0: Must be set to 1. If this bit is set to 0, the PIC will operate in an incompatible 8085 mode.

3.3.5.2 Normal Operation Programming

The operation of the PICs may be modified during normal operation by the Operation Command Words (OCW1 -OCW3).

Every access of OCW1 is to the Interrupt Mask Register (IMR). Following initialization, all of the bits in the IMR are set to 0, indicating that all interrupts are enabled. To mask off any or all of the interrupts, a 1 may be written to the appropriate bit(s) of the IMR using OCW1.

OCW1 is addressed by any read or write cycle to I/O address 0021h for the Master PIC, and I/O address 00A1h for the Slave PIC. These accesses exclude any write cycles which are ICW2-4 cycles.

Operation Command Word 1 (OCW1)

Master PIC (Read/Write, 8-bit, I/O address 0021h)

Slave PIC (Read/Write, 8-bit, I/O address 00A1h)

Bit 7: Interrupt 7 Mask
 0 = Interrupt 7 is unmasked
 1 = Interrupt 7 is masked

Bit 6: Interrupt 6 Mask
 0 = Interrupt 6 is unmasked
 1 = Interrupt 6 is masked

Bit 5: Interrupt 5 Mask
 0 = Interrupt 5 is unmasked
 1 = Interrupt 5 is masked

Bit 4: Interrupt 4 Mask
 0 = Interrupt 4 is unmasked
 1 = Interrupt 4 is masked

Bit 3: Interrupt 3 Mask
 0 = Interrupt 3 is unmasked
 1 = Interrupt 3 is masked

Bit 2: Interrupt 2 Mask
 0 = Interrupt 2 is unmasked
 1 = Interrupt 2 is masked

Bit 1: Interrupt 1 Mask
 0 = Interrupt 1 is unmasked
 1 = Interrupt 1 is masked

Bit 0: Interrupt 0 Mask
 0 = Interrupt 0 is unmasked
 1 = Interrupt 0 is masked

OCW2 sets/controls the priority Rotation and the EOI modes (and combinations of the two). OCW2 is addressed by any write cycle to I/O address 0020h for

the Master PIC and I/O address 00A0h of the Slave PIC when data bits 3 and 4 are set to 0.

Operation Command Word 2 (OCW2)

Master PIC (Read/Write, 8-bit, I/O address 0020h, Bits 4-3 = 00)

Slave PIC (Read/Write, 8-bit, I/O address 00A0h, bits 4-3 = 00)

- Bit 7: Rotate Bit (R)
- Bit 6: Select Bit (SL)
- Bit 5: End of Interrupt Bit (EOI)

These three bits control the priority Rotation and the EOI modes (and combinations of the two) as shown in the following table:

Bit 7 R	Bit 6 SL	Bit 5 EOI	Function
0	0	1	Non-Specific EOI
0	1	1	Specific EOI
1	0	1	Rotate on Non-Specific EOI
1	0	0	Rotate in Automatic EOI mode (Set)
0	0	0	Rotate in Automatic EOI mode (Clear)
1	1	1	Rotate on Specific EOI Command (per L0, L1, L2)
1	1	0	Set Lowest Priority Command (per L0, L1, L2)
0	1	0	No Operation

- Bits 4-3: Must both be 0, otherwise the PIC will not recognize this access as an OCW2.
- Bits 2-0: Level 2-0 (L2, L1, L0) — These bits determine the interrupt level to which the Rotate on Specific Command or Set Lowest Priority Command is addressed. If any of the other options are selected by Bits 7-5, these three bits are don't cares.

Operation Command Word 3 (OCW3)

Master PIC (Read/Write, 8-bit, I/O 0020h, Bits 4-3 = 01)

Slave PIC (Read/Write, 8-bit, I/O 00A0h, Bits 4-3 = 01)

- Bit 7: Reserved: 0
- Bit 6: Enable Special Mask Mode (ESMM)
0 = Prevents writes to Bit 5, SMM
1 = Enables writes to Bit 5, SMM

OCW3 controls the special mask mode and status read pointer. It also affects the POLL command and provides a mechanism to read the In Service Register (ISR) or the Interrupt Request Register (IRR).

OCW3 is addressed by any write cycle to I/O address 0020h for the Master PIC (or I/O address 00A0h for the Slave PIC) when data bit 3 = 1 and bit 4 = 0.

Bit 5: Special Mask Mode (SMM) — (Refer to Priority Nesting later in this section for more information about Special Mask Mode)
 0 = Enable Normal Mask Mode
 1 = Enable Special Mask Mode

Bit 4: Must be set to 0, otherwise the PIC will not recognize this access as a OCW3.

Bit 3: Must be set to 1, otherwise the PIC will not recognize this access as an OCW3.

Bit 2: 0 = No Poll Command Requested
 1 = POLL Command
 Poll Command (P) — The POLL command is used in cases where the INTA sequence is not usable or is not practical. The POLL command, followed by a Read Poll Command, is similar to the INTA sequence. The POLL command is effected by issuing an OCW3 with this bit set to 1. A Read Command following the Poll Command will return a data byte with the following definition:

Bit 7: 1 = active Interrupt
 0 = no Interrupt pending

Bits 6-3: Should be treated as don't cares

Bits 2-0: decode of active (highest priority) requesting interrupt ID (0-7)

The read cycle following a POLL command is a “Read Poll” regardless of the setting of the Bits1-0 (RR, RIS).

Bit 1: Register Read (RR) — When this bit is set to 0, Bit 0 (RIS) is a “don't care” and no ISR or IRR read command will be generated. When this bit is set to 1, the RIS bit will determine if the ISR or IRR is read by the next read cycle to the PIC.
 0 = Disable ISR/IRR read commands
 1 = Enable ISR/IRR read commands

Bit 0: Request In-Service (RIS) — When this bit is set to 0 and bit 1 (RR) is set to 1, the next read cycle from the PIC will read the Interrupt Request Register (IRR). When this bit is set to 1 and bit 1 (RR) is set to 1, the next read cycle from the PIC will read the In-Service Register (ISR). When RR = 0, this bit is a ‘don't care’ and ISR and IRR will not be read.
 0 = The next read cycle to the PIC will access IRR, if RR = 1
 1 = The next read cycle to the PIC will access ISR, if RR = 1
 If the POLL bit is set, it will take precedence over the RR bit and this bit. The next read cycle will be a Read POLL cycle, not a read ISR or IRR cycle.

The Table below summarizes the PIC Registers and how they are accessed.

Table 3-3: Accessing the PIC Registers

PIC Initialization and Programming Registers	Control A0	Signals R/W	Data D4	Requirements D3	Other Requirements
ICW1	0	W	1	X	
ICW2	1	W	X	X	Next write cycle after ICW1
ICW3	1	W	X	X	Next write cycle after IOW2
ICW4	1	W	X	X	Next write cycle after ICW3
OCW1	1	R/W	X	X	Not initialization cycle
OCW2	0	R/W	0	0	Not initialization cycle
OCW3	0	R/W	0	1	Not Initialization cycle

PIC Interrupt Processing Registers	How Accessed
IRR	Read through OCW3
IMR	Read/Write through OCW1
ISR	Read through OCW3 Clear with EOI command by writing to OCW2

3.3.5.6 General Operation

In the most basic mode of operation, the PIC monitors the multiple interrupt requests coming from other system components (generally, I/O elements), and generates the single Interrupt Request (INTR) signal to the CPU if any of the incoming requests have a higher priority than the level (if any) currently being serviced.

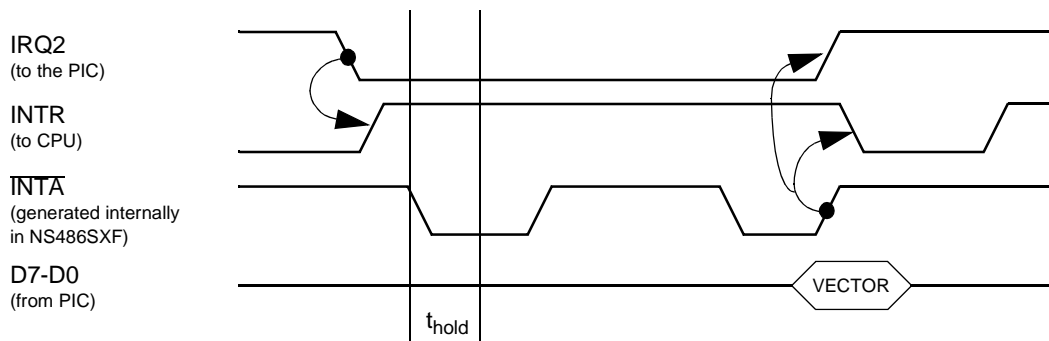
The CPU responds to INTR with Interrupt Acknowledge cycles or POLL sequence. The PIC responds

with the vector address of the requesting level with the highest priority.

3.3.5.7 Interrupt Sequence

One of the most powerful features of the PIC is its programmability. The interrupt addressing rapidly enables direct vectoring of software to a specific routine for each interrupt level.

Figure 3-27 Interrupt Sequence



The normal sequence of an interrupt is as follows:

- 1) One or more Interrupt Request (IR) inputs (IRQ0-IRQ7) become active to set the corresponding bit in the Interrupt Request Register (IRR).
- 2) The PIC evaluates the request(s) and generates the INTR signal to the CPU if any of them are of higher priority than the level (if any) currently being serviced.
- 3) The CPU responds to the INTR with an Interrupt Acknowledge cycle, represented by pulsing INTA low.
- 4) The PIC responds to the first Interrupt Acknowledge cycle by freezing its priority resolving logic, setting the appropriate bit in the ISR (in-Service Register), and resetting the corresponding bit in the IRR (interrupt Request Register) for the requesting level of highest priority.
- 5) During the first Interrupt Acknowledge cycle, the PIC does not drive any data onto the internal Data Bus. The CPU will initiate a second Interrupt Acknowledge cycle by pulsing INTA low a second time.
- 6) During this second and final Interrupt Acknowledge cycle, the PIC releases an 8-bit address

pointer onto the internal Data Bus. The 8-bit value is read by the CPU and becomes a pointer into the interrupt table containing the address of the interrupt service routine.

- 7) The active interrupt level remains 'in-Service' until its respective ISR bit is cleared by an EOI (End Of Interrupt). If the PIC is programmed for AEOI (Automatic EOI), then ISR is cleared on the trailing-edge of the final Interrupt Acknowledge cycle. Otherwise, the ISR remains set until an EOI instruction is issued by the software to end the interrupt sequence.

If no interrupt request (IRQ) is still active during the first command T-state of the first Interrupt Acknowledge cycle (i.e., the request was too short), then the PIC responds as if interrupt level 7 were active (i.e., interrupt 7's vector address will be driven onto the low byte of the local data bus during the second Interrupt Acknowledge Cycle). However, level 7 does not become "In Service" (i.e., ISR7 is not set).

3.3.5.8 End-of-interrupt (EOI) Modes

An “In-Service” interrupt level remains so until its respective ISR bit is reset. Typically, this occurs at the exit of the respective interrupt handler software routine. The active ISR is cleared by some form of EOI command, depending on the programmed EOI mode.

Normal EOI Mode

There are two basic forms of EOI: the Specific EOI and the Non Specific EOI. The Specific EOI Command is a form of OCW2. It clears the ISR of the specific level addressed by the three lowest bits of the OCW2. The Specific EOI must be used in cases where the PIC has been programmed for a mode of operation where a fully nested priority structure does not exist (see the Priority Nesting section).

The Non-Specific EOI command is another form of OCW2. It clears the ISR of the level with the highest priority that is “In Service”. The Non-Specific EOI may be used in cases where the PIC has been programmed for a mode of operation where a fully nested priority structure exists (see the Priority Nesting section).

Automatic EOI Mode

If the PIC is programmed for Automatic EOI operation (Bit 1 (AEOI) of ICW4 = 1), a Non-Specific EOI is automatically performed at the trailing-edge of the final INTA of the interrupt acknowledge sequence.

3.3.5.9 Priority Nesting

Normal Fully Nested Mode

The PIC is in normal fully nested mode following initialization (unless otherwise programmed by Special Fully Nested Diode bit of ICW4). When an interrupt is acknowledged, the vector address of the highest priority request is driven onto the internal data bus and its corresponding ISR bit is set to 1. The ISR bit remains set until cleared by an EOI (normally immediately before the software exits the interrupt service subroutine). While the level is “in Service” (ISR bit set to 1), all unmasked requests to this level or levels of lower priority are inhibited. Unmasked requests to levels of higher-priority are acknowledged.

Special Fully Nested Mode

This mode is used to enable a master PIC to extend its fully nested priority structure to include that of its slaves. **This mode is not supported in NS486SXF.**

Special Mask Mode

This mode is useful in the case where an application may require a certain interrupt routine to alter the priority structure dynamically during its execution. In the Special Mask Mode, when a mask bit is set, further requests to the corresponding level are inhibited, however requests to all other unmasked levels (higher and lower) are enabled. A disabled level is returned to normal operation by clearing its mask bit.

Note: In this mode, a masked ISR bit is not cleared by a non-specific EOI.

3.3.5.10 Priority

Fixed Priority

By default the interrupt levels (0-7) are assigned an ordered priority of 0-7 (0 = highest, 7 = lowest). The priority is ordered by setting a specific level as the lowest level; the other seven levels assume the seven higher priorities in ascending order (level 7 wraps to level 0). Following initialization, by default, the lowest priority (7) is assigned to interrupt level 7. The priority assignment may be changed at any time during operation via OCW2.

Rotating Priority

A specific rotation may be effected by changing the priority assignment during normal operation via OCW2 (Set Priority Command or Rotate on specific EOI).

Automatic Rotation may be effected via OCW2 (Rotate on Non-Specific EOI or setting the "Rotate in Automatic EOI" bit). During automatic rotation, the level being serviced (the highest priority by definition) is reassigned the lowest priority at EOI. In this way interrupt levels have equal access to service.

3.3.5.11 POLL Command

The POLL command is used in cases when the INTA sequence is not usable or is not practical. The POLL command, followed by a Read Poll Command, is similar to the INTA sequence. The POLL command is effected by issuing an OCW3 with POLL Command bit set to 1. The next Read Command returns a data byte with the following definition:

Bit 7:	1 = active Interrupt; 0 = no Interrupt pending
Bits 6-3:	should be treated as don't cares
Bits 2-0:	decode of active (highest priority) requesting interrupt ID (0-7)

The read cycle following a POLL command is "Read Poll" regardless of the setting of the Bits 1-0 (RR, RIS).

Reading Status

The Mask Register may be read via OCW1. Either IRR or ISR may be read via OCW3. If RIS = 1, ISR is read. If RIS = 0, IRR is read. If the POLL command has been issued, the next Read is a "Read Poll", regardless of the current setting or RIS.

3.3.6 The Real Time Clock/Calendar

The NS486SXF Real Time Clock/Calendar is a low power clock that provides a time-of-day clock and 100-year calendar with alarm features and battery operation. Time is kept in BCD or binary format. It includes 50 bytes of general purpose CMOS RAM and 3 maskable interrupt sources. It is backward compatible with both DS1287 and MC146818 RTC/Calendar chips.

3.3.6.1 Memory Map

The RTC contains 15 time/control registers and 50 bytes of general purpose battery backed static RAM. Refer to the table below.

3.3.6.2 Bus Interface

Immediately after system reset, the RTC is not accessible via its I/O mapped registers until access is enabled to it via the Bus Interface Unit (BIU) Control Registers 1 and 2. To enable accesses to the RTC, a

Table 3-4: RTC Memory Map

INDEX	FUNCTION	BCD FORMAT	BINARY FORMAT	COMMENTS
00	Seconds	00-59	00-3b	R/W
01	Seconds Alarm	00-59	00-3b	R/W
02	Minutes	00-59	00-3b	R/W
03	Minutes Alarm	00-59	00-3b	R/W
04	Hours	12hr = 01-12 (AM)	01-0c (AM)	R/W
		12hr = 81-92 (PM)	81-8c (PM)	R/W
		24hr = 00-23	00-17	R/W
05	Hours Alarm	12hr = 01-12 (AM)	01-0c (AM)	R/W
		12hr = 81-92 (PM)	81-8c (PM)	R/W
		24hr = 00-23	00-17	R/W
06	Day of Week	01-07	01-07 (Sun = 1)	R/W
07	Date of Month	01-31	01-1f	
08	Month	01-12	01-0c	
09	Year	00-99	00-63	
0A	Control Register A			bit 7 is Rd. only
0B	Control Register B			R/W
0C	Control Register C			all bits Rd. only
0D	Control Register D			all bits Rd. only
0E-3F	Battery-backed RAM (50 bytes)			R/W
40-7E	None			Unsupported
7F	Dates of Month Alarm		00 - 1f	R/W
80-FF	None			Unsupported

“1” must be written to bit 2 of BIU Control Register 1 (I/O address EF00h) and a “1” must also be written to bit 2 of BIU Control Register 2 (I/O address EF01h). Refer to the BIU section for more information.

The RTC function is mapped to I/O locations 0070h (index) and 0071h (data). This decode is done internal to the NS486SXF.

3.3.6.3 Time Generation

The Time Generation function divides the 32.768 KHz from the RTCX1-RTCX2 pins down to a one Hz signal. This divider chain is controlled by bits 6-4 of Control Register A. Bits 3-0 of Control Register A select one of fifteen taps from the divider chain to be used as a Periodic Interrupt. See Control Register A in the Control Register and Interrupt section for divider configurations and rate selections.

During divider reset (Bits 6-4 of Control Register A = 11x), the divider chain is reset to 0. An update will occur 500ms after the divider chain is activated into normal operational mode (Bits 6-4 of Control Register A = 010). The periodic flag will also become active 1/2 period of the programmed value when the divider chain is activated.

Figure 3-28 represents the external circuitry that is required for the NS486SXF RTC oscillator. The oscillator input may be driven from an external source. If this is desired, the input should be driven rail to rail and be approximately a 50% duty cycle. The oscillator output should be open in this case. The external capacitor values should be chosen to provide the manufacturer’s specified load capacitance for the crystal when combined with the parasitic capacitance of the trace, socket, and package, which can vary from 2 to 8 pF. The rule of thumb in choosing these capacitors is:

$$C_L = (C_1 * C_2)/(C_1 + C_2) + C_{parasitic}$$

$$C_2 > C_1$$

C1 can be trimmed to obtain the 32768.0 Hz

The following considerations should be made with respect to the oscillator circuitry:

1) The oscillator circuitry should be placed as close as possible to the NS486SXF device. This minimizes the effects of any parasitic resistance and capacitance from the board.

2) High frequency lines should not be run near the oscillator circuitry. This includes both parallel and crossing lines. If this is unavoidable, proper sheilding methods should be used.

3) The ground side of the two capacitors should go to a ground plane.

4) The parasitic resistance of the board combined with the parallel resistance (shown in Figure 3-28) should equal 20 M Ohm. This value is critical for the proper operation of the oscillator circuitry.

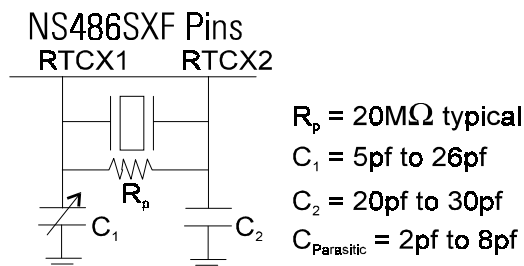


Figure 3-28 Oscillator External Circuitry

The start-up time of this oscillator may vary from two to seven seconds and is due to the high “Q” of the crystal. The parameters below describe the crystal to be used:

Parallel Resonant, tuning fork (N cut) or XY Bar

Q	>= 35K
Load Capacitance (C_L)	9 to 13 pF
Accuracy:	User Choice
Temperature Coefficient:	User Choice

3.3.6.3.1 Time Keeping

The time is kept in BCD or binary format. The format is determined by bit 2 of Control Register B (DM). Either 12 or 24 hour representation for the hours can be maintained as determined by bit 1 of Control Register B (24/12). Note that when changing the format the time registers must be re-initialized to the corresponding data format.

Daylight savings and leap year exceptions are handled by the time keeping function. When bit 0 of Control Register A (DSE) is a “1”, time advances from 1:59:59 AM to 3:00:00 on the first Sunday in April. On the last Sunday of October time changes from 1:59:59 to 1:00:00 when daylight savings is enabled. On leap year February is extended to 29 days.

The time is updated once per second. If a read of the timing registers coincides with an update, data read may not be valid. Also, writes to time registers during an update will have undefined results. The SET bit (bit 7 of Control Register B) when set to a “1”, allows the program to initialize the time registers by stopping an existing update and preventing a new one from occurring. While the SET bit may be used to read the time registers, it is not recommended that it be used. There are three methods for properly reading and writing the time to ensure correct results. These are:

Method 1

Access after detection of an Update Ended Interrupt. This signifies that an update has just completed and there are 999 ms remaining until another one will occur.

Method 2

Poll update-in-progress (bit 7 in Control Register A). The update will occur 244 μ s after the update-in-progress bit goes high. Therefore if a 0 is read, there will be a minimum of 244 μ s in which the time is guaranteed to remain stable.

Method 3

Use of a Periodic interrupt to determine if an update cycle is in progress. The periodic interrupt is first set to a desired period. The program can then use the periodic interrupt to signify that there is (Period of Periodic Interrupt/2 + 244 μ s) remaining until another update will occur.

The Alarm condition is also generated by the Time Keeping function. After each update, the seconds, minutes, and hours are compared with the seconds alarm, minutes alarm, and hours alarm (Note that in the normal operating mode, the Date of Month Alarm register is not used in the determination of an alarm condition. The Date of Month Alarm is discussed further in the following paragraph.). If equal, the Alarm lag is set in Control Register C. This will cause an interrupt condition (IRQZ = “0”) if the Alarm Interrupt Enable bit is set in Control Register B. If both bit 7 and bit 6 of an alarm byte (seconds alarm, minutes alarm, hours alarm) is a “1” then that byte is a “don’t care”.

The Date of Month Alarm register can only be enabled during lockout mode and is supported in binary

mode only. The Date of Month Alarm register is not used in normal mode alarm conditions. The compare of this register is disabled during non-lockout mode or when the Date of Month Alarm either equals 00h, or is greater than 1Fh. Since comparison of the Date of Month Alarm register is disabled when the RTC is in non-lockout mode, care must be taken when programming the alarm registers for alarm generation during lockout mode to avoid a premature alarm generation during non-lockout mode.

3.3.6.4 RAM

The RAM data is accessed at locations 0E-3F.

3.3.6.5 Power Management and System Lockout Features

The Power Management function provides power to the NS486SXF RTC. During system operation, power from the system is used. When system voltage (V_{DD}) falls below 1.2 times battery voltage (V_{BAT}), the RTC Power Management function switches the RTC cell to battery power. Figure 3-29 represents a typical battery configuration and Figure 3-30 represents typical battery current during battery backed mode.

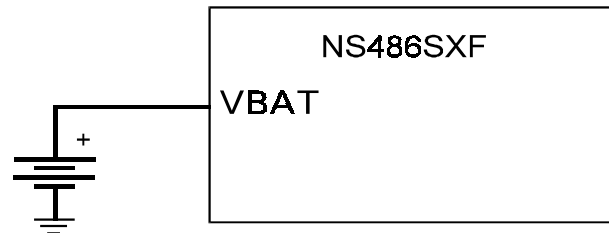


Figure 3-29 Typical Battery Configuration

Note: All of the components necessary for UL certification exist within the analog design of the RTC. They include the 1k-ohm series resistor and the power switching circuitry. The power switching circuitry satisfies the diode criteria.

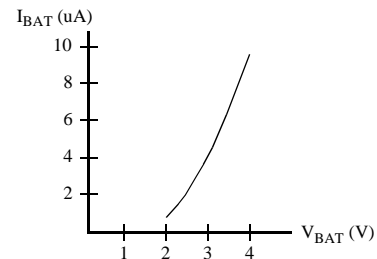


Figure 3-30 Typical Battery Current During Battery Backed Mode

As the RTC switches to battery power and from battery power, all inputs are “locked out” so that the in-

ternal registers cannot be modified. This lockout condition is asserted as long as PWGOOD is inactive (low). **Note that in order to come out of lockout, Vdd must be at least 1.4V higher than Vbat, and Vbat must be at least 2.4V.**

Please ensure that the Vdd ramp-up and down times are met as per Table 7-1 on page 246.

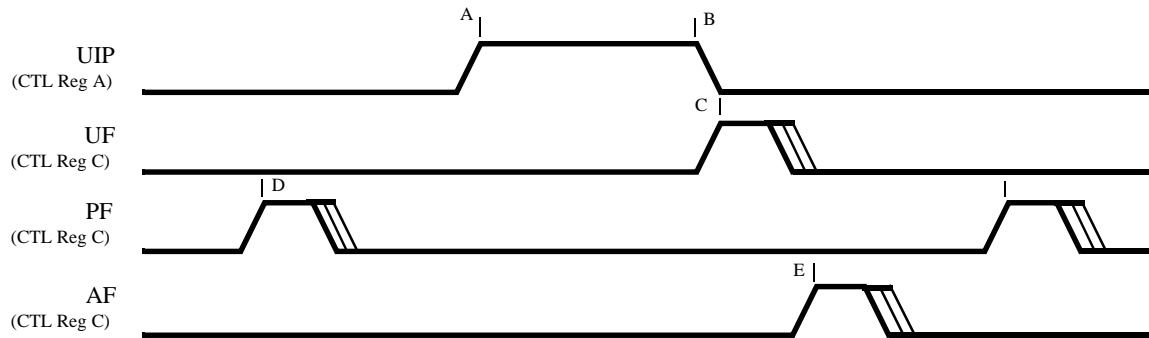
When power is applied to the RTC in the NS486SXF, the RTC Oscillator is operational with the following exceptions:

- 1) The Divider Chain Control (bits 6-4 in Control Register A) is in Oscillator Disabled modes (“000”, and “001”). This provides a means for the user to “shut-down” the oscillator and reduce the power consumption of the RTC cell. The RAM remains functional when the oscillator is disabled

3.3.6.6 Interrupt Handling

The Periodic, Alarm, and Update ended Interrupts are generated (internal_IRQS is driven active) when the respective enable bits in Control Register B are set and an interrupt condition occurs. A read from Register C clears the active interrupt. If a second interrupt condition occurs (other than that which caused the interrupt) during a read from Register C, IRQ will remain active (low). Thus, it is recommended that when multiple interrupts are enabled, the interrupt service routine continues to read (and service interrupts) until bit 7 of Control Register C (IRQ Flag) returns to a “1”. Note that if an interrupt is not serviced before a second condition of the same interrupt occurs, the second interrupt event will be lost.

Figure 3-31 Interrupt Status Timing



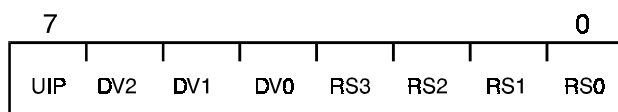
A-B,C	Update In Progress Bit High before Update occurs = 244us
D-C	Periodic interrupt to update = Period (periodic int)/2 + 244us
C-E	Update to Alarm Interrupt = 30.5 us
UIP	Update In Progress Status Bit
UF	Update Ended Flag (Update Ended Interrupt if enabled)
PF	Periodic Flag (Periodic Interrupt if enabled)
AF	Alarm Flag (Alarm Interrupt if enabled)

Flags (and IRQ) are reset at the conclusion of Control Register C read or by RESETC low.

3.3.6.7 Control Registers

The Control Registers/Interrupt Generation contains four registers which can be accessed at any time during non-battery backed operation. Control Registers at address A, B, C, and D are described below.

3.3.6.7.1 Control Register A



Bit 7: **UIP** — Update in Progress (Read only)
 1 = Indicates that the timing registers will be updated within 244us.
 0 = Indicates that an update will not occur before 244us. This bit will read 0 when bit 7 of Control Register B (SET) is a 1.
 Reset has no effect on this bit.

Bits 6-4: **DV[2:0]** — Divider Chain Control (Read/Write)
 These bits control the configuration of the divider chain in the Timing Generation function. Reset has no effect on these bits.

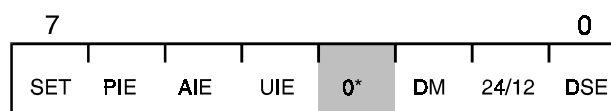
DV[2:0]	Divider Chain Configuration
0 0 0	Oscillator Disabled
0 0 1	Oscillator Disabled
0 1 0	Normal Operation
0 1 1	TEST
1 0 0	TEST
1 0 1	TEST
1 1 0	Divider Chain RESET
1 1 1	Divider Chain RESET

Bits 3-0: **RS[3:0]** — Periodic Interrupt Rate Select (Read/Write)

These bits control the rate of the periodic interrupt. Reset has no effect on these bits.

RS[3:0]	Periodic Interrupt Rate
0 0 0 0	none
0 0 0 1	3.90625 ms
0 0 1 0	7.8125 ms
0 0 1 1	122.070 μs
0 1 0 0	244.141 μs
0 1 0 1	488.281 μs
0 1 1 0	976.562 μs
0 1 1 1	1.953125 ms
1 0 0 0	3.90625 ms
1 0 0 1	7.8125 ms
1 0 1 0	15.625 ms
1 0 1 1	31.25 ms
1 1 0 0	62.5 ms
1 1 0 1	125 ms
1 1 1 0	250 ms
1 1 1 1	500 ms

3.3.6.7.2 Control Register B

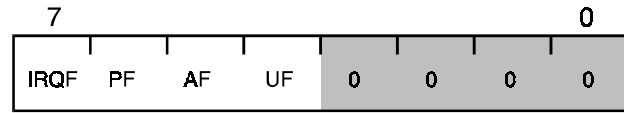


Bit 7: **SET** — Set mode (Read/Write)
 1 = The timing updates are inhibited and the time registers are “frozen.”
 0 = The timing updates occur normally.
 Reset has no effect on this bit.

Bit 6: **PIE** — Periodic Interrupt Enable (Read/Write)
 1 = Enables generation of the periodic interrupt. Bits 3-0 of Control Register A determine the rate of the Periodic interrupt.
 0 = Disables generation of the Periodic interrupt.
 Reset forces this bit to a “0”.

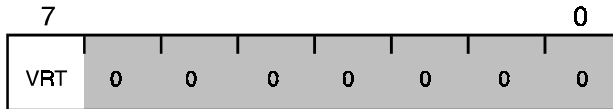
- Bit 5: AIE — Alarm Interrupt Enable (Read/Write)
1 = Enables generation of the alarm interrupt. The alarm interrupt will be generated immediately after a time update in which the Seconds, Minutes, and Hours time compares with their respective alarm counterparts.
0 = Disables generation of the alarm interrupt.
Reset has no effect on this bit.
- Bit 4: UIE — Update Ended Interrupt Enable (Read/Write)
1 = Enables generation of the update ended interrupt. This interrupt is generated at the time an update occurs.
0 = Disables generation of the update ended interrupt.
Reset forces this bit to a “0”.
- Bit 3: 0* — This bit is defined as “Square Wave Enable” by the MC146818 and is not supported by the RTC Cell. This bit will always be read as a “0”.
- Bit 2: DM — Data Mode (Read/Write)
1 = Enables BINARY format.
0 = Enables BCD format.
Reset has no effect on this bit.
- Bit 1: 24/12 — 24 or 12 Hour Mode (Read/Write)
1 = Enables 24 hour format.
0 = Enables 12 hour format.
Reset has no effect on this bit.
- Bit 0: DSE — Daylight Savings Enable (Read/Write)
1 = Enables daylight savings. These two conditions are as follows:
Daylight Savings Spring: Time advances from 1:59:59 to 3:00:00 on the first Sunday in April.
Daylight Savings Fall: Time advances from 1:59:59 to 1:00:00 on the last Sunday in October.
0 = Disables the daylight savings feature.
Reset has no effect on this bit.

3.3.6.7.3 Control Register C



- Bit 7: IRQF — Interrupt Request Flag (Read Only)
1 = When PIE & PF + AIE & AF + UIE & UF = 1. The IRQF bit mirrors the value on internal_IRQ8.
- Bit 6: PF — Periodic Interrupt Flag (Read Only)
1 = When a transition occurs on the selected tap of the divider chain. This bit is reset to a “0” at the conclusion of a read from this register.
Reset forces this bit to a “0”.
- Bit 5: AF — Alarm Interrupt Flag (Read Only)
1 = When an alarm condition is detected. This bit is reset to a “0” at the conclusion of a read from this register.
Reset has no effect on this bit.
- Bit 4: UF — Update Ended Interrupt Flag (Read Only)
1 = When the time registers are updated. This bit is reset to a “0” at the conclusion of a read from this register.
Reset forces this bit to a “0”.
- Bits 3-0: Reserved “0”

3.3.6.7.4 Control Register D



Bit 7: VRT — Valid RAM and Time (Read Only)
 When a “1”, this bit indicates that the contents of the RTC are valid. When a “0”, this bit indicates that the battery source is low and that the RTC data and RAM data are questionable. This bit is set to a “1” at the conclusion of a read from this register.
 Note: Once the RTC_INDEX register has been set to 0Dh, the next read, even a read from the RTC_INDEX, will cause the VRT bit to go to a “1” (valid). In order to ensure a correct read of the VRT bit, you must read the RTC_DATA immediately after writing the RTC_INDEX with 0Dh.

Bits 6-0: Reserved “0”

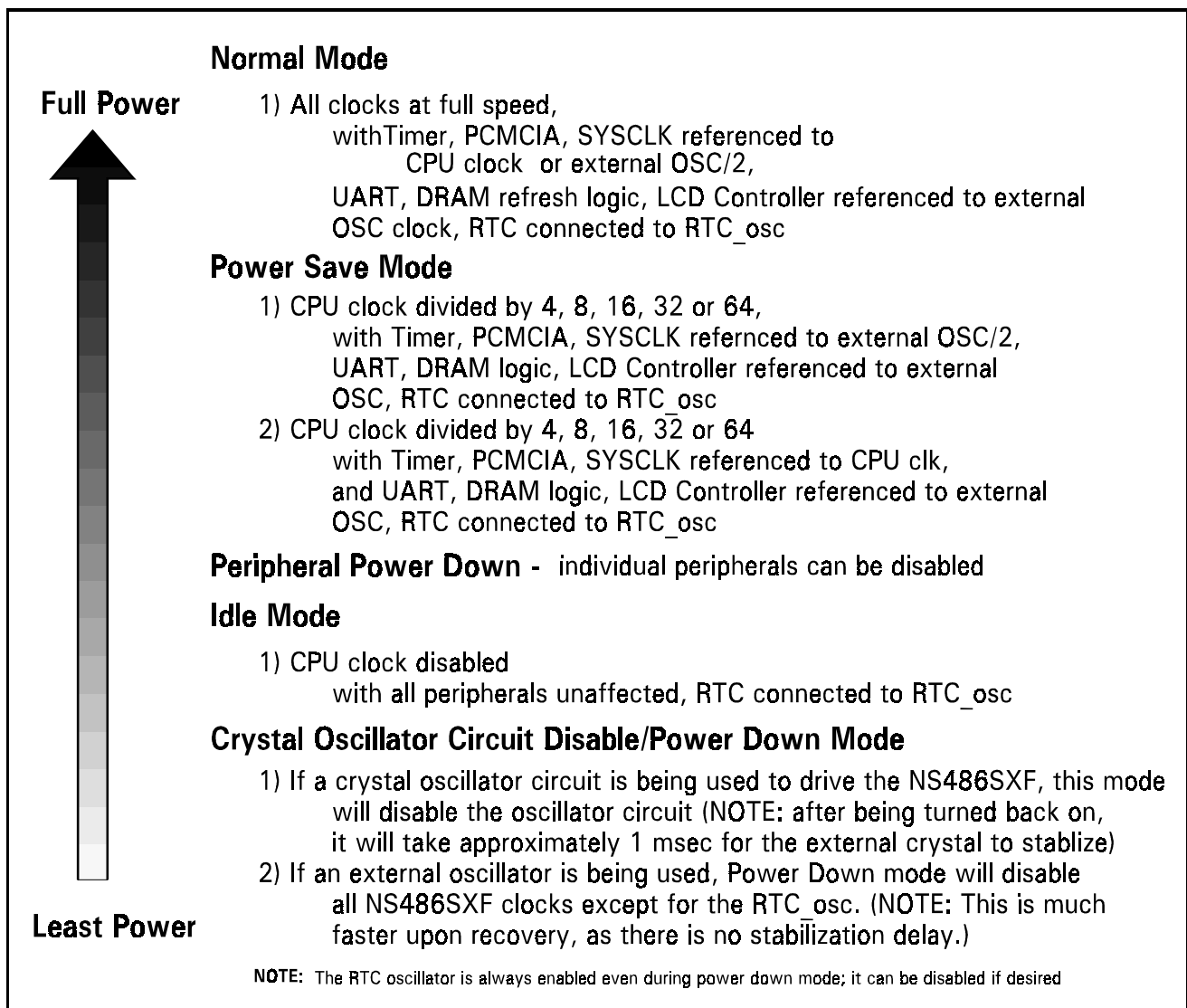
Note: The bits in Control Registers that are not affected by power-good reset can (and will) power up with random results. To guarantee correct operation of the RTC, these bits should be set to known, desired, values the very first time that the RTC is programmed. After that, a valid battery voltage should keep the bits as programmed.

3.3.7 Power Management Features

The NS486SXF power management structure includes a number of power saving mechanisms that can be combined to achieve comprehensive power savings under a variety of system conditions. First of all, the core processor power consumption can be controlled by varying the processor/system clock frequency. The internal CPU clock can be divided by 4, 8, 16, 32 or 64. In addition, in idle mode, the internal processor clock will be disabled. Finally, if a crystal oscillator circuit or external oscillator is being used, it can be disabled. For maximum power savings, all internal clocks can be disabled (even the real-time clock oscillator).

The timer and the PCMCIA interface clock references can be switched between a fixed frequency (external oscillator/2) and the CPU clock (referencing to the CPU clock saves power when running the CPU off of a reduced (divided) clock). The clocks for all sequential on-board peripherals can be individually or globally controlled. By setting bits in the Power Management Control Registers 2 and 3, the internal clocks to the DMA controller, the ECP port, the three-wire interface, the timer, the LCD controller, the DRAM controller, the PCMCIA controller and the UART can be disabled. In addition, the NS486SXF can be programmed to use CMOS level I/Os or TTL level I/O settings. Finally, the external SYSCLK can be disabled via bit 2 of Power Management Control Register 3.

Figure 3-32 Power Management Options



The following table shows what peripherals are connected to which clocks and how those clocks can be disabled/enabled:

<p>CPU</p> <ol style="list-style-type: none"> 1) Uses <code>cpu_clk</code> (Full speed clock = <code>OSC_CLK/2</code>) 2) <code>cpu_clk</code> can be divided by 4, 8, 16, 32 or 64 3) Idle mode: clock disabled <p>Timer, PCMCIA, SYSCLK</p> <ol style="list-style-type: none"> 1) Uses <code>cpu_clk</code> (full speed or divided by 4, 8, 16, 32 or 64) 2) Or can use external <code>OSC_CLK/2</code> (when <code>cpu_clk</code> is divided) 3) Can be individually disabled 	<p>Clock Definitions:</p> <p>OSC_CLK = Oscillator speed</p> <p>cpu_clk = <code>OSC_CLK/2</code> (full speed) or (<code>OSC_CLK/2</code>) divided by 4, 8, 16, 32 or 64</p>
<p>UART, DRAM refresh logic, LCD Controller</p> <ol style="list-style-type: none"> 1) Connected to OSC 2) Can be individually disabled 	<p>Bus Interface Unit</p> <ol style="list-style-type: none"> 1) Uses <code>cpu_clk</code> 2) Can NOT be individually disabled
<p>ECP</p> <ol style="list-style-type: none"> 1) Connected to <code>OSC_CLK/2</code> 2) Can be individually disabled 	
<p>DMA Controller, Three-wire Serial Interface</p> <ol style="list-style-type: none"> 1) Uses <code>cpu_clk</code> (full speed or divided) 2) Can be individually disabled 	
<p>DRAM Controller</p> <ol style="list-style-type: none"> 1) Must use <code>OSC_CLK</code> for DRAM refresh cycles 2) Sequencer can selectably use <code>cpu_clk</code> or $2 * \text{cpu_clk}$ 3) For state machine logic, must use <code>cpu_clk</code> 4) Can be individually disabled. 	
<p>Real-Time Clock</p> <ol style="list-style-type: none"> 1) Uses <code>RTC_osc</code> - typically always enabled, but it can be disabled through the RTC interface 	
<p>Global Peripheral Clock Disable/Enable</p> <ol style="list-style-type: none"> 1) Controls DMA Controller, ECP, Three-wire Interface, and UART 	

3.3.7.1 Power Management Modes

Power saving features include the following:

Idle Mode

In Idle Mode the internal clock to the NS486SXF CPU will be disabled. All enabled peripheral blocks will continue to operate. Any interrupt or reset will re-enable the internal clock to the CPU.

NOTE: When the CPU is in Idle Mode, the instruction cache cannot snoop. Normally, the cache will snoop the addresses to see if a cache address is being updated. If so, it flushes the cache. Therefore, it is the user's responsibility to take the appropriate action when the CPU is idled.

Also, when the CPU is in Idle Mode, the BIU is designed to mimic the CPU during DMA interchanges between memory and peripherals. By responding to DRQs and generating $\overline{\text{DACKs}}$, $\overline{\text{HOLDs}}$ and $\overline{\text{HOLDAs}}$ signals as required, the BIU eliminates the need to re-

activate the CPU during such transfers as screen updates from memory to the LCD controller. This gives the designer added flexibility in conserving power while maintaining basic system functions.

Power-save Mode

The Power-save Mode reduces the internal CPU/system clock's frequency by dividing the internal CPU clock by 4, 8, 16, 32 or 64 (Refer to Power Management Register 1 for more information) The internal clocks for the UART, DRAM refresh logic, LCD Controller and RTC will be unaffected in this mode. The Timer, PCMCIA and SYSCLK all have selectable clock sources between a fixed frequency, which is the external oscillator/2 and `cpu_clk`. Only when a `cpu_clk` source is selected will these clocks be affected by Power-save mode.

Crystal Oscillator Circuit Disable

This function disables the feedback output of the crystal oscillator circuit (i.e. forces OSCX2 low). Normally, the feedback output is used to provide a high-gain feedback to an external crystal to start, stabilize, and maintain a reference oscillation from the crystal. If the feedback is disabled the oscillation will stop. After the feedback output is re-enabled, it takes approximately 1 msec for the external crystal to start and stabilize. On-chip, there is a low-pass filter and counter to insure that none of the start-up and stabilize oscillations are allowed to pass into the rest of the chip. If an external TTL or CMOS oscillator is used then the feedback output can be disabled to save power. Also, the low-pass filter and counter can be bypassed by setting bit 7 of Power Management Configuration Register 4. This latter action may be useful when an external TTL or CMOS oscillator is used.

Power Down Mode

In Power Down Mode all of the internal NS486SXF clocks except the RTC oscillator will be disabled. If a crystal is used to generate the CPU clock, the CPU Oscillator Circuit Disable feature may be used to turn off the clock instead of this mode. If an external oscillator drives CPUX1, then this mode should be used to turn off the NS486SXF internal clocks.

Note: It is very important that power be applied to and removed from the LCD display in proper sequence, otherwise damage can result. To prevent damage to the LCD panels, the external DC power supplied to the LCD Display (V_{EE}) must be disabled before the LCD Controller's clock is disabled.

The power-up sequence is as follows:

- 1) Configure the LCD control registers.
- 2) Apply V_{DD} (5V or 3V) to the display.
- 3) Enable the LCD clocks from the power management registers - this must be done within 20 msec. of applying V_{DD} .
- 4) Enable the LCD controller.
- 5) Within 20 msec. max after applying the LCD clocks, apply V_{EE} (22V/-26V) to the display.

The power-down sequence is as follows:

- 1) Remove V_{EE} from the display.
- 2) Disable the LCD controller.
- 3) Within 20 msec. of removing V_{EE} , disable the LCD clocks.
- 4) Within 20 msec. of removing the LCD clocks, remove V_{DD} from the display.

Never disable the LCD clocks when the LCD is enabled.

3.3.7.2 Enabling/Disabling Individual Peripheral Clocks

The internal clocks for various internal peripherals may be individually enabled/disabled via bits of Power Management Registers 2 and 3. A peripheral's internal clock should only be disabled if that internal peripheral is not to be used.

3.3.7.3 Global Enable/Disable of Peripheral Clocks

When bit 7 of Power Management Register 2 is set to a one, the internal clocks to the DMA Controller, ECP, Three-Wire Interface, and UART logic will all be disabled. When that bit is a zero, the individual peripheral clock enable/disable bits will determine if the individual peripheral clocks are enabled or not. The DRAM and LCD Controllers, PCMCIA, BIU and Timer are not affected by global clock enabling/disabling.

3.3.7.4 Power Supply Level Configuration

The NS486SXF I/Os are power supply-level configurable. The power management block controls voltage sensing and setting for I/O supply-level configuration.

Currently, the Power Management block has the capability to set the operating voltage through firmware only (Bit 5 of Power Management Register 4).

Five volt mode of the NS486SXF provides TTL compatible signalling, both input threshold and output high voltage. In 3.3V mode, some of the output drivers are strengthened to guarantee 3.3V TTL/CMOS output levels and the input buffers are reconfigured for optimum 3.3V performance.

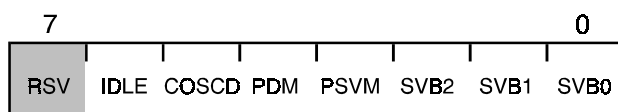
3.3.7.5 Power Management Configuration Registers

Note: In the following descriptions, reference is made to “all resets.” There are three possible ways that a reset is generated in the NS486SXF. PWRGOOD is the reset input to the chip from the external world, the other possibilities are the Watch-Dog Timer generated reset, and the shutdown reset from the CPU.

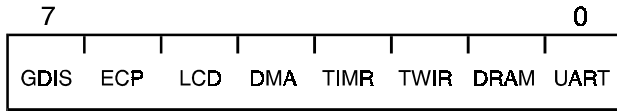
3.3.7.5.1 Power Management Register 1

- Bit 7: Reserved
- Bit 6: IDLE — Idle Mode selection bit
1 = Sets the chip in Idle Mode (cpu_clk disabled). All resets and interrupts force this bit to a “0”
- Bit 5: COSCD — CPU Oscillator Disable bit (used with crystal osc.). **Note:** Even if you are using an external oscillator, asserting this bit will cause all clocking to cease, unless you have previously set the CANOSC bit in PMR4. If you do set CANOSC, then asserting COSCD can save additional power as the crystal osc. feedback gate will be disabled.
1 = Disables CPU oscillator. All resets and interrupts force this bit to a “0”
- Bit 4: PDM — Power-down Mode selection bit (used with external OSC).
1 = Sets the chip to Power-down Mode. All resets and interrupts force this bit to a “0”
- Bit 3: PSVM — Power-save Mode selection bit (divides cpu_clk)
1 = Sets the chip to the Power-save Mode. All resets force this bit to a “0”
- Bits 2-0: SVB[2:0] — Power-save Mode clock division. All resets force these bits to a “0”n.

SVB[2]	SVB[1]	SVB[0]	Divide By
0	0	0	1
0	0	1	4
0	1	0	8
0	1	1	16
1	0	0	32
1	0	1	64
1	1	X	reserved



3.3.7.5.2 Power Management Register 2



I/O Map Address

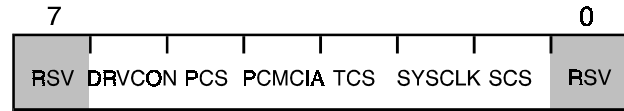
EF91h

Access

R/W

- Bit 7: GDIS — Global peripheral clock disabling selection bit
1 = Causes Global peripheral clock disabling. All resets force this bit to a “0”.
- Bit 6: ECP — ECP clock disable selection bit
1 = Disables the ECP clock. All resets force this bit to a “0”.
- Bit 5: LCD — LCD clock disable selection bit
1 = Disables the LCD clock. All resets force this bit to a “0”. The LCD Controller is not affected by global clock enabling/disabling (GDIS, bit 7).
- Bit 4: DMA — DMA clock disabling selection bit
1 = Disables the DMA clock. All resets force this bit to a “0”.
- Bit 3: TIMR — Timer block clock disabling selection bit
1 = Disables the Timer Clock. All resets force this bit to a “0”. The timer is not affected by global clock enabling/disabling (GDIS, bit 7).
- Bit 2: TWIR — Three-wire block clock disabling selection bit
1 = Disables the Three-wire Clock. All resets force this bit to a “0”.
- Bit 1: DRAM — DRAM block clock disabling selection bit
1 = Disables the DRAM Clock. All resets force this bit to a “0”. The DRAM controller is not affected by global clock enabling/disabling (GDIS, bit 7).
- Bit 0: UART — UART block clock disabling bit
1 = Disables the UART Clock. All resets force this bit to a “0”.

3.3.7.5.3 Power Management Register 3



I/O Map Address

EF92h

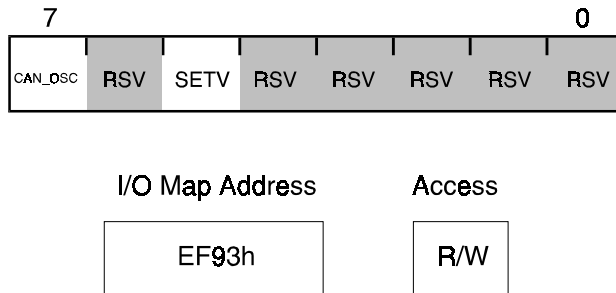
Access

R/W

- Bit 7: Reserved
- Bit 6: DRVCON — External Driver Configuration for system bus and DRAM interface I/Os. This bit only has an affect when the interface SETV bit is set to a one.
1 = Guarantees CMOS level output voltages/drive
0 = Guarantees TTL level output voltage/drive (low noise I/O configuration)
All resets force this bit to a “0”.
- Bit 5: PCS — PCMCIA Clock reference Selection bit
1 = Cpu_clk clock reference (affected by Power Save Mode)
0 = Standard clock reference (not affected by Power Save Mode)
All resets force this bit to a “0”.
- Bit 4: PCMCIA — PCMCIA block clock disabling selection bit
1 = Disables the PCMCIA clock. All resets force this bit to a “0”.
- Bit 3: TCS — Timer Clock reference Selection bit
1 = Cpu_clk clock reference (affected by Power Save Mode)
0 = Standard clock reference (not affected by Power Save Mode)
All resets force this bit to a “0”.
- Bit 2: SYSCLK — SYSCLK clock disabling selection bit
1 = Disables the SYSCLK. Only PWR-GOOD reset forces this bit to a “0”
- Bit 1: SCS — SYSCLK reference Selection bit
1 = Cpu_clk clock reference (affected by Power Save Mode)
0 = Standard clock reference (not affected by Power Save Mode)

Only PWRGOOD reset forces this bit to a “0”
 Bit 0: Reserved. Do not attempt to write a logic 1 to this bit.

3.3.7.5.4 Power Management Register 4



Bit 7: CAN_OSC--External clock source description bit
 1 = CMOS or TTL oscillator
 0 = Crystal oscillator
 Only PWRGOOD reset forces this bit to a “0”
 Bit 6: Reserved
 Bit 5: SETV--Software setting of Operating Voltage
 1 = 5V Operating Voltage
 0 = 3.3V Operating Voltage(default)
 Only PWRGOOD reset forces this bit to a “0”
 Bits 4-0: Reserved

3.3.7.6 Crystal Oscillator Circuitry

The NS486SXF Master Oscillator is designed for fundamental, parallel resonant crystals. A typical connection diagram is provided below:

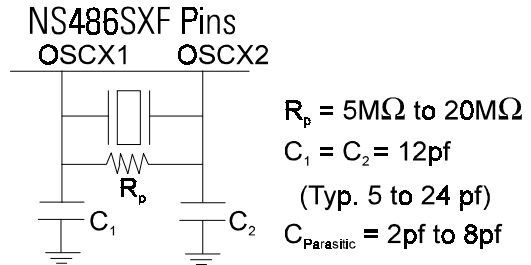


Figure 3-33 Oscillator External Circuitry

Refer to Table 7-1 on page 246 for the specified Vdd ramp rates and Table 7-2 on page 246 for the PWRGOOD timing compatible with the Crystal Oscillator circuitry.

3.3.8 NS486SXF System Bus

The NS486SXF system bus provides the interface to off-chip peripherals and memory. It offers an ISA compatible interface and thus is capable of directly interfacing to many ISA peripheral control devices. The Bus Interface Unit (BIU) is responsible for interfacing the CPU Core to the rest of the world. The BIU generates all of the access signals for both internal and external peripherals and memory. Depending upon whether the access is to internal peripherals, external peripherals or external memory, the BIU generates the timing and control signals to access those resources. The BIU is designed to support a glueless interface to many ISA-type peripherals control devices.

The BIU and DRAM Controller simultaneously monitor the CPU bus. If the access is to DRAM, the BIU stops its attempt to control the bus, and the DRAM Controller takes control. If the access is to internal peripherals or off-chip resources, the DRAM Controller relinquishes its attempt at control and the BIU takes over. This design approach helps speed up access to memory and I/O.

The BIU handles the process of generating valid chip select signals. Internal logic determines valid addresses and asserts chip select signals. There are nine external chip select pins on the NS486SXF ($\overline{CS0}$ - $\overline{CS8}$). ChipSelect 0 should generally be used for the system boot ROM. It always goes active for any access in the upper 64Kbytes of CPU memory address (FFFF0000h to FFFFFFFFh). The remaining chip selects ($\overline{CS1}$ - $\overline{CS8}$) are defined by type of access (I/O or Memory) and by address range. Before output to the external signal pin, chip selects can be combined. This allows a single chip select pin to be used to select external devices that might include multiple I/O controllers.

See Section 4.0 on page 171 for more information on chip selects.

For lowest power consumption, the BIU does not normally indicate internal bus cycles to the external ISA-like bus pins. However, for debugging purposes, the NS486SXF can be programmed to generate external bus cycles at the same time as internal bus cycles. This gives the designer access to internally accessed information.

Access to internal peripherals is performed in three CPU clocks cycles. The fastest access to off-chip resources is also three CPU clock cycles. When accessing off-chip memory and I/O, wait state generation is accomplished through a combination of NS486SXF chip select logic and off-chip peripheral control signals.

See The System Bus on page 171 for more information on the BIU.

3.4 Other On-chip Peripherals

In addition to those peripherals and system control elements needed for an O/S kernel's System Service Elements, the single-chip NS486SXF also includes a number of I/O controllers and resources that make implementing a complete embedded system possible with a minimal chip count. These include an IEEE 1284 Extended Capabilities Port, a serial UART port, an LCD controller, a PCMCIA interface and a MICROWIRE or Access.bus synchronous serial bus interface. For greater design flexibility, when I/O controllers are not being used, their I/O pins can be reconfigured as general purpose bidirectional I/Os. (Refer to Section 3.4.1 "The Reconfigurable I/O Pins".)

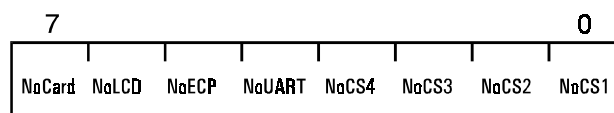
3.4.1 The Reconfigurable I/O Pins

If the NS486SXF UART, ECP Parallel Port, LCD or PCMCIA peripheral functions are not being used, the I/O pins and chip select pins associated with them can be reconfigured as general purpose bidirectional I/O pins. Up to 29 pins, on a pin-by-pin basis can be reconfigured for this purpose. This capability makes the NS486SXF extremely versatile and capable of supporting many different end product configurations with a single NS486SXF device.

Reconfigurable I/O lines are controlled by two registers: the Reconfigurable I/O Control Register at I/O map address EFC0h, and the 32-bit Data Direction Register at I/O map addresses EFC4h-EFC7h.

The RIO Control Register sets the function of the reconfigurable I/O pins available on the NS486SXF. If the PCMCIA, LCD, IEEE 1284 ECP Parallel Port or UART functions are not being used, then their corresponding pins can be altered from their standard function to that of a general purpose inputs and/or outputs, on a pin-by-pin basis using the General Purpose I/O registers. In addition, $\overline{CS}[1]$ to $\overline{CS}[4]$, can be individually reconfigured as General Purpose I/O pins. Upon system reset, the RIO is configured to use the standard functions rather than the RIO mode.

3.4.1.1 Reconfigurable I/O Control Register



	I/O Map Address	Access
	EFC0h	R/W
Bit 7:	NoCard	0 = Normal PCMCIA functionality. 1 = Change the PCMCIA output pin functionality to correspond to General Purpose I/O bits 31-24.
Bit 6:	NoLCD	0 = Normal LCD functionality. 1 = Change the LCD pin functionality to correspond to General Purpose I/O bits 22-16.
Bit 5:	NoECP	0 = Normal ECP functionality. 1 = Change the ECP data pin functionality to correspond to General Purpose I/O bits 15-8.
Bit 4:	NoUART	0 = Normal UART functionality. 1 = Change the UART Rx and UCLK pin functionality to correspond to General Purpose I/O bits 5 and 4.
Bit 3:	NoCS4	0 = Normal Chip Select 4, $\overline{CS}[4]$, functionality. 1 = Change the $\overline{CS}[4]$ pin functionality to correspond to General Purpose I/O bit 3.
Bit 2:	NoCS3	0 = Normal Chip Select 3, $\overline{CS}[3]$, functionality. 1 = Change the $\overline{CS}[3]$ pin functionality to correspond to General Purpose I/O bit 2.
Bit 1:	NoCS2	

0 = Normal Chip Select 2, $\overline{CS}[2]$, functionality.

1 = Change the $\overline{CS}[2]$ pin functionality to correspond to General Purpose I/O bit 1.

Bit 0:

NoCS1

0 = Normal Chip Select 1, $\overline{CS}[1]$, functionality.

1 = Change the $\overline{CS}[1]$ pin functionality to correspond to General Purpose I/O bit 0.

3.4.1.2 Data Direction Register

Each bit in the Data Direction Register determines whether the corresponding pin will be configured as an input or an output. If a bit in the Data Direction Register is a one, the associated pin will be driven as an output by the value in the corresponding value in the Data Port Out register. A zero in a Data Direction register bit makes the corresponding pin an input, and its value can be read by reading the corresponding bit in the Data Port In register.

The 32-bit Data Port In Register is located at address EFC8h in the I/O map. These registers are read-only and always read the value of their associated pins.

The 32-bit Data Port Out Register is located at address EFCCh-EFCFh in the I/O map. These registers are both writable and readable. The value stored in this register is driven onto the corresponding pins if the associated RIO Control bit is set as well as the corresponding RIO Data Direction bit is set to one.

Note that whether a given pin is a general purpose I/O pin or not is determined by the value of the corresponding bit in the Reconfigurable I/O Control Register.

During a system reset the bits in the Data Direction register are set to zeros. This 32-bit register is located at IO addresses EFC4h - EFC7h.

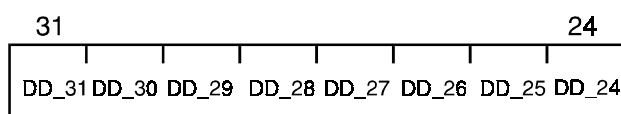
The following table shows the RIO pins and their associated register/bits.

Table 3-5: Reconfigurable I/O Pins and Register bits

I/O Function	Pin #	Name	RIO Ctl. Reg. Bit	Data Dir. Reg. Bit
PCMCIA	79	REG	7	31
PCMCIA	78	ENABLE	7	30
PCMCIA	77	DIR	7	29
PCMCIA	72	GPI	7	28
PCMCIA	71	V _{pp} -SEL2	7	27
PCMCIA	70	V _{pp} -SEL1	7	26
PCMCIA	69	V _{CC} -SEL	7	25
PCMCIA	68	CD_RST	7	24
LCD	48	CLF	6	22
LCD	50	CL2	6	21
LCD	49	CL1	6	20
LCD	51	LCD[3]	6	19
LCD	52	LCD[2]	6	18
LCD	53	LCD[1]	6	17
LCD	54	LCD[0]	6	16
ECP	81	PD[7]	5	15
ECP	82	PD[6]	5	14
ECP	83	PD[5]	5	13
ECP	85	PD[4]	5	12
ECP	86	PD[3]	5	11
ECP	88	PD[2]	5	10
ECP	89	PD[1]	5	9
ECP	90	PD[0]	5	8
UART	58	Rx	4	5
UART	59	UCLK	4	4
CS4	114	$\overline{CS}[4]$	3	3
CS3	115	$\overline{CS}[3]$	2	2
CS2	116	$\overline{CS}[2]$	1	1
CS1	117	$\overline{CS}[1]$	0	0

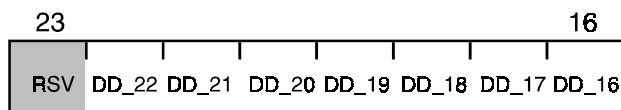
Data Direction Register (bits 31-24):

Note that the operation of the Data Direction Register bits are determined by the state of the RIO Control Register bits for each I/O function. For example, for Data Direction Register bit 31, if the NoCard bit (bit 7) in the RIO Control Register is set to 0, the $\overline{\text{REG}}$ pin remains a PCMCIA output pin and the values in Data Direction Register bit 31 have no effect. If the No-Card bit in the RIO Control Register is set to 1 (disabling the PCMCIA functionality), then the value in the Data Direction Register determine whether the pin is a general purpose input (bit = 0) or a general purpose output (bit = 1).



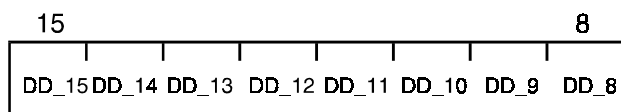
I/O Map Address	Access
EFC7h	R/W

Data Direction Register (bits 23-16):



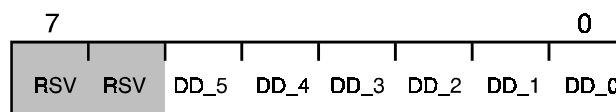
I/O Map Address	Access
EFC6h	R/W

Data Direction Register (bits 15-8):



I/O Map Address	Access
EFC5h	R/W

Data Direction Register (bits 7-0):



I/O Map Address	Access
EFC4h	R/W

3.4.1.3 Data Port Out Register

This 32-bit register is located at IO addresses EFCCh - EFCFh. This register provides the data output storage for the General Purpose

I/O discussed in the previous Data Direction Register section. When a RIO pin is defined as an output (by disabling the initial I/O function in the RIO Control Register and by setting the associated bit with the value of one in the Data Direction Register, its output value is contained in this 32-bit register. For example, if the PCMCIA is disabled by setting bit 7 in the RIO Control Register to 1, and bit 31 in the Data Direction Register is set to 1, pin 78 (the PCMCIA $\overline{\text{REG}}$ signal becomes an output pin. The value of bit 31 in the Data Port Out Register is output to that pin when the I/O write signals are valid.

3.4.1.4 Data Port In Register

This 32-bit register is located at IO addresses EFC8h - EFCBh. When this register is read, it will return the values currently on the corresponding NS486SXF pins described in the prior Data Direction Register section, regardless whether these pins are configured as input only, outputs or under the control of their primary function.

Note: Bits that are Reserved in the Data Direction Register may return indeterminate data.

3.4.2 The IEEE 1284 Bidirectional Parallel Port

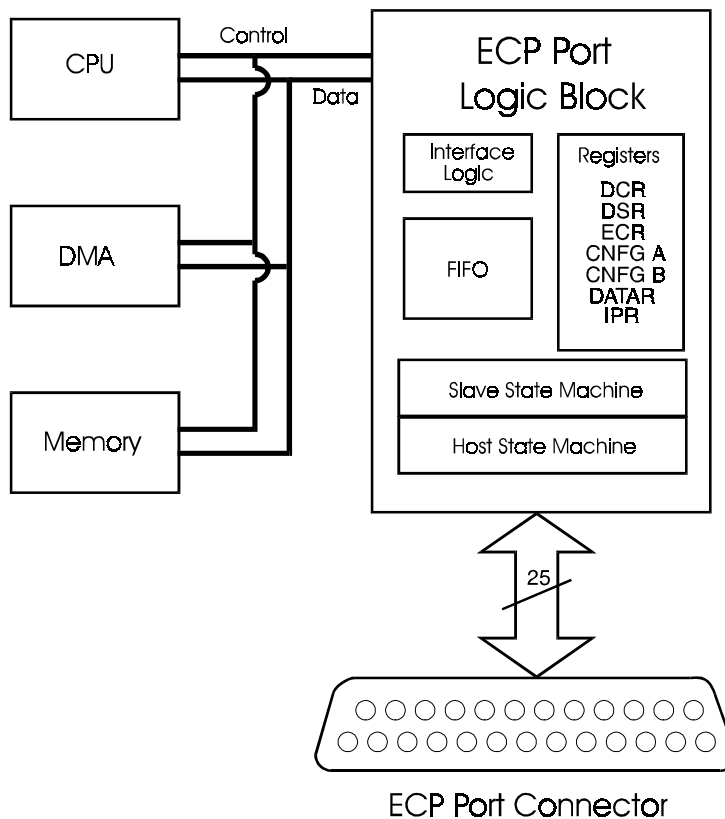
The NS486SXF parallel port is a multifunction 8-bit parallel port that is compatible with the IEEE 1284 bidirectional parallel port standard. The operation of the parallel port is controlled by the content of the NS486SXF parallel port I/O control registers. The port can operate in one of six modes:

- a standard parallel port mode (Centronics compatible with a unidirectional data port),
- a PS/2 compatible mode (same as compatible mode, but with bidirectional I/O lines),
- a parallel port with FIFO mode that adds a FIFO and state machines for handling handshake signals,
- the full Extended Capabilities Port (ECP) mode with state machines used for protocol and handshaking, and

- a configuration mode that permits access to special configuration registers that define the hardware implementation of interrupts (level vs. edge), and the steering of IRQ signals and the selection of DMA channels (ECP and parallel port with FIFO modes only).
- a test mode that makes the contents of the internal FIFOs accessible via a TFIFO register bypassing normal FIFO fill/empty control cycles.

The performance of the port is determined by the mode selected. The standard or Centronics compatible mode and the PS/2 mode support data transfers in the range of 50-150 kbytes/second. These are software limited modes where typically the CPU must respond to an interrupt on every data byte sent or received.

Figure 3-34 ECP Block Diagram



The parallel port with FIFO mode improves performance by limiting the interrupt to the CPU to one every 8-16 bytes. Data is stored in the on-chip 16-byte FIFO and handshake signals flag the CPU only when the buffer is full. In addition, a DMA access can be used to directly access the FIFO buffers, further reducing CPU overhead.

The full ECP mode is also a high performance mode. It uses the internal FIFO and state machine logic to enable the transfer of data using the DMA controller for high speed data transfer rates, with minimum CPU intervention. The ECP port uses an asynchronous handshake, allowing devices to operate as fast or as slow as needed. The timing specification of the ECP port is designed to allow a 2 Mbyte/second transfer rate over a 15 ft. cable. A shorter cable will result in a higher bandwidth. A longer cable will slow down the transfer, but in a non-destructive manner. The NS486SXF ECP port can support both Host and Slave ECP modes making the NS486SXF a versatile I/O controller.

Host/Slave, Data Flow Terminology:

The NS486SXF ECP port can operate as an interface between a Host CPU and a parallel port connector as might be the case for a PC application (Host function). This port can also operate as an interface between a parallel port connector and a peripheral CPU as would be the case in a printer (Slave function). These two modes are referred to as Host mode and Slave mode throughout this document.

The direction of transfer is referred to with respect to the flow of data across the parallel port cable. Forward direction refers to transfers where data flows from the Host to the Slave, while Reverse direction refers to transfers where data flows from the Slave to the Host. Notice that for the ECP port, both the Host-Forward direction and Slave-Reverse directions have the parallel port data pins driven by this block.

ECP mode also implements a simple 64:1 data compression scheme. Single byte RLE (Run Length Encoding) compresses strings of identical bytes. When a run-length count is received, the subsequent data byte is replicated the specified number of times.

ECP mode also supports a channel addressing scheme that provides for 128 channel addresses. Channel addresses can be changed dynamically and the default address is zero.

The following table shows the standard 25-pin, D-type connector definitions for ECP operations. Eight pins are used for grounds, leaving 17 active signal lines.

3.4.2.1 ECP Pinouts

Table 3-6: Parallel Port Pin Out

Connector Pin No.	SPP, ECP Mode	Pin Direction
1	STB	I/O
2	PD0	I/O
3	PD1	I/O
4	PD2	I/O
5	PD3	I/O
6	PD4	I/O
7	PD5	I/O
8	PD6	I/O
9	PD7	I/O
10	ACK	I
11	BUSY	I
12	PE	I
13	SLCT	I
14	ATFD	I/O
15	FAULT	I
16	INIT	I/O
17	SLCTIN	I/O

The 17 signal lines can be divided into three sets - eight data, five status and four control. The data lines are written or read as a single I/O port (Data Register) as are the status lines (Device Status Register) and control lines (Device Control Register). Each register has a separate I/O address location accessed by a com-

bination of a base I/O map address (0278h, 0378h or 03BCh) plus an offset value.

Because the ECP port has multiple functions, some setup and control registers have been added to the standard data, status and control registers. These include five setup registers; the Extended Control Register (ECR) that sets the operating mode and controls interrupt and DMA operations; and two Configuration Registers that control the steering of IRQ and DMA signals. These registers and their addresses are defined in the table on the next page.

The ECP also includes a 16-byte FIFO that can be configured for either direction, command/data FIFO tags (one per byte), a FIFO threshold interrupt for both directions, FIFO empty and full status bits, automatic generation of strobes (by hardware) to fill or empty the FIFO, transfer of commands and data, and a Run Length Encoding (RLE) decompression as explained later in this chapter. Like the data, status and control registers, the FIFO has its own register in the ECP control block, accessed through the base address and an offset. However, since the FIFO can be used in different modes, there are different register names for each type of FIFO access.

The AFIFO, CFIFO, DFIFO and TFIFO registers access the same ECP FIFO. The FIFO is accessed at Base + 000h, or Base + 400h, depending on the mode field of ECR (bits 7-5) and the direction bit (bit 5) in the DCR. The FIFO can be accessed by Host DMA cycles, as well as Host Port I/O (PIO) cycles.

When DMA is configured and enabled (bit 3 of ECR is 1 and bit 2 of ECR is 0), the ECP automatically (by hardware) issues DMA requests to fill the FIFO (in Host-Forward mode (bit 5 of DCR is 0), or Slave-Reverse mode (bit 5 of DCR is 1)) or to empty the FIFO (in Host-Reverse mode (bit 5 of DCR is 1), or Slave-Forward mode (bit 5 of DCR is 0)). All DMA transfers are to or from these registers. The ECP does not assert DMA request for more than 32 consecutive DMA cycles. The ECP stops requesting DMA when Terminal Count (TC) is detected during an ECP DMA cycle.

Writing to a full FIFO, and reading from an empty FIFO, are ignored. The written data is lost, and the read data is undefined. The FIFO Empty and Full status bits are not affected by such accesses.

ECP Operating Modes:

The ECP port operates in different phases, changing from one phase to another based on the state of the bits in the setup and control registers. All operating modes begin with Compatibility mode. In this mode, the ECP port is in Centronics compatible mode and performs host to printer data transfers. The ECP port starts a negotiation phase from the Compatible mode. The Negotiation phase begins with the host initiating a handshaking sequence with the attached peripheral to determine the highest level of ECP compatibility that can be supported.

After a successful negotiation, the ECP enters the Setup phase and configures ECP resources to support the appropriate level. After setup, the ECP port enters the Forward phase and data transfers commence. In the Forward phase, the ECP may switch from Forward to Reverse allowing bidirectional transfers. A Termination phase ends the ECP mode transfer and the port reverts back to Compatible mode.

This procedure is fully documented in the IEEE 1284 standard.

Figure 3-35 Phase Transition States

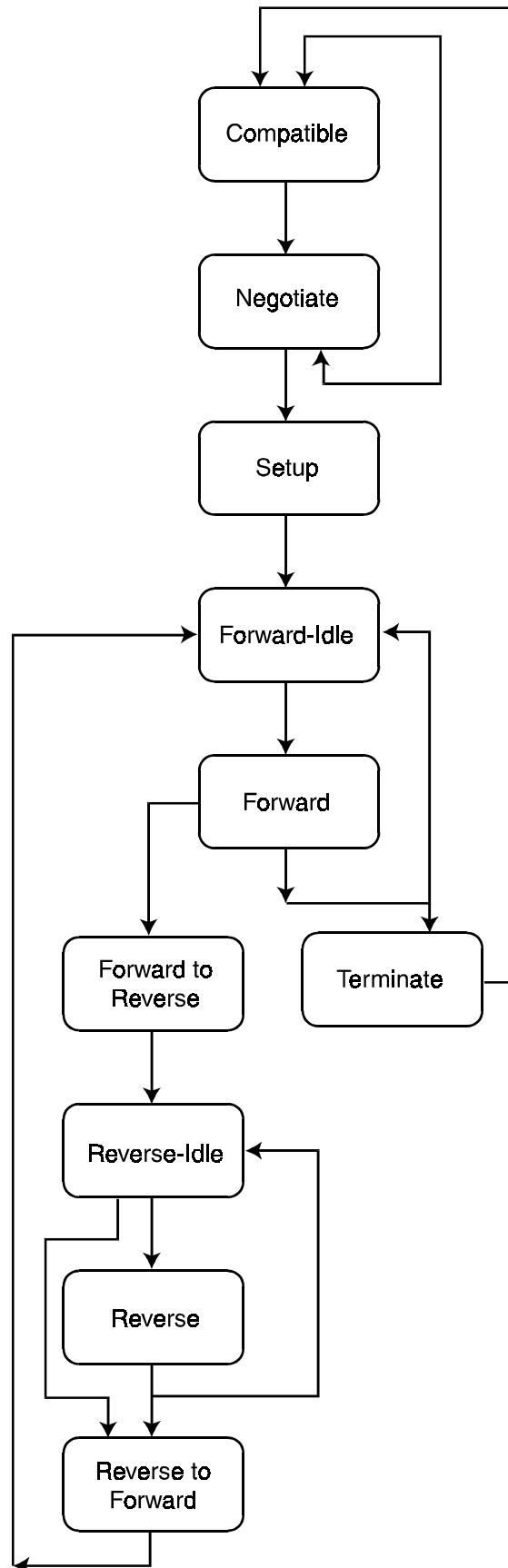


Table 3-7: ECP Registers

Name	Address	I/O	Size	Mode # ECR(5-7)	Function
DATAR	Base + 000h	R/W	byte	000,001	Port Data Register
AFIFO	Base + 000h	R/W	byte	011	ECP Address FIFO
DSR	Base + 001h	R/W	byte	ALL	Device Status Register
DCR	Base + 002h	R/W	byte	ALL	Device Control Register
CFIFO	Base + 400h	R/W	byte	010	Parallel Port Data FIFO
DFIFO	Base + 400h	R/W	byte	011	ECP Data FIFO
TFIFO	Base + 400h	R/W	byte	110	Test FIFO
CNFGA	Base + 400h	R/W	byte	111	Configuration Register A
CNFGB	Base + 401h	R/W	byte	111	Configuration Register B
ECR	Base + 402h	R/W	byte	ALL	Extended Control Register
INDEX	Base + 403h	R/W	byte	ALL	Index Register
SETUP	Base + 404h	R/W	byte	ALL	Setup Registers
IPR	Base + 405h	R	byte	ALL	Trap Certain Events
Base address is 278h, 378h or 3BCh					

3.4.2.2 ECP Registers

Some registers are not accessible in all modes of operation, or may be accessed in one direction only. Accessing a non accessible register has no effect: Data read is undefined, data written is ignored, and the FIFO does not update.

Software operation is detailed in the document *Extended Capabilities Port Protocol and ISA Interface Standard*. However, the functions necessary to setup the ECP block prior to normal operations are not specified. Setup functions required for this block must be derived from this specification (see definitions for Index, and Setup registers). The user should follow the steps below to properly setup the NS486SXF ECP port:

1. The software must setup the ECP block using the Setup Registers, prior to enabling.
2. After ECP setup, the software should enable the ECP block using Setup Register offset 0, bit 0=0.
3. When the ECP block is enabled, and the software wishes to switch modes, it should switch only through Standard Parallel Port or PS/2 modes (ECR bits 7-5 = 000 or 001).
4. When the ECP block is enabled, the software should change direction only in PS/2 mode, or the Idle-Phase in ECP mode (ECR bits 7-5 = 001).
5. The software should switch from Parallel Port FIFO mode (ECR bits 7-5 = 010), or ECP mode (ECR bits 7-5 = 011), to Standard Parallel Port, or PS/2 mode (000, or 001), only when the FIFO is empty.

6. The user software should switch to ECP mode when bits 0 and 1 of DCR are 0. (The \overline{ATFD} signal and the \overline{STB} signal are controlled by ECP hardware and software when these bits are set to 0.)
7. The user software should switch to Parallel Port FIFO mode when only bit 0 of DCR is 0. This puts control of the \overline{STB} signal under the control of the ECP hardware and software. The Parallel Port FIFO mode is for Forward transfers only.
8. The software should disable the ECP port only when in Standard Parallel Port or PS/2 mode.

In Host mode, software may switch from ECP mode Reverse direction to Standard Parallel Port or PS/2 mode when there is an on-going ECP cycle. In this case the read cycle is aborted by deasserting \overline{ATFD} . The FIFO is reset (empty) and a potential data decompression (RLE) is automatically terminated since the new mode is Standard Parallel Port or PS/2 mode. This should be done carefully, however, as changing from Reverse ECP transfers to Standard Parallel Port mode may cause contention on the parallel port data lines.

3.4.2.3 ECP Power Management

The ECP uses the CPU clock, which can be stopped during power down mode. Power Down mode is usable only when operating in Host mode. In power down mode, DMA is disabled, the FIFO is unaccessible (i.e. accesses are ignored) and RLE decompression does not function.

In Power Down mode, all interrupts are masked (except the Standard Parallel Port mode \overline{ACK} interrupt). However, if the ECP block remains enabled, then all registers are accessible, even if the clock is not running.

During this mode, the status and contents of the FIFO are preserved. However, if a read is made of the ECR register, bits 2,1, and 0 will be 011, regardless of the actual FIFO contents. When the ECP block clock is restarted and the ECP block is brought out of Power Down mode, the contents of the above register bits will again reflect the state of the FIFO, and IRQ line.

When the clock is stopped during an ECP cycle, that cycle should be considered corrupted. When the clock is restarted, the cycle will continue the transfers if the FIFO is not empty and the block is in the Forward direction or if the FIFO is not full in the Reverse direction.

When the ECP block is in Power Down mode, the output pads may be TRISTATED, and internal pull-up resistors turned off for further power savings. This is done by setting bit 7 of Setup register (offset = 0) to 1.

Power down should only be applied to a Host interface. The reason for this is that the Host CPU is always responsible for controlling the cycles. The Slave on the other hand must always respond. Since the Slave does not know when the Host will begin a cycle, it must assume that one will be coming at all times. The slave logic must therefore have the state machines involved in monitoring handshaking signals operating at all times. This includes the logic for compatible mode as well as the higher ECP functions.

In Slave mode, if the software is certain that no data transfers will be occurring for some period of time, it can place the slave interface in power down mode. Software should be careful to do this in such a way that the interface does not indicate to the Host that it is ready to perform transfers (e.g. BUSY is low in Forward mode). In power down mode, the DCR, DSR, and DATAR (in compatible-negotiation mode) can be monitored by software periodically to see if the Host is signalling for an ECP negotiation phase.

Notes: Only the FIFO / DMA / RLE are not functional when the clock is frozen. All other registers are accessible and functional. The FIFO / DMA / RLE are affected by ECR modifications. i.e., they are reset even when exits from Parallel Port FIFO mode, or ECP mode are carried out while the clock is frozen.

3.4.2.4 Register Addressing

All registers are accessed through I/O Address space reads and writes. The FIFO data register can also be accessed using DMA cycles. The addresses of individual registers can be computed by taking a base value and adding an incremental value specific to each register to get the resulting register address.

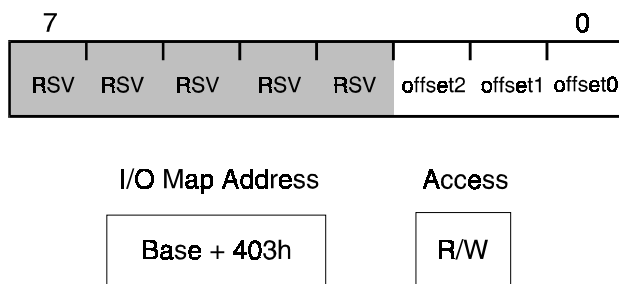
The base address for these registers will be one of the following; 278h, 378h, 3BCh. The currently active base address is defined in the BIU section of the NS486SXF.

Certain addresses access different registers, depending on the currently active mode, defined by bits 7-5 of the Enhanced Control Register (ECR).

In particular, the Setup registers, which control the basic functions of the ECP block, are accessed by a combination of the base address and an offset (403h) as shown in the ECP Register table. In that there are several Setup Registers, which Setup Register will be at the address defined by the base address and offset is determined by the contents of the low order three bits of the Index Register (bits 2-0). These bits point to the individual Setup Register.

3.4.2.5 Index Register (Base address + 403h)

This register controls the accesses to the Setup register. The offset value programmed to bits 2-0 will direct all read/write accesses of the Setup register to one of the four Setup registers.



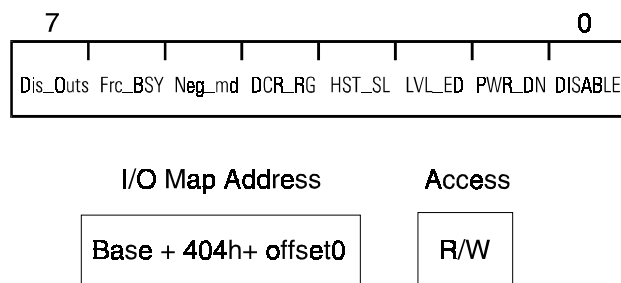
- Bits 7-3: Reserved.
- Bits 2-0: Offset — These bits define the lookup offset for accesses to the Setup register. The reset state of these bits is 000.

3.4.2.6 Setup Register (Base address + 404h)

These registers control basic functions of the ECP block. The setup register set is comprised of four separate registers which can be accessed one at a time. The currently selected register to which all reads and writes will be directed is identified by the programming of bits 2-0 in the Index register. Registers for offsets 101 through 111 are reserved.

3.4.2.6.1 Setup Register-Base Register (Offset 0)

This register defines the base functions to define the behavior of the ECP block.



- Bit 7: Disable Outputs — When this bit is 1, all parallel port outputs will be TRISTATED and all internal pullup resistors will be turned off. The reset state of this bit is 1.
- Bit 6: Force BUSY — When in Slave mode, this bit is used to force the BUSY signal to 1, halting ongoing Forward data transfers. When this bit is 1, and the interface is operating in either ECP, or Parallel Port FIFO mode as a Slave interface, BUSY will only be allowed to transition from low to high. Should the BUSY signal already be 1 when this bit is set, BUSY will not transition to 0, until this bit is cleared. The reset state of this bit is 0.
- Bit 5: Negotiation mode — When this bit is 1, and the interface is operating in Standard Parallel Port mode, as a Slave interface, Reads of the DATA Register (DATAR) will reflect the live states of the parallel port data pins. Additionally, BUSY will not automatically be driven to 1 when STB is low, nor will an IRQ be generated by the STB signal. When this bit is 0,

- parallel port data will be latched into the DATAR register in the rising edge of the \overline{STB} signal (adjusted for noise filtering), and an IRQ will be generated to signal to the CPU that a Standard Parallel Port mode byte transfer has completed. Also BUSY will be driven to 1 when the \overline{STB} signal is seen low (adjusted for noise filtering). The reset state of this bit is 0.
- Bit 4: DCR Registered vs. Live — When this bit is 1, all reads of the DCR in Host mode will reflect the last data written to that register. When this bit is 0, and the port is operating as a Host interface, and the current mode is not ECP mode, all reads of the DCR will reflect the live state of the control pins, and not the last data written to that register. The reset state of this bit is 1.
- Bit 3: Host vs. Slave — When this bit is 1, the ECP block operates as an interface between a system CPU, and an ECP connection (Host mode). When this bit is 0, the ECP block operates as the interface logic between an ECP connection and a peripheral's CPU (Slave mode). The reset state of this bit is 0.
- This bit should only be changed when bit 7 of this register is 1. Alternatively, software should ensure that the ECP block is in compatible mode, with all outputs TRISTATED, prior to changing this bit.
- Bit 2: Level vs. Edge Interrupts — When this bit is 0, the ECP block will cause pulses to be generated for all interrupt conditions. When this bit is 1, the ECP block will hold the IRQ signal active (low) so long as the interrupt condition has not been cleared by the CPU. The reset state of this bit is 0.
- Bit 1: **POWER DOWN** — When this bit is 1, the ECP block will immediately enter a power down state. All DMA activity is halted immediately. All IRQ's are immediately cleared. State machines and register values are unaffected by this bit. This bit should not be set to 1 when in Slave modes other than Compatible-Negotiation mode. The reset state of this bit is 0.
- Bit 0: **Block Disable** — When this bit is 1, the ECP block is disabled. All state machines, and FIFO control logic is reset. All registers are held in their reset state. This bit must be set to 0 to begin any parallel port activity. The reset state of this bit is 1.

3.4.2.6.2 Setup Register-Pulse Width Definition Register (Offset 1)

This register defines the number of clocks to use in defining the pulse widths of DRQ deassertion, and \overline{STB} signal when operating in Host-Parallel Port FIFO mode, or BUSY, and \overline{ACK} when operating in Slave-Parallel Port FIFO mode.



I/O Map Address

Base + 404h+ offset1

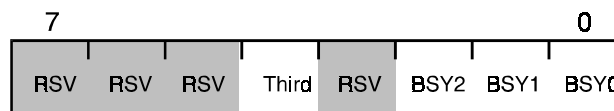
Access

R/W

- Bits 7-4: Minimum DRQ Deassertion Time — These bits define the minimum number of clocks for which the DRQ signal will be held deasserted. The minimum value that can be programmed to this register is 3. Values less than 3 may cause the DRQ behavior to become erratic. The reset value of these bits is 0111
- Bits 3-0: Pulse Width — These bits define the number of clocks for which \overline{STB} will be held active when in Host-Parallel Port FIFO mode, and BUSY and \overline{ACK} will be held active when in Slave-Parallel Port FIFO mode. A value of 0 will result in a one clock wait. The reset value of these bits is 1010

3.4.2.6.3 Setup Register - Host Mode Noise Filtering Register (Offset 2)

This register defines the number of clocks to use to filter out noise on the BUSY and \overline{ACK} signals during Host mode operation.



I/O Map Address

Base + 404h+ offset2

Access

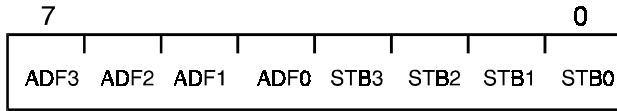
R/W

- Bits 7-5: Reserved.
- Bit 4: Add Third \overline{ACK} Noise Filter Clock. — When this bit is 0, \overline{ACK} must be low (0) for two consecutive clocks prior for the state machines to recognize that the peripheral device has acknowledged a transfer. When this bit is 1, a third clock is added to the filtering. The reset state of this bit is 0.
- Bit 3: Reserved.
- Bits 2-0: BUSY Filtering Clocks — These bits define the number of clocks for which BUSY will be ignored to allow a settling time due to the change of state on the data pins. A value of 0 will result in 1 clock cycle being used to ignore the state of BUSY. The reset value of these bits is 011 (i.e. 3 clock cycles).

Notes: Care should be taken when programming this register. If the operating frequency is low, it is possible to program this register such that all of the pulse widths of the status signals in Host-Standard Parallel Port mode transfers can be missed.

3.4.2.6.4 Setup Register - Slave Mode Noise Filtering Register (Offset 3)

This register defines the number of clocks used to filter out noise on the \overline{ATFD} and \overline{STB} signals during Slave mode operation.



I/O Map Address

Base + 404h+ offset3

Access

R/W

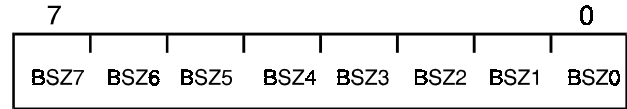
Bits 7-3: \overline{ATFD} Filtering Clocks — These bits define the number of clocks for which \overline{ATFD} must be stable before the state machines will recognize a signal transition. If these bits are programmed to 0, one clock will be used for this purpose. The reset value of these bits is 0100 (i.e. 4 clock cycles).

Bits 3:0: \overline{STB} Filtering Clocks — These bits define the number of clocks for which \overline{STB} must be stable before the state machines will recognize a signal transition. If these bits are programmed to 0, one clock will be used for this purpose. The reset value of these bits is 0010.

Notes: Care should be taken when programming this register. If the operating frequency is low, it is possible to program this register such that all of the pulse widths of the command signals in Slave-Standard Parallel Port mode transfers can be missed.

3.4.2.6.5 Setup Register - DMA Burst Size Register (Offset 4)

This register is defines the DMA burst cycle size. Range can be from 1 cycle per DMA request (value = 01h), to 16 cycles per DMA request (value = 1Fh). The default value is 1Fh.



I/O Map Address

Base + 404h+ offset4

Access

R/W

Bit 7-0: Specifies number of DMA burst cycles per DMA request. Reset value = 00011111, 16 cycles per request.

Suggested Setup Register Values

To help in understanding how to configure the Setup Registers, some default values and suggested settings for these registers are given below to show the flexibility of the ECP block in handling different operating frequencies, system performance and noise levels.

The values in Setup Register, offset 0, are system implementation and task specific so there are no suggested values.

The values in Setup Register, offset 1, adjust the filtering and pulse width clocks for different frequencies. The values in Setup Register, offset 2, set the number of clocks to filter out noise on the \overline{BUSY} and \overline{ACK} signals during Host mode operation. The values in Setup Register, offset 3, sets the number of clocks to filter out noise on the \overline{ATFD} and \overline{STB} signals during Slave mode operation. Which values are best for the specific NS486SXF configuration and clock frequency is answered by looking at the suggested values in the following tables:

Setup Register Values:

Setup Register Offset	Default setting (24 MHz)	30 MHz	25 MHz	20 MHz
1	7Ah	A Eh	8Bh	69h
2	03h	15h	14h	03h
3	42h	63h	53h	42h

Fine tuning the filtering for adjusting noise levels on the ECP table is done with Setup Register, offset2 and offset3. Their values can be adjusted for best and worst case noise levels. (The values in the above table are suggested values for those clock frequencies; the designer can adjust for a specific case using the following tables.)

Setup Register Offset	Default setting	Best noise level	Worst noise level
2	03h	01h	17h
3	42h	11h	FFh

The Setup Register, offset 4 adjusts the DMA burst size. The value may range from 1 to 16 cycles.

Setup Register Offset	Default setting	1 Cycle per DMA Request	8 Cycles per DMA Request
4	1Fh	01h	08h

3.4.2.7 Data Register (DATAR) (Base address + 000h)

This register makes up the bits of the data port interface to the parallel port data pins for Standard Parallel Port and PS/2 modes. In Host-Standard Parallel Port mode and Slave-PS/2 Reverse mode, data written to this register is immediately driven onto the data pins. In Host-PS/2 Reverse mode these bits show the live state of the parallel port data pins. In Slave-Standard Parallel Port mode, when the Negotiation mode bit in Setup register (offset = 0) is 1, these bits will show the live state of the parallel port data pins. When in Slave-Standard Parallel Port mode, and the Negotiation mode bit is 0, reads show the last data clocked into this register by the rising edge of the \overline{STB} signal. This register is not initialized on reset. After reset, data in this register should be treated as indeterminate.



I/O Map Address

Access

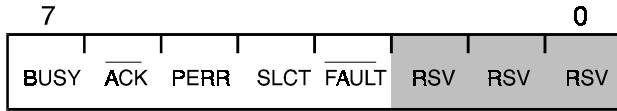
Base + 000h

R/W

Bit 7-0:

Data.

3.4.2.8 Device Status Register (DSR)
(Base address + 001h)



Bit 7: **BUSY** — When in Host mode this bit (BUSY) represents the inverted state of the printer BUSY signal. The printer sets this bit low when it is busy and cannot accept another character.

In Slave mode, the inverted state of this bit is driven onto the BUSY signal. In Slave mode, reads of this bit reflect the inverted state of the BUSY signal. This bit will be set to 0 automatically when the DATAR is loaded with parallel port data in Slave-Standard Parallel Port mode. The reset value of the register for this bit is 0.

Bit 6: **ACK** — This bit represents the current state of the printer acknowledge signal (ACK). The printer pulses this signal low after it has received a character and is ready to receive another one. This bit follows the state of the $\overline{\text{ACK}}$ pin. The pin becomes an output in slave mode. The reset value of the register for this bit is 1.

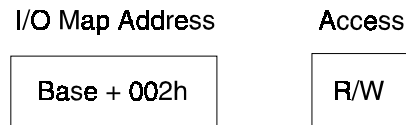
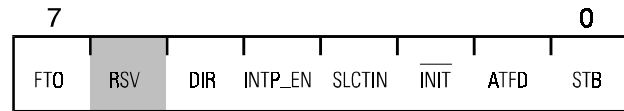
Bit 5: **PERR** — This bit represents the current state of the printer paper end signal (PERR). The printer sets this bit high when it detects the end of the paper. This bit follows the state of the PERR pin. The pin becomes an output in slave mode. The reset value of the register this bit is 1.

Bit 4: **SLCT** — This bit represents the current state of the printer select signal (SLCT). The printer sets this bit high when it is selected. This bit follows the state of the SLCT pin. The pin becomes an output in slave mode. The reset value of the register for this bit is 1.

Bit 3: **FAULT** — This bit represents the current state of the printer error signal (FAULT). The printer sets this bit low when there is a printer error. This bit follows the state of the FAULT pin. The pin becomes an output in slave mode. The reset value of the register for this bit is 1.

Bits 2-0: Reserved. These bits are always a 1.

3.4.2.9 Device Control Register (DCR)
(Base address + 002h)



Bit 7: **FTO -- FIFO Time Out**. This bit indicates that the data in the FIFO has become stale (no change in greater than 256 clocks (~10 microseconds @ 25MHz)). This bit is meaningful only in receive mode and is used to detect the end of a data stream.

Bit 6: Reserved, This bit is always 1.

Bit 5: **Direction** — The direction bit determines whether PD7 - PD0 are driven as outputs or TRISTATED for use as inputs.

When bit 5 of the DCR is 0 the ECP is in Forward direction, and when bit 5 is high (1) the ECP block is operating in the Reverse direction. This will be true for both Host and Slave mode. The default value of this bit is 0, reflecting the Forward direction.

In Host mode, the ECP block drives the PD7-0 pins in the Forward direction but does not drive them in the Reverse direction. In Slave mode, the PD7-0 pins are driven when in the Reverse direction, and are not driven in the Forward direction.

The direction bit, is readable and writ-

able, except in Standard Parallel Port or Parallel Port FIFO modes. In Standard Parallel Port and Parallel Port FIFO modes the direction bit is forced to 0, and data written into this bit is ignored. The reset value of the register for this bit is 0.

Bit 4: Standard Parallel Port IRQ Enable — This bit enables the $\overline{\text{ACK}}$ deassertion interrupt event (1=enable, 0=mask). If a level interrupt is configured clearing this bit clears the $\overline{\text{ACK}}$ pending interrupt request. This bit does not float the IRQ pin.

In ECP mode this bit should be set to 0.

In Slave mode this bit enables the generation of IRQ's in Standard Parallel Port mode, on the rising edge of the $\overline{\text{STB}}$ signal. The reset value of the register for this bit is 0.

Bit 3: $\overline{\text{SLCTIN}}$ — In Host mode, this bit ($\overline{\text{SLCTIN}}$) directly controls the select in signal to the printer via the $\overline{\text{SLCTIN}}$ pin. Setting this bit high selects the printer. It is the inverse of the $\overline{\text{SLCTIN}}$ pin. Reads of this bit will reflect the live state of the $\overline{\text{SLCTIN}}$ pin in Standard Parallel Port mode. In ECP mode, reads of this bit will reflect the last data written.

In Slave mode, this bit reflects the inverted state of the $\overline{\text{SLCTIN}}$ signal.

The reset value of the register for this bit is 0.

Note: this bit must be set to 1 before enabling the ECP or Parallel Port FIFO modes.

Bit 2: $\overline{\text{INIT}}$ — In Host mode, this bit directly controls the signal to initialize the printer via the $\overline{\text{INIT}}$ pin. Setting this bit to low initializes the printer. Reads of this bit will reflect the live state of the $\overline{\text{INIT}}$ pin in Standard Parallel Port mode. In ECP mode, reads of this bit will reflect the last data written.

In Slave mode, this bit reflects the live state of the $\overline{\text{INIT}}$ signal.

The reset value of the register for this bit is 0.

Bit 1: $\overline{\text{ATFD}}$ — In Host mode, this bit directly controls the Automatic Paper Feed signal to the printer via the $\overline{\text{ATFD}}$ pin. Setting this bit high causes the printer to automatically feed paper after each line is printed. Reads of this bit will reflect the inverted state of the $\overline{\text{ATFD}}$ pin in Standard Parallel Port mode. In ECP mode, reads of this bit will reflect the last data written.

In Slave mode, this bit reflects the inverted state of the $\overline{\text{ATFD}}$ signal generated by the Host.

In Host ECP mode the $\overline{\text{ATFD}}$ is controlled by both ECP hardware and software. For this reason, this bit should be 0 when in these modes. The reset value of the register for this bit is 0.

Bit 0: $\overline{\text{STB}}$ — In Host mode, this bit directly controls the data strobe signal to the printer via the $\overline{\text{STB}}$ pin. Reads of this bit will reflect the inverted state of the $\overline{\text{STB}}$ pin in Standard Parallel Port mode. In ECP mode, reads of this bit will reflect the last data written.

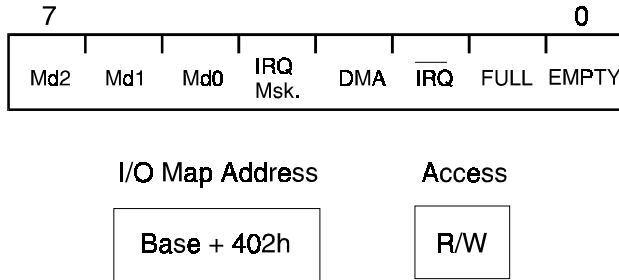
In Slave mode, this bit reflects the inverted state of the $\overline{\text{STB}}$ signal.

In Host - Parallel Port FIFO, or ECP modes the $\overline{\text{STB}}$ is controlled by both ECP hardware and software. For this reason, this bit should be 0 when in these modes. The reset value of the register for this bit is 0.

Signal	Reset Control	State After Reset
SLCTIN	Disable	TRI-STATE
INIT	Disable	Zero
ATFD	Disable	TRI-STATE
STB	Disable	TRI-STATE
Note: Disable is bit 0 of Setup-Base Register (offset = 0).		

3.4.2.10 Extended Control Register (Base address + 402h)

This register controls the ECP and parallel port functions. This register is reset when the port is not enabled. This register is initialized to 00010101.



Bits 7-5: Operating Mode.

000: Standard Parallel Port mode.

Forward transfer cycles are performed under software control. Bit 5 of DCR is forced to 0. (Forward direction) PD7-0 are driven in Host mode, and TRISTATED in Slave mode. The FIFO is reset (empty).

In Slave mode, Data is latched into DATAR and BUSY is automatically driven high when \overline{STB} is sampled low. (Noise filtering clocks apply to \overline{STB} in this mode.)

001: PS/2 mode.

Forward and Reverse cycles are performed under software control. The FIFO is reset (empty). Direction of transfer defined by DCR bit 5.

010: Parallel Port FIFO mode.

Host cycles are performed under hardware control (\overline{STB} is controlled by hardware). Slave cycles are likewise performed under hardware control (BUSY, and \overline{ACK} are controlled by hardware). Bit 5 of DCR is forced to 0 (Forward direction).

011: ECP mode.

The FIFO direction is controlled by bit 5 of DCR. Read and write cycles to the de-

vice are performed under hardware control (In Host mode, \overline{STB} and \overline{ATFD} are controlled by hardware. In Slave mode, BUSY and \overline{ACK} are controlled by hardware).

This mode should only be entered for ECP Data Transfer Phases. Prior to entering this mode, the CPU must have successfully Negotiated, then Setup the ECP transfer, in Host-Standard Parallel Port mode or Slave-Standard Parallel Port mode (with Negotiation mode bit = 1).

100: Reserved.

101: Reserved.

110: FIFO Test mode.

The FIFO is accessible via the TFIFO register. The ECP does not issue ECP cycles to fill/empty the FIFO.

111: Configuration mode.

The CNFGA and CNFGB registers are accessible in this mode.

The reset values for these bits are 000.

Bit 4:

ECP Interrupt Mask bit — When in Host mode, and this bit is 0 an interrupt is generated when \overline{FAULT} is asserted (\overline{FAULT} is 0). Should the \overline{FAULT} signal be asserted when this bit is changed to 0, an interrupt will be generated immediately. When this bit is 1, no interrupt is generated.

This bit is ignored in Slave mode. The reset state of this bit is 1.

Bit 3:

DMA Enable bit — When this bit is 0, DMA is disabled and the internal DRQ pin is 0. When this bit is 1, DMA is enabled and DMA starts when bit 2 of ECR is 0. The reset state of this bit is 0.

Note: Internal DMA Acknowledge is assumed inactive when this bit is 0.

Bit 2:

IRQ Service bit — When this bit is 0, and one of the following three interrupt events occur, an interrupt is generated and this bit is set to 1 by hardware. These

interrupt conditions are valid for both Host and Slave mode operation.

- 1) Bit 3 of ECR is 1, and terminal count is reached during DMA.
- 2) Bit 3 of ECR is 0 and bit 5 of DCR is 0, and there are eight or more bytes free in the FIFO.
- 3) Bit 3 of ECR is 0 and bit 5 of DCR is 1, and there are eight or more bytes to be read from the FIFO.

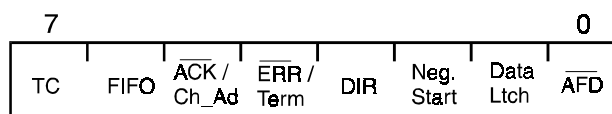
When this bit is 1, DMA and the above three interrupts are disabled. Writing 1 to this bit does not cause an interrupt. When the ECP clock is frozen this bit is read as 0, regardless of its actual value (even though the bit may be modified by software when the ECP clock is frozen). The reset state of this bit is 1.

Bit 1: FIFO Full bit — Read only. This bit is 0 when the FIFO has at least one free byte. This bit is 1 when the FIFO is full. This bit continuously reflects the FIFO state, and therefore can only be read. Data written to this bit is ignored. When the ECP clock is frozen this bit is read as 1, regardless of the actual FIFO state. The reset state of this bit is 0.

Bit 0: FIFO Empty bit — Read only. This bit is 0 when the FIFO has at least one byte of data. This bit is 1 when the FIFO is empty. This bit continuously reflects the FIFO state, and therefore can only be read. Data written to this bit is ignored. When the ECP clock is frozen this bit is read as 1, regardless of the actual FIFO state. The reset state of this bit is 1.

3.4.2.11 Interrupt Pending Register (Base address + 405h)

This register is a read-only register provided to indicate certain pending interrupt conditions.



I/O Map Address

Base + 405h

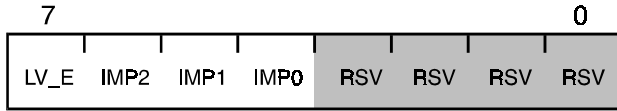
Access

R

- Bit 7: TC event
- Bit 6: FIFO threshold event
- Bit 5: ACK event (host mode) or Chan_Addr event (slave mode)
- Bit 4: ERR falling event (host mode) or Transfer termination event (slave mode)
- Bit 3: Channel direction change event (slave mode)
- Bit 2: Negotiation start event (slave mode)
- Bit 1: Data latch event (slave mode)
- Bit 0: Falling edge of AFD (slave mode)

3.4.2.12 CNFG A Register
(Base address + 400h, in Configuration mode)

This register contains the implementation information for this ECP block.



I/O Map Address

Base + 400h

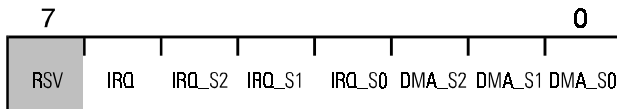
Access

R/W

- Bit 7: Level vs. Edge Interrupts — This read only bit reflects what kind of interrupts the ECP block will generate. When this bit is 1, the ECP block will generate active low, level interrupts. When this bit is 0, the ECP block will generate pulsed interrupts. This bit is read only, and reflects the negated state of bit 2 of Setup register offset 0.
- Bits 6-4: Implementation ID — These read only bits will return the value of 001, reflecting to software that this ECP block is an 8-bit only implementation.
- Bits 3-0: Reserved.

3.4.2.13 CNFG B Register
(Base address + 401h, in Configuration Mode)

This register controls the steering of DMA Request/Acknowledge, and IRQ signaling.



I/O Map Address

Base + 401h

Access

R/W

- Bit 7: Reserved.

- Bit 6: IRQ — This bit reflects the state of the IRQ logic. If an edge IRQ has been generated, or if a Level IRQ is being signaled, this bit will read as a 1. The reset state of this bit is 0.
- Bits 5-3: IRQ Select — These bits define the system IRQ signal which the ECP block will use to signal interrupt conditions.

- 111 - IRQ5
- 110 - IRQ14
- 101 - IRQ13
- 100 - IRQ11
- 011 - IRQ10
- 010 - IRQ9
- 001 - IRQ7 (default)
- 000 - Disables signaling of ALL IRQ's

- Bits 2-0: DMA Select — These bits define the DMA Request/Acknowledge handshake signals the ECP block will use to perform DMA cycles.

- 111 - Reserved
- 110 - Reserved
- 101 - Selects DMA channel 5
- 100 - Reserved
- 011 - Reserved (default)
- 010 - Reserved
- 001 - Selects DMA channel 1
- 000 - Reserved

It should be noted that in most systems, DMA channels 7-5 are 16-bit DMA channels. This register allows the DMA signals to be steered to channel 5, however the ECP block will ONLY operate in 8-bit mode.

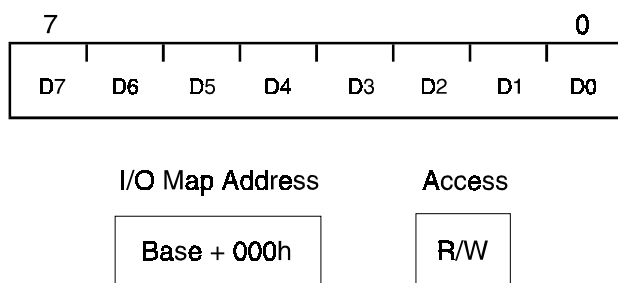
Programming these bits to a reserved value will cause DMA activity to hang, since no DMA Request/Acknowledge can occur.

The reset state of these bits is 011.

3.4.2.14 ECP Address FIFO (AFIFO) (Base address + 000h, in ECP Mode)

In Host mode this register is write only. In the Forward direction (bit 5 of DCR is 0) a byte written into this register is pushed into the FIFO and tagged as command data. Reads of this register give undefined results. Writes to this register during Reverse direction (bit 5 of DCR is 1) have no effect and the data is ignored.

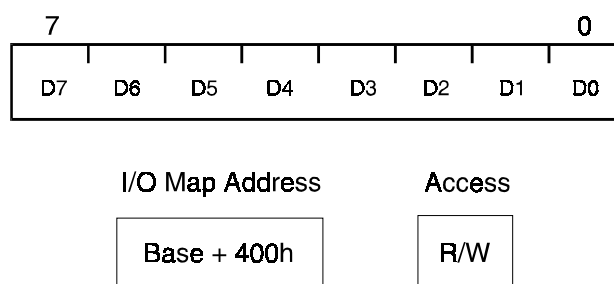
In Slave mode, this register is read only. This is the register where any Channel Addresses issued by the Host will be read. Data written to this register will be ignored.



3.4.2.15 Parallel Port FIFO Register (CFIFO) (Base address + 400h, in Parallel Port FIFO Mode)

In Host mode, this register is write only. A byte written, or DMAed, to this register is written into the FIFO. Reading this register has no effect and the data read is undefined.

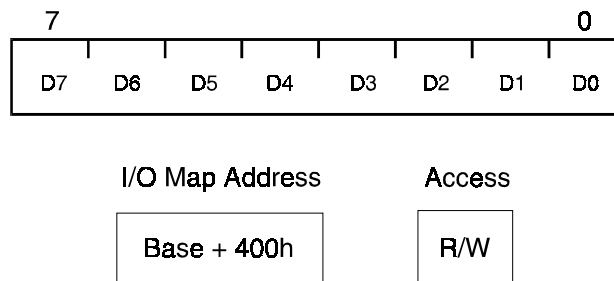
In Slave mode, this register is read only. A byte is written into the FIFO by the state machines for every compatible mode transfer on the parallel port pins. Writing this register has no effect.



3.4.2.16 ECP Data FIFO Register (DFIFO) (Base address + 400h, in ECP Mode)

In Host mode-Forward direction, or Slave mode-Reverse direction, a byte written or DMAed to this register, is written into the FIFO and tagged as data. Reading this register has no effect and the data read is undefined.

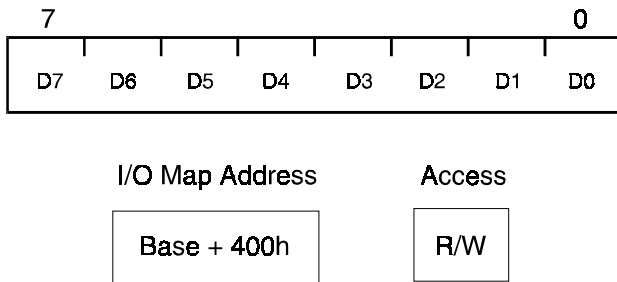
In Host mode-Reverse direction, or Slave mode-Forward direction, the ECP automatically issues ECP read cycles to fill the FIFO. Reading this register pops a byte from the FIFO. Writing this register has no effect and the data written is ignored.



3.4.2.17 Test FIFO Register (TFIFO) (Base address + 400h, in FIFO Test Mode)

A byte written into this register is written into the FIFO. A byte read from this register is emptied from the FIFO. The ECP does not issue any ECP cycles in this mode.

The TFIFO is readable and writable in both directions. In Host-Forward direction and Slave-Reverse direction, PD7-0 are driven but the data is undefined.



3.4.2.18 Software Controlled Data Transfer (Host - Standard Parallel Port and PS/2 modes)

Software controlled Host mode data transfers are supported in Standard Parallel Port and PS/2 modes. The software generates peripheral-device cycles by modifying the DATAR and DCR registers and reading the DSR, DCR and DATAR registers. The negotiation phase and nibble mode transfer, as defined in the IEEE 1284 standard, are performed in Standard Parallel Port mode.

In these modes the FIFO is reset (empty) and is not functional, the DMA and RLE are idle. Standard Parallel Port mode is for the Forward direction only; the direction bit is forced to 0 and PD7-0 are driven. PS/2 mode is for both the Forward and Reverse directions. The direction bit controls whether PD7-0 are TRISTATED for use as inputs.

3.4.2.19 Automatic Data Transfer (Host - Parallel Port FIFO and ECP modes)

Host mode automatic data transfer (data transfers generated by hardware) is supported in Parallel Port FIFO and ECP mode. The ECP block will automatically generate the necessary handshaking signals for Parallel Port FIFO mode. In ECP mode, only the data transfer phases will be handled by hardware. For ECP mode, Software must handle all the Negotiation, Setup, and Termination phases.

Automatic DMA access to fill or empty the FIFO is supported in Parallel Port FIFO, ECP, and FIFO Test (ECR bits 7-5 = 110) modes. Parallel Port FIFO mode is for the Forward direction only; the direction bit is forced to 0 and PD7-0 are driven. ECP mode is for both the Forward and Reverse directions. The direction bit controls whether PD7-0 are driven. Automatic Run Length Encoding (RLE) decompression only is supported in the Reverse direction. FIFO Test mode is a test mode which allows data transfer to and from the FIFO, but not across the parallel port connector.

To improve noise immunity in ECP cycles, the state machine does not examine the control handshake response lines until the data has had time to switch. The amount of time the state machines use for this filtering function is programmable through the setup registers.

3.4.2.19.1 Host - Forward Write Cycle

When the ECP block is in Host mode Forward direction and the FIFO is not full (bit 1 of ECR is 0) the FIFO can be filled by software writes to the FIFO registers (AFIFO and DFIFO in ECP mode, and CFIFO in Parallel Port FIFO mode).

When DMA is enabled (bit 3 of ECR is 1 and bit 2 of ECR is 0) the ECP block automatically issues DMA requests to fill the FIFO with normal data byte.

When the ECP block is in Forward direction and the FIFO is not empty (bit 0 of ECR is 0) the ECP block reads a byte from the FIFO and issues write cycle to the peripheral device. The ECP block drives $\overline{\text{ATFD}}$ according to the operation mode (ECR bits 7-5) and according to the tag of the byte as follows: In Parallel Port FIFO mode (mode 010) $\overline{\text{ATFD}}$ is controlled by bit 1 of DCR. In ECP mode (ECR bits 7-5 = 011) $\overline{\text{ATFD}}$ is controlled by the tag. $\overline{\text{ATFD}}$ is driven high for normal data byte and driven low for command byte.

A Host-ECP mode write cycle starts when the ECP block drives the FIFO tag onto $\overline{\text{ATFD}}$ and the FIFO data byte onto PD7-0. When BUSY is low the ECP block asserts $\overline{\text{STB}}$. In Parallel Port FIFO mode the ECP block deasserts $\overline{\text{STB}}$ to terminate the write cycle. In ECP mode the ECP block waits for BUSY to be high.

When BUSY is high the ECP block deasserts $\overline{\text{STB}}$ then changes $\overline{\text{ATFD}}$ and PD7-0 only after BUSY is low.

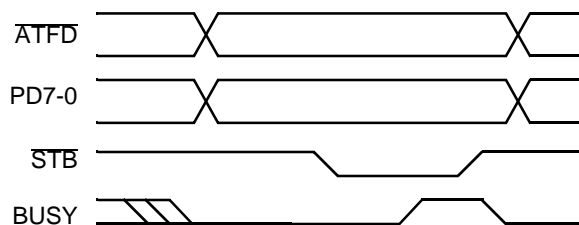


Figure 3-36 Host - Forward Write Cycle

3.4.2.19.2 Host - Reverse Read Cycle

An ECP read cycle starts when the FIFO indicates it is not full, and the ECP block drives $\overline{\text{ATFD}}$ low. The peripheral device drives BUSY high for a normal data read cycle, or drives BUSY low for a command read cycle, and drives the byte to be read onto PD7-0.

When $\overline{\text{ACK}}$ is asserted the ECP block drives $\overline{\text{ATFD}}$ high. When $\overline{\text{ATFD}}$ is high the peripheral device deasserts $\overline{\text{ACK}}$. The ECP block reads the PD7-0 byte, then drives $\overline{\text{ATFD}}$ low. When $\overline{\text{ATFD}}$ is low the peripheral device may change BUSY and PD7-0 states in preparation for the next cycle.

When the ECP block is in the Reverse direction, and the FIFO is not empty (bit 0 of ECR is 0), the FIFO can be emptied by software reads from the FIFO register (only DFIFO in ECP mode, no AFIFO or CFIFO read).

When DMA is enabled (bit 3 of ECR is 1 and bit 2 of ECR is 0) the ECP block automatically issues DMA requests to empty the FIFO.

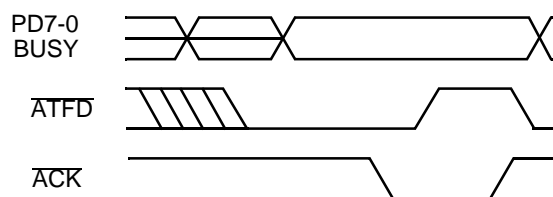


Figure 3-37 Host - Reverse Read Cycle

Notes:

1. FIFO-full condition is checked before every expanded write.
2. A pending DMA request is removed, and a pending RLE expansion is aborted, when you switch from Parallel Port FIFO or ECP modes to other modes.
3. The AFIFO, and the other FIFO ports are neither synchronized nor linked together, except via the empty and full FIFO status bits. The FIFO shall not delay the write and read operations, even when they are performed concurrently. Care must be taken not to corrupt PD7-0 or D7-0 while the other FIFO port is accessed.
4. In the Forward direction, the empty bit is updated when the ECP cycle is completed, not right after the last byte is read out of the FIFO (valid cleared on cycle end).

5. The one-bit command/data tag is used only in Forward direction.

3.4.2.19.3 Run Length Encoding Decompression

When the ECP block is in Host mode, Reverse direction, and the FIFO is not full (bit 1 of ECR is 0), the ECP block issues a read cycle from the peripheral device and monitors the BUSY signal. If BUSY is high the byte is a data byte and it is written into the FIFO. If BUSY is low the byte is a command byte. The ECP block checks bit 7 of the command byte, if it is high the byte is ignored, if it is low the byte is tagged as a Run-Length Count (RLC) byte (not written into the FIFO but used as a Run Length Count to expand the next byte received). Following an RLC read the ECP block issues a read cycle from the peripheral device to read the data byte to be expanded. This byte is considered a data byte, regardless of its BUSY state. This byte is written into the FIFO (RLC+1) times (i.e., RLC=0: write the byte once, RLC=127: write the byte 128 times).

3.4.2.20 Software Controlled Data Transfer (Slave - Standard Parallel Port and PS/2 modes)

Software controlled data transfer is supported in Standard Parallel Port and PS/2 modes. The software generates peripheral-device cycles by modifying the DATAR and DSR registers and reading the DSR, DCR and DATAR registers. The negotiation phase and nibble mode transfer, as defined in the IEEE 1284 standard, are performed in Standard Parallel Port mode, with the "Negotiation mode" bit in Setup register offset 0 set to 1.

In these modes the FIFO is reset (empty) and is not functional, the DMA and RLE functions are idle. Standard Parallel Port mode is for the Forward direction only; the direction bit is forced to 0 and PD7-0 are TRISTATED. PS/2 mode is for both the Forward and Reverse directions. The direction bit controls whether PD7-0 are driven, or TRISTATED for use as inputs.

IMPORTANT NOTE:

Since Standard Parallel Port mode transfers call for a data hold time from the rising edge of the \overline{STB} signal of 500ns (0.5us) an interrupt driven CPU would not normally have sufficient time to read the data prior to

the expiration of the hold time. In Standard Parallel Port mode, data from Forward transfers will be stored in the DATAR register, and the BUSY bit of the DSR will be cleared (causing the BUSY signal to go to 1), when the \overline{STB} signal is sampled low, and adjusted for noise filtering. When the filtered \overline{STB} signal transitions from 0 to 1, an Interrupt is generated. This IRQ is maskable with the DCR Interrupt Enable bit (bit 4)

3.4.2.21 Automatic Data Transfer (Slave Parallel Port FIFO and ECP modes)

Automatic data transfer (data transfers generated by hardware) is supported in Parallel Port FIFO and ECP mode. The ECP block will automatically generate the necessary handshaking signals for Parallel Port FIFO mode. In ECP mode, only the data transfer phases will be handled by hardware. For ECP mode, Software must handle all the Negotiation, Setup, and Termination phases.

Automatic DMA access to fill or empty the FIFO is supported in Parallel Port FIFO mode, ECP mode, and FIFO Test mode. Parallel Port FIFO mode is for the Forward direction only; the direction bit is forced to 0 and PD7-0 are TRISTATED. ECP mode is for both the Forward and Reverse directions. The direction bit controls whether PD7-0 are driven. Automatic Run Length Expanding (RLE) is supported in the Forward direction.

To improve noise immunity in ECP cycles, the state machine does not examine the control handshake response lines until the data has had time to switch. The amount of time the state machines use for this filtering function is programmable through the setup registers

3.4.2.21.1 Slave - Forward Write Cycles

When the ECP block is in Slave-Forward mode, and the FIFO is not full, the ECP block drives BUSY low, indicating it is ready to accept a cycle. The Host will then drive the data byte on the data pins, and drive the \overline{ATFD} signal to indicate what kind of data is being transferred. The Host will also drive \overline{STB} low, indicating data is valid.

When \overline{STB} transitions from low to high, the ECP block will write the byte on the connector data pins to one of three places. If the \overline{ATFD} signal is high, the

byte is a normal data byte, and the ECP block will write it into the FIFO. If the $\overline{\text{ATFD}}$ signal is low, the byte is a command byte.

Command bytes are handled in one of two ways. If the most significant data bit (PD7) is 0, then the byte is a RLC (Run-Length Count) which the ECP block will write into the RLE counter for data expansion. If the most significant data bit is 1, then the byte is a Channel Address.

When the ECP block recognizes a Channel Address transfer, it will generate an interrupt to the CPU. The state machines will then wait in the data transfer state, not driving BUSY high, until the Channel Address is read by the CPU. (i.e. the AFIFO is read) The Interrupt will be cleared automatically when the AFIFO is read.

Once the byte has been transferred either to the FIFO, or the RLE counter, or the Channel Address has been read from the AFIFO, the ECP block then drives BUSY high, indicating it is ready to terminate the transfer. When the Host drives $\overline{\text{STB}}$ high, terminating the transfer, the ECP block checks that the RLE counter is zero, and the FIFO is not full prior to driving BUSY low to indicate it is ready to continue receiving data.

When in ECP mode, and $\overline{\text{SLCTIN}}$ goes low, the Slave interprets that as a signal to terminate the session. At this point an interrupt is generated and software handles termination handshaking.

3.4.2.21.2 Slave - Reverse Read Cycles

When the ECP block is in Slave mode Reverse direction and the Host drives $\overline{\text{ATFD}}$ low, the ECP block waits for the FIFO to indicate it is not empty then drives data onto the data pins and sets the BUSY signal to the proper state. BUSY will be driven high for normal data transfers and low for transferring RLE Count (RLC) values. When the data byte and BUSY signals are driven the ECP block drives $\overline{\text{ACK}}$ low and waits for the Host to drive $\overline{\text{ATFD}}$ high, indicating the byte has been transferred. When $\overline{\text{ATFD}}$ is sampled high, the ECP block will drive $\overline{\text{ACK}}$ high, terminating the transfer. The ECP block will wait for the $\overline{\text{ATFD}}$ signal to again be driven low before it drives the next data on the connector pins.

3.4.2.22 FIFO Test Access

This mode is used to test DMA and PIO access and writes to the ECP block's 16 byte FIFO and is entered by setting ECR bits 7-5 to 110. Once in this mode, data can be read or written to the FIFO regardless of the direction bit in the DCR register.

In Host-Forward and Slave-Reverse directions the FIFO data is driven onto the Parallel Port data lines, however no other signals (e.g. $\overline{\text{STB}}$, $\overline{\text{ACK}}$) are generated.

This mode can be used to determine performance statistics, the FIFO depth and interrupt thresholds.

Reading an empty FIFO will not cause an underflow, and writing to a full FIFO will be ignored. The FIFO status bits in the ECR can be monitored to avoid these two conditions.

3.4.2.23 Configuration Registers Access

This mode is selected when ECR bits 7-5 = 111. The two configuration registers, CNFGA and CNFGB, are accessible only in this mode.

3.4.2.24 Interrupts

An interrupt is generated when any of the following events occurs in Host mode operations:

1. When bit 2 of ECR is 0, bit 3 of ECR is 1 and TC is asserted during DMA cycle.
2. When bit 2 of ECR is 0, bit 3 of ECR is 0, bit 5 of DCR is 0 and there are eight or more bytes free in the FIFO. It includes the case when bit 2 of ECR is cleared to 0 and there are already eight or more bytes free in the FIFO (Parallel Port FIFO mode, ECP mode, and FIFO Test mode only).
3. When bit 2 of ECR is 0, bit 3 of ECR is 0, bit 5 of DCR is 1 and there are eight or more bytes to be read from the FIFO. It includes the case when bit 2 of ECR is cleared to 0 and there are already eight or more bytes to be read from the FIFO (ECP and FIFO Test modes only).
4. When bit 4 of ECR is 0 and $\overline{\text{FAULT}}$ is asserted (high to low edge) or $\overline{\text{FAULT}}$ is asserted when bit 4 of ECR is modified from 1 to 0.
5. When bit 4 of DCR is 1 and $\overline{\text{ACK}}$ is deasserted (low-to-high edge).

Note: Interrupt events #2, #3 and #4 are level

events, thus they are shaped as interrupt pulses. These interrupts are masked (inactive) when the ECP clock is frozen. Interrupt event #1 is a pulse event. The last interrupt event behaves as in the Standard Parallel Port mode: the IRQ signal follows the $\overline{\text{ACK}}$ signal transition. Note that interrupt event #4 may be lost when the ECP clock is frozen.

An interrupt is generated when any of the following events occurs in Slave mode operations:

1. When bit 2 of ECR is 0, bit 3 of ECR is 1 and TC is asserted during DMA cycle.
2. When bit 2 of ECR is 0, bit 3 of ECR is 0, bit 5 of DCR is 1 and there are eight or more bytes free in the FIFO. It includes the case when bit 2 of ECR is cleared to 0 and there are already eight or more bytes free in the FIFO (ECP, and FIFO Test modes only).
3. When bit 2 of ECR is 0, bit 3 of ECR is 0, bit 5 of DCR is 0 and there are eight or more bytes to be read from the FIFO. It includes the case when bit 2 of ECR is cleared to 0 and there are already eight or more bytes to be read from the FIFO (Parallel Port FIFO mode, ECP mode, and FIFO Test mode only).
4. When in ECP mode, and a Channel Address is received from the Host. This interrupt condition will clear itself when the AFIFO is read.
5. When in ECP mode, $\overline{\text{SLCTIN}}$ transitions from high to low, indicating the end of 1284 transfer mode and the beginning of the termination handshaking done in software. This interrupt condition will clear itself when the DCR is read.
6. When in ECP mode, $\overline{\text{INIT}}$ transitions from either high to low or low to high. This interrupt condition will clear itself when the DCR is read.
7. When in Parallel Port FIFO mode, and the Host indicates a negotiation phase by driving $\overline{\text{SLCTIN}}$ high, and $\overline{\text{ATFD}}$ low. $\overline{\text{SLCTIN}}$ and $\overline{\text{ATFD}}$ must be in these states for the number of clocks defined for $\overline{\text{ATFD}}$ noise filtering in SETUP register offset 3. This interrupt condition will clear itself when the DCR is read.
8. When in Standard Parallel Port mode, and the Host causes the $\overline{\text{STB}}$ signal to transition from 0 to 1.

3.4.3 The PCMCIA Interface

The NS486SXF PCMCIA interface supports the direct connection of a single PCMCIA 2.0 IC card. Exchange Card Architecture (ExCA release 1.50) compatibility and eExecute In Place (XIP) capability is also provided. For documentation of registers and operation not detailed in this section, please refer to the Intel 82365SL PC Card Interface Controller (PCIC) Datasheet.

Accessing the PCMCIA interface switches the I/O bus automatically into the PCMCIA mode and permits Memory Window Mapping and Address Offsets to be handled inside the NS486SXF device. Should power management and/or “hot” card insertion/removal options be required, external buffering is required.

If the PCMCIA block is being used, two chip selects (CS1 and CS2) and IRQ5 become PCMCIA pins. Therefore those functions are not available when using the PCMCIA controller.

The NS486SXF provides 13 control and status signals to interface to a single PCMCIA Card Slot (Slot A), 26 address lines and 16 data lines as shown in the figure.

The NS486SXF programming interface and register set is exactly compatible with the Intel82365SL PC Card Interface Controller with the following exceptions:

- 1) The NS486SXF supports only a single PCMCIA Card Slot (A).
- 2) The NS486SXF supports only a 26-bit wide external address space.
- 3) There is no support for the 82365SL power control scheme.

Table 3-8 lists the NS486SXF PCMCIA register set and their addresses. The base address for the PCMCIA registers is set in Bus Interface Unit Control Register 2.

Figure 3-38 PCMCIA Interface

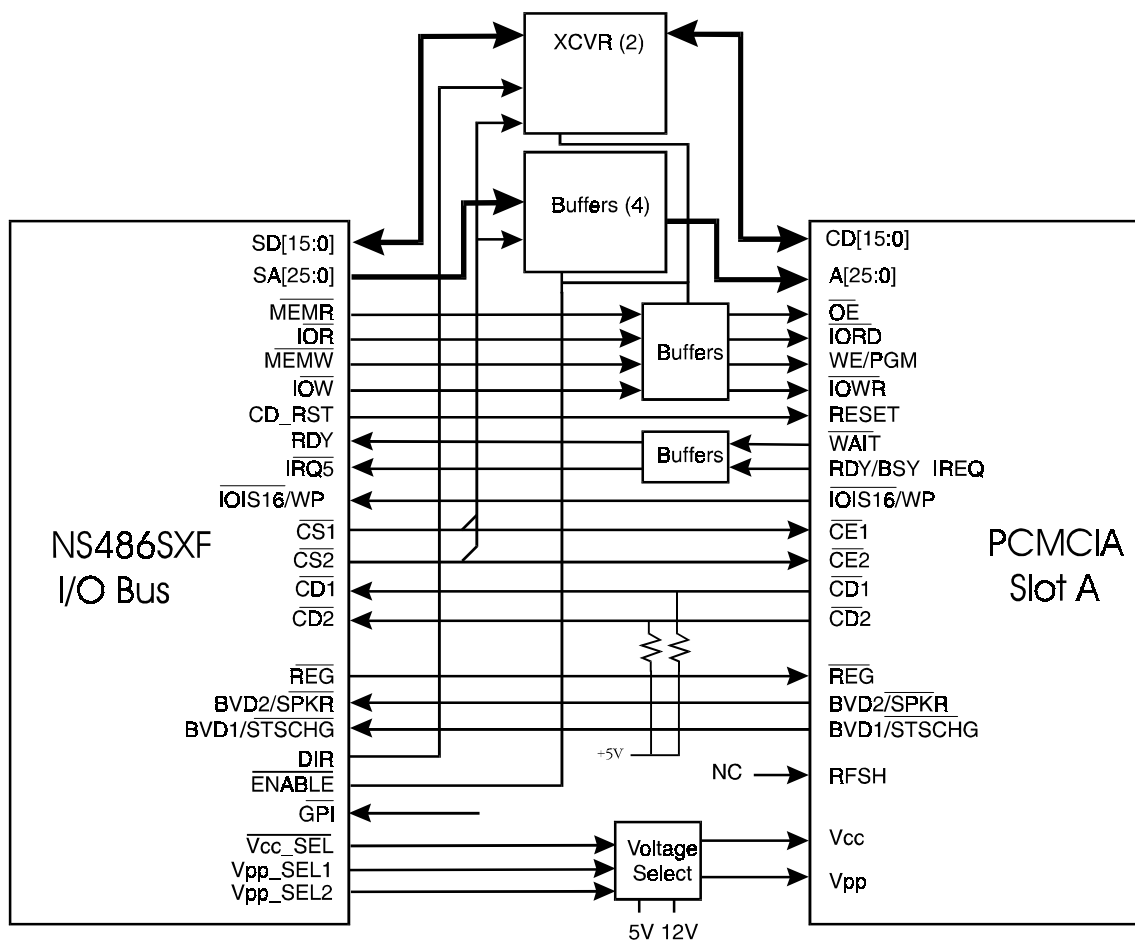


Table 3-8: NS486SXF PCMCIA Registers

Address	Register Name
GENERAL SETUP REGISTERS	
Base + 00h	Identification and Revision
Base + 01h	Interface Status
Base +02h	Power and RESETDRV Control
Base +04h	Card Status Change
Base +06h	Address Window Enable
Base +16h	Card Detect and General Control Register
Base +1Eh	Global Control Register
INTERRUPT REGISTERS	
Base +03h	
Base +05h	
I/O REGISTERS	
Base +07h	I/O Control
Base +08h	I/O Address 0 Start Low Byte
Base +09h	I/O Address 0 Start High Byte
Base +0Ah	I/O Address 0 Stop Low Byte
Base +0Bh	I/O Address 0 Stop High Byte
Base +0Ch	I/O Address 1 Start Low Byte
Base +0Dh	I/O Address 1 Start High Byte
Base +0Eh	I/O Address 1 Stop Low Byte
Base +0Fh	I/O Address 1 Stop High Byte
MEMORY REGISTERS	
Base +10h	System Memory Address 0 Mapping Start Low Byte
Base +11h	System Memory Address 0 Mapping Start High Byte
Base +12h	System Memory Address 0 Mapping Stop Low Byte
Base +13h	System Memory Address 0 Mapping Stop High Byte
Base +14h	Card Memory Offset Address 0 Low Byte
Base +15h	Card Memory Offset Address 0 High Byte
Base +17h	Reserved
Base +18h	System Memory Address 1 Mapping Start Low Byte

Address	Register Name
Base +19h	System Memory Address 1 Mapping Start High Byte
Base + 1Ah	System Memory Address 1 Mapping Stop Low Byte
Base + 1Bh	System Memory Address 1 Mapping Stop High Byte
Base +1Ch	Card Memory Offset Address 1 Low Byte
Base +1Dh	Card Memory Offset Address 1 High Byte
Base +1Fh	Reserved
Base +20h	System Memory Address 2 Start Low Byte
Base +21h	System Memory Address 2 Start High Byte
Base +26h	Reserved
Base +27h	Reserved
Base +28h	System Memory Address 3 Mapping Start Low Byte
Base +29h	System Memory Address 3 Mapping Start High Byte
Base +2Ah	System Memory Address 3 Mapping Stop Low Byte
Base +2Bh	System Memory Address 3 Mapping Stop High Byte
Base +2Ch	Card Memory Offset Address 3 Low Byte
Base +2Dh	Card Memory Offset Address 3 High Byte
Base +2Eh	Reserved
Base +2Fh	Reserved
Base +30h	System Memory Address 4 Mapping Start Low Byte
Base +31h	System Memory Address 4 Mapping Start High Byte
Base +32h	System Memory Address 4 Mapping Stop Low Byte
Base +33h	System Memory Address 4 Mapping Stop High Byte
Base +34h	Card Memory Offset Address 4 Low Byte
Base +35h	Card Memory Offset Address 4 High Byte

3.4.3.1 PCMCIA Address Register

The base address page of the PCMCIA card memory space must be programmed. The memory space can be located on any 2^{24} page boundary. See Section 4.0 The System Bus for details.

3.4.3.2 PCMCIA Clock Selection Register

The PCMCIA controller requires a SYSCLK, typically near 8MHz. This clock selection register allows for choosing the SYSCLK rate as a divided down clock from the CPU clock. See Section 4.0 The System Bus for details.

3.4.3.3 PCMCIA Interrupt Selection Registers

3.4.3.3.1 Interrupt and General Control Register (Base Address + 03h)

Note that the $\overline{\text{INTR}}$ Enable bit documented in the Intel datasheet is used to enable the Card Status Change Interrupt onto internal interrupt request 6. If $\overline{\text{INTR}}$ Enable is set to a 0 then the Card Status Change Interrupt Configuration Register settings are used to steer the Status Change interrupt. For the $\overline{\text{INTR}}$ interrupt to actually drive the internal interrupt request 6, you must program bit 0 in the register documented by Section 3.3.5.4.3 “Miscellaneous Interrupt Selection Register 3”.

Note that for the PC I/O Card Interrupt Level routing (bits 3 through 0 in this register) to work, the appropriate enable must be set in Section 3.3.5.4 “Miscellaneous (PCMCIA and Extended Capabilities Port (ECP)) Interrupt Selection Registers”.

3.4.3.3.2 Card Status Change Interrupt Configuration Register (Base Address + 05h)

The routing bits in this register (bits 3 to 0) are ignored if the $\overline{\text{INTR}}$ Enable bit is set in the Interrupt and General Control Register documented above. Note that for the Card Status Change Interrupt Level routing (bits 3 through 0 in this register) to work, the appropriate enable must be set in Section 3.3.5.4 “Miscellaneous (PCMCIA and Extended Capabilities Port (ECP)) Interrupt Selection Registers”.

3.4.4 The MICROWIRE or Access.bus Interface

The NS486SXF MICROWIRE/Access.bus interface provides for full support of both the three wire MICROWIRE™ and the two wire Access.bus synchronous serial interfaces. These industry standard interfaces permit easy interfacing to a wide range of low-cost specialty memories and I/O devices. These include EEPROMs, SRAMs, timers, A/D converters, D/A converters, clock chips, and peripheral drivers.

The MICROWIRE or Access.bus interfaces share the use of two NS486SXF pins, so only one of them may be selected at any time. The two pins shared by the two interfaces are:

SCLK (pin #43), which both the MICROWIRE and Access.bus interfaces use as the serial clock.

SI (pin #42), which MICROWIRE uses as a serial input data pin, while Access.bus uses it as its I/O data pin.

The MICROWIRE interface requires a third pin:

SO (pin #41), which is used as a serial output data pin.

When both the MICROWIRE and Access.bus interfaces are disabled, these three pins may be used as modem control lines ($\overline{\text{DCD}}$, $\overline{\text{CTS}}$ and RI).

3.4.4.1 MICROWIRE Interface

MICROWIRE is a three-wire synchronous serial bus. All data transfers are synchronous with the serial clock (SCLK) and data is read into the NS486SXF MICROWIRE interface on the serial input pin (SI). Data written out will be driven on the serial output pin (SO).

Several MICROWIRE devices can be connect to the same three-wire system, as shown in Figure 3.38. One, and only one of these devices operates in what is called master mode and supplies the synchronous serial clock (SCLK) signal for the entire system. The master is also responsible for initiating all data transfers. All other MICROWIRE devices must operate in

slave mode, where SCLK is an input signal and the slave device provides(receives) the serial data for read(write) cycles from(to) it. The slave device uses the master's SCLK for serially shifting data out(in), while the master device shifts the data in(out).

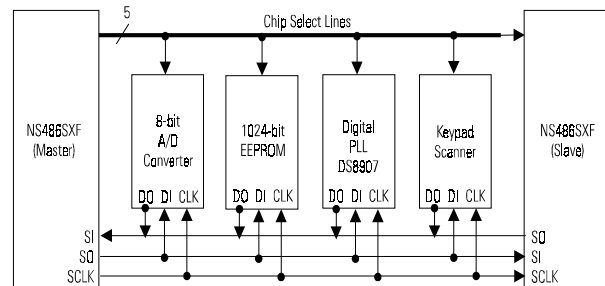


Figure 3-39 A MICROWIRE System Example

The NS486SXF MICROWIRE interface may be configured to operate in either master or slave mode. When the Configuration bit (CONFIG, bit 1 in the MICROWIRE Control Register) is a one, the NS486SXF MICROWIRE interface will be configured in master mode; when CONFIG is a zero, the NS486SXF MICROWIRE interface will operate in slave mode.

When there is only a single MICROWIRE slave connected to the MICROWIRE master, the slave's chip select signal can be tied active since all MICROWIRE accesses should be to the slave device.

When more than one MICROWIRE slave is connected to the MICROWIRE master, each slave's chip select should be driven with an unique chip select signal. When configured as the MICROWIRE master, the NS486SXF MICROWIRE interface does not have any chip select logic designed into it. Instead it is suggested that the system designer configure the appropriate number of Reconfigurable I/O (RIO) pins to provide the necessary chip selects for the MICROWIRE slaves. These chip selects would then be under software control via these general purpose RIO pins and the software would be responsible for selecting the appropriate device for each MICROWIRE

transfer. See Section 3.4.1 on page 103 for information on programming the RIO pins.

When the **NS486SXF** MICROWIRE interface is configured in slave mode, one of the RIO pins will operate as the MICROWIRE interface chip select input pin. The pin which will act as the input MICROWIRE chip select will be selected by the MICROWIRE Slave Chip Select Register. See “MICROWIRE Slave Chip Select Register” on page 141.

NOTE: In slave mode, the input MICROWIRE chip select is an active low signal, so the selected pin must be driven active low to select the **NS486SXF** MICROWIRE interface as a slave.

3.4.4.2 MICROWIRE Transfers

An 8-bit shift register, called Serial Input/Output (SIO) register, is used for both transmitting and receiving data. For either type of transfer, the bits of SIO are shifted left through the register. When the data byte is being transmitted, the bits are shifted out through the SO output pin (most significant bit first). When a data byte is being received, the bits are shifted in through the SI input pin (most significant bit first).

3.4.4.2.1 MICROWIRE Master Transmit/Receive

The **NS486SXF** MICROWIRE interface performs a transmit and receive at the same time. In master mode, the following steps will generate an 8-bit transfer:

- 1) Software must enable the MICROWIRE interface and configure it appropriately via the MICROWIRE Control Register.
- 2) If the **NS486SXF** MICROWIRE interface will be used to transmit data out, software must write the desired transmit data into the SIO register. If the **NS486SXF** MICROWIRE interface is only receiving data, one can ignore this step.
- 3) Software must set the BUSY bit in the MICROWIRE Control Register to a one. This will start the transfer.
- 4) At the end of the transfer, the MICROWIRE Interrupt Flag bit (uWI, bit 0 of the MICROWIRE Control Register) will be set to a one, an interrupt will be generated and the BUSY bit will be cleared to a zero. For re-

ceive transfers, the software should read the SIO register; this will clear both the interrupt request and the MICROWIRE Interrupt Flag bit. The other way to clear the interrupt request and uWI is to write a one to the BUSY bit; this is the typical method for multiple back to back Master transmit cycles.

- 5) If the next MICROWIRE transfer is a transmit go to Step 2. If the next MICROWIRE transfer is a receive, proceed to Step 3.

In master mode, whenever the BUSY bit is a zero, SCLK will remain low. Once the BUSY bit is set to a one, the MICROWIRE interface will proceed to generate eight periodic clock pulses on SCLK. For normal SCLK mode, on the rising edge of SCLK, the MICROWIRE interface will shift in the data presented on the SI pin and on the falling edge, the next bit in the SIO register will be shifted out onto the SO pin as depicted in Figure 3.39.

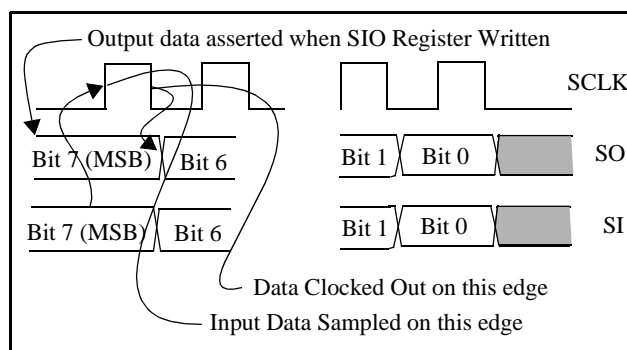


Figure 3-40 MICROWIRE Normal SCLK Mode Timing Diagram

The **NS486SXF** MICROWIRE interface also supports an alternate SCLK clocking mode. The timing of the serial transfer in alternate SCLK mode is slightly different. In alternate SCLK mode, the SIO register data being shifted out onto the SO pin occurs on the rising edge of SCLK. While the data being shifted in off of the SI pin, will be shifted in on the falling edge of SCLK. The timing of alternate SCLK mode is shown in Figure 3.40. The SCLK mode is determined

by the SCTL bit (bit 2 of the MICROWIRE Control Register).

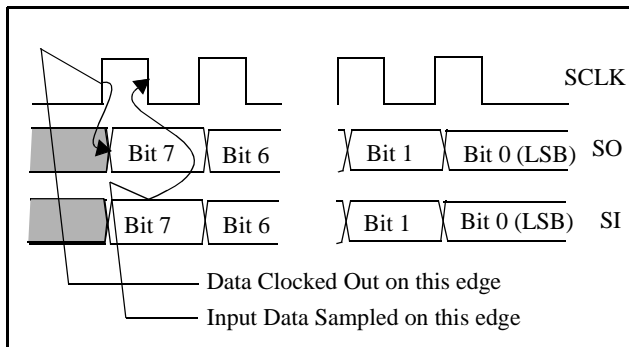


Figure 3-41 MICROWIRE Alternate SCLK Mode Timing Diagram

At the end of the transfer, the data shifted in on the SI pin will be readable via the SIO register and the BUSY bit will be cleared; thus SCLK will remain low until the BUSY bit is set to a one again. Also, the MICROWIRE transmit/receive interrupt flag (uWI, bit 0 of the MICROWIRE Control Register) will be set and an interrupt request will be generated.

3.4.4.2.2 MICROWIRE Slave Transmit/Receive

When the NS486SXF MICROWIRE interface is configured in slave mode, it must still perform the two steps required for a master transfers.

However, there are two major differences from master mode. First, the NS486SXF MICROWIRE interface will not respond unless its chip select pin is driven active low and second, the NS486SXF MICROWIRE interface will not generate the SCLK pulses. Instead, in slave mode the MICROWIRE interface must wait for the MICROWIRE master to generate these SCLK pulses and present (receive) the data appropriately.

At the end of a transmit cycle, the BUSY bit will be cleared to a zero, the uWI bit will be set and an interrupt generated. It is the responsibility of the software to guarantee the BUSY bit is set and the appropriate transmit data is written to the SIO register before the MICROWIRE master attempts to read from the NS486SXF MICROWIRE interface again. If the MICROWIRE master attempts another transfer before the BUSY bit is set, the NS486SXF MICROWIRE interface will not provide the appropriate data. This is

an overall system software issue, which is beyond the scope of this document other than to note the issue.

At the end of the receive, the BUSY bit will be cleared to a zero, the uWI bit will be set and an interrupt generated. It is the responsibility of the software to guarantee that the SIO register is read and the BUSY bit is set before the MICROWIRE master attempts to transmit to the NS486SXF MICROWIRE interface again. If the MICROWIRE master attempts another transfer before the BUSY bit is set, the NS486SXF MICROWIRE interface will not clock in the appropriate data. User software should make sure that this case does not arise.

NOTE: In slave mode, it is important that the software programs the SL1-SL0 and Con2-Con0 bits to select a serial clock frequency greater than or equal to the transfer rate used on the MICROWIRE interface. A serial clock frequency of 1-5 MHz is suggested. Even though this serial clock will not drive the SCLK signal, it is used internally in the NS486SXF MICROWIRE interface to synchronize signals. If the internal serial clock is not programmed to meet this requirement, the MICROWIRE interface may not work as desired, in slave mode.

3.4.4.3 The Access.Bus Interface

Access.bus is a two wire interface that allows bidirectional communications between ICs. The two interface lines are the serial data line (SI), and the serial clock line (SCLK). Access.bus system configuration examples are shown in Figures 3.41 and 3.42. As illustrated, Access.bus supports multiple masters and multiple slaves.

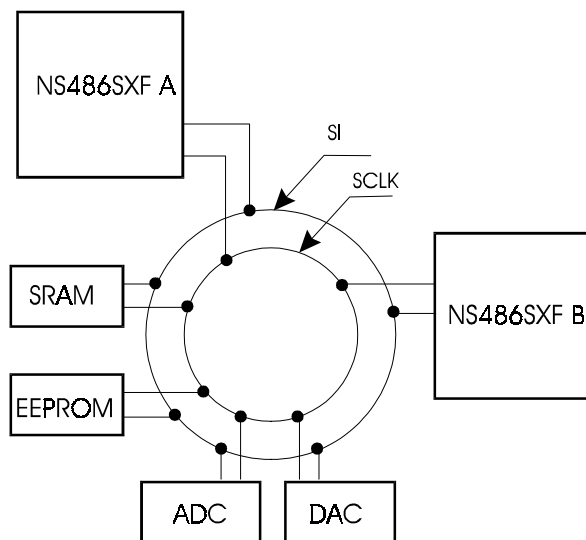


Figure 3-42 A Sample Access.bus System

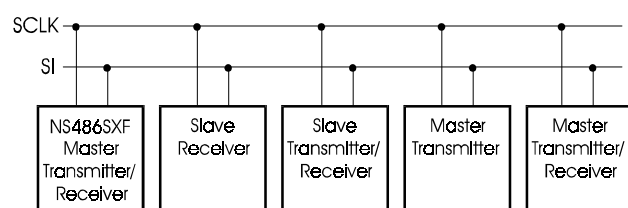


Figure 3-43 Multiple Resources

The Access.bus protocol includes software addressing and data transfer protocols in addition to hardware definitions.

The two serial lines (SI and SCLK) should be connected to a positive supply via a pull-up resistor and are to remain HIGH when the bus is not busy. Each device has a unique address and can operate as a transmitter and/or a receiver. (Note that some peripherals will only be receivers, and most controllers (in-

cluding the NS486SXF) will be both transmitters and receivers.)

All Access.bus devices must drive the SI and SCLK lines with open-collector or open-drain drivers. In this manner, multiple devices may share these signals without any signal contention.

During data transfers, a device can be either a master or a slave. The initiating device is considered the master, it also generates the clock (SCLK) and the start condition for the transfer. The addressed device is considered the slave during the transfer. When the NS486SXF is initiating a data transfer with an attached Access.bus peripheral, it is the master and the transmitter. However, when the Access.bus peripheral in question responds and sends data to the NS486SXF, then the peripheral is the transmitter (even though it remains the slave) and the NS486SXF is the receiver (even though it remains the master). In other words, the master can be both a transmitter and a receiver. Likewise, a slave can be both a transmitter and a receiver. The key is that the initiator is the provider of the clock signal (SCLK) and is considered the present Access.bus master.

As shown in Figures 3.41 and 3.42, it is possible to have more than one master on the bus. Because of this, an arbitration protocol has been included in the Access.bus implementation. The arbitration protocol depends on a wired-AND connection to the devices on the bus and clock synchronization. If two or masters attempt to start a transfer at the same time, the master transmitting to the lowest address will win the arbitration.

3.4.4.4 Access.bus Data Transfers

Data is transferred during the high state of the serial clock (SCLK). Data can only change during the low state of SCLK. This is shown in Figure 3.43.

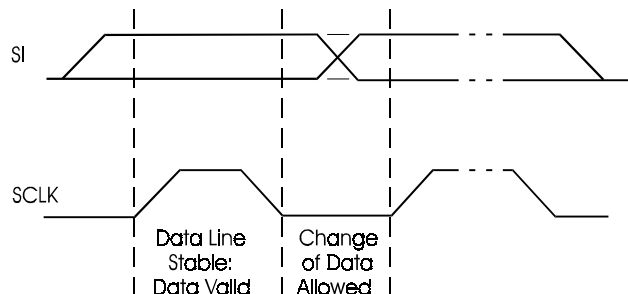


Figure 3-44 Bit Transfer

One data bit is transferred during each clock pulse. The data on the SI line must remain stable during the HIGH period to be valid. Transitions on SI during the entire high period of SCLK are interpreted as control signals. In this manner, a single data line is used to transfer both command/control information as well as data.

A high-to-low transition on SI while SCLK is high indicates a START condition as shown in Figure 3.44.

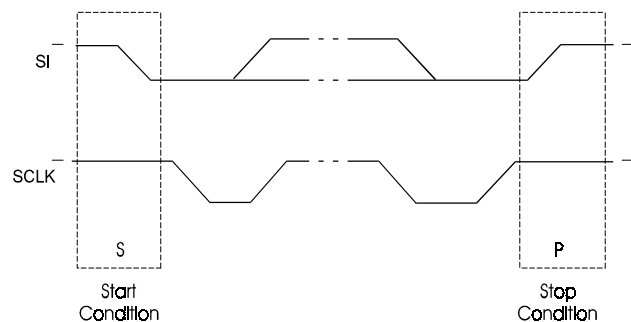


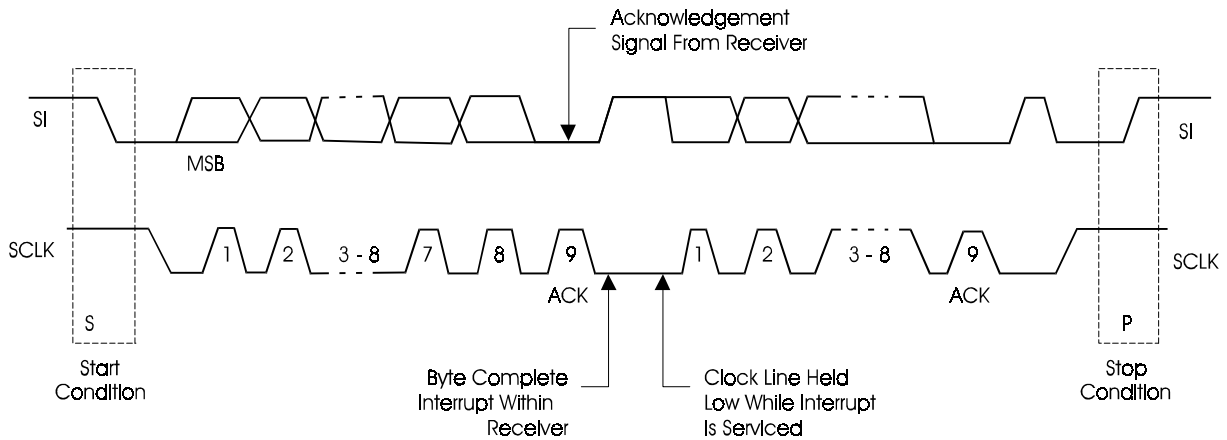
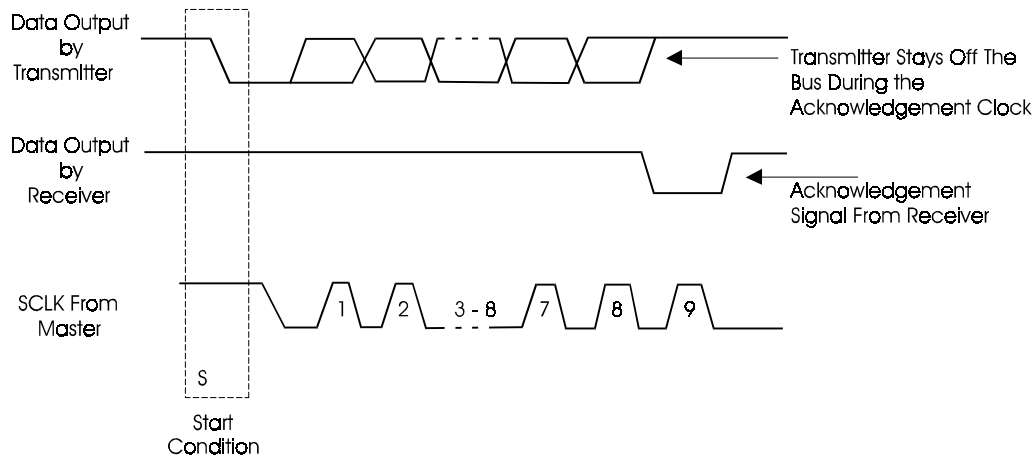
Figure 3-45 Start and Stop Conditions

Similarly, a low-to-high transition on SI while SCLK is high indicates a STOP condition.

The bus is considered to be busy after the START condition and free again after a certain time interval after the STOP condition. START and STOP conditions are generated by the present Access.bus master.

During a data transfer, the smallest unit of transfer is a byte. Any number of data bytes can be transferred in

a given transfer sequence. Each byte is transferred with the most significant bit first and the least significant bit last. After the transfer of the least significant bit of each byte, an Acknowledge bit must be generated by the receiving device; as shown in Figure 3.45. During the Acknowledge bit time, the transmitting device stops driving SI and the receiving device drives SI low. If the receiving device does not drive SI low during the Acknowledge bit time, SI will be pulled high by the pull-up resistor and the cycle will not be acknowledged.

Figure 3-46 Access.bus Data Transfer**Figure 3-47 Access.bus Acknowledge Cycle**

A slave receiver must generate an acknowledge after the reception of each byte, and a master must generate an acknowledge after the reception of each byte clocked out of the slave transmitter. If the receiving device cannot receive the data immediately, it can force the transmitter into a wait state by holding SCLK low. This can happen when the receiver is busy with some peripheral related task or is otherwise occupied with higher order system tasks. When this happens, the master, after a time-out of 2-3 msec., will abort the transfer and go to the idle state. This will also result in setting the Overflow bit in the MICROWIRE Control Register to a one. This type of cycle may also indicate that a device on Access.bus is in an unrecoverable condition and needs to be reset. User software should ensure that this condition is detected and resolved.

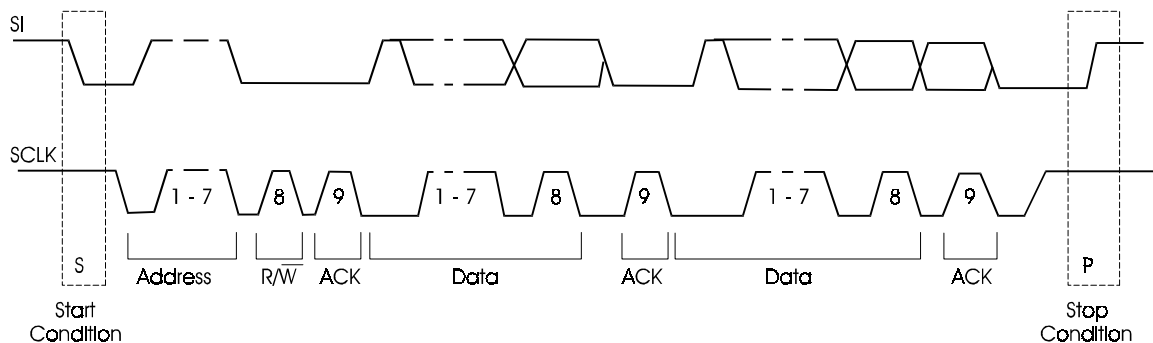
There are two exceptions to the “acknowledge after every byte” rule. The first is when the master is the receiver. It must signal an end of data to the transmitter by NOT generating an acknowledge to the last byte that is to be clocked out of the slave. This “negative acknowledge” still includes a SCLK pulse, (generated by the master), but SI will be pulled up by the pull-up resistor. The second exception is when a slave sends a negative acknowledge to indicate that it can no longer accept additional data bytes. This may occur when the receiver FIFO is full, the receiver is otherwise occupied or the receiver is recovering from an error condition. If the slave receiver sends a negative acknowledge, the master will abort the transfer by generating the STOP condition.

3.4.4.5 Addressing Transfer Formats

Each device on the Access.bus must have a unique address. Before any data is transmitted, the master transmits the address of the slave being addressed. A slave device, once it recognizes its address, acknowledges the address. The **NS486SXF** Access.bus address is contained in the Own Address Register and may be modified via software. The **NS486SXF** Access.bus interface will also acknowledge an address of all zeros (called a “general call”), as required of all enabled Access.bus devices.

The address is the first seven bits after a START condition. The eighth bit is the direction (read = 1/ write=0, R/ \overline{W}) indicator. A complete data transfer is shown in Figure 3-48 and shows the start condition, followed by a seven bit address, a one bit R/ \overline{W} indicator, a one bit Acknowledge, followed by the first data byte, a one bit Acknowledge and so on. A low-to-high transition on SI during a SCLK high period indicated the STOP condition and ends the transfer.

Figure 3-48 A Complete Access.bus Data Transfer

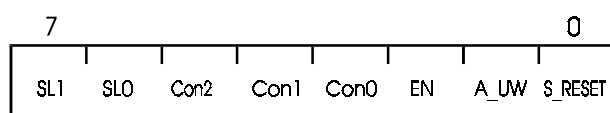


When the address is sent, each device connected to the Access.bus compares its address with the transmitted address. If there is a match, the device will consider itself addressed, and will generate an acknowledge. Depending upon the state of the R/ \overline{W} bit (read=1, write=0), the device will act as a transmitter (send data) or a receiver (receive data), respectively.

3.4.4.6 MICROWIRE/Access.bus Control Register (MACON)

The MICROWIRE interface supports a 0 - 1 MHz serial clock frequency and the Access.bus interface supports a 0-100KHz serial clock frequency. Bits 7-3 of this register may be programmed to generate these required serial clock frequencies.

Meanwhile, bits 2-0 allow the enabling of one of these interfaces, the selection of MICROWIRE or Access.bus and a soft reset bit, respectively.



I/O Map Address

0050h

Access

R/W

Bits 7-6: SL1,2 — Selects the input OSCX1 clock divided by 8, 16, 32, or 64, as the prescaled serial clock.

Bits 5-3: Con2-0 — The prescaled serial clock generated by SL1 and SL2 will be further divided by (the value in these three bits plus one, all multiplied by 2). The result will be the serial clock to be used by either the MICROWIRE interface or the Access.bus interface (whichever is selected).

SL0 and SL1 are two select signals used to select a prescaled clock equal to the input OSCX1 clock divided by 8, 16, 32 or 64. This prescaled clock goes to a counter controlled by Con0, Con1 and Con2. With these five select bits, one can chose the OSCX1 clock divided by 16, 24, 32, 40, 48, 56, 64, 80, 96, 112, 128, 160, 192, 224, 256, 320, 384, 448, 512, 640, 768, 896 and 1024 as the clock source for either the MICROWIRE interface or the Access.bus interface (whichever is selected).

SL1	SL0	Input OSCX1 Clock divided by
0	0	8
0	1	16
1	0	32
1	1	64

Con0, Con1, Con2, SL1, and SL0 should be programmed according to the frequency of the input OSCX1 clock. These five bits determine frequency of the clock used in this block, and are used to optimize performance at differ oscillator speeds.

Con2	Con1	Con0	SL1 and SL0 selected prescaled Clock divided by
0	0	0	2
0	0	1	4
0	1	0	6
0	1	1	8
1	0	0	10
1	0	1	12
1	1	0	14
1	1	1	16

- Bit 2: EN - When this bit is a one, it enables either the MICROWIRE interface or the Access.bus interface (whichever is selected by bit 1, A_UW). When this bit is a zero, neither the MICROWIRE interface nor the Access.bus interface will be enabled. When this bit is a zero, the SCLK, SI and SO pins may be used as modem control pins (DCD, CTS and RI).
- Bit 1: A_UW — When this bit is a zero the MICROWIRE interface is selected. When this bit is a one the Access.bus interface is selected.
- Bit 0: S_RESET - Soft reset. When this bit is written to a one, it will reset the selected interface logic. Software must write a zero to this bit to clear it once it has been set to a one.

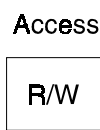
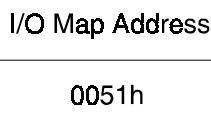
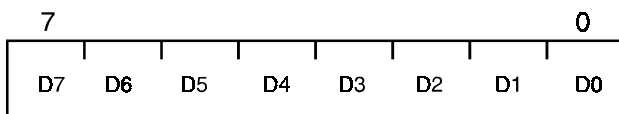
3.4.4.7 MICROWIRE Registers

MICROWIRE is a synchronous serial three-wire interface communication system that allows the NS486SXF to communicate with any other device that also supports the MICROWIRE interface.

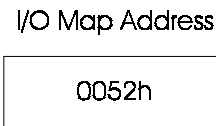
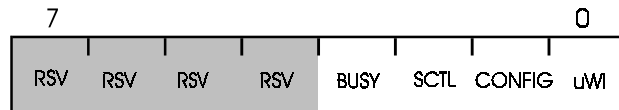
There are three 8-bit MICROWIRE registers:

3.4.4.7.1 Serial Input/Output Register (SIO)

An 8-bit shift register, called the SIO (Serial Input/Output) register, is used for both transmitting and receiving data. In MICROWIRE systems, the most significant bit is transmitted(received) first and the least significant bit is transmitted(received) last. The NS486SXF CPU reads and writes to the SIO Register via an 8-bit parallel data bus.



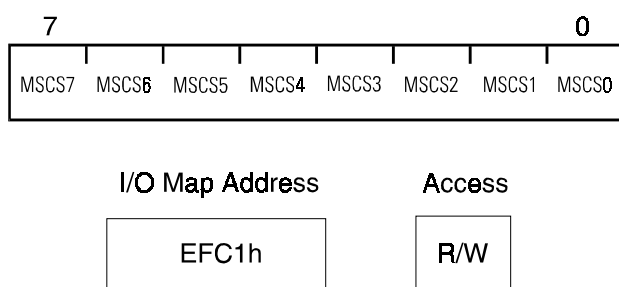
3.4.4.7.2 MICROWIRE Control Register (uWCON)



- Bits 7-4: Reserved.
- Bit 3: BUSY — Software must set this bit to a one before the MICROWIRE interface can transmit or receive data. This bit will be cleared to a zero by hardware at the end of a MICROWIRE transfer cycle. While this bit is set to a one, it indicates that the MICROWIRE shift register is busy shifting data out/in.
- Bit 2: SCTL — SCLK mode ConTroL. When this bit is a zero (standard SCLK mode), the output data on SO is clock out on the falling edge of SCLK and the input data on SI is sampled on the rising edge of SCLK.
When this bit is a one (alternate SCLK mode), the output data on SO is clock out on the rising edge of SCLK and the input data on SI is sampled on the falling edge of SCLK.
- Bit 1: CONFIG — Configuration bit. When one, the MICROWIRE interface will operate in Master mode. When zero, the MICROWIRE interface operates in slave mode.
- Bit 0: uWI — MICROWIRE transmit/receive interrupt flag. This bit is set to a one at the end of SIO register shifting. This bit will be cleared to a zero by reading the SIO register or when the BUSY bit is set to a one. (This is a read only bit.)

3.4.4.7.3 MICROWIRE Slave Chip Select Register

The eight bits in this register select which pin will be used as the input MICROWIRE chip select, when the NS486SXF MICROWIRE interface is in slave mode. When the NS486SXF MICROWIRE interface is in master mode, the bits in this register will have no function. It is suggested that only one pin be selected as the input MICROWIRE chip select, but if more than one is selected, then an active low on any one of the selected pins will enable the NS486SXF MICROWIRE interface.



Reset Value: 00000000h

Bits 7: MSCS7-MSCS0 — MICROWIRE Slave Chip Selection bits 7-0. When a one is written to a bit in this register it will select the corresponding pin listed below as the input MICROWIRE chip select. The input MICROWIRE chip select is an active-low signal which must be driven to a zero to select the NS486SXF MICROWIRE interface, when it is in slave mode.

Bit	Pin used as the input MICROWIRE chip select
7	REG
6	CLF
5	PD[7]
4	Rx
3	$\overline{CS}[4]$
2	$\overline{CS}[3]$
1	$\overline{CS}[2]$
0	$\overline{CS}[1]$

Note: User software must also configure the chosen pin as a RIO input as well.

3.4.4.7.4 Access.bus Serial Interface

The two wires, serial data(SI) and serial clock (SCLK), carry information between the devices connected to the Access.bus. Each device is recognized by a unique address and can operate as either a transmitter or receiver, depending on the function of the device. It is a multi-master bus.

Four modes are supported:

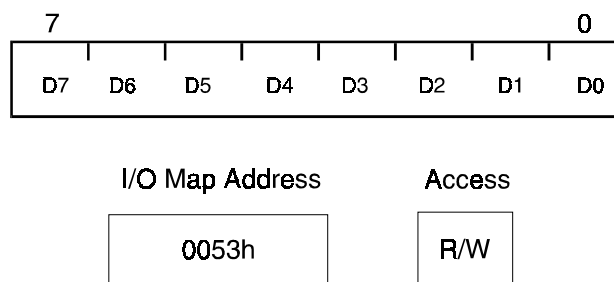
- 1) Master Transmit
- 2) Master Receive
- 3) Slave Transmit
- 4) Slave Receive

There are five registers inside the Access.bus module:

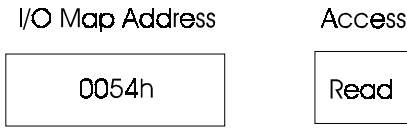
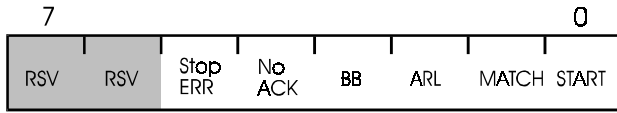
- 1) Serial Input/Output Data Register (SDA)
- 2) Access.bus Status Register
- 3) Access.bus Control Register 1
- 4) Access.bus Control Register 2
- 5) Own Address Register

3.4.4.7.5 Serial Input/Output DATA Register (SDA)

An 8-bit shift register, called the Serial Input/Output register (SDA), is used for both transmitting and receiving data. The most significant bit is transmitted(received) first and the least significant bit is transmitted(received) last. The CPU reads and writes SDA via an 8-bit parallel data bus at I/O address 0053h.



3.4.4.7.6 Access.bus Status Register

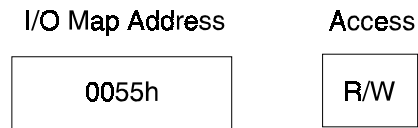


Reset Condition: 00h; all bits are set by hardware.

- Bits 7,6: Reserved
- Bit 5: Stop_ERR — Premature Stop Error Flag. This bit will be set to a one whenever a STOP condition is detected in the middle of a data transfer phase. When this error flag is set to a one, it indicates a possible system problem which must be addressed by the system software. It can be cleared by system reset, soft reset a START condition and by software writing a “1” to this bit.
- Bit 4: No_ACK — No Acknowledge Error flag. This bit will be set to a one, in Master mode when a transmission is not acknowledged on the ninth clock. It can be cleared by system reset, soft reset a START condition and by software writing a “1” to this bit.
- Bit 3: BB — Bus Busy flag. When set to a one, it indicates that the Access.bus is currently busy. It is set by a START condition and cleared by Reset or a STOP condition. (This is a read only bit.)
- Bit 2: ARL — Arbitration Loss flag. When this bit is set to a one, it indicates that this device lost arbitration while trying to take control of Access.bus. It is cleared by system reset, soft reset, a STOP condition or software writing a “1” to this bit.
- Bit 1: MATCH — Address match flag. In slave mode, this bit is set to one when the address byte (the first byte transferred) matches the 7-bit address in the Own Address Register. It is cleared by Reset, a START condition, STOP condition or software writing a “1” to this bit.
- Bit 0: START flag. It will be set by hardware when a START condition is detected. It

is cleared at the end of the first byte transfer, Reset or a STOP condition. It is cleared at the end of the first byte transfer, system reset, soft reset a STOP condition and by software writing a “1” to this bit.

3.4.4.7.7 Access.bus Control Register 1 (Write/Read)



Reset Condition: 00h

- Bits 7,6: Reserved.
- Bit 5: OVERFLOW — Overflow flag. This bit will be set to one when the serial clock (SCLK) is held low for more than 2-3 msec. In Master mode, when this bit is set to a one, the transfer should be aborted and the Access.bus interface logic should be reset by a soft reset (that is writing a one to the SR bit). This bit will be cleared by Reset or software writing a zero to this bit.
- Bit 4: MASTRQ — Software must set this bit to a one to request control of the bus as a master.
- Bit 3: ABINT — Access.bus interrupt flag. Will be set to a 1 at the end of a transfer by hardware. Must be cleared by software writing a zero to this bit.
- Bit 2: STOP_W — STOP request. In master mode, setting this bit to a one results in the generation of a STOP condition. This bit is cleared by Reset or by software writing a zero to this bit.
- Bit 1: BUSY — Busy Transmitting flag. Software must set this bit to a one before the Access.bus interface can transmit or receive data. When this bit is a zero the SCLK signal is driven low, preventing any further Access.bus transfers.
Once set to a one, this bit will be cleared to a zero by hardware at the end

of an Access.bus transfer cycle from/to this logic. When this bit is set to a one, it indicates that Access.bus shift register is busy shifting data out/in.

Bit 0: ACK — Acknowledge flag. This bit is a status bit for transmit modes and a control bit for receive modes.

If this Access.bus interface logic is in Master Transmit mode and a transmitted byte is not acknowledged this bit will not be set; the Access.bus interface logic will generate a master abort (STOP condition). If this bit is set to a one, then software should clear it by writing it to a zero before the next transfer begins.

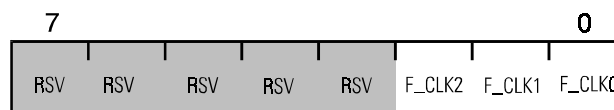
If this Access.bus interface logic is Slave Transmit mode and a transmitted byte is not acknowledged this bit will not be set; the Access.bus logic will simply stop its transfer and return to an idle mode. If this bit is set to a one, then software should clear it by writing it to a zero before the next transfer begins.

If this Access.bus interface logic is in either the Master or Slave Receive mode, this bit is under software control. Setting this bit to a one, will result in the generation of an Acknowledge at the appropriate time. Clearing this bit to a zero will result in no Acknowledge generation.

A Reset, STOP condition or software writing a zero to this bit, resets this bit to zero.

3.4.4.7.8 Access.bus Control Register 2 (Write/Read)

This register defines the number of clocks to use to filter out noise on SCLK and SI signals. An edge on either SCLK or SI will start a delay counter programmed by bits 2-0 of this register. If after the delay time, the value of SCLK or SI is the same (indicating that the edge was not a noise glitch), then at that time the change in SCLK or SI will be recognized.



I/O Map Address

0057h

Access

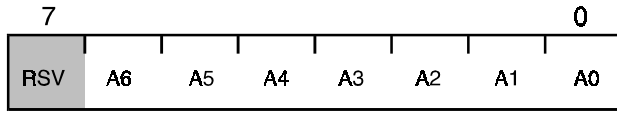
R/W

Bits 7-3: Reserved.

Bit2-0: F_CLK — Filtering Clocks. These bits define the number of input OSCX1 clocks a new value on SCLK or SI must be maintained to be recognized. If the new value on SCLK or SI is less than this delay time, it will be filtered out and ignored.

Bits 2-0	OSCX1 Clock Periods
000	5
001	7
010 (reset value)	9
011	11
100	13
101	15
110	17
111	19

**3.4.4.7.9 Own Address Register
(Write/Read):**



I/O Map Address

0056h

Access

R/W

Bit 7: Reserved.

Bits 6-0: Loaded with the 7-bit Access.bus address. When addressed as a slave, the first seven bits transferred will be compared with this register to determine if this device is being addressed. This register should be programmed before the Access.bus interface is enabled, if the Access.bus interface is programmed to operate in Slave mode.

**3.4.4.8 MICROWIRE/Access.bus
Interface Programming Notes**

If the current bus master wants to continue to use the Access.Bus to talk to another slave without possible arbitration, then Sr can be used. To program Repeat Start (Sr):

- a) After the current transfer is finished (BUSY=0, ABINT=1); set MASTRQ=1 for repeat start
- b) Set BUSY
- c) Clear ABINT
- d) Program SDA (Data Register) with the address of the slave.

During an Access.bus transfer, software must not write to the SDA register or the transferred data will be corrupted.

Do not reprogram the SCON (System Register) clock divide chain while the Access.bus is still busy.

The SIO register must be programmed only after the MICROWIRE interface is enabled.

STOP_W needs to be set before the last transfer finishes. A good time to set STOP_R is before the last byte transfer starts.

3.4.5 The Serial UART Port

The NS486SXF UART provides full NS16550 (PC standard) serial communications port compatibility. It performs serial-to-parallel conversion from external devices and parallel-to-serial conversion from data from the NS486SXF going to external peripherals. Full modem control is supported. A 16-bit FIFO buffer improves performance.

The NS486SXF UART is a standard PC UART, but implementation choices raise some issues to be considered by the user. These include:

- 1) The standard UART uses a 1.843 MHz clock. The NS486SXF UART uses the oscillator clock divided down to near 1.843 MHz. The NS486SXF system clock may be any of many frequencies, depending upon implementation. User software must know this frequency and set the appropriate clock divisor (at I/O map location EF70h) to achieve a near 1.843 MHz UART clock. (The faster the system clock, the closer to 1.843 MHz will be the final divided clock.)
- 2) The NS486SXF provides for full modem control signals, but they are shared with other I/O functions. The Rx and Tx signals are always available, but the $\overline{\text{RTS}}$, $\overline{\text{DTR}}$, and $\overline{\text{DSR}}$ signals are shared with the auxiliary communications processor port interface. Likewise, the $\overline{\text{CTS}}$, $\overline{\text{DCD}}$, and $\overline{\text{RI}}$ modem interface signals are shared with the three-wire synchronous serial interface. If either or both sets of these control signals are required, then the primary function must be disabled.
- 3) The NS486SXF provides a HP-SIR or IrDA compatible infrared communications port capability. The single NS486SXF serial port can be selected as either the standard UART or the serial infrared port at any time. In serial infrared mode, the NS486SXF uses transmitter LEDs and receiver photodiodes to provide two-way wireless communications.

General Operation:

The UART performs serial-to-parallel conversion on data characters received from a peripheral device and parallel-to-serial conversion on data characters received from the CPU. The CPU can read the complete status of the UART at any time during the functional operation. Status information includes the type and condition of the transfer operations being performed by the UART, as well as any error conditions (parity, overrun, framing, or break interrupt).

The UART has a programmable baud rate generator that is capable of dividing the internal reference clock by divisors of 1 to $2^{16}-1$, and producing a 16x clock for driving the transmitter logic. Provisions are also included to use this 16x clock to drive the receiver logic. The UART has a prioritized interrupt system. Interrupts can be programmed to the user's requirements, minimizing the computing required to handle the communications link.

3.4.5.1 Serial Port Registers

There are twelve UART control registers located at one of four I/O address map locations: 03F8-03FFh (usual PC COM1 I/O address), 02F8-02FFh (usual PC COM2 I/O address), 03E8-03EFh (COM3), 02E8-02EFh (COM4). The default location for the UART is the COM1 address range, the other locations can be selected by programming the Bus Interface Unit Control Register #2 (See Section 4.0). Location references to UART registers in the following text assume an offset to one of the four base locations. The twelve registers are placed in only 8 locations by the technique of sharing the same I/O address for some registers (i.e., one register is read only at the address and one register is write only at the address) and by using Bit 7 in the Line Control Register to gain access to the Divisor Latch Registers at location 000-001 instead of the Receiver Buffer, Transmitter Holding, and Interrupt Enable Registers.

Use the following table to determine the address of any of the registers. Note that the Bit7 of the Line Control Register (DLAB) must be set to 1 to gain access to the Divisor Latch registers. Otherwise access to the first two locations will gain access to the Receive Buffer (if the access is a read to location 0), the Transmitter Holding register (if the access is a write to location 0), or the Interrupt Enable register (if the access is to location 1).

Table 3-9, “Summary of NS486SXF UART Registers,” on page 147 is a complete summary of all the serial port registers, showing the register address and the the setting of the DLAB bit in each case.

DLAB1	A2	A1	A0	Selected Register
0	0	0	0	Receiver Buffer-RBR (Read), Transmitter Holding-THR (Write)
0	0	0	1	Interrupt Enable-IER
0	0	1	0	Interrupt Identification-IIR (Read), FIFO Control-FCR (Write)
x	0	1	1	Line Control-LCR
x	1	0	0	MODEM Control-MCR
x	1	0	1	Line Status-LSR
x	1	1	0	MODEM Status-MSR
x	1	1	1	Scratch
1	0	0	0	Divisor Latch (Least Significant Byte)
1	0	0	1	Divisor Latch (Most Significant Byte)

Table 3-9: Summary of NS486SXF UART Registers

Register Address												
	0 DLAB=0	0 DLAB=0	1 DLAB=0	2	2	3	4	5	6	7	0 DLAB=1	1 DLAB=1
Bit No.	Receiver Buffer Register (Read Only)	Transmitter Holding Register (Write Only)	Interrupt Enable Register	Interrupt Ident. Register (Read Only)	FIFO Control Register (Write Only)	Line Control Register	MODEM Control Register	Line Status Register	MODEM Status Register	Scratch Pad Register	(LS)	(MS)
	RBR	THR	IER	IIR	FCR	LCR	MCR	LSR	MSR	SCR	DLL	DLM
0	Data Bit 0 (Note 1)	Data Bit 0	Enable Received Data Available Interrupt	"0" if Interrupt Pending	FIFO Enable	Word Length Select Bit 0	Data Terminal Ready (DTR)	Data Ready (DR)	Delta Clear to Send	Bit 0	Bit 0	Bit 8
1	Data Bit 1	Data Bit 1	Enable Transmitter Holding Register-Empty Interrupt	Interrupt ID Bit	RCVR FIFO Reset	Word Length Select Bit 1	Request to Send (RTS)	Overrun Error (OE)	Delta Data Set Ready	Bit 1	Bit 1	Bit 9
2	Data Bit 2	Data Bit 2	Enable Receiver Line Status Interrupt	Interrupt ID Bit	XMIT FIFO Reset	Number of Stop Bits	Out 1 Bit	Parity Error (PE)	Trailing Edge Ring Indicator	Bit 2	Bit 2	Bit 10
3	Data Bit 3	Data Bit 3	Enable MODEM Status Interrupt	Interrupt ID Bit (Note 2)	Reserved	Parity Enable	IRQ Enable (Note 3)	Framing Error (FE)	Delta Data Carrier Detect	Bit 3	Bit 3	Bit 11
4	Data Bit 4	Data Bit 4	0	0	Reserved	Even Parity Select	Loop	Break Interrupt (BI)	Clear to Send (CTS)	Bit 4	Bit 4	Bit 12
5	Data Bit 5	Data Bit 5	0	0	Reserved	Stick Parity	0	Transmitter Holding Register (THRE)	Data Set Ready (DSR)	Bit 5	Bit 5	Bit 13
6	Data Bit 6	Data Bit 6	0	FIFOs Enabled (Note 2)	RCVR Trigger (MSB)	Set Break	0	Transmitter Empty (TEMT)	Ring Indicator (\overline{RI})	Bit 6	Bit 6	Bit 14
7	Data Bit 7	Data Bit 7	0	FIFOs Enabled (Note 2)	RCVR Trigger (MSB)	Divisor Latch Access Bit (DLAB)	0	Error in RCVR FIFO (Note 2)	Data Carrier Detect (DCD)	Bit 7	Bit 7	Bit 15

Note 1: Bit 0 is the least significant bit. It is the first bit serially transmitted or received.

Note 2: These bits are always 0 in the NS16450 Mode.

Note 3: This bit does not have a pin associated with it. See the Interrupt Source Selection Section concerning how to disable the UART IRQ signal.

3.4.5.1.1 Line Control Register

The system programmer uses the Line Control Register (LCR) to specify the format of the asynchronous data communications exchange and set the Divisor Latch Access bit. This is a read and write register. Table 3-9 shows the contents of the LCR. Details on each bit follow:

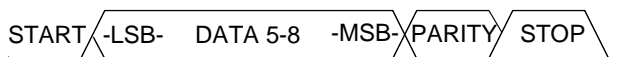


Figure 3-49 NS486SXF Composite Serial Data

Bit 7: This bit is the Divisor Latch Access Bit (DLAB). It must be set high (logic 1) to access the Divisor Latches of the Baud rate Generator during a Read or Write operation. It must be set low (logic 0) to access any other register.

Bit 6: This bit is the Break Control bit. It causes a break condition to be transmitted to the receiving UART. When it is set to 1, the serial output (SOUT) is forced to the Spacing state (0). The break is disabled by setting bit 6 to 0. The Break Control bit acts only on SOUT and has no effect on the transmitter logic.

Note: This feature enables the CPU to alert a terminal. If the following sequence is used, no erroneous characters will be transmitted because of the break.

1. Wait for the transmitter to be idle, (TEMT = 1).
2. Set break for the appropriate amount of time. If the transmitter will be used to time the break duration then check that TEMT = 1 before clearing the Break Control bit.
3. Clear break when normal transmission has to be restored.

During the break, the Transmitter can be used as a character timer to accurately establish the break duration by sending characters and monitoring THRE and TEMT.

Bit 5: This bit is the Stick Parity bit. When parity is enabled it is used in conjunction with bit 4 to select Mark or Space Parity. When LCR bits 3, 4 and 5 are 1 the Parity bit is transmitted and checked as a 0

(Space Parity). If bits 3 and 5 are 1 and bit 4 is a 0, then the Parity bit is transmitted and checked as 1 (Mark Parity). If bit 5 is 0 Stick Parity is disabled.

Bit 4: This bit is the Even Parity Select bit. When parity is enabled and bit 4 is 0, an odd number of logic 1s is transmitted or checked in the data word bits and Parity bit. When parity is enabled and bit 4 is a 1, an even number of logic 1s is transmitted or checked.

Bit 3: This bit is the Parity Enable bit. When it is 1, a Parity bit is generated (transmit data) or checked (receive data) between the last data bit and Stop bit of the serial data. (The Parity bit is used to produce an even or odd number of 1s when the data bits and the Parity bit are summed.)

Bit 2: This bit specifies the number of Stop bits transmitted with each serial character. If it is 0, one Stop bit is generated in the transmitted data. If it is 1, when a 5-bit data length is selected, one and a half Stop bits are generated. If it is 1, when either a 6-, 7-, or 8-bit word length is selected, two Stop bits are generated. The receiver checks the first Stop bit only, regardless of the number of Stop bits selected.

Bits 1, 0: These two bits specify the number of data bits in each transmitted or received serial character. The encoding of bits 0 and 1 is as follows:

Bit 1	Bit 0	Data Length
0	0	5 Bits
0	1	6 Bits
1	0	7 Bits
1	1	8 Bits

Table 3-10: UART Reset Configuration

Register Signal	Reset Control	Reset State
Interrupt Enable	Master Reset	0000 0000 (Note 1)
Interrupt Identification	Master Reset	0000 0001
FIFO Control	Master Reset	0000 0000
Line Control	Master Reset	0000 0000
MODEM Control	Master Reset	0000 0000
Line Status	Master Reset	0110 0000
MODEM Status	Master Reset	XXXX 0000 (Note 2)
SOUT	Master Reset	High
INTR (RCVR Errs)	Read LSR MR	Low/TRI-STATE
INTR (RCVR Data Ready)	Read RBR MR	Low/TRI-STATE
INTR (THRE)	Read IIR Write THR MR	Low/TRI-STATE
INTR (Modem Status Changes)	Read MSR MR	Low/TRI-STATE
Interrupt Enable Bit	Master Reset	Low
RTS	Master Reset	High
DTR	Master Reset	High
RCVR FIFO	MR/FCR1•FCR0/ΔFCR0	All Bits Low
XMIT FIFO	MR/FCR1•FCR0/ΔFCR0	All Bits Low

Note 1: Boldface bits are permanently low.

Note 2: Bits 7-4 are driven by the input signals.

3.4.5.1.2 Programmable Baud Rate Generator

NS486SXF contains a programmable UART clock generator and a programmable Baud rate Generator. The oscillator clock signal is divided by the divisor contained in the CPU clock divisor register (see following table), and is sent to the Baud rate Generator, then divided by the divisor of the associated UART. The output frequency of the Baud rate Generator is 16x the baud rate.

$$\text{divisor \#} = (\text{frequency input}) / (\text{baud rate} \times 16)$$

The output of the Baud rate Generator drives the transmitter and receiver sections of the serial channel. Two 8-bit latches store the divisor in a 16-bit binary format. These Divisor Latches must be loaded during initialization to ensure proper operation of the Baud rate Generator. Upon loading the Divisor Latches, a 16-bit Baud Counter is loaded. The following table provides decimal divisors to use with potential oscillator frequencies for the NS486SXF. Using a divisor of zero is not recommended.

Table 3-11: NS486SXF UART Divisors, Baud Rates and Clock Frequencies

		50 MHz Div. By 27	48 MHz Div. By 26	40 MHz Div. By 22	25 MHz Div. By 14
Baud Rate	Divisor	%Err.	%Err.	%Err.	%Err.
300	384	0.469	0.160	1.357	3.119
1200	96	0.469	0.160	1.357	3.119
2400	48	0.469	0.160	1.357	3.119
9600	12	0.469	0.160	1.357	3.119
14400	8	0.469	0.160	1.357	3.119
19200	6	0.469	0.160	1.357	3.119
38400	3	0.469	0.160	1.357	3.119
57600	2	0.469	0.160	1.357	3.119
115200	1	0.469	0.160	1.357	3.119

Note: The CPU Clock Divisor Register (see Section 3.4.5.1.8 on page 154) must be programmed with the appropriate value from the top of the chosen column.

3.4.5.1.3 Line Status Register

This 8-bit register provides status information to the CPU concerning the data transfer. Table 3.10 shows the contents of the Line Status Register.

- Bit 7: In the FIFO Mode LSR7 is set when there is at least one parity error, framing error or break indication in the FIFO. LSR7 is cleared when the CPU reads the LSR, if there are no subsequent errors in the FIFO.
- Bit 6: This bit is the Transmitter Empty (TEMT) indicator. It is set to 1 whenever the Transmitter Holding Register (THR) and the Transmitter Shift Register (TSR) are both empty. It is reset to 0 if either the THR or TSR contains a data character. In the FIFO mode this bit is set to 1 whenever the transmitter FIFO and the shift register are both empty.

- Bit 5: This bit is the Transmitter Holding Register Empty (THRE) indicator. It indicates that the UART is ready to accept a new character for transmission. In addition, it causes the UART to issue an interrupt to the CPU when the Transmit Holding Register Empty Interrupt enable is set high. The THRE bit is set to 1 when a character is transferred from the Transmitter Holding Register into the Transmitter Shift Register. The bit is reset to 0 whenever the CPU loads the Transmitter Holding Register. In the FIFO mode it is set when the XMIT FIFO is empty; it is cleared when at least 1 byte is written to the XMIT FIFO.

Note: Bits 1 through 4 are the error conditions that produce a Receiver Line Status interrupt whenever any of the corresponding conditions are detected and that interrupt is enabled.

- Bit 4: This bit is the Break Interrupt (BI) indicator. It is set to 1 whenever the received data input is held in the Spacing (0) state for longer than a full word transmission time (i.e., the total time of Start bit + data bits + Parity + Stop bits). The BI indicator is reset whenever the CPU reads the contents of the Line Status Register. In the FIFO mode this error is associated with the particular character in the FIFO that it applies to. This error is revealed to the CPU when its associated character is at the top of the FIFO. When break occurs only one character is loaded into the FIFO. To Restart after a break is received, the SIN pin must be 1 for at least 1/2 bit time.
- Bit 3: This bit is the Framing Error (FE) indicator. It indicates that the received character did not have a valid Stop bit. It is set to 1 whenever the Stop bit following the last data bit or parity bit is a 0 (Spacing level). The FE indicator is reset whenever the CPU reads the contents of the Line Status Register. In the FIFO mode this error is associated with the particular character in the FIFO that it applies to. This error is revealed to the CPU when its associated character is at the top of the FIFO. The UART will try to resynchronize after a framing error by assuming that the error was due to the next start bit. It samples this “start” bit twice and then takes in the bits following it as the rest of the frame.
- Bit 2: This bit is the Parity Error (PE) indicator. It indicates that the received data character does not have the correct parity, as selected by the even-parity select bit. The PE bit is set to 1 upon detection of a parity error and is reset to 0 whenever the CPU reads the contents of the Line Status Register. In the FIFO mode this error is associated with the particular character in the FIFO that it applies to. This error is revealed to the CPU when its associated character is at the top of the FIFO.
- Bit 1: This bit is the Overrun Error (OE) indicator. It indicates that data in the Receiver Buffer Register was not read by the CPU before the next character was transferred into the Receiver Buffer Register, thereby destroying the previous character. The OE indicator is set to 1 upon detection of an overrun condition, and reset whenever the CPU reads the contents of the Line Status Register. If the FIFO mode data continues to fill the FIFO beyond the trigger level, an Overrun error will occur only after the FIFO is completely full and the next character has been received in the shift register. OE is indicated to the CPU as soon as it happens. The character in the shift register is overwritten, but it is not transferred to the FIFO.
- Bit 0: This bit is the receiver Data Ready (DR) indicator. It is set to 1 whenever a complete incoming character has been received and transferred into the Receiver Buffer Register or the FIFO. It is reset to 0 by reading the data in the Receiver Buffer Register or the FIFO.

Note: The Line Status Register is intended for read operations only. Writing to this register is not recommended as this operation is only used for factory testing. In the FIFO mode the software must load a data byte in the Rx FIFO via the Loopback Mode in order to write to LSR2 - LSR4. LSR0 and LSR7 can't be written to in FIFO Mode.

3.4.5.1.4 FIFO Control Register

This is a write-only register at the same location as the IIR (the IIR is a read-only register). This register is used to enable the FIFOs, clear the FIFOs and to set the RCVR FIFO trigger level.

Bits 7, 6: FCR6 and FCR7 are used to designate the interrupt trigger level. When the number of bytes in the RCVR FIFO equals the designated interrupt trigger level, a Received Data Available Interrupt is activated. This interrupt must be enabled by setting IER0.

FCR Bits		RCVR FIFO
7	6	Trigger Level (Bytes)
0	0	01
0	1	04
1	0	08
1	1	14

- Bits 4,5: FCR4 to FCR5 are reserved for future use.
- Bit 3: Writing to FCR3 does not change UART operations.
- Bit 2: Writing 1 to FCR2 clears all bytes in the XMIT FIFO and resets its counter logic to 0. The shift register is not cleared. The 1 that is written to this bit position is self-clearing.
- Bit 1: Writing 1 to FCR1 clears all bytes in the RCVR FIFO and resets its counter logic to 0. The shift register is not cleared. The 1 that is written to this bit position is self-clearing.
- Bit 0: Writing a 1 to FCR0 enables both the XMIT and RCVR FIFOs. Resetting FCR0 clears all bytes in both FIFOs. When changing from FIFO Mode to NS16450 Mode and vice versa, data is automatically cleared from the FIFOs. This bit must already be 1 when other FCR bits are written to or they will not be programmed.

3.4.5.1.5 Interrupt Identification Register

In order to provide minimum software overhead during data character transfers, the UART prioritizes interrupts into four levels and records these in the Interrupt Identification Register. The four levels of interrupt conditions in order of priority are Receiver Line Status; Received Data Ready; Transmitter Holding Register Empty; and MODEM Status.

When the CPU accesses the IIR, the UART freezes all interrupts and indicates the highest priority pending interrupt to the CPU. While this CPU access is occurring, the UART records new interrupts, but does not change its current indication until the current access is complete. Table 3-10 shows the contents of the IIR.

- Bits 7, 6: These two bits are set when FCR0 = 1. (FIFO Mode enabled.)
- Bits 5, 4: These bits of the IIR are always 0.
- Bit 3: In the FIFO mode it is set along with bit 2 when a time-out interrupt is pending.
- Bits 2, 1: These two bits of the IIR are used to identify the highest priority interrupt pending as indicated in Table 3-11.
- Bit 0: This bit can be used in an interrupt environment to indicate whether an interrupt condition is pending. When it is 0, an interrupt is pending and the IIR contents may be used as a pointer to the appropriate interrupt service routine. When it is 1, no interrupt is pending.

3.4.5.1.6 Interrupt Enable Register

This register enables the five types of UART interrupts. Each interrupt can individually activate the appropriate interrupt signal. It is possible to totally disable the interrupt system by resetting bits 0 through 3 of the Interrupt Enable Register (IER). Similarly, setting bits of this register to 1, enables the selected interrupt(s). Disabling an interrupt prevents it from being indicated as active in the IIR and from activating the interrupt signal. All other system functions operate in their normal manner, including the setting of the Line Status and MODEM Status Registers. Table 3-10 shows the contents of the IER. Details on each bit follow. See MODEM Control Register bit 3 for more information on enabling the interrupt.

Bits 7-4:	These four bits are always logic 0.	Bit 3:	This bit has no affect on the NS486SXF UART interrupt. Refer to Section 3.3.5.3 “Interrupt Source Selection” on how to control the NS486SXF’s UART IRQ signal. In Local Loopback Mode, this bit controls bit 7 of the MODEM Status Register.
Bit 3:	This bit enables the MODEM Status Interrupt when set to logic 1.	Bit 2:	This bit is the OUT1 bit. It does not have an output pin associated with it. It can be written to and read by the CPU. In Local Loopback Mode, this bit controls bit 6 of the Modem Status Register.
Bit 2:	This bit enables the Receiver Line Status Interrupt when set to logic 1.	Bit 1:	This bit controls the Request to Send (RTS) output. Its effect on the RTS output is identical to that described above for bit 0. In Local Loopback Mode, this bit controls bit 4 of the MODEM Status Register.
Bit 1:	This bit enables the Transmitter Holding Register Empty Interrupt when set to 1.	Bit 0:	This bit controls the Data Terminal Ready (DTR) output. When it is set to 1, the \overline{DTR} output is forced to a logic 0. When it is reset to 0, the \overline{DTR} output is forced to 1. In Local Loopback Mode, this bit controls bit 5 of the MODEM Status Register.
Bit 0:	When set to 1 this bit enables the Received Data Available Interrupt and Timeout Interrupt in the FIFO Mode.		

3.4.5.1.7 MODEM Control Register

This register controls the interface with the MODEM or data set (or a peripheral device emulating a MODEM). The contents of the MODEM Control Register (MCR) are indicated in Table 3-10 and are described below.

Bits 7-5:	These bits are permanently set to 0.
Bit 4:	This bit provides a Local loopback feature for diagnostic testing of the UART. When it is set to 1, the following occur: the transmitter Serial Output (SOUT) is set to the Marking (1) state; the receiver Serial Input (SIN) is disconnected; the output of the Transmitter Shift Register is “looped back” (connected) to the Receiver Shift Register; the four MODEM Control inputs (\overline{DSR} , \overline{CTS} , \overline{RI} and \overline{DCD}) are disconnected; and the DTR, RTS, OUT1, IRQ ENABLE bits in MCR are internally connected to DSR, CTS, \overline{RI} and DCD in MSR, respectively. The MODEM Control output pins are forced to their high (inactive) states. In the Loopback Mode, data that is transmitted is immediately received. This feature allows the processor to verify the transmit-and-received-data paths of the serial port. In the Loopback Mode, the receiver and transmitter interrupts are fully operational. The MODEM Status Interrupts are also operational, but the interrupts’ sources are the lower four bits of MCR instead of the four MODEM control inputs. Writing a 1 to any of these 4 MCR bits will cause an interrupt. In Loopback Mode the interrupts are still controlled by the Interrupt Enable Register. The IRQ3 and IRQ4 pins will be TRI-STATE in the Loopback Mode.

Note: The \overline{DTR} and RTS output of the UART may be applied to an EIA inverting line driver (such as the DS1488) to obtain the proper polarity input at the MODEM or data set.

**3.4.5.1.8 Clock Divisor Register
(I/O Address EF70)**

This register contains the divisor to produce the (near) 1.8432 MHz clock to the UART from the oscillator clock. The lower six bits of this register should be programmed with a divisor value. Example values are shown at the top of each column in Table , “,” on page 150. **Note:** Do not change the upper two bits of this register. The reset value of the lower six bits is undefined.

**3.4.5.1.9 Modem Signal Control
Register (I/O Address EF71)**

This register controls the multiplexing of the auxiliary communications processor signal pins with either a set of the UART MODEM functions or additional chip selects. It also controls the multiplexing of the 3-Wire functions with the remaining UART MODEM functions. This configuration presents two levels of control for MODEM interfaces, allowing either a minimum MODEM interface, and separately, the additional pins to provide a full MODEM interface. At reset this register is initialized to 31h.

- Bits 7-6: Reserved.
- Bit 5: Infrared transmitter drive control bit.
 - 1 = The IRTX signal will be active for 3/16 baud (this is the HP-SIR standard mode).
 - 0 = The IRTX signal will be active for 1.6 microseconds (this is the iRDA mode).
- Bit 4: Infrared Half/Full Duplex configuration bit.
 - 0 = Full Duplex; both transmitter and receiver are enabled.
 - 1 = Half Duplex. The receiver input is blocked to ‘1’ while the transmitter is busy: from the beginning of the start bit until the end of the stop bit(s).

This prevents noise during transmission. Otherwise the receiver may inadvertently pick up stray photons from the transmitter.

- Bit 3: UART interface Mode bit. This bit is used for run time mode selection of either Normal (UART) Mode or IR Mode.
 - 1 = Infrared (IR) Mode
 - 0 = Normal (UART) Mode
- Bit 2: Setting this bit to 1 places signals \overline{CTS} , \overline{DCD} , and \overline{RI} on the SI, SO, and SCLK pins respectively. Setting this bit to 0 places the Three-Wire functions onto their respective pins.
- Bit 1: Setting this bit to 1 selects the additional chip selects ($\overline{CS6}$, $\overline{CS7}$ & $\overline{CS8}$) onto the \overline{EREQ} , \overline{DRV} , and \overline{EACK} pins. Setting this bit to 0 places the External Request, Drive Enable, and External Acknowledge functions onto their respective pins. This bit is ignored if Bit 0 is set to 1.
- Bit 0: Setting this bit to 1 places signals \overline{RTS} , \overline{DTR} , and \overline{DSR} on the \overline{EREQ} , \overline{DRV} , and \overline{EACK} pins respectively. Setting this bit to 0 places the functions selected in Bit 1 onto the \overline{EREQ} , \overline{DRV} , and \overline{EACK} pins.

Table 3-12: NS486SXF UART Interrupt Control Functions

FIFO Mode Only	Interrupt Identification Register			Interrupt Set and Reset Functions				
	Bit 3	Bit 2	Bit 1	Bit 0	Priority Level	Interrupt Type	Interrupt Source	Interrupt Reset Control
0	0	0	1		-	None	None	—
0	1	1	0		Highest	Receiver Line Status	Overrun Error, Parity Error, Framing Error or Break Interrupt	Reading the Line Status Register
0	1	0	0		Second	Received Data Available	Receiver Data Available	Read Receiver Buffer
1	1	0	0		Second	Character Time-out Indication	No Characters Have Been Removed from or input to the RCVR FIFO During the Last 4 Char. Times and there is at least 1 Char. in it during this Time.	Reading the Receiver Buffer Register
0	0	1	0		Third	Transmitter Holding Register Empty	Transmitter Holding Register Empty	Reading the IIR Register if Source of Interrupt) or Writing the Transmitter Holding Register
0	0	0	0		Fourth	MODEM Status	Clear to Send or Data Set Ready or Ring Indicator or Data Carrier Detect	Reading the MODEM Status Register

3.4.5.1.10 MODEM Status Register

This register provides the current state of the control lines from the MODEM (or peripheral device) to the CPU. In addition to this current-state information, four bits of the MODEM Status Register provide change information. These bits are set to a logic 1 whenever a control input from the MODEM changes state. They are reset to logic 0 whenever the CPU

reads the MODEM Status Register. Table 3-10 shows the contents of the MSR.

- Bit 7: This bit is the complement of the Data Carrier Detect ($\overline{\text{DCD}}$) input. If bit 4 of the MCR is set to a 1, this bit is equivalent to $\overline{\text{IRQ ENABLE}}$ in the MCR.
- Bit 6: This bit is the complement of the Ring Indicator ($\overline{\text{RI}}$) input. If bit 4 of the MCR is set to a 1, this bit is equivalent to $\overline{\text{OUT 1}}$ in the MCR.
- Bit 5: This bit is the complement of the Data Set Ready ($\overline{\text{DSR}}$) input. If bit 4 of the MCR is set to a 1, this bit is equivalent to $\overline{\text{DTR}}$ in the MCR.
- Bit 4: This bit is the complement of the Clear to Send ($\overline{\text{CTS}}$) input. If bit 4 (loopback) of the MCR is set to a 1, this bit is equivalent to $\overline{\text{RTS}}$ in the MCR.

Bit 3:	This bit is the Delta Data Carrier Detect (DDCD) indicator. Bit 3 indicates that the $\overline{\text{DCD}}$ input to the chip has changed state.
Bit 2:	This bit is the Trailing Edge of Ring Indicator (TERI) detector. Bit 2 indicates that the $\overline{\text{RI}}$ input to the chip has changed from a low to a high state.
Bit 1:	This bit is the Delta Data Set Ready (DDSR) indicator. Bit 1 indicates that the $\overline{\text{DSR}}$ input to the chip has changed state since the last time it was read by the CPU.
Bit 0:	This bit is the Delta Clear to Send (DCTS) indicator. Bit 0 indicates that the $\overline{\text{CTS}}$ input to the chip has changed state since the last time it was read by the CPU.

Note: Whenever bit 0, 1, 2, or 3 is set to logic 1, a MODEM Status Interrupt is generated.

3.4.5.1.11 Scratchpad Register

This 8-bit Read/Write Register does not control the UART in any way. It is intended as a scratchpad register to be used by the programmer to hold data temporarily.

3.4.5.2 IrDA v1.0 Compatible Infrared Link

The SIR is a two-way wireless communication port using infrared as a transmission medium. The SIR connects the IRRX and IRTX signals via transmitter LED(s) and receiver photo diode(s): SOUT is encoded by the NS486SXF into the IRTX signal. The received signal (IRRX) is decoded back by the NS486SXF to generate the required SIN signal. There are two UART interface modes:

1. Normal (Modem) mode: The normal 16550 UART interface.
2. SIR mode: Serial Infrared mode, complies with IrDA standard.

The system can select either the UART or the SIR at any time. The SIR is in either Full Duplex or Half Duplex configuration:

1. Full Duplex: both transmitter and receiver are enabled simultaneously.
2. Half Duplex: the receiver is blocked when the transmitter is on.

The SIR output signal (IRTX - Infrared Transmitter), when active, is active for intervals of either 1.6 ms or 3/16 baud. This is to provide compatibility with both Hewlett-Packard SIR and IrDA. See Section 3.4.5.1.9 "Modem Signal Control Register (I/O Address EF71)"

The NS486SXF directly interfaces with an infrared transceiver analog front-end device. The NS486SXF can also be interfaced with discrete LED and photo-diode devices via an external analog circuit.

3.4.6 The LCD Controller

The NS486SXF LCD controller controls a variety of supertwist LCD panels. Supported configurations include 320x240, 320x200 and 480x320 with monochrome or grayscale graphics LCD modules equipped with self-contained screen drivers. NS486SXF uses a video frame buffer in system DRAM with either a 1- or 2-bit per pixel grayscale. A 60 - 90 Hz frame refresh rate is supported. Refresh rates can be fine-tuned to optimize image quality for a given LCD screen.

There is no separate video frame buffer - the LCD Controller uses system memory (DRAM or SRAM) for its video data storage. NS486SXF uses DMA transfers (using DMA Channels 1 or 5) to transfer video data from memory to a FIFO in the LCD Controller, minimizing CPU overhead (and permitting video display to continue even with the CPU in idle or power save modes).

The video data begins with several (up to 16) words of GLUT (Gray-scale Look-Up Table)

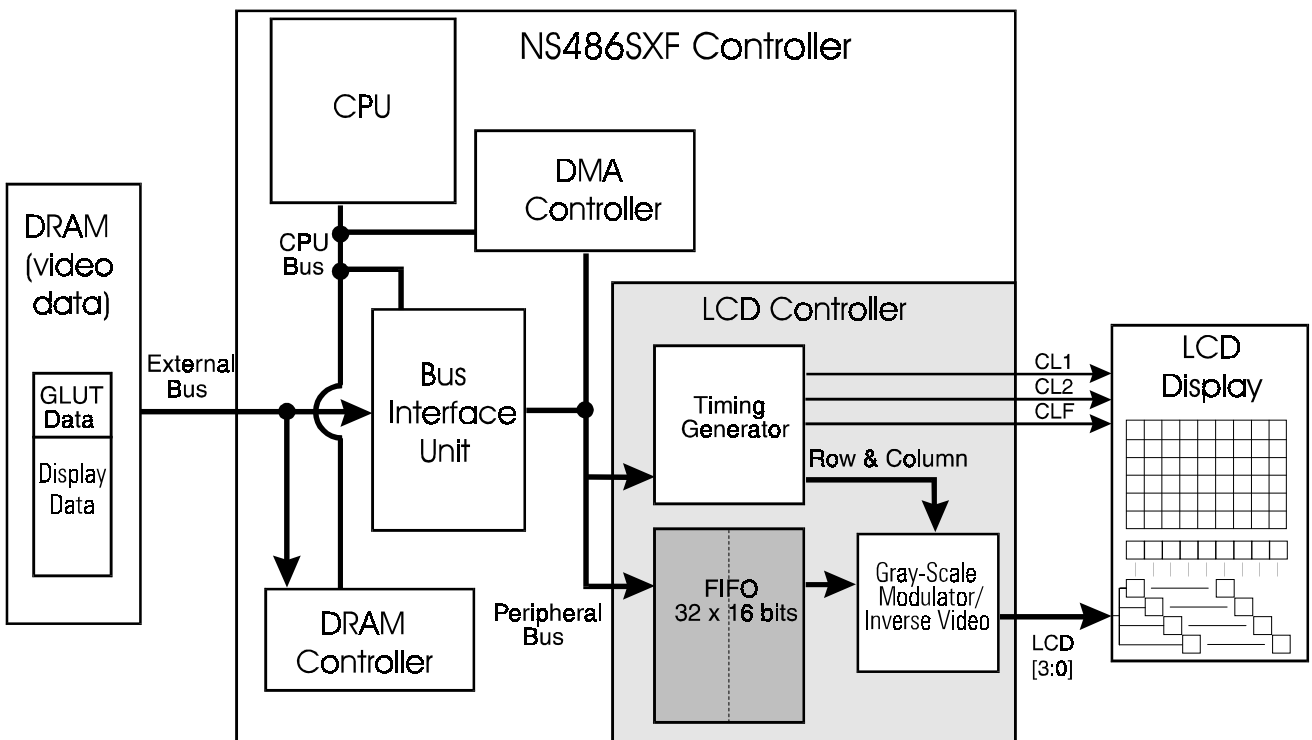
data that control the gray-scale operation, followed by the graphics data for the current frame.

The base address of the frame buffer is programmable, with resolution at least as fine as 64K bytes. With a 50 MHz crystal frequency, through the DRAM Controller, the LCD refresh operations are estimated to consume from 2.2% to 11.3% of the system bus bandwidth, depending on the size of screen, frame rate and number of gray scales.

Applying Power:

Note: It is very important that power be applied to the LCD display in proper sequence, otherwise damage can result. To prevent damage to the LCD panels, the external DC power supplied to the LCD Display (V_{EE}) must be disabled before the LCD Controller's clock is disabled.

Figure 3-50 LCD Controller Diagram



The power-up sequence is as follows:

- 1) Configure the LCD control registers.
- 2) Apply V_{DD} (5V or 3V) to the display.
- 3) Enable the LCD clock from the power management registers - this must be done within 20 msec. of applying V_{DD} .
- 4) Enable the LCD controller.
- 5) Within 20 msec. max after applying the LCD clock, apply V_{EE} (22V/-26V) to the display.

The power-down sequence is as follows:

- 1) Remove V_{EE} from the display.
- 2) Disable the LCD controller.
- 3) Within 20 msec. of removing V_{EE} , disable the LCD clock.
- 4) Within 20 msec. of removing the LCD clock, remove V_{DD} from the display.

Never disable the LCD clock when the LCD is enabled.

This sequence is shown in Figure 3.47.

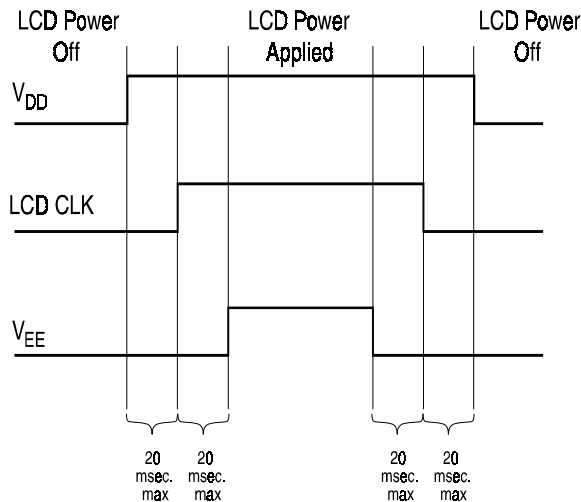


Figure 3-51 LCD Power Application

3.4.6.1 LCD Display Basics

An LCD display panel is a rectangular grid of rectangular or square dots. Acting as a thin double-paned window, the LCD grid is actually transparent electrodes laid out in horizontal rows on one thin pane, and in vertical columns on the other. The liquid crystal formula trapped in between the panes reacts to an electrical field applied to each electrode in the rows and columns. The reaction rotates the polarization of light transmitted through the LCD display. Polarizing layers outside the panes cause the dots to appear light or dark as the polarization changes. There are small gaps between the rows and columns, giving each dot a clear definition.

The display is controlled by continuously feeding dot data to the display. The data is organized as follows into individual pixels, rows of pixels, and full-page frames. Pixels are the individual data dots or bits. These bits are put together into rows. A set of rows makes up a frame. A frame is one full page of the display. LCD data is continuously sent (four bits or pixels at a time) to the LCD panel to refresh the display frame.

In the NS486SXF, as shown in Figure 3.46, the LCD display data is stored in system memory, transferred using the DMA controller from the memory to the LCD controller and finally sent to the LCD panel through the LCD [3:0] signal lines. The sequencing of the data is controlled with three clock signals, (CL1, CL2, CLF).

The display data is output from the NS486SXF in the four LCD [3:0] signals. A Frame signal, (CLF) is sent to indicate the start of the first Frame of data. The dot data (LCD [3:0]) is sent into a shift register in the LCD panel using a Dot clock (CL2) to clock the data bits into the LCD panel circuitry.

As shown in Figure 3-52, in the LCD panel, the data is sequentially organized into a full column (or row) of data. A Row pulse clock (CL1) is used to indicate when a full row of pixels has been sent. A frame consists of a given number of rows of a given number of pixels. For example, a 320x240 display would have a

row consisting of 320 pixels. A set of 240 rows would consist of a complete display frame of 320 x 240. Figure 3-53 shows this configuration.

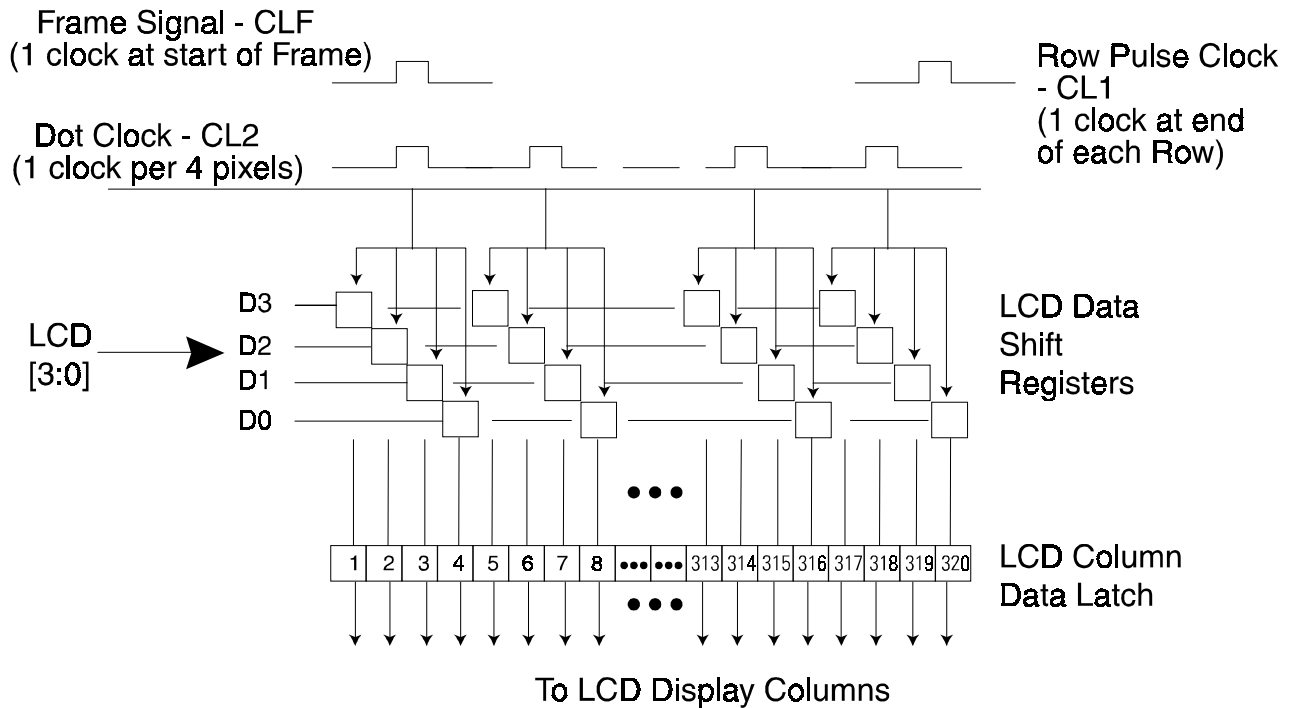


Figure 3-52 LCD Row/Pixel Organization

Note that a Frame Signal (CLF) indicates the beginning of the frame data; The dot clock (CL2) clocks in the LCD data four pixels at a time (LCD [3:0]), and the Row Pulse clock (CL1) indicates the end of the row data.

A complete frame of data makes up one full display screen as shown in Figure 3-53.

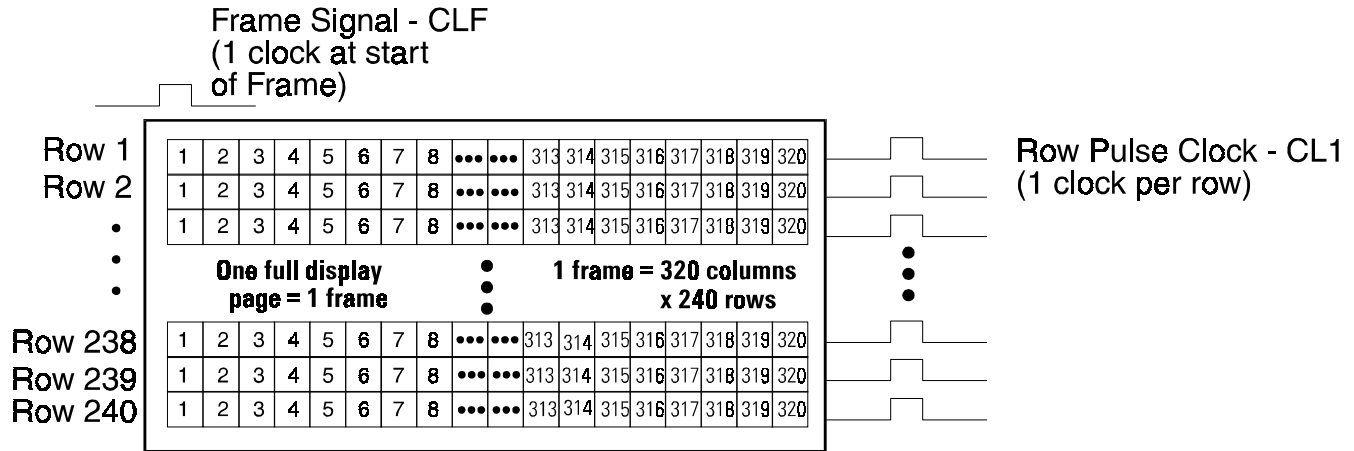


Figure 3-53 A Complete Frame

Since the LCD Display has no on-board frame buffer memory, the display data must be continuously refreshed. To get a stable, flicker-free image, the display data must be sent to the panel at least 60 times per second or 60 Hz. The optimum frequency for a given display varies, so the NS486SXF supports a wide range of refresh frequencies, from 60 Hz. to over 90 Hz.

As previously noted, the NS486SXF LCD controller sends data to the LCD panel in four pixel nibbles. The LCD panel stores up the nibbles until it has an entire row (320 in the previous example). With the arrival of the CL1 Row Clock pulse, the panel outputs the contents of the Column Shift Register to the internal column drivers. The Row counter is incremented and the next batch of pixels (in nibbles) is stored in the shift register. At the next CL1, that row is output to the column drivers. In this way the entire screen data is sequentially written.

Simple monochrome black (or blue) and white data is just as described above. Each data bit translates into a single pixel in the display. A one = full on display of either black or blue; a zero = off or white.

The NS486SXF LCD controller also supports gray scale modulation of the LCD data. Because of the nature of the LCD displays, this modulation is done in a temporal or time modulated way. Four levels of display are possible - ON (black or blue), dark gray, light gray, and OFF. A gray scale pixel map is used to modulate the various pixels. Gray scale pixels are turned on and off during successive frame scans. The rate in which they are turned on and off determines how dark or light they appear. Further, in order to prevent flickering, adjacent pixels of the same gray value will be modulated at different frequencies.

Finally, NS486SXF supports inverse video displays with programmable blinking rates.

3.4.6.2 Direct Control Signals

The NS486SXF LCD Controller provides pins for the common timing signals and data inputs for LCD control (see Figure 3.50). These are:

CL1, CL2 - Clocking Signals

CLF - Frame Signal

LCD[3:0] - Data Signals

The CL2 signal is the Dot Clock for LCD data. Display data is clocked out of the chip on the rising edge of this signal, to be shifted into the LCD panel module column drivers (Xdrivers) on each falling edge. Each CL2 pulse clocks four pixels into the panel's internal shift register. The pixels are taken from pins LCD[3:0], with the left most pixel on LCD[3]. CL2 is derived from the external oscillator clock with bits 7-3 of LCD Configuration Register 2 defining the level of clock division. Using a 32 x 16 bit FIFO, the maximum specified DRQ to $\overline{\text{DACK}}$ bus latency for a 480 x 320 screen with 4 gray levels is 20 usec (for a 320 X 240 screen, 40 usec) for 2 bits per pixel gray scale and a 72 Hz frame refresh rate.

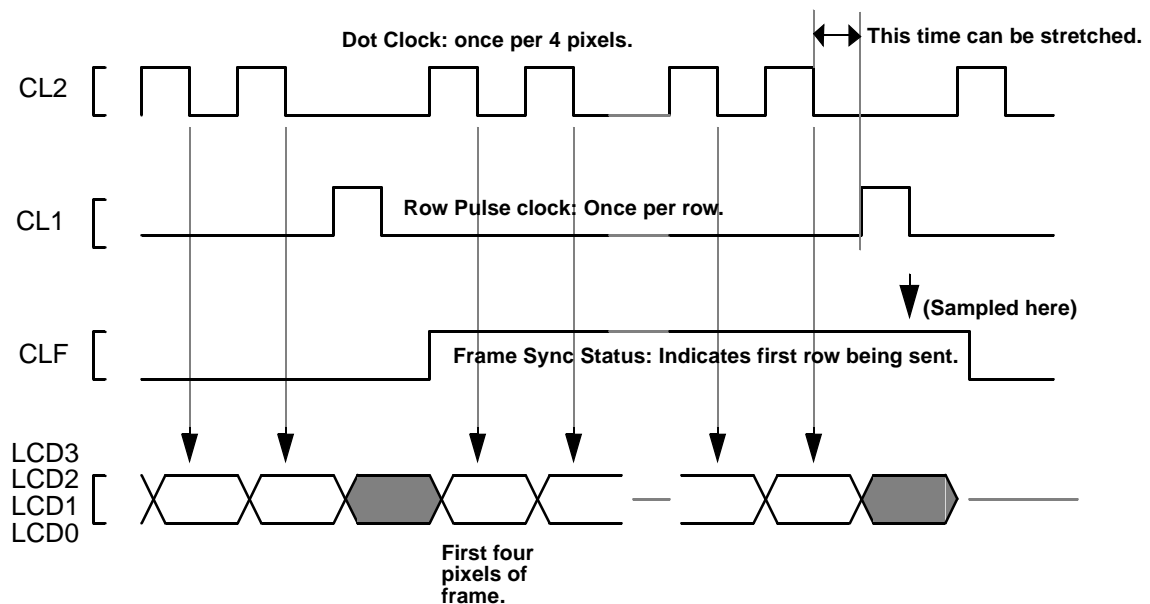
Once all the pixels of a row have been sent, the LCD Controller applies a pulse on the CL1

clock (also called Row Pulse Clock). This writes the row onto the display and advances to the next row. CL1 is generated by counting the number of CL2 cycles per row (i.e. 80 cycles for 320 pixels - one CL2 pulse for every 4 pixels) and adding any necessary untransmitted offset cycles to guarantee an acceptable Frame rate (see bits 3-0 of LCD Configuration Register 3). As data is presented for the first row of the frame, the signal CLF is brought high, and is held through the first CL1 pulse, as shown.

It should be noted that because of the varying characteristics of each LCD display, the exact frame refresh rate has a significant bearing on the final image quality of the display. Thus, the NS486SXF allows the designer to experiment with the precise frame refresh rate required to optimized image quality. This is accomplished by setting bits 3-0 of the LCD Configuration Register 3 to add an amount of time to the time between the last Dot clock of a Row and the Row Pulse (CL1) as shown in Figure 3.50. Up to 16 additional CL2 times (not pulses) can be added to create a precise frame refresh rate.

Note that this time can be varied "on-the-fly", so that the programmer can see the real-time effect of different values in these bits.

Figure 3-54 LCD Controller Timing



Pixels are modulated for gray-scale by presenting their data bits high and low in successive frame scans. Although the duty cycles are the same, adjacent gray pixels will not be modulated identically, in order to avoid flickering.

Two types of screen display modes are selectable. The first type is inverse video display (bit 1, LCD Configuration Register 1). The second type is display in blink mode where the duration and background are selectable (bits 7-5, LCD Configuration Register 4).

The LCD Controller supports a large number of 1/4 and 1/2 size VGA type LCD screens with excellent image quality. NS486SXF allows the designer to vary the timing of the Dot Clock and Row Pulses by a few pulses in order to optimized the visual image for the given display characteristics. The dot clock start pulse can be time shifted or stretched from as little as one clock pulse to as many as 16 clock pulses. This fine-tunes the frame refresh rate and results in excellent image quality regardless of the LCD display characteristics.

Power Management

Power can be saved by disabling the LCD Controller clock input, or by enabling one of the NS486SXF power-save mode clock division levels (up to divide by 16). Since the display data is coming from DRAM memory via the DMA Controller, the display can still display data, even when the NS486SXF CPU is in idle mode. The Bus Interface Unit mimics the CPU HOLD/HOLDA handshake signal timing with the DMA Controller to maintain DMA transfers.

Since the FIFO can only hold a small amount of display memory, it needs refilling on the order of every 20 to 100 ms depending on the number of gray scales, screen size, and frame rate. The size of the FIFO is fixed at 32 x 16, but the threshold limit is variable (bits 5-4, LCD Configuration Register 3).

Clock Division

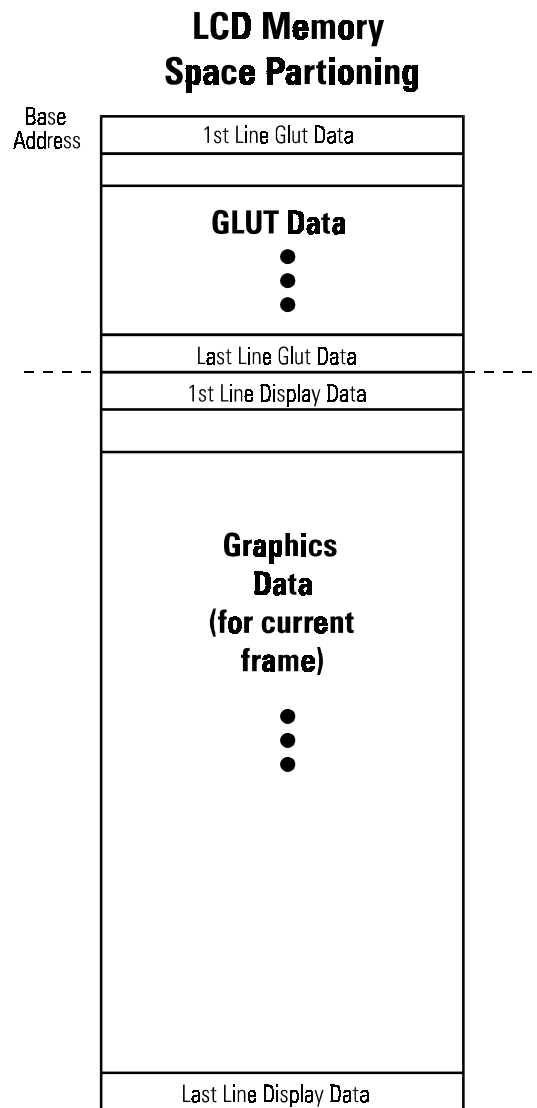
The LCD controller supports a configurable clock division scheme suitable for different types and sizes of LCD screens, for example, a standard 480 X 320 pixel LCD screen running at 14.2 ms frame rate requires a 2.7 MHz dot clock which is obtained by dividing down the system OSC_CLK. Using the data from the

clock frequency configuration location, user software must set an appropriate divisor to obtain 2.7 Mhz. The clock divisor can be programmed on the fly letting the designer easily optimize the screen frequency for the specific display screen being used.

Data Format

The graphics data to be displayed is held in memory (usually DRAM), and consists of up to 16 words of GLUT data, followed by the graphics data for the current frame as shown in the figure below.

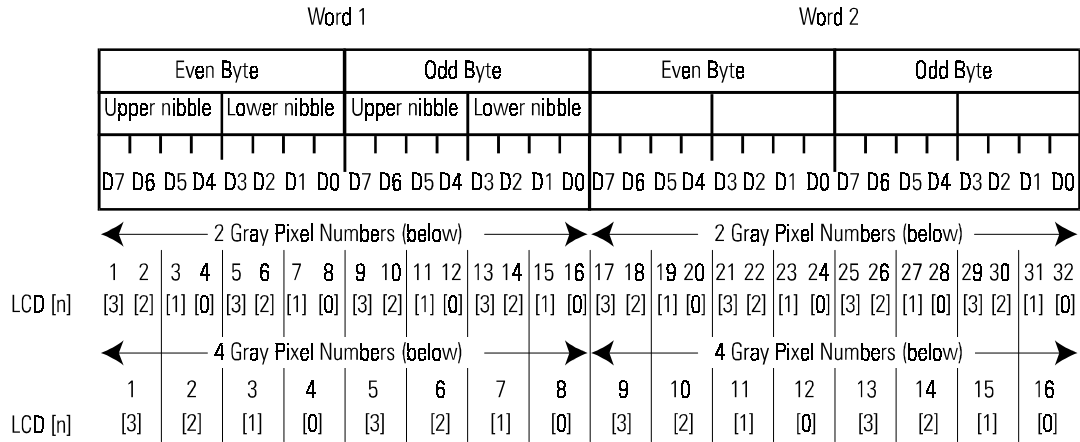
Figure 3-55 LCD Memory Space



Each frame is similarly structured. In some cases, the GLUT data can be the same for all data frames. In this case, the GLUT data will be looped for successive display frames.

DRAM video data mapping to the LCD data bits is shown in Figure 3-56

Figure 3-56 DRAM Video Data to LCD Pixel Mapping for 2 and 4 Gray Levels



Gray-Scale Modulation

For enhanced image quality, the NS486SXF LCD Controller supports gray-scale modulation. An off-chip gray-scale data lookup table (in the DRAM) contains gray-scale data. Gray-scale values can be one of the following:

- 00 = Full bright
- 01 = light gray
- 10 = dark gray
- 11 = off

The Full bright value, 00, is mapped to a pixel value of 0. The Off value, 11, is mapped to a pixel value of 1. The pixel values of light and dark gray, 01 and 10 are determined by GLUT word decoding map. The gray scale is achieved through modulation of the applied voltage pulses to the display. Since adjacent pixels cannot be modulated in exactly the same way, or unwanted flickering will occur, an odd and even mapping scheme is used. For example, for a dark gray pixel on an even row, certain bits will be used to determine the graphic value. For a dark gray pixel on

the next odd row, different bits will be used to determine the graphic value. In this way, no two rows will be modulated exactly the same.

Note that by the time you get to the next even row, the frequencies can be the same, as no flickering will be perceived by the eye with a row difference (in space and in time). The exact values of the gray scale pixel is determined by the values of a GLUT decode table as shown in Figure 3.53.

Essentially, gray scale modulation can be understood if one realizes that each pixel is being refreshed approximately 60 times per second. Naturally, if a pixel is white, the value of zero will be sent 60 times for each second (for that bit. If the pixel is black (or blue), a 1 will be sent for 60 times.

If a pixel is light or dark gray, however, a value of 1 will be sent for a percentage of the time. For example, if a one is sent for 45 times, and a zero is sent for 15 times (during that 60 Hz refresh), a dark gray will appear on the screen.

GLUT WORD

Even Byte		Even Byte	
Even Row	Even Row	Odd Row	Odd Row
dark gray decode nibble	light gray decode nibble	dark gray decode nibble	light gray decode nibble
Bits: D7, D6, D5, D4	Bits: D3, D2, D1, D0	Bits: D15, D14, D13, D12	Bits: D11, D10, D9, D8

NOTE: Graphic bit pairs ON (0,0) and OFF (1,1) are mapped to decoded values of (0) and (1), respectively

LIGHT AND DARK GRAY
GLUT WORD DECODING MAP

Odd/Even graphic bit pair values byte position of bit pairs	even row dark pixel (1,0) encoding	even row light pixel (0,1) encoding	odd row dark pixel (1,0) encoding	odd row light pixel (0,1) encoding
[7:6]	D7	D3	D15	D11
[5:4]	D6	D2	D14	D10
[3:2]	D5	D1	D13	D9
[1:0]	D4	D0	D12	D8

Figure 3-57 GLUT Word Mapping

If a one is sent for 15 times, and a zero is sent for 45 times, a light gray will appear. The GLUT table is used to provide the appropriate ones and zeros needed to display gray scale for any given pixel in the frame. Because of the desire to keep adjacent pixels of the same gray value from blinking in sync (causing flickering), the table values will be varied for even and odd rows. The user is given the opportunity to select their own optimum GLUT values for any given display.

The modulation of the applied voltage pulses to the LCD display module determines the brightness of each pixel.

The gray scale modulation approach used on the NS486SXF is to use gray scale modulation data read from a dynamically updated off-chip lookup table.

The table is maintained in the at-minimum 64K DRAM address space allocated for the LCD controller's bit-map data. The size of the table is programmable from 0-16 words (bits 1- 4 of configuration register 4).

The principle of operation is that after an EOP is generated (by the LCD Controller, following the loading of the last word of bit-map data (i.e. for the end of the row on line 240/200/320)^{Note 1}, the DMA will auto-initialize to the LCD channel's base address. (See DMA Section for explanation of DMA registers and their usage.) On the next DMA transfer to the LCD, the data coming into the LCD controller will be the Gray Scale Lookup Table data (GLUT), except for the case where zero GLUT words are programmed which should be the case for display applications with only 2 gray levels(i.e. on and off, only). The GLUT words coming into the DMA will be counted and only the word used for modulation of the next frame will be

stored. It is identified by a GLUT word address counter that is automatically incremented each new frame. When the GLUT counter reaches the number GLUT words programmed, the LCD controller generates an interrupt^{Note 2} to signal the CPU to update the GLUT in the DRAM. This interrupt can be disabled within the LCD should the current GLUT programming be adequate for an extended time.

The GLUT word for the current frame is transferred from the FIFO to the GLUT register where it is used for bitmap data decoding gray scale modulation. The GLUT word is comprised of 2 light gray and 2 dark gray nibbles of data, where one nibble is for odd rows and the other for even rows. The nibble data stores the value(1 or 0) that should be placed on the D[3] - D[0] data ports for that shade.

NOTE 1: After the LCD EOP is received by the DMA it removes the LCD DACK. [The LCD controller can hold DREQ active during the time the DMA is going through auto-initialization.] Because the LCD is released after sending out an EOP, a higher priority DMA slave could take over the DMA after the LCD is released even though LCD DREQ is still active.

NOTE 2: Each frame is specified to be at least 13.6ms long (at a 73.5Hz frame rate).So, assuming that a 16 word GLUT space has been allocated, the GLUT update interrupt would occur at least every 218ms.

The GLUT is accessed from the first 2 to 16 DRAM word locations pointed to by the base address stored in the dedicated LCD DMA channel's (Channels 1 or 5) base address register. Initially, at LCD enabling, the current and next frame's GLUT data is loaded into the LCD controller's GLUT registers.

After an EOP cycle is complete (a DMA transfer complete signal), the next DMA access will start at the beginning of the LCD memory space where the next frame's GLUT data will be loaded into the GLUT registers. (The next DMA access after an EOP will start at the base address previously loaded when DMA auto-initialization is being used.)

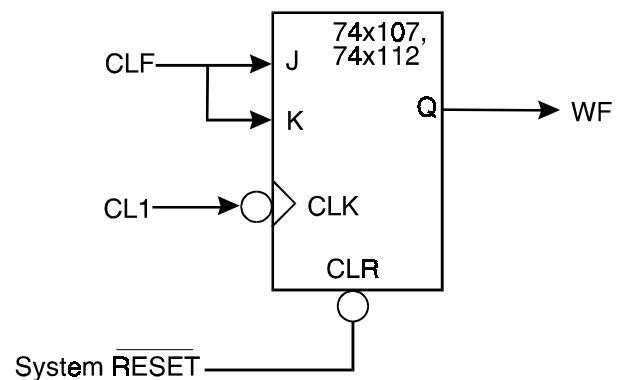
Each frame needs one word of GLUT, so if 16 GLUT words are specified, then it will be 16 frames before the GLUT data will need to be updated by the CPU.

When 16 or whatever number of programmed GLUT words have been reached, an internal GLUT counter generates a CPU interrupt. This interrupt can be programmably turned off within the LCD controller if periodic GLUT updating is not needed. In this case, the current GLUT data is continuously looped through from frame to frame.

3.4.6.3 Indirectly Applied Signals

Some panels require a signal "WF", which is used for switching the polarity of the AC signal that sustains the images on the screen. Where required, it can be provided by a negative edge-triggered J-K flip-flop function, as shown in Figure 3.53. Reference your panel's design documentation to determine if this signal is required.

Figure 3-58 Generating the WF Signal



The V_{EE} (-22V) and V_{DD} (+5V or 3V) supplies, and the V_{OFF} (Contrast control) voltage to the panel must be sequenced to avoid damage and/or degradation of the display. Control of these functions is performed by off-chip power switches, which can be handled by software using an external addressable latch. See Figure 3.54.

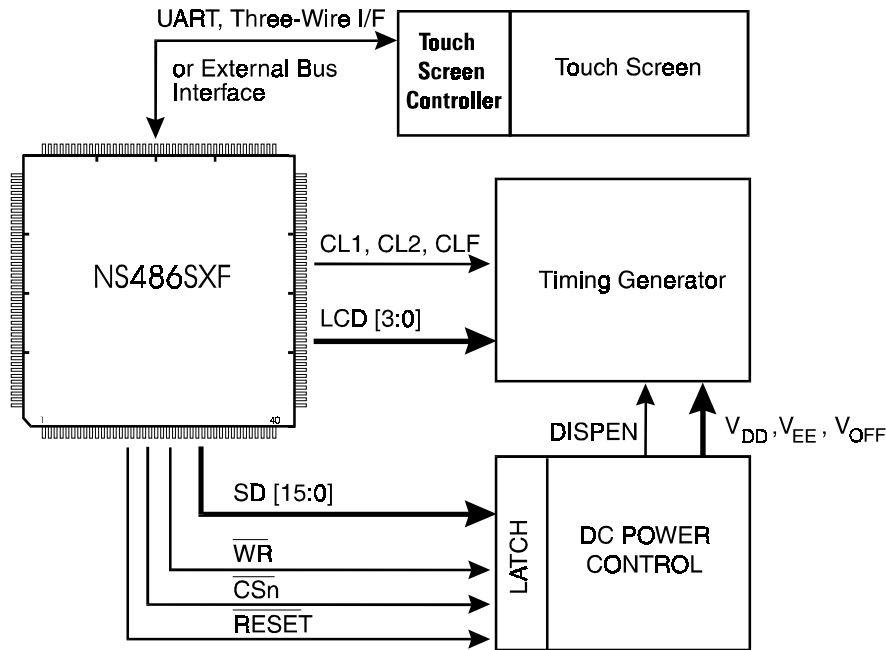
DC voltage drive on the inputs of a powered up display leads to rapid deterioration of the liquid crystal, so after the display is powered up CL1, CL2 and CLF will begin oscillating and LCD[n] data will begin loading into the display. When the display is powered down, the signals CLx and LCDn will be forced to a low DC state to prevent latch-up within the display. A reset forces the LCD controller and the power switches to a powered down state.

In addition, some panels have a Display Enable signal, which must be sequenced with the application and removal of power. This also can be handled by software, using an external addressable latch.

All panels are assumed to present standard CMOS input loading, accepting a V_{IH} of 0.8 VDD and V_{IL} of

0.2 VDD. Some panels specify a large capacitive load on the three pins WF, CLF and CL1. These may require buffering off-chip.

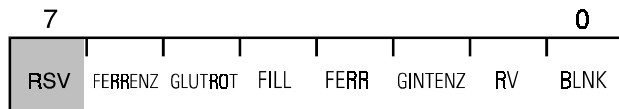
Figure 3-59 Typical LCD and Touch Screen Interface



3.4.6.4 LCD Configuration Registers

There are four LCD configuration registers that control the operation of the LCD controller and provide status information to the CPU. Some bits must be set prior to enabling the LCD controller and left unchanged during LCD operation, other bits can be updated on-the-fly during LCD operation. Updatable bits are bits 7-3 of Configuration Register 2, (controlling the dot clock divisors), bits 3-0 of Configuration Register 3, (controlling the CL1 pulse offset for adjusting the refresh rate), and bits 7-5 of Configuration Register 4, (controlling inverse video and blink rates).

3.4.6.4.1 LCD Configuration Register 1



I/O Map Address

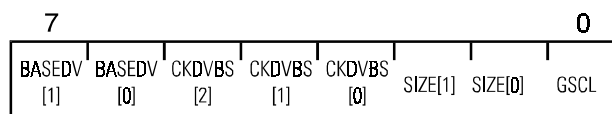
EFA0h

Access

R/W

- Bit 7: Reserved
- Bit 6: FERRENZ — FIFO error interrupt disable selection bit. 1 = Disable LCD FIFO empty interrupt. Reset forces this bit to a “0”.
- Bit 5: GLUTROT — Fixed GLUT word rotation selection bit. 1 = Enables the rotation of the current GLUT word. No new GLUT words are loaded into the current GLUT register when this mode is enabled. At the beginning of each new frame, the even row nibble portions of the GLUT word are shifted right and the odd row nibble portions are shifted left. Reset forces this bit to a “0”.
- Bit 4: FILL — GLUT Interrupt Status bit
1 = Indicates that the DRAM GLUT entries should be updated. Reset forces this bit to a “0”.
- Bit 3: FERR — FIFO Interrupt Status bit
1 = Indicates that the LCD FIFO has run dry. Reset forces this bit to a “0”.
- Bit 2: GINTENZ — GLUT update interrupt disabling selection bit
1 = Disables the interrupt for signaling DRAM GLUT entry updates. Reset forces this bit to a “0”.
- Bit 1: RV — Reverse Video enable selection bit
1 = Enables reverse video images on the LCD screen. Reset forces this bit to a “0”.
- Bit 0: BLNK — Blink enable selection bit
1 = Enables blinking images on the LCD screen. Reset forces this bit to a “0”.

3.4.6.4.2 LCD Configuration Register 2



I/O Map Address

EFA1h

Access

R/W

Bits 7,6: BASEDV[1:0] — Binary clock division of basis selection. Reset forces these bits to “1”. **Do not use 00h (/1) for Oscillator frequencies of 16MHz and above.**

BASEDV[1]	BASEDV[0]	Binary Division
0	0	/1
0	1	/2
1	0	/4
1	1	/8

Bits 5-3: CKDVBS[2:0] — Binary clock division of basis selection Reset forces these bits to “0”

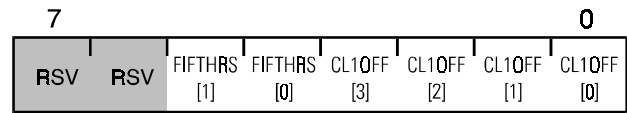
CKDVBS [2]	CKDVBS [1]	CKDVBS [0]	Integer Division
0	0	0	/2
0	0	1	/3
0	1	0	/5
0	1	1	/7
1	0	0	/9
1	0	1	/11
1	1	0	/13
1	1	1	reserved

Bits 2,1: SIZE[1:0] — Screen size selection Reset forces these bits to “0”

3.4.6.4.3 LCD Configuration Register 3

SIZE[1]	SIZE[0]	Screen Size
0	0	320 x 240
0	1	320 x 200
1	0	480 x 320
1	1	reserved

Bit 0: GSCL — 1 or 2 bit per pixel selection
 1 = 2 bit per pixel gray scale encoding
 0 = 1 bit per pixel gray scale encoding
 Resets forces this bit to a “0”.



I/O Map Address

EFA2h

Access

R/W

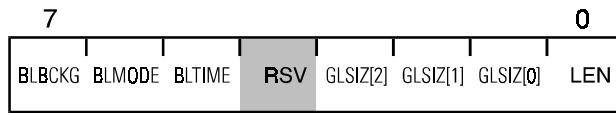
Bits 7,6: Reserved

Bits 5,4: FIFTHRS[1:0] — Fraction LCD FIFO may empty before a DRQ is generated
 Reset forces these bits to “0”

FIFTHRS[1]	FIFTHRS[0]	FIFO fill threshold
0	0	Eighth
0	1	Quarter
1	0	Half
1	1	Three_quarters

Bits 3,0: CL1OFF[3:0] — Selection of CL1 pulse offset after the last CL2 shift clock. A single offset is equal to one period of the CL2 clock. The number of offsets is equal to the binary equivalent of CL1OFF [3:0] + 1. This provides for a range of 1 to 16 offsets. Reset forces these bits to “0”.

3.4.6.4.4 LCD Configuration Register 4



I/O Map Address

EFA3h

Access

R/W

1 = Enables the LCD controller (clocks and data lines are active)
 0 = Disables the LCD controller (clocks and data lines are held low)
 Reset forces this bit to a “0”.

- Bit 7: BLBCKG — Background shade selection bit for blinking
 1 = Background shade is “1”
 0 = Background shade is “0”
 Reset forces this bit to a “0”.
- Bit 6: BLMODE — Blink to inverse video or background selection bit
 1 = Blink to inverse video
 0 = Blink to the background shade
 Reset forces this bit to a “0”.
- Bit 5: BLTIME — Period of blink selection bit
 1 = Blink period = 72 frames(50/50 duty cycle)
 0 = Blink period = 36 frames(50/50 duty cycle)
 Reset forces this bit to a “0”.
- Bit 4: Reserved.
- Bits 3-1: GLSIZ[2:0] — GLUT table size in DRAM from 0-16 words. The table size is selected by the value of GLSIZ[2:0] (possible values are: 0, 2, 4, 6, 8, 10, 12, and 16). Reset forces these bits to “0”

GLSIZ[2]	GLSIZ[1]	GLSIZ[0]	# GLUT Words
0	0	0	0
0	0	1	2
0	1	0	4
0	1	1	6
1	0	0	8
1	0	1	10
1	1	0	12
1	1	1	16

Bit 0: LEN — LCD controller enable selection bit

4.0 The System Bus

4.1 Bus Interface Unit (BIU)

The system bus, through a Bus Interface Unit (BIU) provides the interface for the CPU to on-chip peripherals, to memory and to external peripherals, such as a hard disk, a floppy disk or a modem. Furthermore, the BIU provides the necessary Chip Select logic and signals for those external peripherals. Wait states are supported through the Chip Select logic and by external peripheral feedback signaling. Though the bus does not fully emulate the ISA bus, it is capable of directly interfacing to most ISA peripheral control devices. See Section 4.2 for a more complete discussion of ISA-bus interfacing issues.

The BIU decodes all CPU accesses to internal and external resources and generates the appropriate control signals based on the type of access as well as the address of the access. The BIU always monitors the CPU local bus and if the access is to DRAM, the BIU will suspend operations and let the DRAM controller take over and generate the appropriate DRAM control signals. For all other accesses, the BIU will take control and generate the necessary control signals.

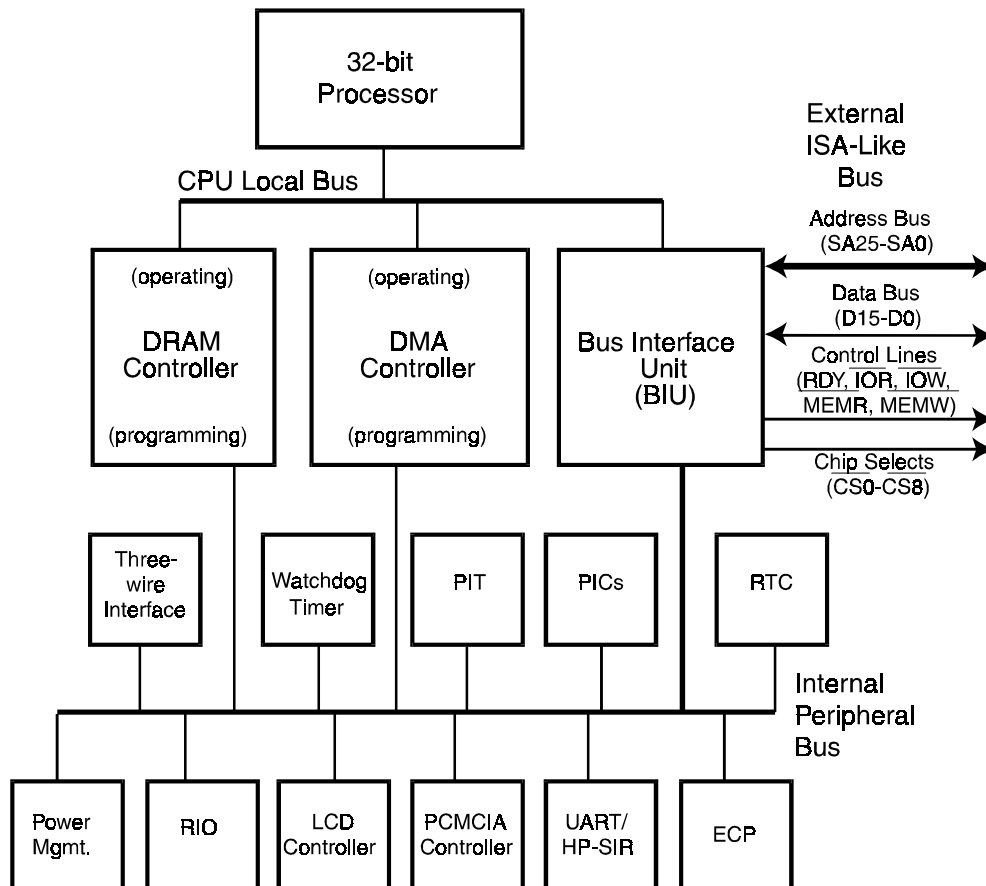


Figure 4-1 NS486SXF Internal Buses

The Bus Interface Unit contains a number of control registers (explained later in this section) that allow the user to:

- 1) Program chip selects (A set of Chip Select Address and Mask Registers, in conjunction with a Chip Select Enable Register and a Chip Select Access Type Register, control the generation of external chip select signals.)
- 2) Program which chip selects decode cacheable memory range(s) (via the Cacheable Chip Selects Register).
- 3) Program which chip selects are connected to 16-bit devices (via the 16-bit Logical Chip Select Register).
- 4) Extend the timing characteristics of control signals to the ISA-like external bus (via the Chip Select Access Time Registers).
- 5) Enable accesses to the following internal resources:
 - a) The Three-wire interfaces (MICROWIRE and Access.bus)
 - b) The UART
 - c) The DMA Controller
 - d) The Interrupt Controllers
 - e) The Programmable Interval Timer
 - f) The Real Time Clock
 - g) The PCMCIA controller
 - h) The ECP port(via the Bus Interface Unit Control Registers 1 and 2).
- 6) Monitor internal bus cycles to assist in tracing and debugging internal accesses (via a bit in the Bus Interface Unit Control Register 1).

4.1.1 Internal Peripheral Accesses

The BIU decodes all CPU accesses to internal peripherals and generates the appropriate internal address and control signals. In the default configuration of the BIU, no external indication of these internal bus cycles will be made; this mode of operation provides for lower power consumption. However, if the Internal Cycle Debug bit (ICD, bit 0 in the BIU Control Reg-

ister 1) is set to a one, an external bus cycle will be performed at the same time the internal access is being made; this mode of operation provides information indicating when accesses are being made to the various internal peripherals. Furthermore, during reads from internal peripherals, with ICD = 1, the value being read by the CPU will be driven out on the SD15-0 data bus pins. It is important that no external peripheral drives SD15-0 during these bus cycles.

Accesses are made to internal peripherals in three CPU T-states; this corresponds to the shortest possible external bus access timing.

4.1.1.1 NS486SXF Peripheral Control Registers

The peripheral functions within the **NS486SXF** may be broken into two general categories: **NS486SXF** specific functions and PC industry standard functions.

The control registers for all of the **NS486SXF** specific functions reside in the I/O address range EF00h-EFFFh. These I/O addresses will always be accessible by the **NS486SXF** CPU and must never be used by externally connected peripherals.

The PC industry-standard functions have historically resided in certain I/O address ranges. To remain consistent with the historical implementation, the **NS486SXF** has whenever possible mapped the PC industry standard functions into their historical I/O address ranges.

An issue with using the historical I/O address ranges is that if one of these peripherals is not to be used, its I/O address range may conflict with an I/O address range of an external peripheral. To eliminate this **potential** problem, following an IC reset, the access to all PC industry standard peripherals will be disabled. A one must be written to the appropriate enable bits in the BIU Control Register 1 and the BIU Control Register 2, to enable access to each internal peripheral. When an internal peripheral is disabled, accesses to its address range will become external bus cycles.

The BIU Control Registers 1 and 2, only enable/disable the ability of the CPU to access these internal peripherals and does not enable/disable the function itself. This fact is especially important with regards to the Real Time Clock (RTC), since it may be battery

backed and will continue to operate even when system power is lost. When the RTC is battery backed, the RTC will continue to operate after an IC reset, but the CPU must enable its access to the internal RTC via the BIU Control Registers 1 and 2 before the CPU will be able to read and write to the RTC.

4.1.1.2 Global Enable

Bit 2 (GPE) of BIU Control Register 2 (I/O address EF01h) is a global access enable bit for PC industry-standard internal peripherals. When GPE is a zero, the CPU will not be able to read or write these internal peripherals. When GPE is a one, the access enable bit associated with each peripheral will determine if the peripheral may be accessed or not.

4.1.1.3 Accessing The Internal UART

The NS486SXF internal UART can be configured to reside in the four different industry standard COM Port address ranges. Bits 7 and 6 (UART_S1 and UART_S0, respectively) of the BIU Control Register 2 determine which of the four possible I/O address ranges is used.

UART_S1	UART_S0	I/O Address Range
0	0	03F8h-03FFh (COM1)
0	1	02F8h-02FFh (COM2)
1	0	03E8h-03EFh (COM3)
1	1	02E8h-02EFh (COM4)

4.1.1.4 Accessing The Internal Extended Capabilities Port (ECP)

The NS486SXF internal ECP can be configured to reside at three different industry standard I/O address ranges for Parallel Ports. Bits 5 and 4 (ECP_S1 and ECP_S0, respectively) of BIU Control Register 2 determine which of the three possible I/O address range combinations are used.

ECP_S1	ECP_S0	I/O Address Ranges
0	0	0278h-027Ah
		0678h-067Dh
0	1	0378h-037Ah
		0778h-077Dh
1	0	03BCh-03BEh
		07BCh-07C1h
1	1	reserved

The registers associated with I/O addresses 067Bh-067Dh(option 1), 077Bh-077D(option2) and 07BFh-07C1h(option 3) define new options for the ECP. Refer to section 3.4.2 on page 107 for more information regarding their functions.

4.1.1.5 Accessing Other Internal Peripherals

The PC industry standard peripherals which only map into a single set of address ranges include the Three-Wire Interfaces, the Timer, the Real Time Clock, the Interrupt Controllers and the DMA Controller. When they are enabled, these peripherals map into the following I/O address ranges.

	I/O Address Range
Three-Wire	0050h-0057h
Timer	0040h-0047h
Real Time Clock	0070h-0071h
Interrupt Controllers	0020h-0021h
“	00A0h-00A1h
DMA Controller	0000h-001Fh
“	0080h-008Fh
“	00C0h-00DFh

4.1.1.6 Accessing The Internal PCMCIA Controller

The NS486SXF internal PCMCIA Controller can be configured so that its control registers may be accessed in two different locations. Bit 3 (PCM_S0) of BIU Control Register 2 determines which I/O address range is used.

CM_S0	I/O Address Range
0	03E0h-03E1h
1	03E2h-03E3h

4.1.2 PCMCIA Memory Accesses

The **NS486SXF** PCMCIA Controller is exactly compatible with the Intel82365SL PC Card Interface Controller. Typically, this type of PCMCIA Controller connects to an ISA bus. The ISA bus only generates bus cycles for memory mapped into the least significant 16 Mbytes of the CPU's memory range. As a result, this PCMCIA Controller type usually only translates accesses to the least significant 16 Mbytes of memory to the inserted PCMCIA Card. To avoid this limitation, the **NS486SXF** provides an 8-bit register that allows software to control the base memory address translated by the internal PCMCIA Controller.

The PCMCIA Memory Base Address Register is located at I/O address EF4Eh and is reset to 00h by an IC reset. **The eight most significant address bits of any CPU memory access (CPU Address 31-24) must match the value in this register or the BIU will not generate memory access strobes to the internal PCMCIA Controller and the memory cycle will be sent out onto the ISA-like bus by the BIU.**

The reset value of the PCMCIA Memory Base Address Register (00h), defaults to the least significant 16 Mbytes of CPU memory. Each of the 256 possible values that may be written into this register selects an unique 16 Mbyte range of memory in the 4 Gbyte memory range of the CPU.

4.1.2.1 PCMCIA Access Timing

The BIU and the PCMCIA Controller share the ISA-like bus address (SA25-0) and control signals (\overline{IOR} , \overline{IOW} , \overline{MEMR} and \overline{MEMW}).

The BIU will take control of all CPU bus cycles not claimed by the DRAM Controller. To determine if such a cycle is meant for the PCMCIA Controller, the BIU will insert two extra command delay T-states into the bus cycle when access to the PCMCIA Controller has been enabled. These two command delay T-states provide the PCMCIA Controller with enough time to determine if the cycle belongs to it or not.

If the access is not for the PCMCIA Controller, then the BIU will perform the appropriate bus cycle. It should be noted that if access to the PCMCIA Controller is not enabled (implying that the PCMCIA

Controller is not to be used), then the BIU will not insert the two command delays; instead the BIU will perform the appropriate bus cycle without delay.

If the access is for the PCMCIA Controller, the BIU will give control of the ISA-like bus address and control signals over to the PCMCIA Controller and the PCMCIA Controller will perform the appropriate access cycle.

NOTE: It is suggested that one does not program any of the logical chip selects to coincide with the I/O or memory ranges supported by the PCMCIA Controller. If one does program a logic chip select to coincide with any I/O or memory ranges supported by the PCMCIA Controller, one must program the logical chip select to have one command delay. If this is not done, the PCMCIA accesses will not work correctly.

The fundamental operating frequency of the PCMCIA Controller is determined by Bits 0 and 1 (PCCLK0 and PCCLK1, respectively) of the PCMCIA Clock Selection Register (I/O address EF4Fh). The PCMCIA Controller may be programmed to operate at one-fourth, one-third or one-half the frequency of the **NS486SXF** CPU. When the CPU enters a power saving mode, the clock to the PCMCIA controller will also slow down respectively.

PCCLK1	PCCLK0	Operating Frequency
0	0	CPUCLK/4
0	1	CPUCLK/3
1	X	CPUCLK/2

NOTE: The reset setting is CPUCLK/4. In an ISA bus system, the PCMCIA Controller is usually driven by SYSCLK, which is a ~8 MHz clock. Also, the switching of the clock frequency into the PCMCIA Controller is guaranteed to be glitch free, so one may switch the frequency of operation at any time.

4.1.3 Interrupt Acknowledge Cycles

The BIU decodes all interrupt acknowledge cycles for both internal and external interrupt controllers. The Interrupt Acknowledge signal (\overline{INTA}) strobes active low during each interrupt acknowledge cycle. Simultaneously, the three most significant bits of the System Address (SA25-23) will be driven with the three Cascade Line signals (CAS2-0, respectively) from the internal master interrupt controller. In this manner an external interrupt controller can determine if it is supposed to respond with the appropriate interrupt vector.

When a cascaded external interrupt controller provides the appropriate interrupt vector (during the second \overline{INTA} strobe) in an interrupt acknowledge sequence, the BIU reads the vector from the SD7-0 pins and presents it to the CPU as required.

Likewise, when an internal interrupt controller provides the appropriate interrupt vector (during the second \overline{INTA} strobe) in an interrupt acknowledge sequence, the BIU will read that 8-bit vector and present it to the CPU as required.

Refer to Appendix D: External Interrupt Controller for information regarding how a cascaded external interrupt controller should be connected to **NS486SXF**.

4.1.4 HALT Cycles

The BIU decodes and acknowledges a HALT cycle from the CPU, but no other action will be taken.

4.1.5 SHUTDOWN Cycles

The BIU decodes and acknowledges a SHUTDOWN cycle from the CPU and when the SHUTDOWN Reset bit (SHR, bit 1 of the BIU Control Register 1) is a one, an internal CPU reset will be performed. When SHR is a zero, the BIU will decode and acknowledge a SHUTDOWN cycle, but no CPU reset will be performed.

When a SHUTDOWN cycle results in the generation of a CPU reset, only the CPU will be reset, the on chip peripherals will not be reset and the RESET and \overline{RESET} pins will remain in their inactive states.

4.2 External Bus Cycles

To reduce pin count, the DRAM controller and BIU share address and data pins (SA12-1 and SD15-0). DRAM accesses are more frequent than any other, so to optimize the DRAM access times, the logic is designed to anticipate a DRAM access cycle. When a DRAM access is made, the minimum number of T-states are used. However, when a non-DRAM access is performed the first T-state of each cycle allows control of the access to be transferred from the DRAM Controller to the Bus Interface Unit. Figure 4-6 shows the timing diagram associated with the shortest possible non-DRAM read cycle.

4.2.1 ISA-like Bus Operation

NS486SXF's External Interface Bus can be described as ISA-like because **NS486SXF's** external bus can operate with most ISA bus peripherals and it meets many of the most widely used specifications of the ISA bus. It is not exact, however, and individual usage must be examined to ensure adequate compatibility.

Figure 4-2 ISA-like Bus Cycle Timing shows the basic functional signal relationships of the **NS486SXF** ISA-like bus.

To insert extra wait states into a ISA-like bus cycle, an external device may pull the ready pin (RDY) inactive low. The BIU will insert wait states indefinitely, until the RDY signal returns high. Typically, in the system design, the RDY signal is pulled up with a resistor and multiple devices can drive this signal with open-collector, open-drain or TRISTATE drives; thus allowing multiple devices the ability to insert additional wait states via the same input signal pin. Figure 4-11 through Figure 4-13 on page 187 illustrate the insertion of wait states using the RDY feedback signal.

The **NS486SXF's** Bus Interface Unit (BIU) also accepts a feedback signal $\overline{CS16}$, which functions similarly to the $\overline{TOCS16}$ and $\overline{MEMCS16}$ signals on the ISA bus. One important difference between the **NS486SXF's** $\overline{CS16}$ and the ISA bus signals is that the **NS486SXF** samples $\overline{CS16}$ near the end of the command strobe (i.e. \overline{MEMR} , \overline{MEMW} , \overline{IOR} , \overline{IOW}), rather than the leading edge as in the ISA bus. This is done to allow the user to gate $\overline{CS16}$ with the command strobe for qualification. When, the $\overline{CS16}$ signal is

driven active low, the BIU will perform a 16-bit device access for the present cycle. On the other hand, if $\overline{CS16}$ is inactive high (indicating an 8-bit device), the BIU will steer the ISA-like bus to perform 8-bit transfers on the lower byte of the System Data Bus (SD7-0). The BIU will also perform any required 16-bit to two 8-bit cycle translations, similar to a standard ISA bus controller.

Note: The Programmable 16-bit Logical Chip Select Register (I/O address EF57h) contains control bits that affect 16-bit accesses. When any Chip Select bit in this register is set to 1, its associated logical chip select access will be treated as a 16-bit access regardless of the state of the $\overline{CS16}$ signal. This is to provide support for memory devices (EPROM's, SRAMs, etc.) that would require extra external logic to generate the appropriate feedback signal on the $\overline{CS16}$ pin. This allows 16-bit memories to be connected to the **NS486SXF** with no additional external logic.

Figure 4-14 depicts the timing of a 16-bit access being broken into two 8-bit cycles (the minimum cycle times are assumed for this figure).

Other ISA-like Bus Features

Any of the standard ISA Bus Interrupt Requests (i.e. IRQ15, IRQ14, IRQ12, IRQ11, IRQ10, IRQ9, IRQ7, IRQ6, IRQ5, IRQ4 and IRQ3) may be steered to the six interrupt pins provided on the **NS486SXF** (refer to the Interrupt Source Selection section). The **NS486SXF** only supports six interrupt requests at any one time, so only six of the eleven of the ISA bus interrupt requests can be supported at the same time.

The 'x86 architecture supports byte, word, and double word accesses to memory. The DRAM controller automatically handles these accesses to DRAM, and the BIU will convert 32-bit accesses into sequential 8 or 16 bit accesses. However, external 16-bit memory must support both 8 and 16 bit accesses. The SA0 and \overline{SBHE} signals can be used during memory and I/O cycles to determine the width of an access. The 16-bit memory system should interpret these signals as follows:

SA0	\overline{SBHE}	Meaning
0	0	16-bit transfer
1	0	8-bit transfer (odd/high byte)
0	1	8-bit (even/low byte)

4.2.2 ISA-like Bus Timing

This section provides information about the ISA-like bus timing.

Note: The numbers used in this text are approximations, and are provided only to give the user a better understanding of the design.

Important ISA and NS486SXF terms include:

Command Delay Time — This is the amount of time from a valid address until a command strobe (i.e. $\overline{\text{IOR}}$, $\overline{\text{IOW}}$, $\overline{\text{MEMR}}$ or $\overline{\text{MEMW}}$) goes active low. See Section 4.4.6 on page 184 and Section 4.5.7 on page 193.

Access Time — This is the amount of time that a command strobe (i.e. $\overline{\text{IOR}}$, $\overline{\text{IOW}}$, $\overline{\text{MEMR}}$ or $\overline{\text{MEMW}}$) is active low. See Section 4.4.6 on page 184 and Section 4.5.7 on page 193.

Address Hold Time — This is the amount of time that a valid address is held after the trailing edge of a command strobe (i.e. $\overline{\text{IOR}}$, $\overline{\text{IOW}}$, $\overline{\text{MEMR}}$ or $\overline{\text{MEMW}}$).

Data Hold Time — This is the amount of time that the write data is held after the trailing edge of a write command strobe (i.e. $\overline{\text{IOW}}$ or $\overline{\text{MEMW}}$).

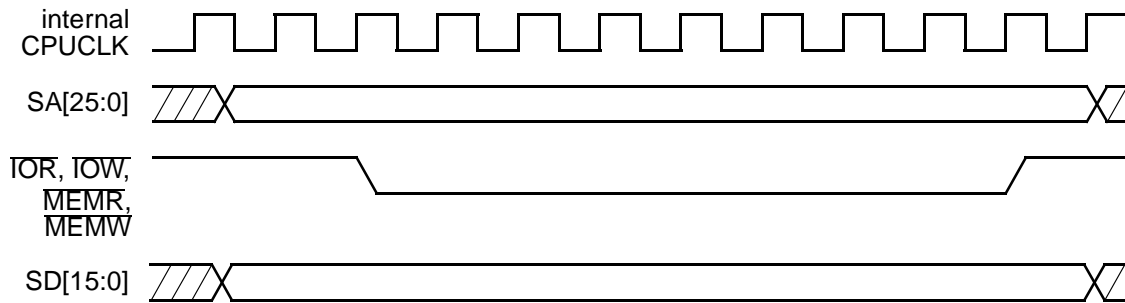
RDY Extension Time — This is the amount of time from the RDY signal going active high until the trailing edge of a command strobe (i.e. $\overline{\text{IOR}}$, $\overline{\text{IOW}}$, $\overline{\text{MEMR}}$ or $\overline{\text{MEMW}}$). See “Boot ROM Access Time Register” on page 195.

To be the most ISA-Like, the following options should be selected:

- 1) Maximum number of Command Delay States (one)
- 2) Maximum number of Programmed Wait States (seven)
- 3) Maximum number of RDY Extension Clock Periods (two)

The timing diagram in Figure 4-2 shows the result when RDY remains active high.

Figure 4-2 ISA-like Bus Cycle Timing



In general the **NS486SXF**'s ISA-like External Interface bus may be programmed to support a wide range of ISA Bus components and boards.

The following numerical calculations assume an internal CPU operating frequency of 25 MHz.

NS486SXF Command Delay Time:

$$\begin{aligned}
 &= (\text{Programmed \# of Command Delays} \\
 &\quad + 0.5) \times (\text{CPUCLK period}) \\
 &= \sim 1.5 \text{ CPUCLK periods} \\
 &= 1.5 (40 \text{ nsec}) \\
 &= 60 \text{ nsec}
 \end{aligned}$$

For ISA Command Delay time:

$$\begin{aligned}
 &= \sim 40 \text{ nsec for 16-bit boards or devices} \\
 &= \sim 100 \text{ nsec for 8-bit boards of devices}
 \end{aligned}$$

NS486SXF Access Time:

$$\begin{aligned}
 &= (\text{Programmed \# of Wait States} + 1) \times \\
 &\quad (\text{CPUCLK period}) \\
 &= \sim 8 \text{ CPUCLK periods} \\
 &= 8(40 \text{ nsec}) \\
 &= 320 \text{ nsec}
 \end{aligned}$$

For ISA Access Time:

$$\begin{aligned}
 &= \sim 170 \text{ nsec for 16-bit IO boards or} \\
 &\quad \text{devices} \\
 &= \sim 225 \text{ nsec for 16-bit Memory boards} \\
 &\quad \text{or devices} \\
 &= \sim 500 \text{ nsec for 8-bit boards or devices}
 \end{aligned}$$

NS486SXF Address Hold Time:

$$\begin{aligned}
 &= \sim 1 \text{ CPUCLK period (always)} \\
 &= 40 \text{ nsec}
 \end{aligned}$$

For ISA Address Hold Time:

$$= \sim 40 \text{ nsec}$$

NS486SXF Data Hold Time:

$$\begin{aligned}
 &= \sim 1 \text{ CPUCLK period (always)} \\
 &= 40 \text{ nsec}
 \end{aligned}$$

For ISA Data Hold Time:

$$= \sim 35 \text{ nsec}$$

NS486SXF RDY Extension Time:

$$\begin{aligned}
 &= (\text{Programmed \# of RDY Extension} \\
 &\quad \text{Clock Periods}) \times (\text{CPUCLK period}) \\
 &= \sim 2 \text{ CPUCLK periods} \\
 &= 2(40 \text{ nsec}) \\
 &= 80 \text{ nsec}
 \end{aligned}$$

For ISA RDY Extension Time:

$$= \sim 120 \text{ nsec}$$

4.3 Device Configuration at RESET

At the end of a POWERGOOD (PWGOOD) reset, the **NS486SXF** must select each of the following functions to be supported:

- 1) 8-bit Boot-up ROM BIOS
vs. 16-bit Boot-up ROM BIOS
- 2) The **NS486SXF** operates in normal mode vs. the **NS486SXF** will TRI-STATE all of its output drivers because it recognizes that an In Circuit Emulator (ICE) pod is attached to the system.

These two functions must be determined before any code is fetched or executed. To achieve this requirement, two pins which are normally outputs or reserved ($\overline{\text{SBHE}}$, and SA[0]), become inputs while the PWGOOD input signal is inactive low. Upon the rising edge of the PWGOOD signal, the values on these pins will be latched into two separate registers, which control the above functions as discussed below. These two functions are considered “strapping options” because they are set immediately upon coming out of reset.

4.3.1 Boot-up ROM BIOS Size

When the PWGOOD signal is inactive low (indicating the **NS486SXF** is being reset), the $\overline{\text{SBHE}}$ pin becomes an input. If $\overline{\text{SBHE}}$ is pulled high with a 10 Kohm resistor, an internal latch will latch in a one upon the rising edge of PWGOOD, that selects 16-bit Boot-up ROM BIOS support. On the other hand, if $\overline{\text{SBHE}}$ is pulled low by a 10 Kohm resistor, the internal latch will latch in a zero, selecting 8-bit Boot-up ROM BIOS support.

When 16-bit Boot-up ROM BIOS is selected, all memory accesses to the boot-up address segment FFFF0000h-FFFFFFFFh will be performed as 16-bit accesses with all even address code bytes fetched on SD[7:0], while all odd address code bytes would be fetched off of SD[15:8]. This mode of operation fetches code twice as fast as 8-bit Boot ROM BIOS, but it also requires either two 8-bit EPROMs or a 16-bit wide EPROM.

When 8-bit Boot-up ROM BIOS is selected, all memory accesses to the boot-up address segment FFFF0000h-FFFFFFFFh will be performed as 8-bit accesses with all code bytes fetched on SD[7:0]. This reduces the component count. Also it should be noted that in some systems, once the **NS486SXF** has booted up, it will no longer access this memory range, so doubling the boot-up time may not have any impact on the system performance at other times.

The 10 Kohm pull-up or pull-down resistor will have no impact on the normal operation of the **NS486SXF** after PWGOOD becomes active high. Following a small delay after PWGOOD rises, the $\overline{\text{SBHE}}$ signal will become a driven output in the **NS486SXF** normal mode of operation.

4.3.2 Normal Mode vs. ICE Mode

When the PWGOOD signal is inactive low (indicating the **NS486SXF** is being reset), the SA[0] pin becomes an input. If left floating, an internal latch will latch in a one upon the rising edge of PWGOOD, that selects **NS486SXF** normal mode (there is a weak pull-up resistor internal to the **NS486SXF**). On the other hand, if SA[0] is driven low during this time the internal latch will latch in a zero. If a zero is latched from SA[0] and the **TEST** signal is asserted (low) then the **NS486SXF** will enter ICE TRI-STATE Mode.

NS486SXF Normal Mode is the normal operation of the **NS486SXF** that is discussed throughout this Data Sheet.

NS486SXF ICE TRI-STATE Mode results in all of the output drivers on the **NS486SXF** being placed in TRI-STATE. An In-Circuit-Emulator should be the only thing that drives SA[0] low via its pod connection. The **TEST** pin should be pulled high through a resistor in the target design, so that the ICE can assert it low as well. An ICE may now be directly connected to a board under development without removing the **NS486SXF** in that system.

After a small delay from when PWGOOD rises, the SA[0] signal will become a driven output in the **NS486SXF** normal mode of operation.

4.4 Chip Select Logic

There are nine external chip select pins on the **NS486SXF** ($\overline{CS}[0]$ - $\overline{CS}[8]$). Chip Select 0 ($\overline{CS}[0]$) should be connected to the system Boot ROM. The other eight programmable chip selects may be used to interface to external slave devices connected to the **NS486SXF** interface bus. Chip Select 0 ($\overline{CS}[0]$) is a unique chip select that is designed to generate the chip select for the system boot ROM. As a result, $\overline{CS}[0]$ will always go active for any CPU access to the CPU's reset address segment (that is the most significant 64 Kbytes of CPU memory address, FFFF0000h - FFFFFFFFh). Additional memory ranges may be added to $\overline{CS}[0]$ via the programmable chip select logic. Reference the following sections and the Boot ROM Selection Register for more information regarding how the additional memory range(s) may be programmed to the $\overline{CS}[0]$ Chip Select pin.

4.4.1 Boot ROM Data Bus Size

The \overline{SBHE} pin of the **NS486SXF** is normally an output, but during a system reset (when PWGOOD equals zero), \overline{SBHE} will be an input to the **NS486SXF**. Externally, this signal should be pulled up or down via a 10 Kohm resistor. When the PWGOOD signal rises, the value on the \overline{SBHE} pin will be latched into an internal strapping latch. The value of this \overline{SBHE} strapping latch will determine if the **NS486SXF** will access the Boot ROM memory address range (FFFF0000h-FFFFFFFh) with 16-bit cycles or 8-bit cycles.

\overline{SBHE} Strapping Latch	Boot ROM Size
0	8-bit
1	16-bit

An 8-bit ROM size indicates that each byte of the Boot ROM (memory address range FFFF0000h-FFFFFFFh) will be accessed via the low byte of the system data bus (SD7-0).

A 16-bit ROM size indicates that all even address bytes of the Boot ROM will be accessed via the low byte of the system data bus (SD7-0) and all of the odd address bytes of the Boot ROM will be accessed via the high byte of the system data bus (SD15-8). This

will allow the **NS486SXF** to access twice as many bytes as it would in the 8-bit access mode.

By implementing this feature, the **NS486SXF** can boot-up out of 8-bit or 16-bit Boot ROM without any additional hardware to generate the external $\overline{CS}[0]$ feedback signal.

4.4.2 Other Chip Selects' Data Bus Size

The logic associated with the programmable chip selects also provides an 8-bit register which allows software to select whether each logical chip select supports 16-bit data size independent of the $\overline{CS}[0]$ feedback signal. This 8-bit register is located at I/O address EF57h and is called the Programmable 16-bit Logical Chip Select Register. For more information regarding the function of each bit of this register, refer to its register description in section 4.5.10 on page 199.

By providing this feature, the **NS486SXF** provides a way that the system designer can connect 16-bit external memories (EPROMs or SRAMs) to the ISA-like bus without any additional hardware to generate the external $\overline{CS}[0]$ feedback signal.

The system designer may connect a 16-bit memory to the ISA-like bus and have the software program a logical chip select to decode the appropriate address range for the memory, as well as setting the appropriate bit in the Programmable 16-bit Logical Chip Select Register. As a result the **NS486SXF** will always generate the appropriate memory access signals to the 16-bit memory; placing all of the even byte data on SD7-0 and all of the odd byte data on SD15-8.

4.4.3 Programmable Chip Select Architecture

NS486SXF chip select pins ($\overline{CS}[8:0]$) are driven using a two step process. First, a "Logical" chip select is created using an address range and the access type (I/O or Memory). Then a single logical chip select or combination of logical chip selects can be used to drive a given external Chip Select pin. This allows a single Chip Select pin to select a device such as the National PC87332 SuperI/O™ multifunction peripheral controller. These devices contain multiple I/O controllers (dual UARTs, a floppy disk controller, an ECP port, and a hard disk controller) in a single package. With the **NS486SXF** chip select structure, it can

activate a single Chip Select pin with accesses to several different I/O address ranges. This process can be seen in Figure 4-3 and Figure 4-4.

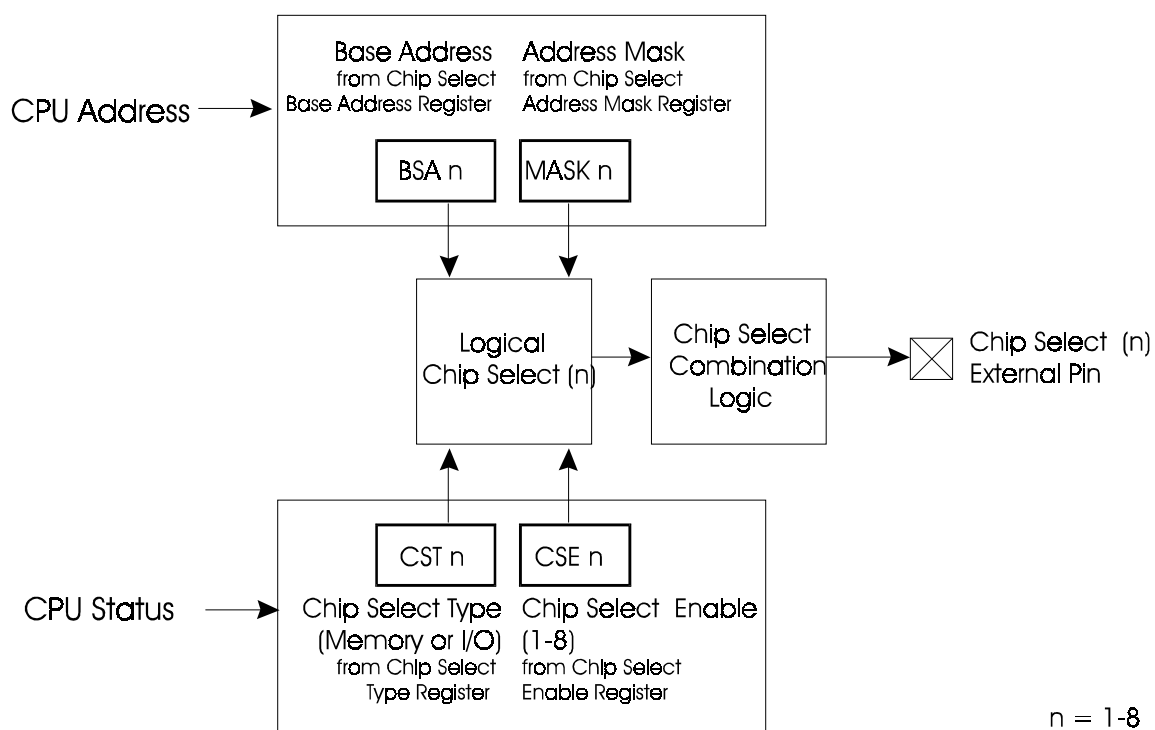


Figure 4-3 Logical Chip Selection

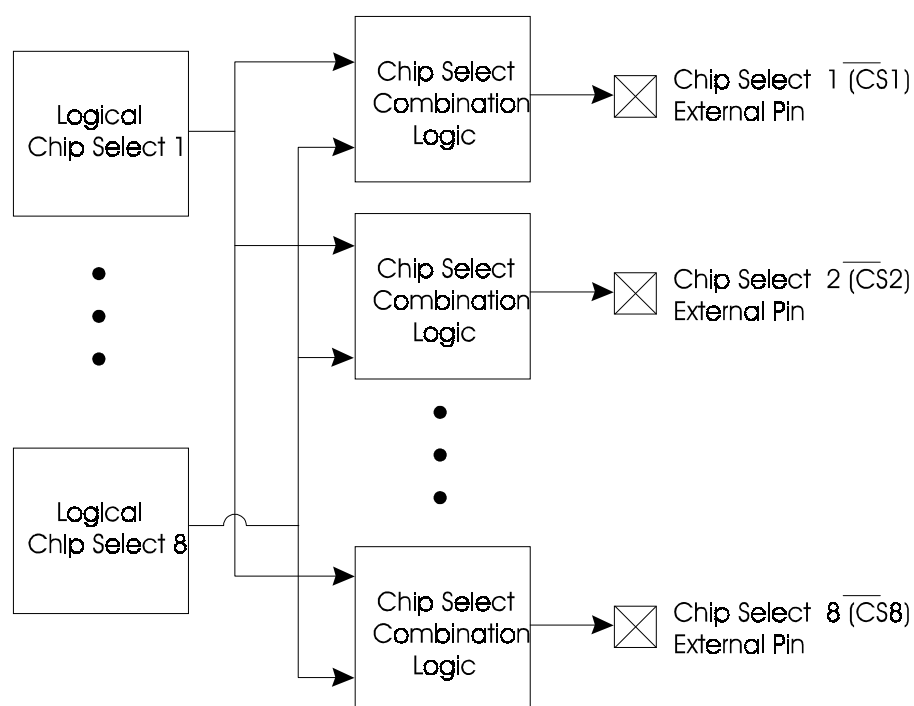


Figure 4-4 External Chip Selection

4.4.4 Programming Logical Chip Selects

When programming a Logical Chip Select, two functions must be programmed:

- 1) The type of access (I/O or Memory) associated with the chip select.
- 2) The address range associated with the chip select.

The user must program the 8-bit Chip Select Type Register (I/O address EF03h) to set the type of access associated with each Logical Chip Select. Bit 0 corresponds to Logical Chip Select 1, bit 1 corresponds to Logical Chip Select 2, and so on, up to bit 7 and Logical Chip Select 8. If a bit of the Chip Select Type Register is a 0, then the Logical Chip Select associated with that bit will decode I/O addresses; if that bit is a 1, then the associated Logical Chip Select will decode Memory addresses.

To program the active address range associated with a Logical Chip Select, the user must program its 32-bit Base Address Register (at I/O addresses EF04h-EF23h) and its 26-bit Address Mask Register (I/O addresses EF24h-EF43h).

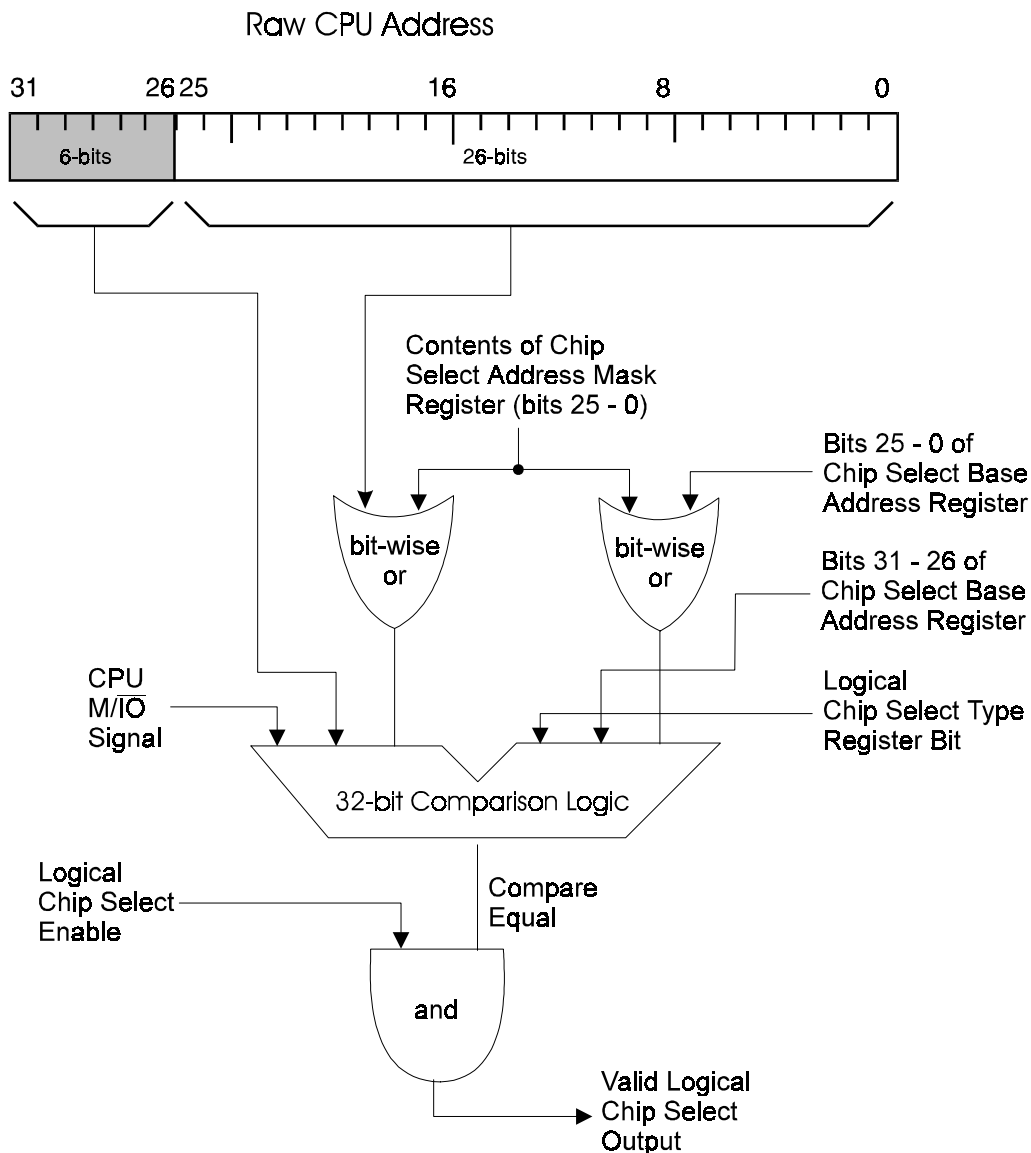


Figure 4-5 Address Masking

The Base Address Registers provide a complete 32-bit address that is compared to CPU generated addresses. The Address Mask Registers are 26-bit in size (since only the lower 26 address bits are driven off-chip) and they define the size or range of addresses associated with each Logical Chip Select. A zero in a bit position in the Address Mask Register indicates that the corresponding address line must match the CPU address. A one in a bit position in the Address Mask Register indicates that address bit is a “*don’t care*”. This means that these address bits that toggle throughout a given address range are “masked off” and a valid chip select is generated whenever any of these addresses are selected.

This is shown in Figure 4-5 and explained in the following examples.

EXAMPLE 1:

Logical Chip Select Active Addresses:

0070h -0071h

Base Address Register:

0000_0070h

Address Mask Register:

0000_0001h

OR

Base Address Register:

0000_0071h

Address Mask Register:

0000_0001h

The user should note from the above example that the least significant address bit is a “*don’t care*”; this is indicated by the 1 in the least significant bit of the Address Mask Register. As a result, it does not matter whether the Base Address Register is loaded with either 0000_0070h or 0000_0071h (the least significant bit being a one or a zero).

It is suggested (but not required) that any bit programmed as a “*don’t care*” in the Address Mask Register have its corresponding Base Address bit programmed as a 0.

EXAMPLE 2:

Logical Chip Select Active Addresses

F_0000h - F_7FFFh

Base Address Register:

000F_0000h

Address Mask Register:

0000_7FFFh

The 15 least significant bits of the Address Mask Register are all 1’s, so the 15 least significant bits of the address decode are “*don’t cares*”. So, for simplicity and clarity, the 15 least significant bits of the Base Address Register are all 0’s.

Contrast the following example to the previous example.

EXAMPLE 3:

Logical Chip Select Active Addresses:

F_8000h - F_FFFFh

Base Address Register:

000F_8000h

Address Mask Register:

000_7FFFh

The only modification is that bit 15 of the Base Address Register was changed from a 0 to a 1. As a result the decode of the active address range moved to the next 32 Kbyte range.

The user must remember that the bits programmed as 1’s in the Address Mask Register, indicate only which address bits are “*don’t cares*”. They do **not** indicate the number of addresses to be decode. The following example may help in this regard:

EXAMPLE 4:

Logical Chip Select Active Addresses:

0060h and 0064h

Base Address Register:

0000_0060h

Address Mask Register:

000_0004h

Only address bit 2 is a “*don’t care*”; indicated by a 1 in bit 2 of the Address Mask Register (04h). However, address bits 1-0 are not “*don’t cares*”; as a result the chip select will only be active for addresses 0060h and 0064h, and will not be active for addresses 0061h - 0063h.

Furthermore, to decode a single address location, one should program the Address Mask Register with all 0’s.

EXAMPLE 5:

Logical Chip Select Active Address
1234_5678h
Base Address Register:
1234_5678h
Address Mask Register:
000_0000h

None of the address bits are “*don’t cares*”, so the address generated by the CPU and the address in the Base Address Register, must exactly match for the Logical Chip Select to become active.

4.4.5 Combining Chip Selects

One limitation associated with each logical chip select is the fact that only a single I/O or memory address range may be selected by each logical chip select.

To overcome this limitation, the **NS486SXF** provides circuitry which allows the user to combine multiple logical chip selects into a single signal, which is then driven out onto one of the generic chip select pin ($\overline{CS}[1]$ - $\overline{CS}[8]$).

An 8-bit register (External Chip Selection Registers 1 through 8, at I/O addresses EF45h - EF4Ch) associated with each generic chip select pin ($\overline{CS}[1]$ - $\overline{CS}[8]$) defines which logical chip select(s) are to be combined to generate the signal driven onto the output chip select pin.

This provides the ability to generate a chip select with a minimum of a single memory or I/O address range; or, on the other hand it provides the ability to generate a chip select with a maximum of eight different memory and/or I/O address ranges.

Eight logical chip selects provide a maximum of eight address ranges, that can be combined in any desired fashion to drive the generic chip select pins ($\overline{CS}[1]$ - $\overline{CS}[8]$). Refer to Figure 4-4.

It should be noted that if the user programs a chip select decode to coincide with either DRAM memory or an enabled internal peripheral, the external command strobes will not go active when such an access takes place, but the associated chip select signal will still go active.

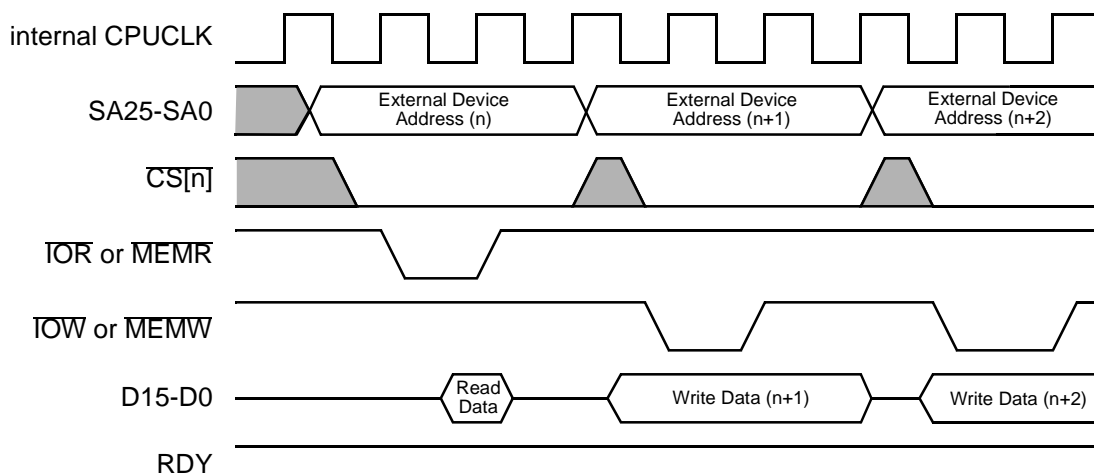
4.4.6 Programmable Access Timing

The Chip Select Access Time Registers 1 through 4 (at I/O address locations EF50h - EF53h) provide the means to program access timing parameters. Each Logical Chip Select has three bits which determine the number of additional programmed wait states associated with an access to its decoded locations. The number of added wait states may be any number between 0 and 7; resulting in a total of between 1 and 8 access states per cycle. More wait states may be added by any external device driving the Ready signal (RDY) inactive low. RDY is sampled at the end of the last programmed wait state, and if it is inactive low, another wait state will be added to the cycle. The Bus Interface Unit will continue to insert wait states until it samples RDY active high, then the access will be terminated.

Also in the Chip Select Access Time registers there is a bit associated with each Logical Chip Select which determines if the default command delay value is zero or one clock period. At higher frequencies one command delay may be required, otherwise the chip select may have negative setup time.

An External Bus cycle with the maximum number of wait states (seven) and the maximum number of inactive command states (one) will be performed if the access is not to DRAM, an internal peripheral or associated with a chip select decode.

The timing diagrams that follow show the timing for external device accesses.



**Figure 4-6 No Wait States, No Command Delays, External Interface Bus Cycles
(Internal access timing is similar)**

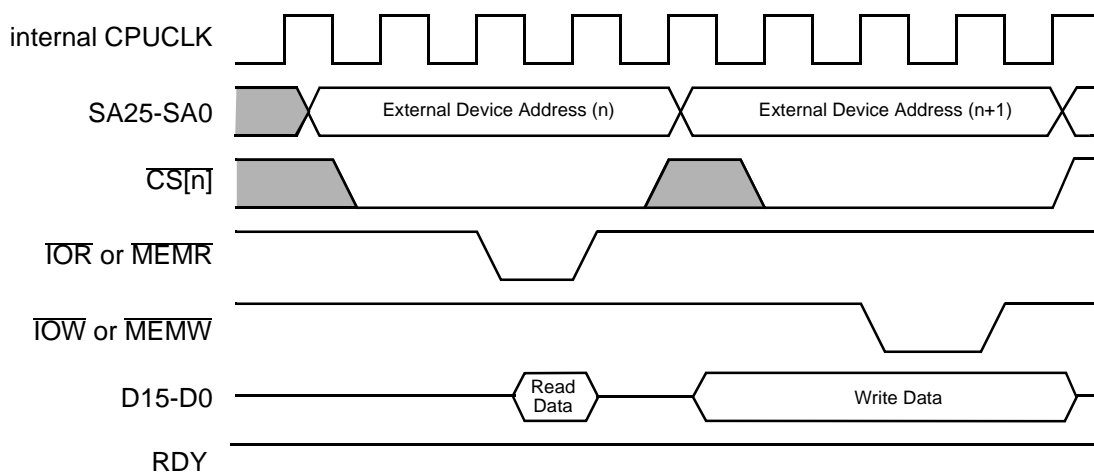


Figure 4-7 No Wait States, One Command Delay, External Bus Cycles

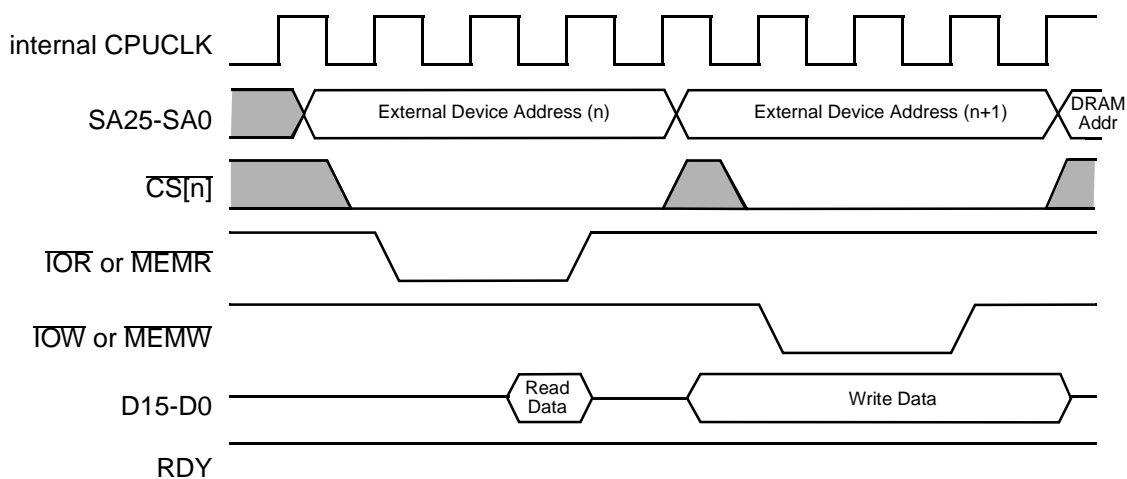


Figure 4-8 One Wait State, No Command Delays, External Interface Bus Cycles

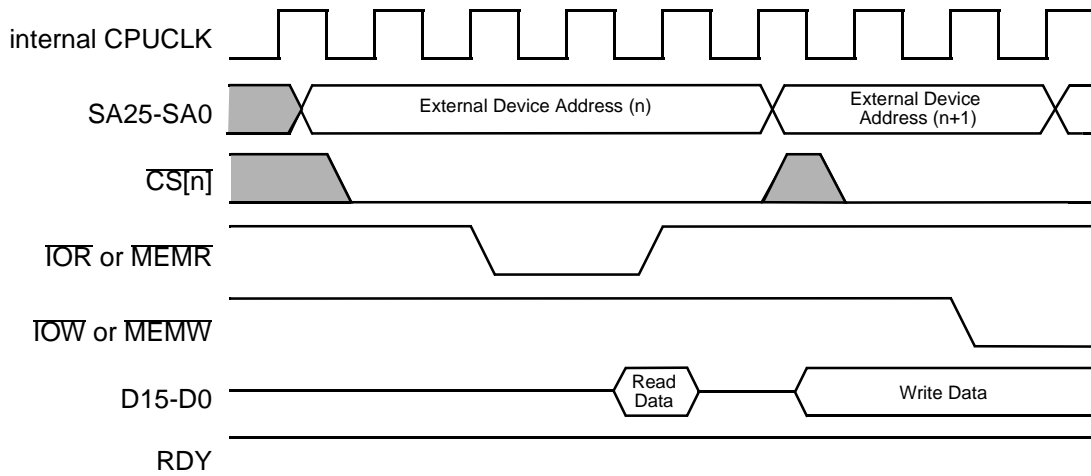


Figure 4-9 One Wait State, One Command Delay, External Interface Bus Cycles

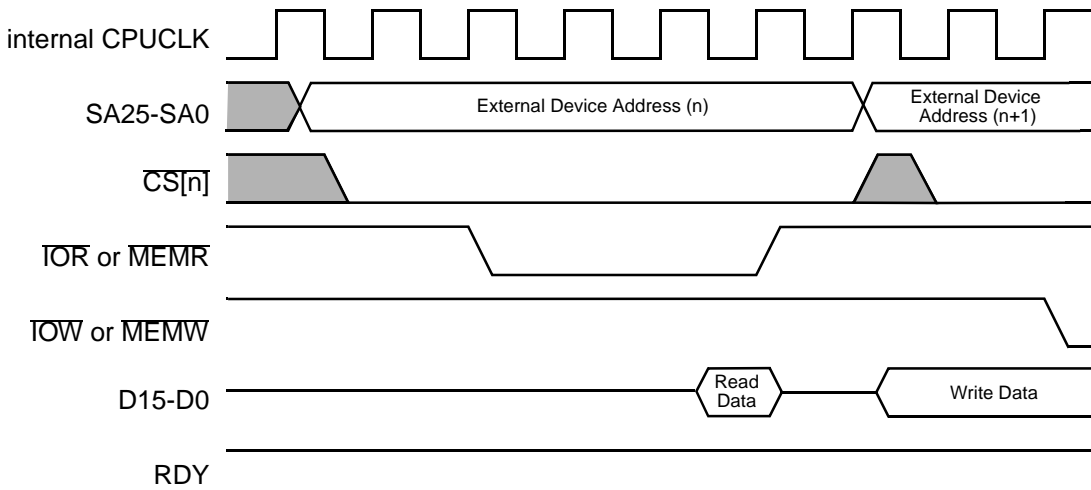


Figure 4-10 Two Wait States, One Command Delay, External Interface Bus Cycles

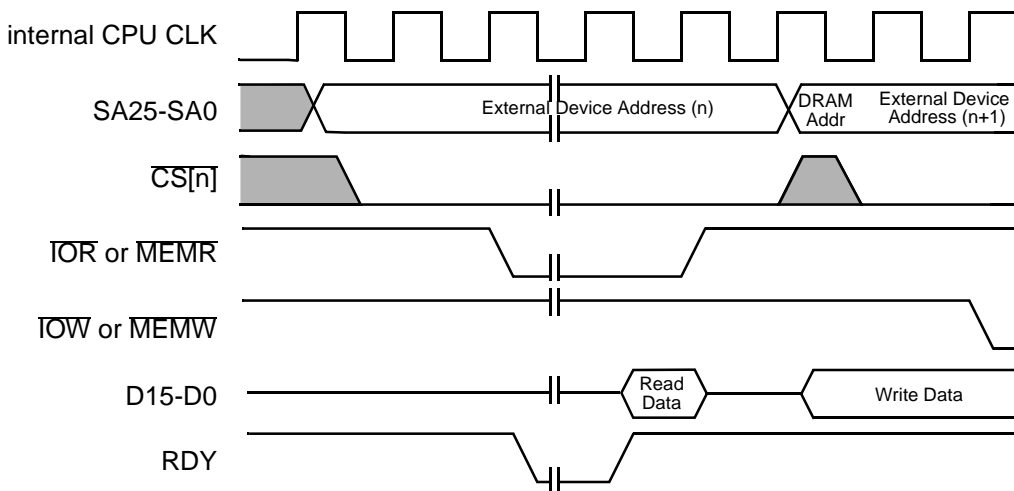


Figure 4-11 One Command Delay, Ext. Device Inserts Wait State, External Interface Bus Cycles

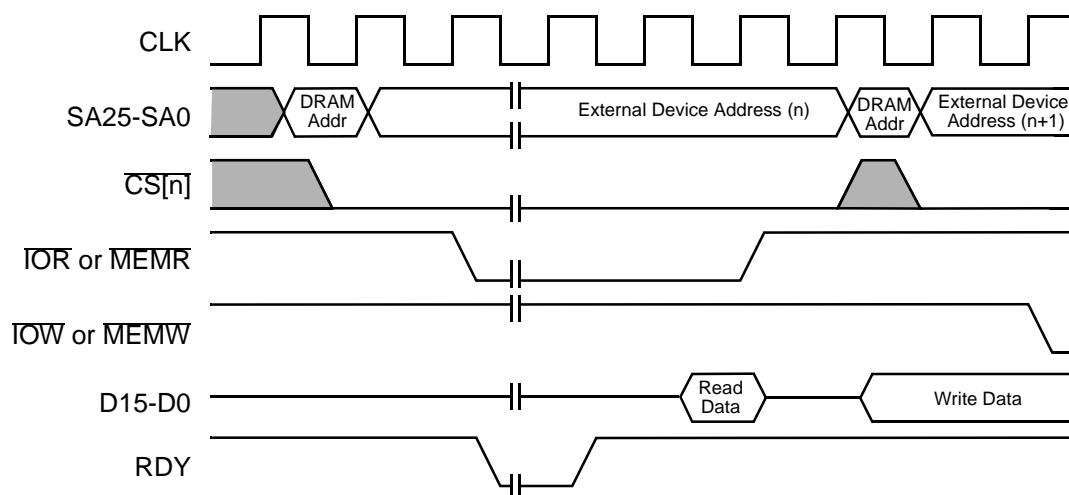


Figure 4-12 One Command Delay, Ext. Device Inserts Wait State, One Clock Period of RDY Extension, External Interface Bus Cycles

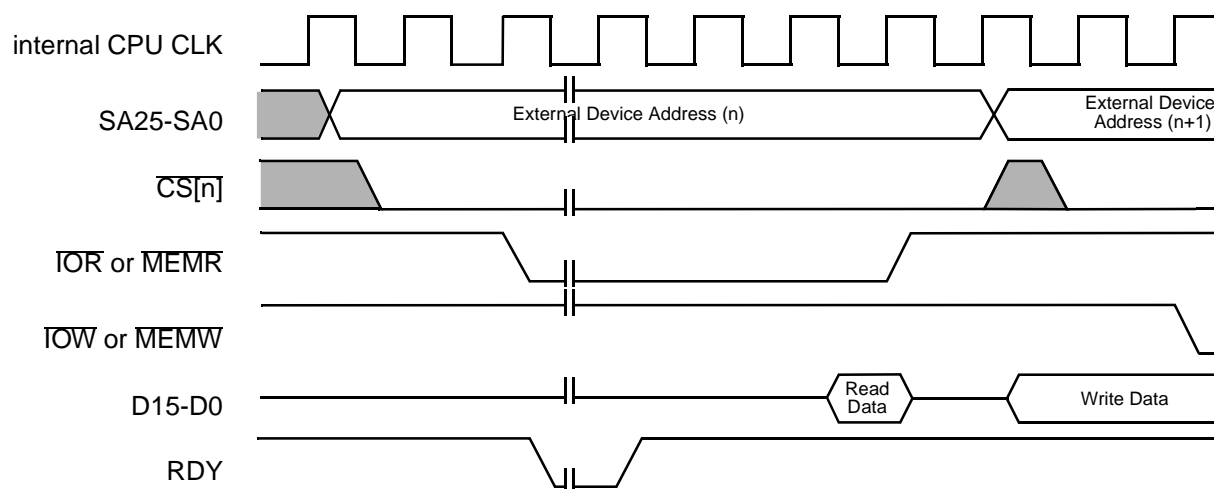
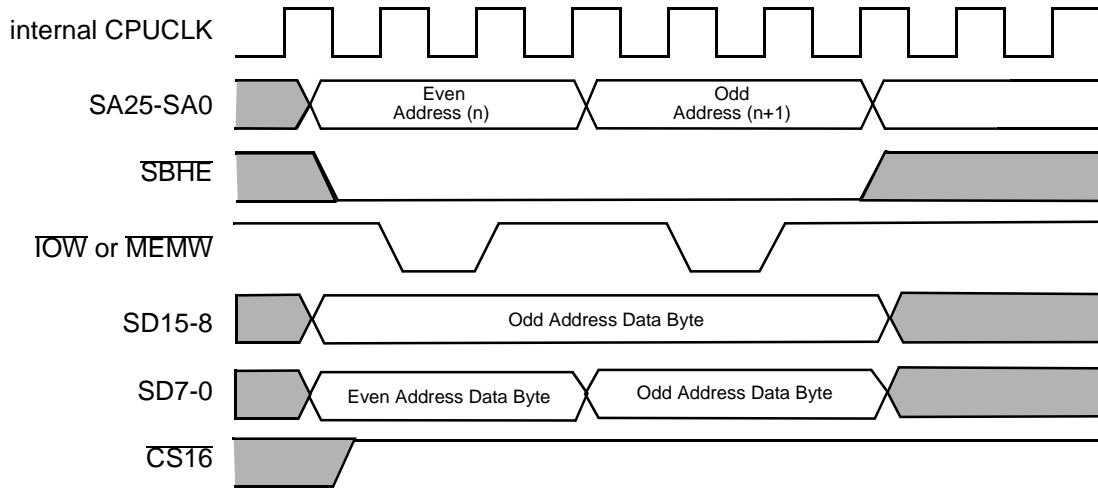
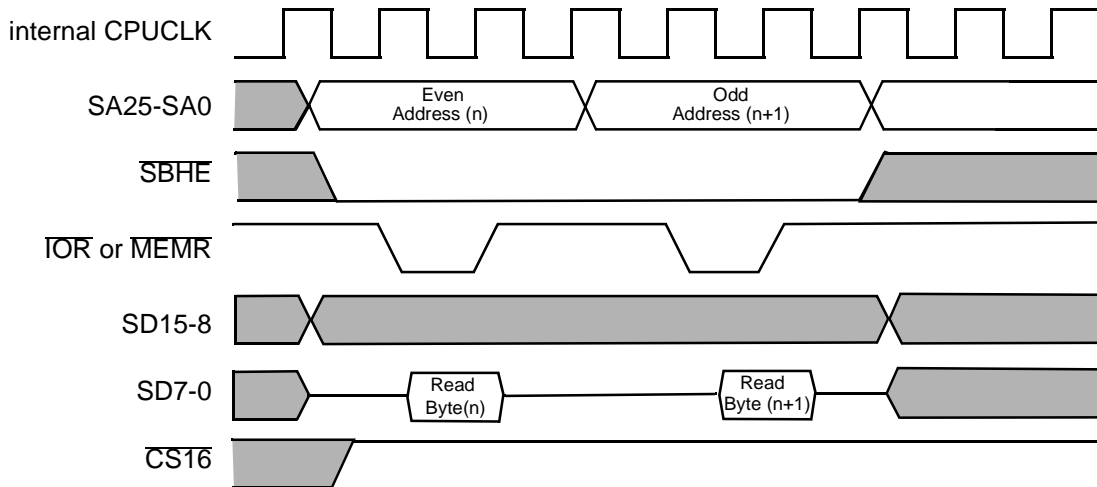


Figure 4-13 One Command Delay, Ext. Device Inserts Wait State, Two Clock Periods RDY Extension, External Interface Bus Cycles



(a) 16-bit to 8-bit Write Cycle Translation



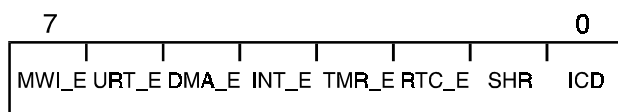
(b) 16-bit to 8-bit Read Cycle Translation

Figure 4-14 16-bit to 8-bit Cycle Translations

4.5 Bus Interface Unit Registers

4.5.1 Bus Interface Unit Control Register 1

During a system reset the bits in this register are set to 02h. This 8-bit register is located at I/O address EF00h.



I/O Map Address

EF00h

Access

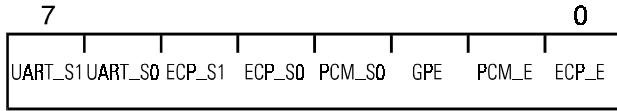
R/W

- Bit 7: MWI_E — MICROWIRE Interface Enable. When this bit is a one (and GPE = 1), the Bus Interface Unit will decode the CPU accesses and generate the proper access signals to the three wire interface logic. When this bit is a zero, no accesses to the three wire interface logic will be allowed. This bit is reset to zero by a system reset.
- Bit 6: URT_E — UART Enable. When this bit is a one (and GPE = 1), the Bus Interface Unit will decode the CPU accesses and generate the proper access signals to the on-board UART. When this bit is a zero, no accesses to the UART will be allowed. This bit is reset to zero by a system reset.
- Bit 5: DMA_E — DMA Enable. When this bit is a one (and GPE = 1), the Bus Interface Unit will decode the CPU accesses and generate the proper access signals to the on-board DMA Controller. When this bit is a zero, no accesses to the DMA Controller will be allowed. This bit is reset to zero by a system reset.
- Bit 4: INT_E — Interrupt Controller Enable. When this bit is a one (and GPE = 1), the Bus Interface Unit will decode the CPU accesses and generate the proper access signals to the on-board Interrupt Controllers. When this bit is a zero, no accesses to the Interrupt Controllers will be

- allowed. This bit is reset to zero by a system reset.
- Bit 3: TMR_E — TiMeR Enable. When this bit is a one (and GPE = 1), the Bus Interface Unit will decode the CPU accesses and generate the proper access signals to the on-board Timer. When this bit is a zero, no accesses to the Timer will be allowed. This bit is reset to zero by a system reset.
- Bit 2: RTC_E — Real Time Clock access Enable. When this bit is a one (and GPE = 1), the Bus Interface Unit will decode the CPU accesses and generate the proper access signals to the on-board RTC. When this bit is a zero, no accesses to the RTC will be allowed. This bit is reset to zero by a system reset.
- Bit 1: SHR — SHUTDOWN Reset. When this bit is a one, a CPU reset will be generated when the Bus Interface Unit decodes a Shutdown cycle by the CPU. When this bit is zero, no CPU reset is generated when a Shutdown cycle is decoded. This bit is set to one by a system reset.
- Bit 0: ICD — Internal Cycle Debug. When this bit is a one, an external bus cycle will be generated at the same time the internal bus cycle is performed. When this bit is a zero, there will be no external indication of internal bus cycles. This bit is set to zero by a system reset.
- Note:** PCMCIA cycles may be corrupted when this feature is enabled. Data driven out during internal IOR cycles could contend with external buffers.

4.5.2 Bus Interface Unit Control Register 2

During a system reset the bits in this register are set to 00h. This 8-bit register is located at I/O address EF01h.



I/O Map Address

EF01h

Access

R/W

Bits 7-6: UART_S1 and UART_S0 — UART IO address Selection bits 1-0. These two bits determine the I/O address range associated with the internal UART. These bits are only valid when both the UART Enable bit (Bit 6 of the Bus Interface Unit Control Register 1) and the Global Peripheral Enable bit are both set to one.

UART_S1	UART_S0	I/O Address Range
0	0	03F8h - 03FFh (COM1)
0	1	02F8h - 02FFh (COM2)
1	0	03E8h - 03EFh (COM3)
1	1	02E8h - 02EFh (COM4)

Bits 5,4: ECP_S1 and ECP_S0 -- ECP IO address Selection bits 1-0. These two bits determine the I/O address ranges associated with the internal Extended Capabilities Port. These bits are only valid when both the ECP Enable bit (Bit 0 of the Bus Interface Unit Control Register 2) and the Global Peripheral Enable bit are both set to one.

ECP_S1	ECP_S0	I/O Address Ranges
0	0	0278h - 027Ah 0678h - 067Ah
0	1	0378h - 037Ah 0778h - 077Ah
1	0	03BCh - 03BEh 07BCh - 07BEh
1	1	reserved

Bit 3: PCM_S0 — PCMCIA Controller address Selection bit. This bit determines the I/O address range of the PCMCIA control registers. This bit is only valid when both the PCMCIA Controller bit (bit 1 of this register) and the Global Peripheral Enable bit are both set to one.

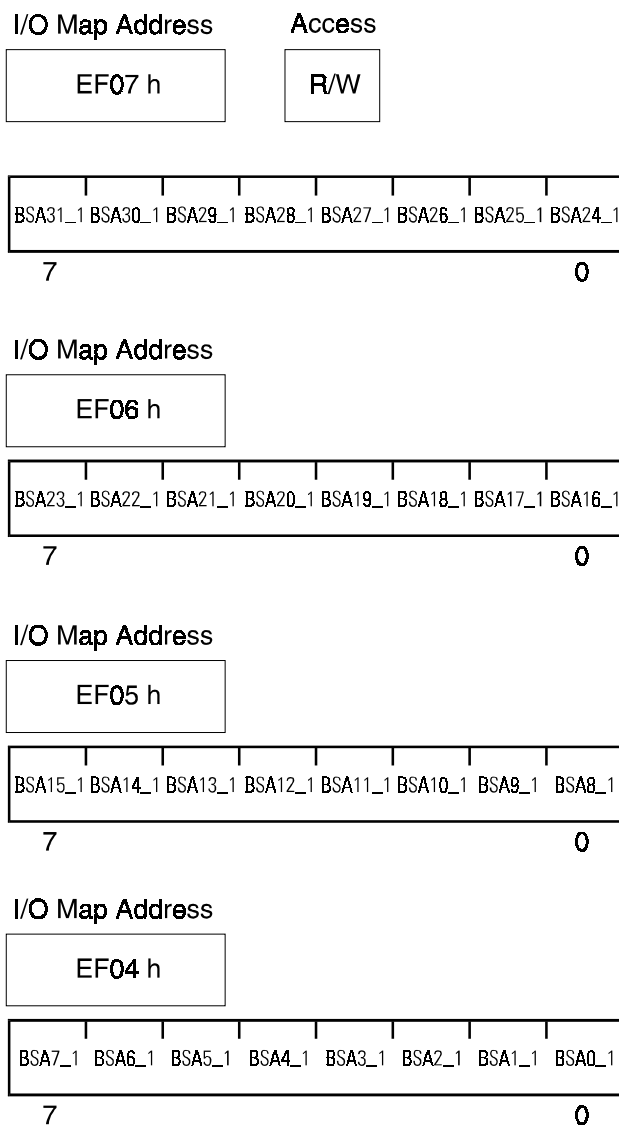
PCM_S0	I/O Address Range
0	03E0h-03E1h
1	03E2h-03E3h

Bit 2: GPE — Global Peripheral Enable. When this bit is a zero, all accesses to peripherals will be disabled. When this bit is a one, the enable bit associated with each peripheral will determine if the peripheral may be accessed or not.

Bit 1: PCM_E — PCMCIA Controller Enable. When this bit is a one, the Bus Interface Unit will decode the CPU accesses and generate the proper access signals to the on-board PCMCIA Controller. When this bit is a zero, no accesses to the PCMCIA Controller will be allowed and the pins associated with this function may be programmed to operate as general purpose I/O ports. This bit is reset to zero by a system reset.

Bit 0: ECP_E — Extended Capabilities Port Enable. When this bit is a one, the Bus Interface Unit will decode the CPU accesses and generate the proper access signals to the on-board Extended Capabilities Port. When this bit is a zero, no accesses to the Extended Capabilities Port will be allowed and the pins associated with this function may be programmed to operate as general purpose I/O ports. This bit is reset to zero by a system reset.

4.5.3 Chip Select Base Address Registers



Bits 31-0:BSA31_1- BSA0_1 — These thirty-two bits determine the address used to provide support for one of eight logical chip select decodes.

4.5.3.1 Chip Select Base Address Register 1

During a system reset the bits in these registers are not set. These four 8-bit registers are located at I/O addresses EF04h - EF07h, and may be accessed one at a time or two at a time.

4.5.3.2 Chip Select Base Address Register 2

During a system reset the bits in these registers are not set. These four 8-bit registers are located at I/O addresses EF08h - EF0Bh, and may be accessed one at a time or two at a time. These registers function like Chip Select Base Address Register 1.

4.5.3.3 Chip Select Base Address Register 3

During a system reset the bits in these registers are not set. These four 8-bit registers are located at I/O addresses EF0Ch - EF0Fh, and may be accessed one at a time or two at a time. These registers function like Chip Select Base Address Register 1.

4.5.3.4 Chip Select Base Address Register 4

During a system reset the bits in these registers are not set. These four 8-bit registers are located at I/O addresses EF10h - EF13h, and may be accessed one at a time or two at a time. These registers function like Chip Select Base Address Register 1.

4.5.3.5 Chip Select Base Address Register 5

During a system reset the bits in these registers are not set. These four 8-bit registers are located at I/O addresses EF14h - EF17h, and may be accessed one at a time or two at a time. These registers function like Chip Select Base Address Register 1.

4.5.3.6 Chip Select Base Address Register 6

During a system reset the bits in these registers are not set. These four 8-bit registers are located at I/O addresses EF18h - EF1Bh, and may be accessed one at a time or two at a time. These registers function like Chip Select Base Address Register 1.

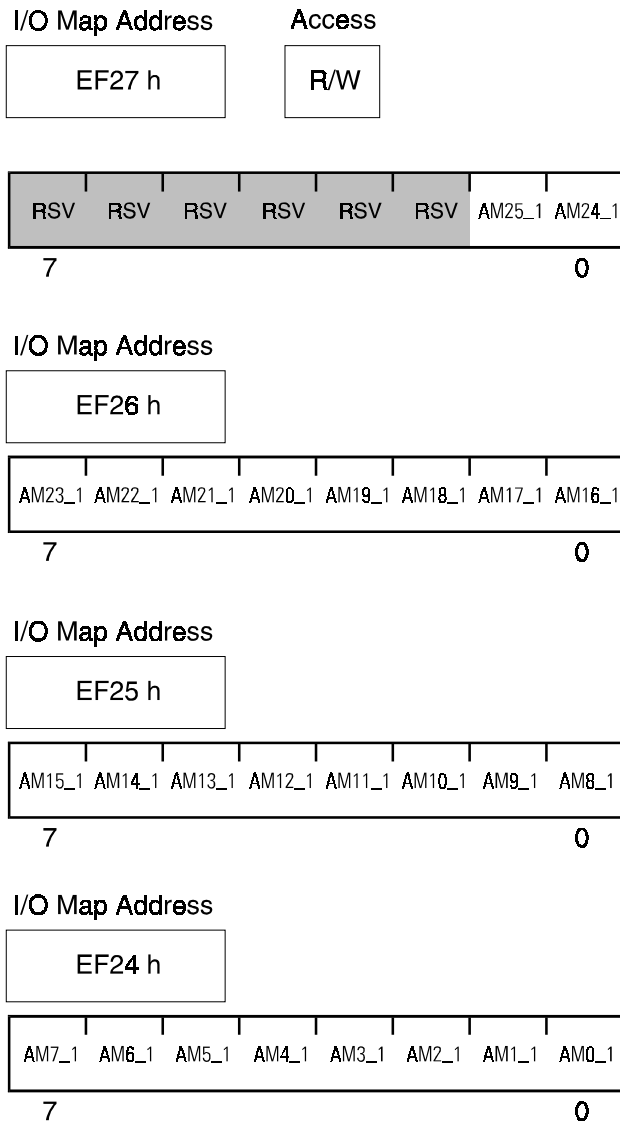
4.5.3.7 Chip Select Base Address Register 7

During a system reset the bits in these registers are not set. These four 8-bit registers are located at I/O addresses EF1Ch - EF1Fh, and may be accessed one at a time or two at a time. These registers function like Chip Select Base Address Register 1.

4.5.3.8 Chip Select Base Address Register 8

During a system reset the bits in these registers are not set. These four 8-bit registers are located at I/O addresses EF20h - EF23h, and may be accessed one at a time or two at a time. These registers function like Chip Select Base Address Register 1.

4.5.4 Chip Select Address Mask Registers



Bits 25-0:AM25_1-AM0_1 — These twenty-six bits indicate the size or range of addresses associated with Logical Chip Select 1. The user indicates which address bits are “don’t cares” by writing a 1 to the corresponding bit(s) in this register.

Only the lower twenty-six address bits are driven off-chip (i.e. SA25-0), so no address range greater than 64 MBytes may be programmed because external devices could not dis-

tinguish which address is actually being accessed. This means that the six most significant address bits in the Chip Select Base Address Register 1 must match the corresponding CPU generated address bits for Logical Chip Select 1 to go active. Likewise this is the reason the upper six Address Mask bits are unnecessary.

4.5.4.1 Chip Select Address Mask Register 1

During a system reset the bits in these registers are not set. These four 8-bit registers are located at I/O addresses EF24h - EF27h, and are used as a mask for the associated Chip Select Address.

4.5.4.2 Chip Select Address Mask Register 2

During a system reset the bits in these registers are not set. These four 8-bit registers are located at I/O addresses EF28h - EF2Bh, and are used as a mask for the associated Chip Select Address. These registers function like Chip Select Address Mask Register 1.

4.5.4.3 Chip Select Address Mask Register 3

During a system reset the bits in these registers are not set. These four 8-bit registers are located at I/O addresses EF2Ch - EF2Fh, and are used as a mask for the associated Chip Select Address. These registers function like Chip Select Address Mask Register 1.

4.5.4.4 Chip Select Address Mask Register 4

During a system reset the bits in these registers are not set. These four 8-bit registers are located at I/O addresses EF30h - EF33h, and are used as a mask for the associated Chip Select Address. These registers function like Chip Select Address Mask Register 1.

4.5.4.5 Chip Select Address Mask Register 5

During a system reset the bits in these registers are not set. These four 8-bit registers are located at I/O addresses EF34h - EF37h, and are used as a mask for the associated Chip Select Address. These registers function like Chip Select Address Mask Register 1.

4.5.4.6 Chip Select Address Mask Register 6

During a system reset the bits in these registers are not set. These four 8-bit registers are located at I/O addresses EF38h - EF3Bh, and are used as a mask for

the associated Chip Select Address. These registers function like Chip Select Address Mask Register 1.

4.5.4.7 Chip Select Address Mask Register 7

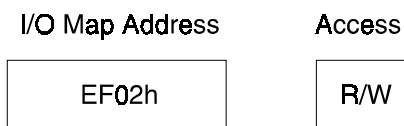
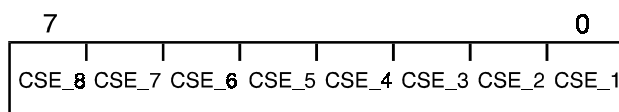
During a system reset the bits in these registers are not set. These four 8-bit registers are located at I/O addresses EF3Ch - EF3Fh, and are used as a mask for the associated Chip Select Address. These registers function like Chip Select Address Mask Register 1.

4.5.4.8 Chip Select Address Mask Register 8

During a system reset the bits in these registers are not set. These four 8-bit registers are located at I/O addresses EF40h - EF43h, and are used as a mask for the associated Chip Select Address. These registers function like Chip Select Address Mask Register 1.

4.5.5 Chip Select Enable Register

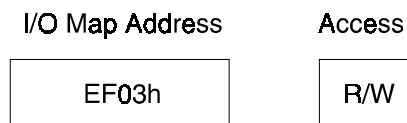
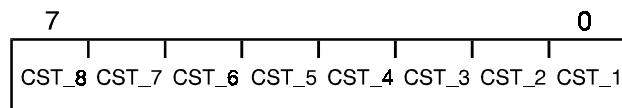
During a system reset the bits in this register are reset to zero. This 8-bit register is located at I/O addresses EF02h and contains the enables for each of the eight Logical Chip Select decodes.



Bits 7-0: CSE_8-CSE_1 — These eight bits enable or disable (1 or 0, respectively) the associated Logical Chip Select decodes. Before programming or changing either the Chip Select Base Address or Address Range registers one should always disable the Logical Chip Select decode by writing a zero to the appropriate bit(s).

4.5.6 Chip Select Type Register

During a system reset the bits in this register are not set. This 8-bit register is located at I/O addresses EF03h and contains the type of cycle (memory or IO) for each of the eight Chip Select decodes.

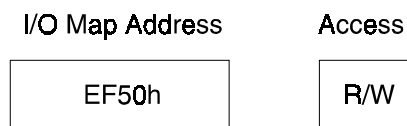
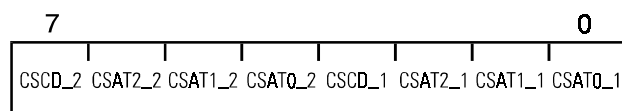


Bits 7-0: CST_8-CST_1 — These eight bits determine the type of cycle (memory when a 1 and IO when a 0) associated with each Chip Select decode. **Before programming or changing either the Chip Select Type registers one should always disable the Chip Select decode by writing a zero to the Chip Select Enable bit appropriate bit(s).**

4.5.7 Chip Select Access Time Registers

4.5.7.1 Chip Select Access Time Register 1

During a system reset the bits in this register are all set to ones. This 8-bit register is located at I/O address EF50h and contains the access timing information for Logical Chip Select decodes 1 and 2.



Bit 7: CSCD_2 — Chip Select 2 Command Delay. When this bit is a one, an extra clock period of address setup time will be inserted into the accesses associated with this Logical Chip Select decode.

Bits 6-4: CSAT2_2-CSAT0_2 — Chip Select 2 Access Time. These three bits determine the number of clock cycles the external bus interface command strobes will be programmed to be active low. The number of clock cycles the command strobes are active will always be one greater than the value in these bits.

Bit 3: CSCD_1 — Chip Select 1 Command Delay. When this bit is a one, an extra clock period of address setup time will be inserted into the accesses associated with this Logical Chip Select decode.

Bits 2-0: CSAT2_1-CSAT0_1 — Chip Select 1 Access Time. These three bits determine the number of clock cycles the external bus interface command strobes will be programmed to be active low. The number of clock cycles the command strobes are active will always be one greater than the value in these bits.

4.5.7.2 Chip Select Access Time Register 2

During a system reset the bits in this register are all set to ones. This 8-bit register is located at I/O address EF51h and contains the access timing information for Logical Chip Select decodes 3 and 4.



Bit 7: CSCD_4 — Chip Select 4 Command Delay. When this bit is a one, an extra clock period of address setup time will be inserted into the accesses associated with this Logical Chip Select decode.

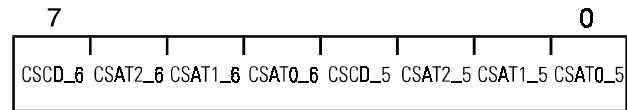
Bits 6-4: CSAT2_4 — Chip Select 4 Access Time. These three bits determine the number of clock cycles the external bus interface command strobes will be programmed to be active low. The number of clock cycles the command strobes are active will always be one greater than the value in these bits.

Bit 3: CSCD_3 — Chip Select 3 Command Delay. When this bit is a one, an extra clock period of address setup time will be inserted into the accesses associated with this Logical Chip Select decode.

Bits 2-0: CSAT2_3-CSAT0_3 — Chip Select 3 Access Time. These three bits determine the number of clock cycles the external bus interface command strobes will be programmed to be active low. The number of clock cycles the command strobes are active will always be one greater than the value in these bits.

4.5.7.3 Chip Select Access Time Register 3

During a system reset the bits in this register are all set to ones. This 8-bit register is located at I/O address EF52h and contains the access timing information for Chip Select decodes 5 and 6.



Bit 7: CSCD_6 — Chip Select 6 Command Delay. When this bit is a one, an extra clock period of address setup time will be inserted into the accesses associated with this Logical Chip Select decode.

Bits 6-4: CSAT2_6-CSAT0_6 — Chip Select 6 Access Time. These three bits determine the number of clock cycles the external bus interface command strobes will be programmed to be active low. The number of clock cycles the command strobes are active will always be one greater than the value in these bits.

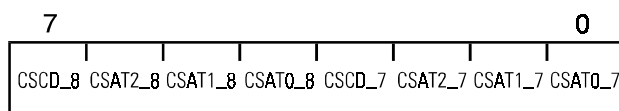
Bit 3: CSCD_5 — Chip Select 5 Command Delay. When this bit is a one, an extra clock period of address setup time will be inserted into the accesses associated with this Logical Chip Select decode.

Bits 2-0: CSAT2_5-CSAT0_5 — Chip Select 5 Access Time. These three bits determine the number of clock cycles the external

bus interface command strobes will be programmed to be active low. The number of clock cycles the command strobes are active will always be one greater than the value in these bits.

4.5.7.4 Chip Select Access Time Register 4

During a system reset the bits in this register are all set to ones. This 8-bit register is located at I/O address EF53h and contains the access timing information for Logical Chip Select decodes 7 and 8.



I/O Map Address

EF53h

Access

R/W

Bit 7: CSCD_8 — Chip Select 8 Command Delay. When this bit is a one, an extra clock period of address setup time will be inserted into the accesses associated with this Logical Chip Select decode.

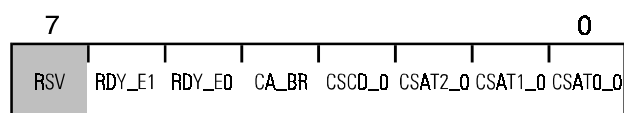
Bits 6-4: CSAT2_8-CSAT0_8 — Chip Select 8 Access Time. These three bits determine the number of clock cycles the external bus interface command strobes will be programmed to be active low. The number of clock cycles the command strobes are active will always be one greater than the value in these bits.

Bit 3: CSCD_7 — Chip Select 7 Command Delay. When this bit is a one, an extra clock period of address setup time will be inserted into the accesses associated with this Logical Chip Select decode.

Bits 2-0: CSAT2_7-CSAT0_7 — Chip Select 7 Access Time. These three bits determine the number of clock cycles the external bus interface command strobes will be programmed to be active low. The number of clock cycles the command strobes are active will always be one greater than the value in these bits.

4.5.7.5 Boot ROM Access Time Register

During a system reset this register is set to 6Fh. This 7-bit register is located at I/O address EF54h and contains the access timing information for Boot ROM Chip Select 0 ($\overline{CS}[0]$). Also bits 5 and 6 of this register determine the minimum number of CPUCLK periods following the assertion of RDY, that the associated command strobe (i.e., \overline{IOR} , \overline{IOW} , \overline{MEMR} , or \overline{MEMW}) may go inactive high to terminate an access. The default number is two CPUCLK periods.



I/O Map Address

EF54 h

Access

R/W

Bits 7: Reserved.

Bits 6-5: RDY_E1 and RDY_E0 — ReaDY Extension bits 1-0.

RDY_E1	RDY_E0	CPUCLK Periods of Extension
0	0	Zero
0	1	One
1	0	Two (reset default)
1	1	Three

Bit 4: CA_BR — Cacheable Boot ROM Chip Select. When a one is written into this bit, it indicates that the Boot ROM address range (FFFF_0000h - FFFF_FFFFh) is cacheable. When this bit is a zero, this address range is uncacheable.

Bit 3: CSCD_0 — Chip Select 0 Command Delay. When this bit is a one, an extra clock period of address setup time will be inserted into the accesses to the Boot ROM address range (FFFF_0000h - FFFF_FFFF).

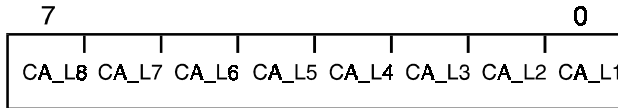
Bits 2-0: CSAT2_0-CSAT0_0 — Chip Select 0 Access Time. These three bits determine the number of clock cycles the external bus interface command strobes will be

programmed to be active low. The number of clock cycles the command strobes are active will always be one greater than the value in these bits.

4.5.8 Cacheable Chip Selects Register

During a system reset the bits in this register are set to 00h. This 8-bit register is located at I/O address EF55h.

Writing a 1 into a bit in this register indicates that the corresponding Logical Chip Select decodes a cacheable memory range. The NS486SXF CPU will only allow the caching of instruction code fetched from cacheable memory ranges.



I/O Map Address	Access
EF55 h	R/W

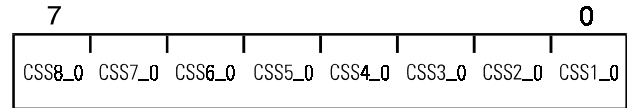
Bit 7-0: CA_L8 - CA_L1 — CacheAble Logical Chip Select 8-1. When a 1 is written into a bit in this register, it indicates that the corresponding Logical Chip Select decodes a cacheable memory range.

4.5.9 External Chip Selection Registers

4.5.9.1 Boot ROM Selection Register

During a system reset the bits in this register are all set to zeros. This 8-bit register is located at I/O address EF44h and contains selection bits for determining which Logical Chip Selects will be associated with the External Chip Select 0 ($\overline{CS}[0]$).

NOTE: $\overline{CS}0$ will always go active for memory accesses to the most significant 64 Kbytes of memory (FFFF0000h - FFFFFFFFh). This register allows the user to specify additional memory range(s) that will cause $\overline{CS}[0]$ to go active low.

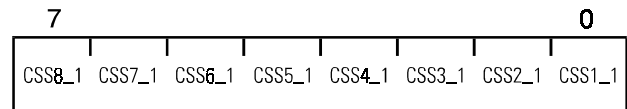


I/O Map Address	Access
EF44 h	R/W

Bits 7-0: CSS8_0- CSS1_0 — Chip Select Selection. When a bit is a one in this register, it indicates that when there is an active Logical Chip Select associated with that decode logic, the external Chip Select 0 ($\overline{CS}[0]$) signal will go active.

4.5.9.2 External Chip Select Selection Register 1

During a system reset the bits in this register are all set to zeros. This 8-bit register is located at I/O address EF45h and contains selection bits for determining which Logical Chip Selects will be associated with the External Chip Select 1 ($\overline{CS}[1]$).

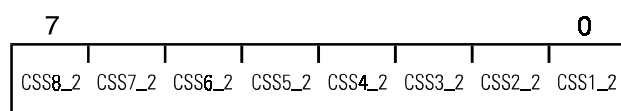


I/O Map Address	Access
EF45 h	R/W

Bits 7-0: CSS8_1-CSS1_1 — Chip Select Selection. When a bit is a one in this register, it indicates that when there is an active Logical Chip Select associated with that decode logic, the external Chip Select 1 ($\overline{CS}[1]$) signal will go active.

4.5.9.3 External Chip Select Selection Register 2

During a system reset the bits in this register are all set to zeros. This 8-bit register is located at I/O address EF46h and contains selection bits for determining which Logical Chip Select decodes will be associated with the External Chip Select 2 ($\overline{CS}[2]$).



I/O Map Address

EF46 h

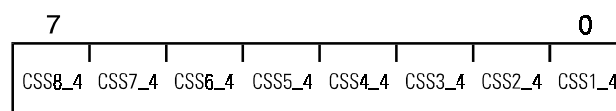
Access

R/W

Bits 7-0: CSS8_2 - CSS1_2 — Chip Select Selection. When a bit is a one in this register, it indicates that when there is an active Logical Chip Select associated with that decode logic, the external Chip Select 2 ($\overline{CS}[2]$) signal will go active.

4.5.9.5 External Chip Select Selection Register 4

During a system reset the bits in this register are all set to zeros. This 8-bit register is located at I/O address EF48h and contains selection bits for determining which Logical Chip Select decodes will be associated with the External Chip Select 4 ($\overline{CS}[4]$).



I/O Map Address

EF48 h

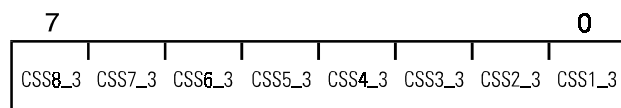
Access

R/W

Bits 7-0: CSS8_4 - CSS1_4 — Chip Select Selection. When a bit is a one in this register, it indicates that when there is an active Logical Chip Select associated with that decode logic, the external Chip Select 4 ($\overline{CS}[4]$) signal will go active.

4.5.9.4 External Chip Select Selection Register 3

During a system reset the bits in this register are all set to zeros. This 8-bit register is located at I/O address EF47h and contains selection bits for determining which Logical Chip Select decodes will be associated with the External Chip Select 3 ($\overline{CS}[3]$).



I/O Map Address

EF47 h

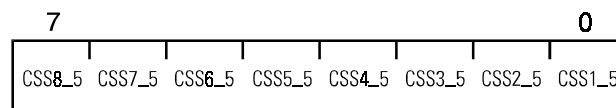
Access

R/W

Bits 7-0: CSS8_3 - CSS1_3 — Chip Select Selection. When a bit is a one in this register, it indicates that when there is an active Logical Chip Select associated with that decode logic, the external Chip Select 3 ($\overline{CS}[3]$) signal will go active.

4.5.9.6 External Chip Select Selection Register 5

During a system reset the bits in this register are all set to zeros. This 8-bit register is located at I/O address EF49h and contains selection bits for determining which Logical Chip Select decodes will be associated with the External Chip Select 5 ($\overline{CS}[5]$).



I/O Map Address

EF49 h

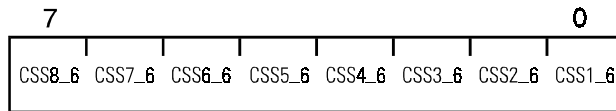
Access

R/W

Bits 7-0: CSS8_5 - CSS1_5 — Chip Select Selection. When a bit is a one in this register, it indicates that when there is an active Logical Chip Select associated with that decode logic, the External Chip Select 5 ($\overline{CS}[5]$) signal will go active.

4.5.9.7 External Chip Select Selection Register 6

During a system reset the bits in this register are all set to zeros. This 8-bit register is located at I/O address EF4Ah and contains selection bits for determining which Logical Chip Select decodes will be associated with the External Chip Select 6 ($\overline{CS}[6]$).

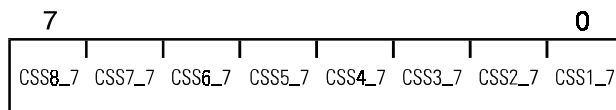


I/O Map Address	Access
EF4A h	R/W

Bits 7-0: CSS8_6 - CSS1_6 — Chip Select Selection. When a bit is a one in this register, it indicates that when there is an active Logical Chip Select associated with that decode logic, the external Chip Select6 ($\overline{CS}[6]$) signal will go active.

4.5.9.8 External Chip Select Selection Register 7

During a system reset the bits in this register are all set to zeros. This 8-bit register is located at I/O address EF4Bh and contains selection bits for determining which Logical Chip Select decodes will be associated with the External Chip Select 7 ($\overline{CS}[7]$).

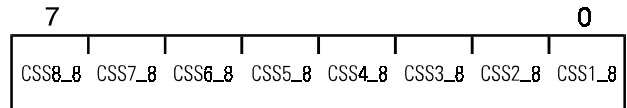


I/O Map Address	Access
EF4B h	R/W

Bits 7-0: CSS8_7 - CSS1_7 — Chip Select Selection. When a bit is a one in this register, it indicates that when there is an active Logical Chip Select associated with that decode logic, the external Chip Select 7 ($\overline{CS}[7]$) signal will go active.

4.5.9.9 External Chip Select Selection Register 8

During a system reset the bits in this register are all set to zeros. This 8-bit register is located at I/O address EF4Ch and contains selection bits for determining which Logical Chip Select decodes will be associated with the External Chip Select 8 ($\overline{CS}[8]$).

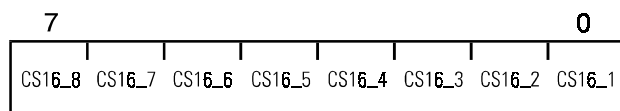


I/O Map Address	Access
EF4C h	R/W

Bits 7-0: CSS8_8 - CSS1_8 — Chip Select Selection. When a bit is a one in this register, it indicates that when there is an active Logical Chip Select associated with that decode logic, the external Chip Select8 ($\overline{CS}[8]$) signal will go active.

4.5.10 Programmable 16-bit Logical Chip Select Register

During a system reset the bits in this register are all set to zeros. This 8-bit register is located at I/O address EF57h. When a bit in this register is set to a one, its associated Logical Chip Select access will be treated as a 16-bit access, regardless of the state of the $\overline{CS16}$ signal. When a bit in this register is zero, its Logical Chip Select access will generate the appropriate bus cycles based on the $\overline{CS16}$ signal.



I/O Map Address

EF57 h

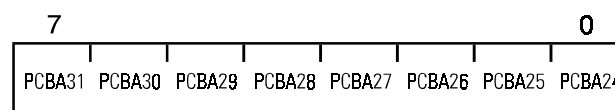
Access

R/W

Bits 7-0: CS16_8 - CS16_1 — Logical Chip Select 8-1, 16-bit device. Bit 7 is associated with logical chip select 8, bit 6 with logical chip select 7, and so on. When set to 1, an access to the selected logical chip select will be treated as a 16-bit access, regardless of the status of $\overline{CS16}$. When set to 0, an access to the logical chip select will generate the appropriate bus cycles based upon the $\overline{CS16}$ signal.

After an IC reset, this register will be all zeros, so memory accesses to 00000000h - 00FFFFFFh may be translated and sent to an attached PCMCIA card.

For another example, if this register was written with the value 12h, then memory accesses to 12000000h-12FFFFFFh could be translated and sent to an attached PCMCIA card.



I/O Map Address

EF4E h

Access

R/W

Bits 7-0: PCBA31 - PCBA24 — PCMCIA Base Address 31-24. These eight bits determine the base address of all memory cycles translated by the **NS486SXF** PCMCIA Controller. These eight bits must match those produced by the CPU or the memory cycle will not be seen by the PCMCIA Controller, in which case the Bus Interface Unit would run an ISA-like bus cycle.

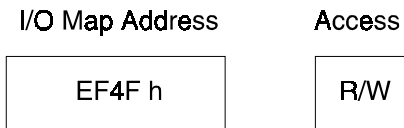
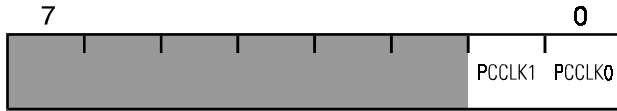
4.5.11 PCMCIA Memory Base Address Register

During a system reset the bits in this register are all set to zeros. This 8-bit register is located at I/O address EF4Eh. The bits of this register (PCBA_31 - PCBA_24) define the base address of the PCMCIA Controller's memory translation window. See also: PCMCIA Memory Accesses on page 174

An Intel82365SL type PCMCIA Controller can only translate from a 16 Mbyte memory address range to an attached PCMCIA card. When the bits of this register (PCBA_31 - PCBA_24) match the eight most significant bits of the CPU's address (31-24, respectively), then the access may be translated by the **NS486SXF** PCMCIA Controller.

4.5.12 PCMCIA Clock Selection Register

During a system reset the bits in this register are all set to zeros. This 2-bit register is located at I/O address EF4Fh. The bits of this register (PCCLK1 and PCCLK0) determine the operating frequency of the PCMCIA Controller.



Bits 7-2: Reserved.

Bit 1-0 PCCLK1 - PCCLK0 — PCMCIA Clock Selection bits 1 and 0. These two bits the frequency of operation for the PCMCIA Controller.

PCCLK1	PCCLK0	Operating Frequency
0	0	CPUCLK/4
0	1	CPUCLK/3
1	X	CPUCLK/2

NOTE: In an ISA bus system, the PCMCIA Controller is usually driven by SY-SCLK, which is a ~8 MHz clock. The reset setting is CPUCLK/4. Also the switching of the clock frequency into the PCMCIA Controller is guaranteed to be glitch free. So one may switch the frequency of operation at any time.

4.6 Auxiliary Processor, Shared Memory Interface

NS486SXF supports the use of an Auxiliary Processor, with a sharing of memory space. The Auxiliary Processor interface provides a low cost interface for sharing memory belonging to an Auxiliary Processor or any other processor including another NS486SXF. Only a small amount of additional logic, along with a set of TRI-STATE buffers for address, data and control signals, is required.

4.6.1 Auxiliary Processor Communications

There are two methods of transferring information to/from shared memory with the NS486SXF; the first is under CPU control and the second is via the DMA Controller.

Note: The Auxiliary Processor Communications interface supports only memory mapped shared memory; IO mapping of the shared memory at present is not supported.

4.6.2 CPU Controlled Transfers

When under CPU control, one or more of the programmable Logical Chip Selects should be programmed to indicate that they are to be used as Auxiliary Processor Communication chip select(s). **Furthermore, a command delay must be programmed for any Logical Chip Select(s) used to support the Auxiliary Processor shared memory space.** (The command delay requirement is made to allow the use of more efficient arbitration logic within the NS486SXF.) The Auxiliary Processor Chip Select Selection Register (located at I/O address EF56h) determines which Logical Chip Selects support Auxiliary Processor shared memory.

After the appropriate Logical Chip Select(s) have been programmed, any CPU access to the address range associated with given Logical Chip Select(s) will generate a External Bus Request ($\overline{\text{EREQ}}$). Until an External Bus Acknowledge ($\overline{\text{EACK}}$) is received in reply the NS486SXF's bus will loop in a command delay state. Once $\overline{\text{EACK}}$ is driven low, the combination of both $\overline{\text{EREQ}}$ and $\overline{\text{EACK}}$ both being low will result in the Shared Memory Drive Control signal ($\overline{\text{DRV}}$) going active low. After one more CPUCLK period command delay, the memory strobe ($\overline{\text{MEMR}}$ or $\overline{\text{MEMW}}$) will be driven low for its programmed period of time, plus any RDY inserted wait states. After the rising edge of the $\overline{\text{MEMR}}$ or $\overline{\text{MEMW}}$ strobe, one CPUCLK period of hold time will be associated with the access; after which $\overline{\text{EREQ}}$ (and thus $\overline{\text{DRV}}$) will be taken inactive high.

The exception to the above description is when a NS486SXF 16-bit CPU access to an 8-bit shared memory is performed, or when the CPU performed a locked bus cycle sequence. In such cases the $\overline{\text{EREQ}}$ signal will stay active low until the transfer sequence is completed and will only return inactive high at the end of the transfer cycle.

It should be noted that in general, this method of accessing shared memory only allows a single transfer with the shared memory by the NS486SXF's CPU before it releases the shared memory by taking $\overline{\text{EREQ}}$ inactive high. Furthermore, it should be noted that any back to back accesses to the shared memory by the NS486SXF's CPU must take into consideration the latency issue of the $\overline{\text{EREQ}}$ going inactive high to the $\overline{\text{EACK}}$ going inactive. If this period of time is greater than two NS486SXF CPUCLK periods, then back to back bus cycles must be avoided by software means.

The following block diagram shows a conceptual shared memory interface between the NS486SXF and an Auxiliary Processor. External circuitry must use the $\overline{\text{DRV}}$ and $\overline{\text{MEMR}}$ signals to determine the direction of the data buses; otherwise all of the other signals are driven from the NS486SXF's bus to the Auxiliary Processor's Bus.

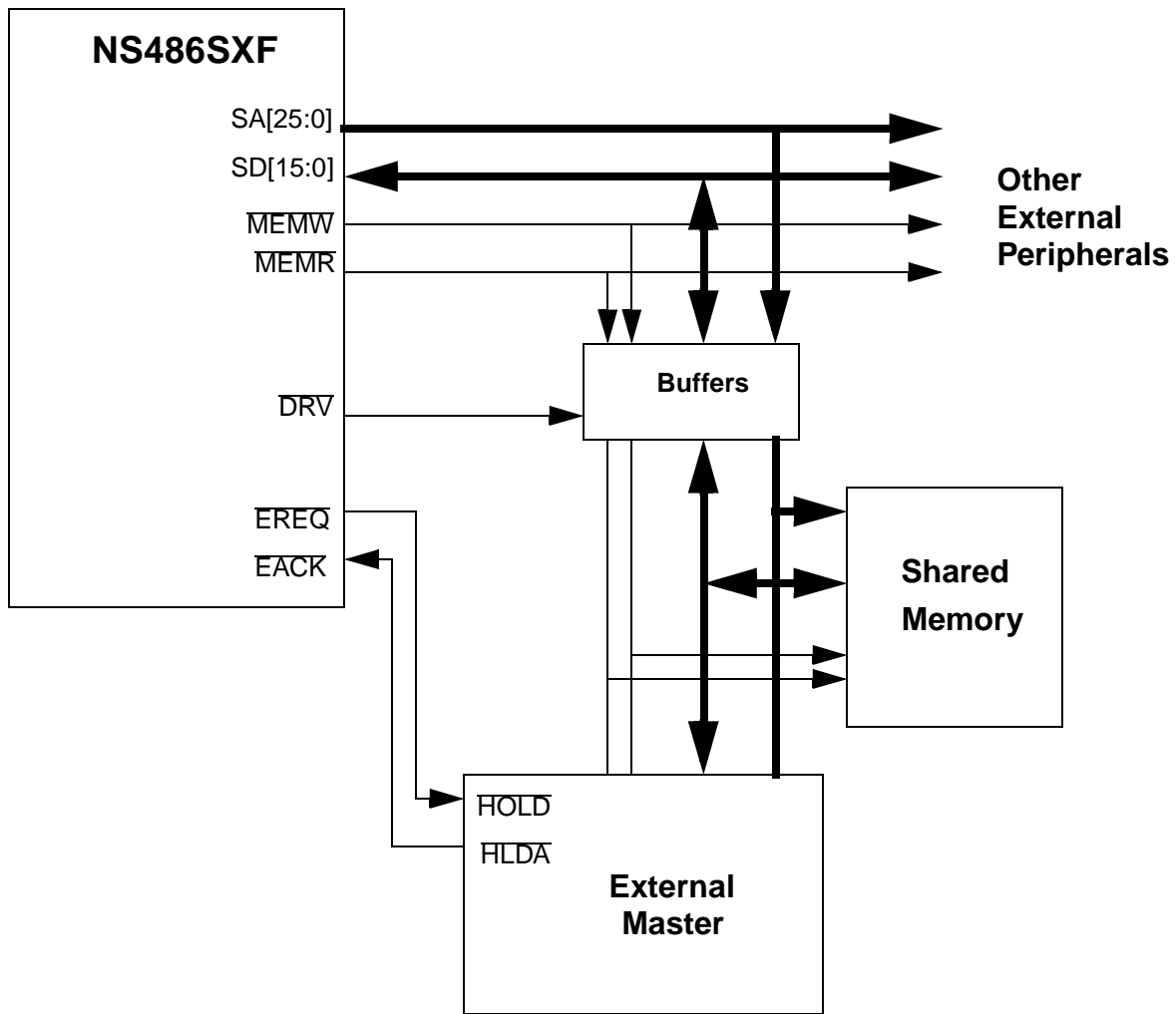
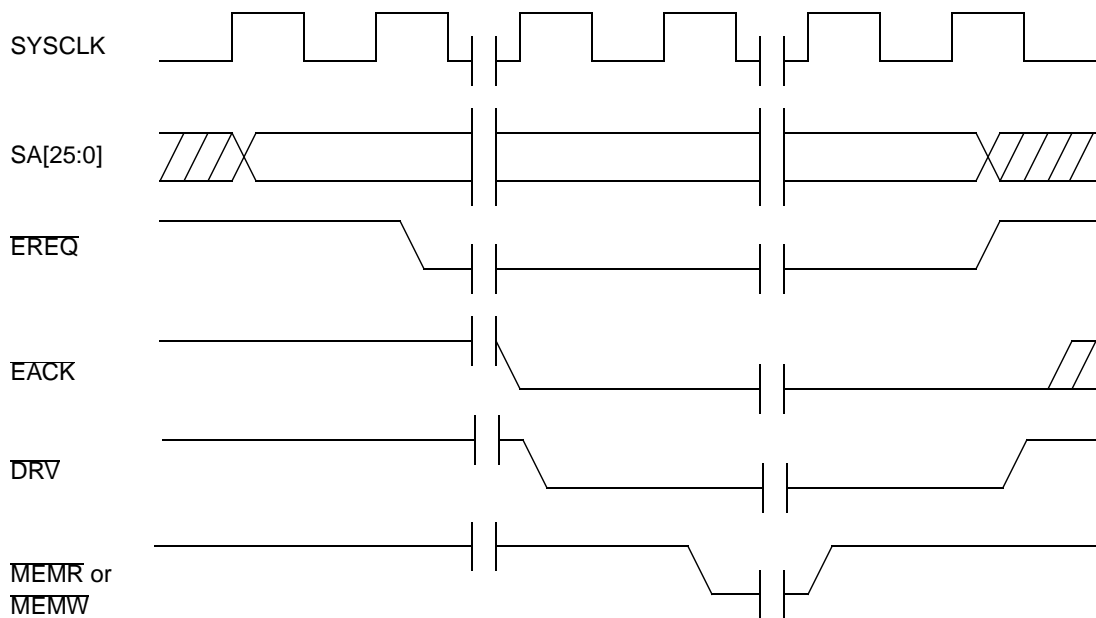


Figure 4-15 Auxiliary Processor Shared Memory Block Diagram



Note: The SYSCLK is provided for reference only.

Figure 4-16 Shared Memory Single Access (CPU Controlled)

4.6.3 DMA Controlled Transfers

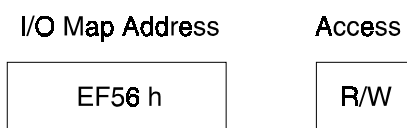
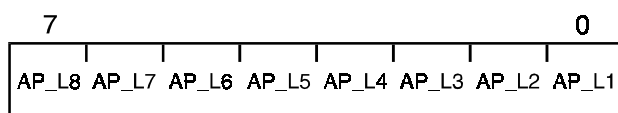
The DMA controlled shared memory transfer is very similar to the CPU controlled transfer except for three differences:

- 1) The **NS486SXF**'s DMA Controller may retain control over the shared memory to allow large blocks of data to be transferred at a time.
- 2) The initiation of the DMA controlled shared memory block transfer must be a DMA Request. That requires either the Auxiliary Processor Controller to make a DMA request, or the **NS486SXF**'s CPU can make a DMA request via the CPU DMA Request Register. If the request is made via the CPU DMA Request Register, the entire block transfer will be performed before control is returned to the CPU.
- 3) After the last transfer more than one CPUCLK period may pass before $\overline{\text{EREQ}}$ is deasserted.

4.6.4 Auxiliary Processor Chip Select Selection Register

During a system reset the bits in this register are set to 00h. This 8-bit register is located at I/O address EF56h.

Writing a 1 into a bit in this register indicates that the corresponding Logical Chip Select decodes a Auxiliary Processor shared memory range. When an access is made to that Auxiliary Processor shared memory range the Bus Interface Unit will perform the arbitration for the Auxiliary Processor shared memory and perform the appropriate bus cycle.



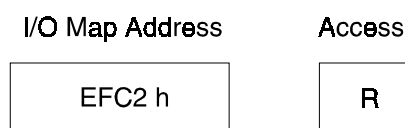
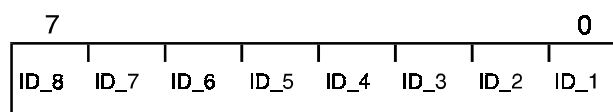
Bit 7-0: AP_L8 - AP_L1 — Auxiliary Processor Logical Chip Select 8-1. When a 1 is written into a bit in this register, it indicates that the corresponding Logical Chip Select decodes an Auxiliary Processor shared memory range.

4.7 Device ID and Revision Registers

The **NS486SXF** contains two 8-bit read-only registers that contain the Part ID and Revision, respectively. These two registers allow software the ability to determine specifically determine the device type and revision.

4.7.1 Device ID Register

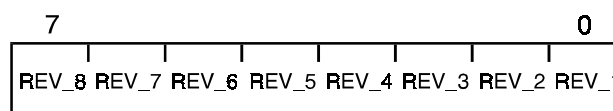
The **NS486SXF** device ID number is contained in the read-only Device ID Register at I/O address EFC2h. The **NS386SXF** device ID number is 01h.



Bits 7-0: ID_7-ID_1 — These eight bits provide the device ID number of the **NS486SXF**, which is 01h.

4.7.2 Device Revision Register

The **NS486SXF** device revision number is contained in the read-only Device Revision Register at I/O address EFC3h. The present **NS386SXF** device ID number is 02h.



Bits 7-0: REV_7-REV_1 — These eight bits provide the device revision number of the **NS486SXF**, which is currently 02h.

5.0 NS486 CPU Overview

5.1 Architectural Overview

The NS486 32-bit Processor Core is a low-cost Microsoft at Work compatible CPU, designed for embedded control applications. It features:

- Small die size
- Low power consumption (high “MIPS per mA”)
- Low voltage capability (3.3V or 5V operation)
- Efficient 3-stage pipeline, giving performance between 386 and 486
- On-chip 1k-byte instruction cache
- Write buffer

The NS486’s core processor 32-bit architecture provides powerful programming resources to match the powerful on-board hardware resources. NS486 supports segmented memory management and multitasking via hardware. A sophisticated protection scheme includes privilege level checking (with four levels) segment type checking (code and data segments), access rights checking, and other protection mechanisms in hardware. Finally, internal debug registers support fast program development and debug.

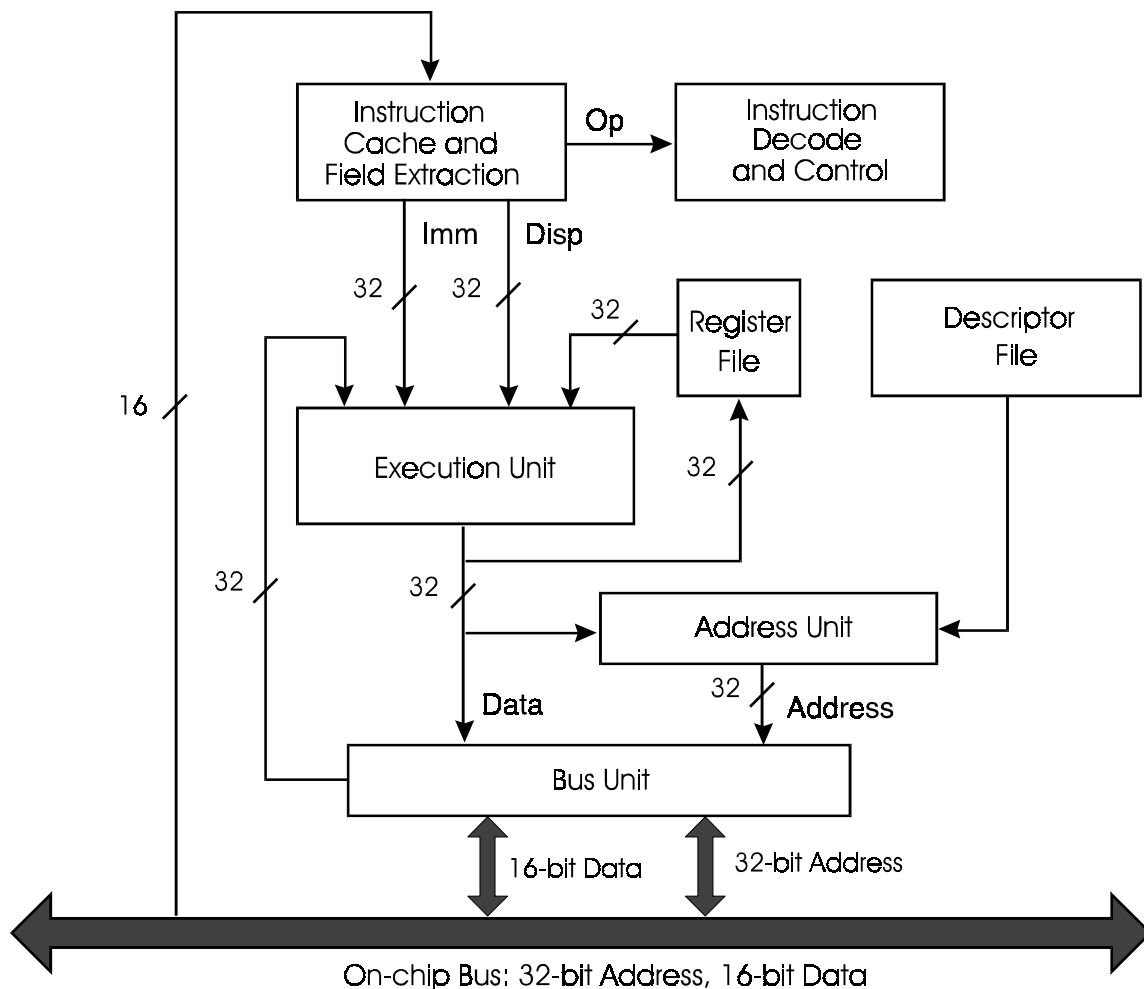


Figure 5-1 Processor Core Block Diagram

5.2 Key Differences From The 486

Some 386/486 CPU functions not required for embedded control type applications are not implemented in the NS486 processor core. However, NS486's 32-bit processor is instruction set compatible with the standard Intel486 processor with the following exceptions:

- 1) NS486 operates in protected mode only. This means that NS486 does not support virtual 8086 mode or real mode operation. Note that it also means that upon reset, the NS486 starts up in protected mode versus the standard '486's real address mode state. This means that internal register values will be different at reset.
- 2) NS486 does not support virtual memory paging and there is no Page Unit or Translation Lookaside Buffer.
- 3) NS486 does not support the use of a hardware floating point co-processor. This means that NS486 will execute floating point instructions in software only. There are no Floating Point instructions. All coprocessor ESCAPE instructions will trap instead, allowing software emulation of all Floating-Point instructions. The WAIT instruction will execute as a no-operation instruction; in effect, the emulated co-processor always appears to be finished.
- 4) NS486 has a 1 Kbyte on-board instruction cache instead of an 8 Kbyte unified cache featured on the Intel 486.
- 5) NS486 does not include the register resources to support those functions that are not implemented. In some cases, this means some bits in some registers are reserved and not used (the EFLAGS register and Control Register 0). In some other cases, it means that the registers controlling that resource do not exist (Control Registers 2 and 3).

Other instructions whose purpose is to affect non-existent registers or control bits will act as no-operation instructions. The result of reading a non-existent register or bit is unde-

fined. The WBINVD (Write Back and Invalidate Data Cache) instruction is not implemented in NS486. The INVLPG (Invalidate TLB Entry) instruction is not implemented in NS486.

Otherwise, the NS486 core processor executes all '486 instructions (appropriate to protected mode), including debug instructions and provides the same programming model and register set as the standard '486.

At reset, unlike the standard '486, the NS486 starts up in 32 bit protected mode instead of real mode (since real mode is not implemented in the NS486 CPU). The processor sets the memory management registers to point to all of memory and loads the EIP register to jump to a location in the user ROM. From there, the processor jumps to the initialization code.

Where changes have been made to the standard '486 processor, they are minor and have been made to improve performance, reduce cost and eliminate unneeded resources.

Using advanced sub-micron semiconductor processes increases the number of operations performed in each clock cycle. This reduces the number of instruction pipeline stages from five to only three. The ability to perform multiple operations simultaneously in one clock cycle improves both performance and power efficiency. By eliminating the extra locking and control mechanisms needed for a five-stage pipeline, significant silicon real estate space is saved, power consumption is cut and there are fewer stalls. Further, the control logic needed for full 8086 compatibility has been eliminated. With the elimination of Virtual Memory (paging), linear addresses are identical with the actual physical addresses. These changes have little impact in embedded applications and save power and lower device cost.

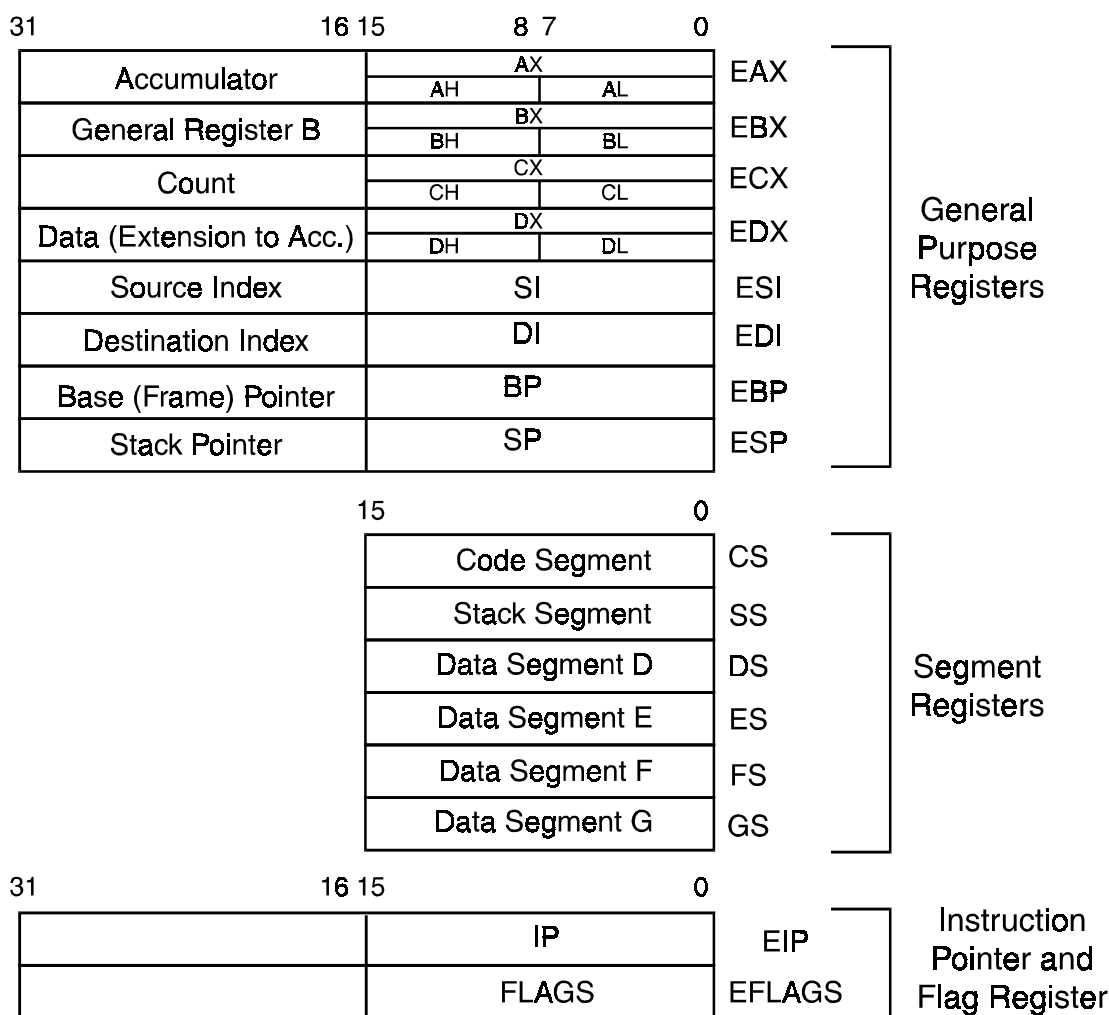
5.3 Register Set

The NS486 Processor Core implements an industry standard 486 architecture. NS486 processor resources are controlled through several sets of registers including application program registers (the general register set), memory management registers, debug registers and control registers.

The NS486 processor application registers include eight 32-bit general purpose registers, six 16-bit segment registers, a 32-bit Instruction Pointer register

and a 32-bit Extended Flags register. Memory management registers include the GDTR (Global Descriptor Table Register), IDTR (Interrupt Descriptor Table Register), LDTR (Local Descriptor Table Register) and the TR (Task Register). There are six Debug Registers and one Control Register.

Figure 5-2 General Purpose, Segment and Flag Registers



5.3.1 Application Registers

The NS486 General Registers, Segment Registers, Instruction Pointer Register and EFLAGS Register are shown in Figure 5.2. The General Purpose Registers typically contain arithmetic and logical instruction operands and are the major user registers of the processor. The Segment Registers hold segment selectors

that point to tables stored in memory that contain the base address of each segment, plus some other memory addressing information. The EFLAGS Register contains status bits that reflect the status of the processor after execution of instructions and some control bits that control the operation of some instructions. The Instruction Pointer Register points to the next instruction to be executed.

General Purpose Registers

The General Purpose Registers can be divided into four data registers, EAX, EBX, ECX, and EDX, two pointer registers, ESP and EBP, and two index registers, ESI and EDI.

The data registers, EAX, EBX, ECX and EDX are primarily used as data registers, but can also be used as Base or Index pointers for addressing memory. They take on some dedicated functions as well: EAX serves as the Accumulator where appropriate, ECX holds an iteration count for repeated (String) instructions and for some instructions that implement looping, and EDX serves as an overflow extension to the Accumulator, for example in multiplication. These registers may be accessed in units of **bytes** (8 bits), **words** (16 bits) or **dwords** (32 bits: “double words”). When accessed as dwords, the registers are referred to as EAX, EBX, ECX and EDX. When accessed as words, they are called AX, BX, CX and DX, and their least-significant 16 bits are used. When accessed as bytes, either of their two least-significant bytes may be referenced: AL, BL, CL and DL refer to the least-significant byte (bits 7–0), and AH, BH, CH and DH refer to their next more significant byte (bits 15–8). When a value is written to a subsection of a register, the remainder of the register is unaffected.

The registers ESI, EDI and EBP are primarily used as Indexes or Base pointers for addressing, but may also be manipulated as 16-bit or 32-bit data (bytes cannot be accessed individually). ESI and EDI are named for their special role in string instructions: Source Index and Destination Index, respectively. String instructions use them as addresses, and auto-increment or auto-decrement them in their execution. The EBP register is the “Base Pointer”, or “Frame Pointer”, and is tailored for addressing a fixed point in the stack (the “activation record”), which holds the parameters and local variables for the current procedure. When these registers are accessed as 16-bit values, they are referred to as SI, DI and BP.

The NS486 processor implements a stack using the ESP register. The stack is accessed during the PUSH and POP instructions, procedure calls, procedure returns, interrupts, exceptions, and interrupt/exception returns. During these operations, the NS486 processor automatically sets the value of the ESP register.

The ESP register may be used only as a Base register in address generation, and it may be manipulated as 16-bit or 32-bit data. When referenced as a 16-bit value, it is called “SP”.

Segment Registers

Segmentation is a means of grouping data structures within the memory space of the NS486 processor. There are three basic types of segments - code, data, and stack. The processor uses six 16-bit segment registers to find the code, data and stack information in memory. The NS486 segment registers hold a segment selector that references a segment descriptor defined in the descriptor table for each segment. This segment selector contains a 13-bit index, a table indicator bit (TI), and a two-bit requested privilege level (RPL) field.

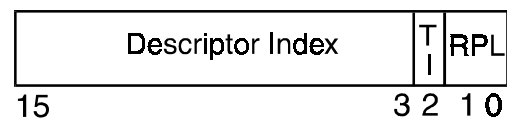


Figure 5-3 Segment Selector

Loading a selector value into a Segment Register triggers a reference to an entry of one of two Descriptor Tables, both of which are provided by the Operating System. This **descriptor** entry (containing the Segment’s base address, length, protection level and other attributes) is loaded into a **shadow register** associated with that Segment Register, where it is held until that Segment Register is changed again. The shadow registers are not directly accessible by user code.

The set of valid selector values for a Segment Register is determined by the Operating System. The resulting Segment Base address is assigned by the Operating System and is generally invisible to user code; this is the means by which code is made relocatable. The selector value placed into a Segment Register consists of three fields, as shown in Figure 5.3:

- RPL = Requested Privilege Level. These two bits can be used to restrict the privilege level of accesses to the selector’s segment. Normally, this field is zero.

- **TI = Table Identifier:** A zero in this bit means that the Global Descriptor Table (**GDT**, pointed to by the register GDTR) is to be referenced, and a one in this bit means that the current task's Local Descriptor Table (**LDT**, pointed to by the register LDTR) is to be referenced.
- **Descriptor Index** is the entry number of the specified Descriptor Table to be referenced. Entry number 0 of the GDT is a "null" descriptor: placing a selector value of all zeroes into a Segment Register will cause a trap to occur on any subsequent use of that Segment Register.

The CS register is special in that it cannot be directly loaded; it is altered instead by any instruction that performs an intersegment jump or call, or any instruction that causes an interrupt or exception to occur. The intersegment form of the **CALL** or **JUMP** instruction provides a 16-bit Selector value to identify the segment into which control is to be transferred, and an Offset value to identify the entry point of the desired code within the segment. These values may be supplied as immediate information, or may be supplied from memory using a general addressing mode.

Although the CS register is manipulated differently, it can still be used with a Segment Override prefix to read data within the current Code Segment, if the segment is marked to allow such use. However, a Code Segment is never available for writing data, and a trap will occur if this is attempted.

Addressing: Segments

In addressing an operand, the CPU uses certain Segment Registers by default. Code is always fetched using the Segment Base information associated with the CS register's contents. Data is accessed using the DS register by default, with a few exceptions:

- Any instruction accessing memory using either the ESP or EBP register as the Base Register will use the SS (Stack Segment) register by default. Push and Pop instructions always use SS.
- Some string instructions (e.g. **MOVS**) will always use the ES register for the destination operand.

Default selections may be overridden by placing the name of the desired segment register and a colon character in front of a memory addressing expression (for example, "FS:13[EBX]"). This causes a one-byte

Segment Override prefix to be added before the opcode, which temporarily overrides the default Segment Register selection for that instruction, and causes the requested Segment Register to be used instead.

Data and Address Size Defaults and Overrides

Code may be designated as belonging to a 16-bit segment, or a 32-bit segment. Depending on the type of Code Segment, certain types of operations are more efficient.

32-Bit vs. 16-Bit Data

In 32-bit mode, an instruction will access 8-bit and 32-bit operands most efficiently. Accessing a 16-bit operand (for example, **BX** instead of **EBX**) requires the use of an Operand Size Override prefix byte, and takes more time.

In 16-bit mode, an instruction will access 8-bit and 16-bit operands most efficiently; this mode implements accesses that are similar to (but still a superset of) the 8086 and 80286 architectures. 32-bit accesses are still possible (for example, to access the entire **EAX** register), but this requires the use of an Operand Size Override prefix byte, and takes more time. This mode may still be more desirable if more 16-bit than 32-bit data is being manipulated in the program.

32-Bit vs. 16-Bit Addressing

In 32-bit Address Mode, addresses are calculated as 32-bit quantities. The entire 32-bit contents of base and index registers are used, and displacement values may be either 8 bits or 32 bits in length. To use register and displacement values that are only 16 bits in size, an Address Size Override prefix byte must be appended to each instruction that does so. This, however, also restricts the selection of addressing modes to the 16-bit subset described below.

In 16-bit Address Mode, a subset of the addressing modes are used, which are characteristic of the earlier 8086 and 80286 architectures. All of the addressing modes perform 16-bit address calculations to generate the Offset portion of an address. Only the lower 16 bits of addressing registers are used (e.g., **BX** instead of **EBX**). Displacement values may be either 8 bits or 16 bits in length. Carries out of bit 15 of the offset cal-

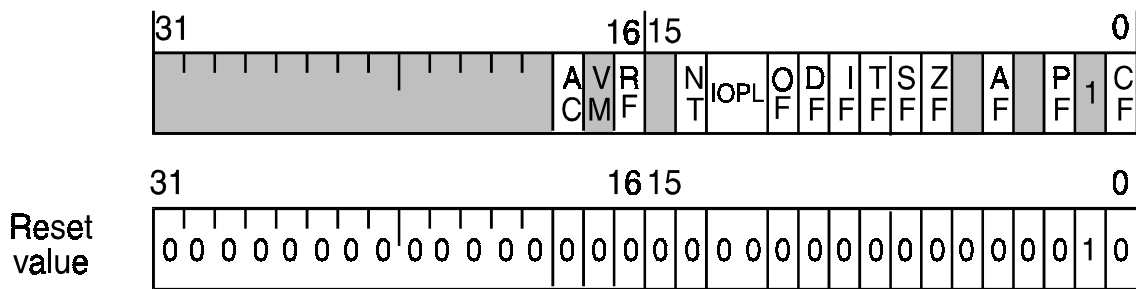
ulation are lost. Scaling of indexes is not available. Only the BX and BP registers may be used as Base registers, and only the SI and DI registers may be used as Index registers. To escape from these restrictions, and use register and displacement values that are 32 bits in size, an Address Size Override prefix byte must be appended to each instruction that does so. The addressing modes available in 16-bit and 32-bit Address Modes, and their encodings, are shown in Tables 5.5 and 5.6.

The EFLAGS Register

The EFLAGS Register contains control and status bits that indicated the current state of the NS486 processor and that control the operations of some instructions.

The EFLAGS Register cannot be directly accessed and therefore it is accessed using the PUSHF and POPF instructions. The following table shows the EFLAGS bits and their meanings.

Figure 5-4 EFLAGS Register



Reserved. Do not change these bits from their original values.
The VM bit (bit 17) is read-only, always zero.

Table 5-1: Functions of EFLAG Register Bits

Flag	Function
CF	Carry Flag: Carry or Borrow from most-significant bit.
PF	Parity Flag: Exclusive NOR of lower 8 bits of result.
AF	Auxiliary Flag: Carry or borrow from Bit 3.
ZF	Zero Flag: Zero result sets ZF to 1; else ZF is cleared.
SF	Sign Flag: set to most-significant bit of result. (1 = negative number)
TF	Trap Enable Flag: When set by software, causes Interrupt 1 (Debug) at completion of each instruction.
IF	Interrupt Enable Flag: When set by software, enables maskable interrupts. This bit may be altered only if the program is authorized to perform I/O instructions (see IOPL below).
DF	Direction Flag: Set by software to control String instructions. DF = 0: Post-Increment ESI and/or EDI DF = 1: Post-Decrement ESI and/or EDI
OF	Overflow Flag: Set to 1 if 2's Complement overflow occurs; else cleared.
IOPL	I/O Privilege Level: 2 bits. Determines at what privilege level a program must be running in order to perform I/O instructions without port-by-port checks. Only Level 0 programs may alter this field.
NT	Nested Task: Determines whether an IRET instruction will return directly (NT=0) or via a task switch (NT=1).
RF	Restart Flag: RF = 0 signals that the Debug interrupt should occur before the first instruction begins (i.e., immediately), and RF = 1 signals that the Debug interrupt should occur only after the first instruction finishes. This bit can be altered by software action only with an IRET instruction or a task switch.
VM	Virtual 8086 Mode: Not implemented in the NS486 processor. This bit is permanently zero. NOTE: For upward software compatibility, do not attempt to write a 1 to this bit.
AC	Alignment Check. Cleared to zero at reset. This bit is non-functional as Alignment Check (which is not implemented in the NS486SXF). Read/Write.

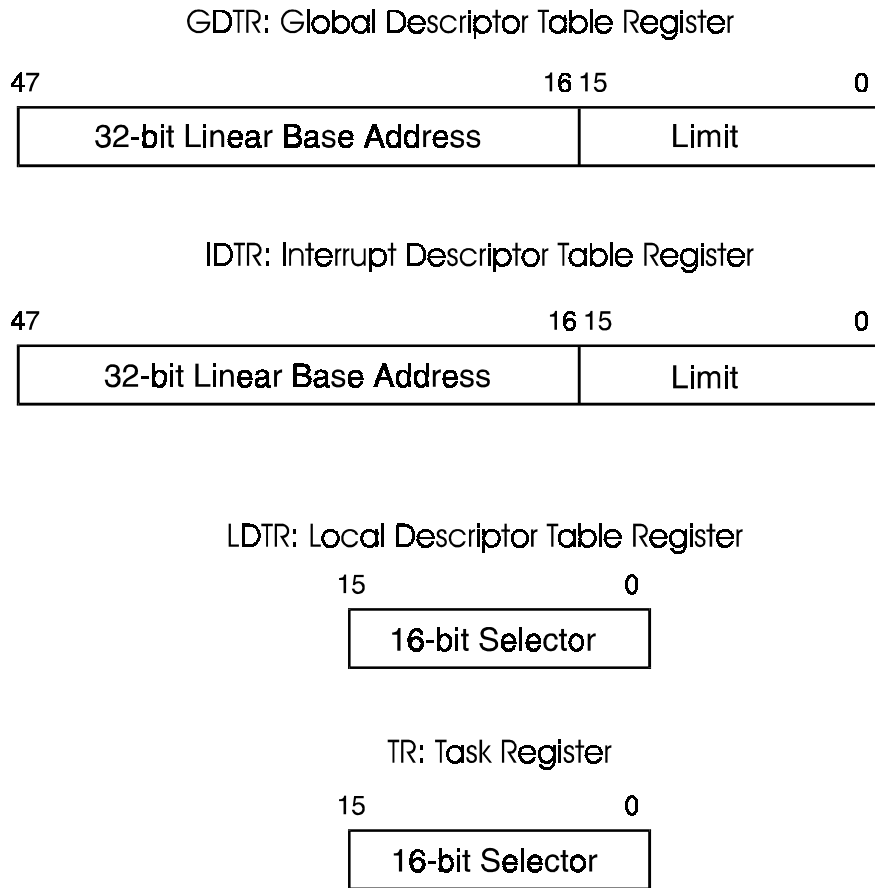
The Instruction Pointer Register

The Instruction Pointer Register (EIP) holds the offset into the current code segment of the next instruction to be executed. This offset is typically incremented when the instruction is being executed. The contents of this register can be modified by an interrupt routine, an exception routine, or an instruction that calls for a jump or call to a non-sequential instruction address.

5.3.2 Memory Management Registers

Figure 5.5 shows the Memory Management Registers: GDTR, IDTR, LDTR and TR. These registers each point to tables that are important to the segmentation features of the architecture. The tables pointed to by the registers GDTR and LDTR contain the segment descriptors that are referenced whenever a selector value is placed into a Segment Register.

Figure 5-5 Memory Management Registers



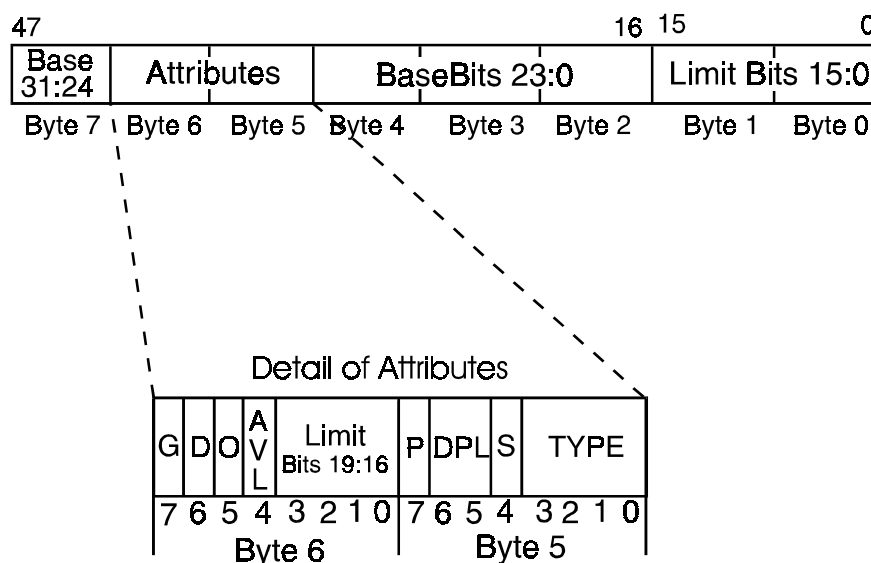
Memory Descriptors

The elements of the tables GDT, LDT and IDT are 8-byte values called **descriptors**. Each provides the necessary information for accessing a segment. Descriptors in the NS486 are in all respects identical to those used in the 486 architecture. They consist of:

- A 32-bit Base Address
- A 20-bit Limit
- A 12-bit set of Attributes (e.g., Protection Level, Access Rights)

The format of a typical descriptor (one which is being used to describe a memory segment) is shown in Figure 5.6. The 32-bit Segment Base value is split for compatibility with older 286 segment descriptor formats, as is the Limit field.

Figure 5-6 Memory Segment Descriptor Format



In the Attribute field, the subfields have the following meanings:

- G is the Granularity of the Limit field: G=0 means that the Limit is expressed in bytes, giving a maximum segment size of 1 Mbyte. G=1 means that the Limit is expressed in 4-Kbyte units, giving a maximum segment size of 4 Gbytes. The Granularity bit does not affect the granularity of the Base Address, which has one-byte resolution regardless.
- D is the Default bit, used for backward compatibility with the 16-bit architectures of the 8086 and 80286 processors. D=0 forces backward compatibility, and D=1 allows all of the 486-class architectural features within the segment. The uses of this bit vary with the circumstances of the segment's use:
 - In segments containing code, the D bit indicates the default Data Size and Address Size modes for the code in that segment.
 - D=0 means that the defaults are 16-bit Data and 16-bit Address, requiring use of the Address Size and Data Size override prefix on any instruction that uses 486-class extended features. D=1 means that the default Address Size and Data Size are 32-bit, requiring use of override prefixes for instructions to use the 16-bit modes.
 - In segments used for stacks, where memory usage expands downward to lower addresses, the D bit indicates whether the limit is applied on a maximum of 4 Gbytes (D=1), or 64 Kbytes (D=0). Also, when the segment is being referenced through the SS Segment Register, the D bit determines whether the entire ESP register is used when pushing or popping data (D=1) or only the 16-bit SP portion of it (D=0).
- O is the Default Operation Size; 0 = 16-bit segment; 1 = 32-bit segment.
- AVL is a bit that is "Available" for software use. CPU hardware does not use or alter it.
- P is the "Present" bit. It must be a "1" when the segment is referenced, or a trap is triggered. The P bit could be cleared for debugging purposes, or for gathering statistics on segment usage.
- DPL is the Descriptor Privilege Level: the privi-

lege level of the segment.

- S is the Descriptor Type. DT = 1 means that the descriptor describes a memory segment. S = 0 means that the descriptor describes a Gate or a System segment.
 - TYPE is a 4-bit field, divided into two subfields: the most-significant three bits define the types of accesses that are allowed on the contents of the segment:
 - 000x = Read-Only
 - 001x = Read/Write
 - 010x = Read-Only, with inverted interpretation of limit (stack-oriented)
 - 011x = Read/Write, with inverted interpretation of limit (stack-oriented)
 - 100x = Execute-Only
 - 101x = Execute/Read
 - 110x = Execute-Only, conforming Privilege Level to match calling code's level.
 - 111x = Execute/Read, conforming Privilege Level to match calling code's level.
- The least-significant bit of the TYPE field (the "Accessed" bit) indicates whether this segment has been accessed by the processor since the last time operating-system software cleared the bit. It is set to a 1 by the processor hardware on an access, if it is not already set.

A memory segment descriptor is made compatible with 16-bit 286 code by setting the top 16 bits of the descriptor to zeroes.

Other Descriptors

Two other classes of descriptor exist: **system segments** and **gates**. They are distinguished from memory segments by the fact that the S bit in their descriptor is a 0 instead of a 1.

A **system segment** contains memory information that is not directly accessible to running code. This will be either a Task State Segment (**TSS**) or a Local Descriptor Table (**LDT**). Figure 5.7 shows the format of the descriptor for a system segment. The overall format is the same as a memory segment descriptor, except that the Type field is interpreted differently: the Type field encodings are listed in Figure 5.7.

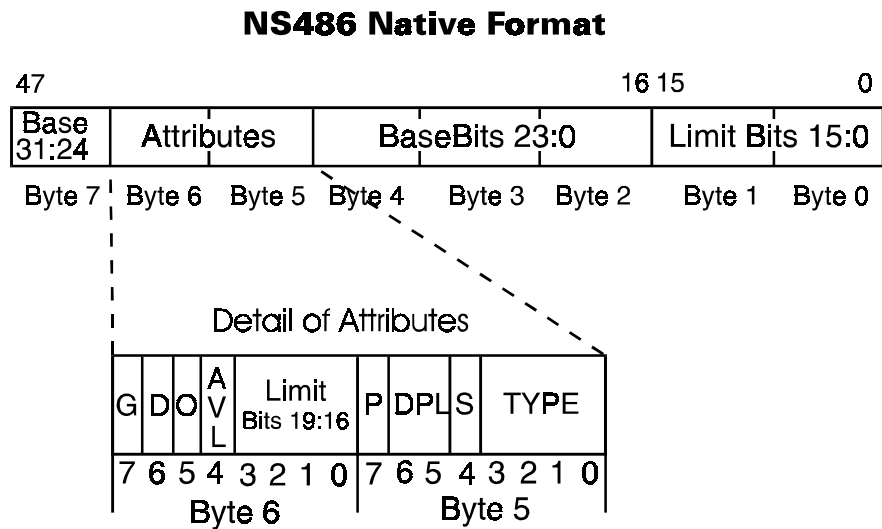
A **gate** is a method of regulating calls. It substitutes for an executable memory segment's descriptor, and

limits an intersegment call to a specific entry point in the destination segment. Transfer of control between differing privilege levels requires use of a gate. A gate can regulate entry to a task, an interrupt or trap service routine, or a called procedure.

The format of the descriptor for a gate is shown in Figure 5.8. It is distinguished from normal memory descriptors by the fact that the S bit is zero. It is distinguished from system segment descriptors by the contents of the Type field, which are mutually exclusive with those for system segments. The fields within a gate descriptor are significantly different from other descriptors. Instead of providing a Base Address and Limit, the gate descriptor provides a selector value, which identifies the descriptor of the executable segment to which calls will be made. It also provides a 32-bit offset into that segment, which identifies the entry point (this replaces the offset from the calling instruction).

The Attribute field of a gate descriptor is significantly different from other types of descriptors. The P, DPL, S and Type subfields are in the same positions, but the G, D, AVL and upper Limit fields do not exist. Instead, because a call through a gate will often involve switching stacks, a Dword Count field is provided, which specifies how many dwords of information (0–31) will be copied from the top of the old stack to the new stack.

Figure 5-7 System Segment Descriptor Format

**Possible Types:**

- 1 = 286-Compatible Task State Segment (Available)
- 2 = Local Descriptor Table (LDT) Segment
- 3 = 286-Compatible Task State Segment (Busy)
- 9 = Elentari Task State Segment (Available)
- B = Elentari Task State Segment (Busy)

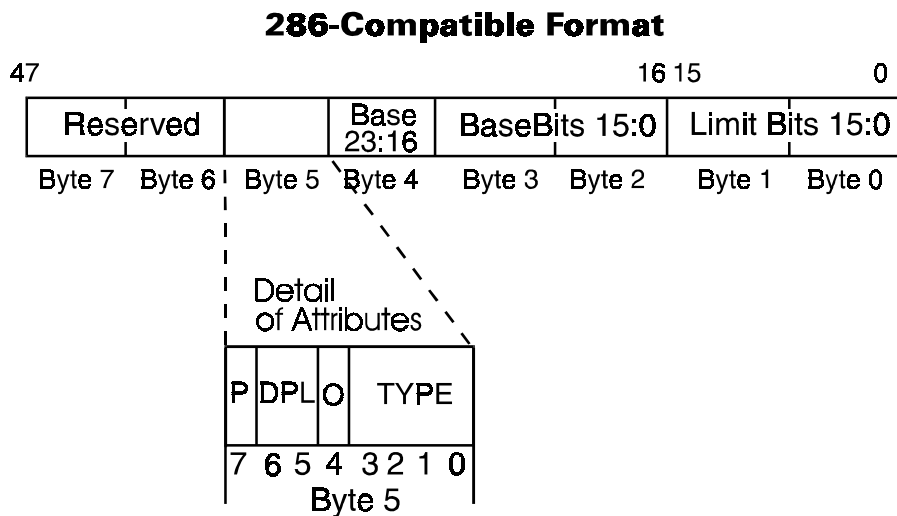
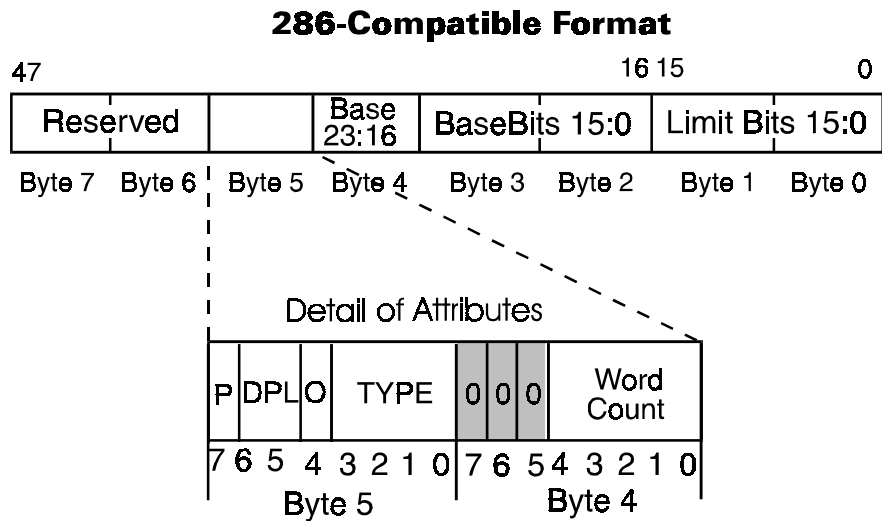
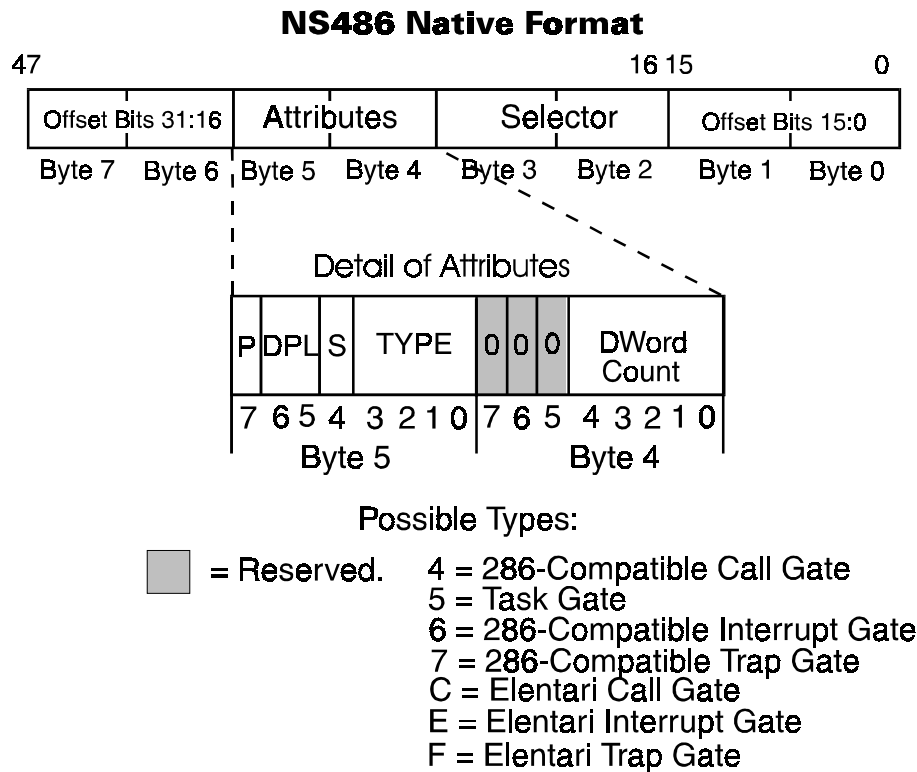


Figure 5-8 Gate Descriptor Format



5.3.2.1 GDTR: Global Descriptor Table Register

The 48-bit register **GDTR** points to the beginning of the GDT, and defines in the Limit field the length of the table. If there are N descriptors in the table, the Limit field will be encoded as $(8 \times N) - 1$.

A segment listed in the GDT may be referenced by a program by loading a Selector value into one of the Segment Registers, as described in “Addressing: Segments” above.

5.3.2.2 IDTR: Interrupt Descriptor Table Register

The 48-bit register **IDTR** points to the beginning of the IDT, and defines in the Limit field the length of the table. If there are N descriptors in the table, the Limit field will be encoded as $(8 \times N) - 1$.

The descriptors are used to describe the segment to which control is transferred by an interrupt. The table entry is selected by means of the interrupt vector number, as given in the table on the next page.

Table 5-2: Exceptions and Interrupts

Vector Number (IDT Index)	Description
0	Divide Error: Divide by Zero, or Divide Overflow
1	Debug Trap; Hardware Breakpoint
2	Non-Maskable Interrupt (NMI)
3	Software Breakpoint
4	INTO Instruction: Overflow Detected
5	BOUND Instruction: Range Exceeded
6	Invalid Opcode
7	Device Not Available: Coprocessor Operation
8	Double Fault
9	(Reserved for Future Use)
10	Invalid Task State Segment
11	Segment Not Present
12	Stack Exception
13	General Protection Fault
14	Page Fault (Unused by NS486)
15	(Reserved for Future Use)
16	Floating Point Error (Unused by NS486 Hardware)
17	Alignment Check (Unused by NS486)
18—31	(Reserved for Future Use)
32—255	Available for Maskable Interrupts
	Any interrupt “n” (0–255) may also be triggered by an “INT n” instruction.

5.3.2.3 LDTR: Local Descriptor Table Register

This 16-bit register contains a Selector value, identifying a segment which contains a table of descriptors. In a multitasking system, there will be multiple tables of this kind. The selector placed in this register will be referencing an element of the GDT.

A segment listed in a task’s LDT may be referenced by a program by loading a Selector value into one of the Segment Registers, as previously described.

5.3.2.4 TR: Task Register

This 16-bit register contains a GDT selector value, identifying a segment whose purpose is to hold the state of the CPU and other task context information. This type of segment is referred to as a Task State Segment (TSS). The first 104 bytes of the TSS are used by the CPU hardware; these locations are shown

in Figures 5.9 and 5.10. Additional information needed by Operating System software may occupy any space beyond these locations.

Figure 5-9 Task State Segment (TSS) Format

NS486 Native Format

I/O Permission Bitmap offset	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	T	64h
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	LDT Selector		60h
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	GS		5Ch
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	FS		58h
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	DS		54h
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	SS		50h
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	CS		4Ch
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	ES		48h
EDI			44h
ESI			40h
EBP			3Ch
ESP			38h
EBX			34h
EDX			30h
ECX			2Ch
EAX			28h
EFLAGS			24h
EIP			20h
			1Ch
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	SS2		18h
ESP2			14h
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	SS1		10h
ESP1			0Ch
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	SS0		08h
ESP0			04h
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	LINK (Selector of Previous TSS)		00h

- This area is vacant because the CR3 register (PDBR) does not exist. This is a difference from the 486 architecture.
- Static information, not written during a task switch. (Same as 486.)

An optional I/O Permissions Bitmap may also appear after these fields, pointed to by the word at location 66 hex; this is used to grant unprivileged programs access to I/O locations on a port-by-port basis (0 grants access, 1 denies access). All I/O can be denied by set-

ting location 66 hex to point beyond the end of the TSS: no Permissions bitmap is then required.

The NS486 processor uses the same format as that documented for the 486 architecture, except for locations 1C–1F (CR3/PDBR), which are not used because the CR3 register does not exist.

Figure 5-10 286-Compatible Task State Segment (TSS) Format

NS486 286-Compatible Format

Note: Recommended for use only in all 16-bit systems.

15	0
LDT Selector	2Ah
DS	28h
SS	26h
CS	24h
ES	22h
DI	20h
SI	1Eh
BP	1Ch
SP	1Ah
BX	18h
DX	16h
CX	14h
AX	12h
FLAGS	10h
IP	0Eh
SS2	0Ch
SP2	0Ah
SS1	08h
SP1	06h
SS0	04h
SP0	02h
LINK (Selector of Previous TSS)	00h

Static information, not written during a task switch. (Same as 486.)

5.3.3 Debug Registers

The Debug Registers, DR0–DR3, DR6 and DR7, are shown in Figure 5.11. They implement fully the documented functions of the 486 register set.

The registers DR0 through DR3 specify four addresses, in linear form, which will be monitored to generate

Debug traps (Interrupt 1). Registers DR6 and DR7 provide status and control fields for the breakpointing functions. These functions are listed in Table 5.3 and Table 5.4.

Figure 5-11 Debug Registers

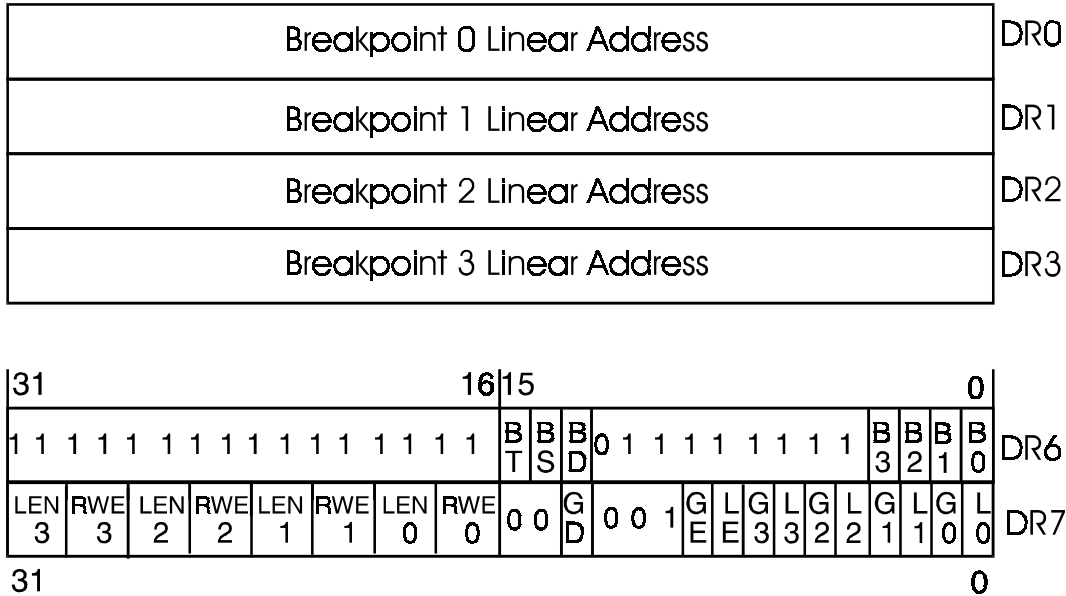


Table 5-3: Functions of Debug Status Register Bits (DR6)

Field	Function
These bits are only set by hardware; software must clear them.	
B0–B3	These bits are set to 1 by processor hardware when the breakpoint condition in the corresponding register (DR0–3) is met. The Debug Trap need not be enabled for these bits to be set.
BD	1 = Debug Trap was triggered because an attempt was made to access one of the Debug registers while the DR7 register’s GD bit was set.
BS	1 = Debug Trap was triggered by the TF bit in the EFLAGS register (Single-Stepping).
BT	1 = Debug Trap was triggered by a task switch into a task whose T bit was set (in its TSS segment).

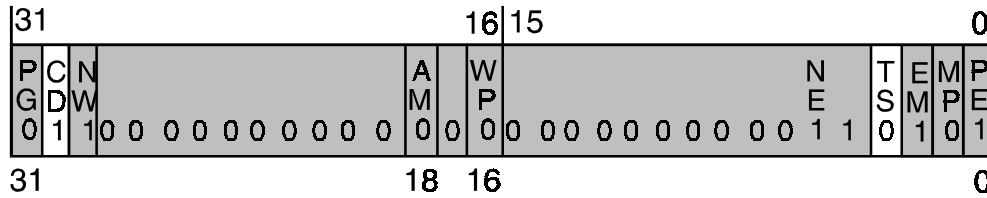
Table 5-4: Functions of Debug Ctrl. Reg. Bits (DR7)

Field	Function
LEN0 LEN1 LEN2 LEN3	<p>For each of the registers DR0–DR3, there is a corresponding two-bit LEN field (LEN0, LEN1, LEN2 and LEN3, respectively). The LEN field specifies the operand length for which data accesses are monitored. Encodings are:</p> <p>00 = one byte (Required for all Execution breakpoints)</p> <p>01 = two bytes</p> <p>10 = (reserved)</p> <p>11 = four bytes</p> <p>Note that for valid breakpointing, the address in the corresponding DR0–DR3 register must be aligned to a multiple of the length chosen here.</p>
RWE0 RWE1 RWE2 RWE3	<p>For each of the registers DR0–DR3, there is a corresponding two-bit RWE field (RWE0, RWE1, RWE2 and RWE3, respectively). The RWE field specifies the conditions under which the corresponding address in DR0–DR3 will report a match. Encodings are:</p> <p>00 = Execution as an instruction</p> <p>01 = Data Write</p> <p>10 = (reserved)</p> <p>11 = Data Read or Write</p>
G0 / L0 G1 / L1 G2 / L2 G3 / L3	<p>Global / Local Breakpoint Enables. These bits enable the breakpoint conditions to trigger the Debug Trap (Interrupt 1). For each Debug register (DR0–DR3), a trap is enabled if either the L or G bit contains a 1. The L bits differ from the G bits in that they are cleared whenever the processor performs a task switch.</p>
GE / LE	<p>Global / Local Exact Data Breakpoint Mode. These bits are remnants of the earlier 386 architecture, and are not used by the NS486.</p>
GD	<p>Global Debug Access Detect. When set to a 1, this bit causes any instruction that accesses any of the Debug registers to cause a Debug trap instead. The trap clears the GD bit, allowing code within the Debug trap service routine to access these registers, and sets the BD bit in DR6. This feature allows an In-Circuit Emulator (ICE) or other development-mode monitor to emulate these registers for the application system's code, while preventing that code from accessing them directly.</p>

5.3.4 Control Registers

Due to the fact that the NS486 is not designed to support Paged virtual memory, nor a Floating Point co-processor, most of the 486 Control Register set is not

used. Registers CR1, CR2 and CR3 do not exist, and accesses to them will generate an undefined result.



■ = Read Only. For compatibility, rewrite only their existing values into them.

0 = cleared to 0 on Reset
1 = Set to 1 on Reset

Figure 5-12 Control Register CR0

The CR0 register, however, does exist; its bit map is shown in Figure 5.12. Most of its bits are read-only, with the following exceptions:

- CD = Cache Disable. Setting this bit will freeze the contents of the on-chip instruction cache. Setting the CD bit will not prevent the cache from being accessed. It will prevent any further writes to the cache, thus “locking” the data in the cache. Any instruction fetch from an address that has been cached will still come from the cache. There are two ways to prevent the cache from being accessed:
 - 1) Never enable the cache, thus there will never be valid data in it.
 - 2) After disabling the cache, execute the INVD instruction which will invalidate the cache and no further access will be from the cache. Note the INVD must be executed in Protection Level 0.
- TS = Task Switch status. This bit is set to one whenever a task switch occurs.

Other bits are read-only, and reflect the following states:

- Paging off (PG = 0)
- Non-Write-through Cache feature on (NW = 1)
- Alignment Trap Mask Off (AM = 0)

- Page Write Protect Off (WP = 0)
- Numeric Error processing is Standard (NE = 1)
- Trap on Coprocessor Operations (EM = 1)
- Do Not Trap WAIT instructions (MP = 0)
- Protected Mode Operation (PE = 1)

5.4 General Operands and Memory

The NS486 architecture can be described as a two-operand, one-address machine. In the most common instructions (for example, MOV and two operand arithmetic instructions), the two operands consist of a **reg operand**, which may be any of the General Registers, and a **general operand** (or **r/m operand**), which may also be one of the General Registers, but is allowed to reside in memory as well.

Memory is addressed in units of bytes. The architecture is “Little-Endian,” meaning that the least-significant byte of a multibyte value is stored at the lowest byte address. That byte address is interpreted as the address of the operand. The least-significant bit of a value is designated Bit 0. There is no architectural requirement that operands be aligned to specific addresses, but most efficient operation is obtained (and the on-chip Debug features operate best) when operands are aligned to addresses that are integral units of

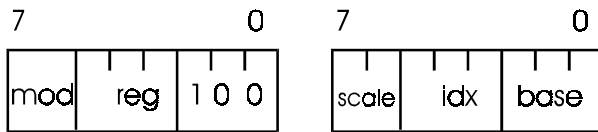
Table 5-5: 16-Bit Addressing Modes by mod and r/m Fields

mod	r/m	Addressing Mode
00	000	DS:[BX+SI]
	001	DS:[BX+DI]
	010	SS:[BP+SI]
	011	SS:[BP+DI]
	100	DS:[SI]
	101	DS:[DI]
	110	DS:disp16 (Absolute within Segment)
	111	DS:[BX]
01	000	DS:[BX+SI]+disp8
	001	DS:[BX+DI]+disp8
	010	SS:[BP+SI]+disp8
	011	SS:[BP+DI]+disp8
	100	DS:[SI]+disp8
	101	DS:[DI]+disp8
	110	SS:[BP]+disp8
	111	DS:[BX]+disp8
10	000	DS:[BX+SI]+disp16
	001	DS:[BX+DI]+disp16
	010	SS:[BP+SI]+disp16
	011	SS:[BP+DI]+disp16
	100	DS:[SI]+disp16
	101	DS:[DI]+disp16
	110	SS:[BP]+disp16
	111	DS:[BX]+disp16
<p>The following modes reference registers rather than memory. The same encodings apply in the r/m field (with mod = 11) or in the reg field. The register, or portion of the register, is determined by the width of the operand: 32, 16 or 8 bits as determined by other fields of the opcode and/or the Default Data Size attribute of the code segment being executed.</p>		
11	000	EAX or AX or AL
	001	ECX or CX or CL
	010	EDX or DX or DL
	011	EBX or BX or BL
	100	ESP or SP or AH
	101	EBP or BP or CH
	110	ESI or SI or DH
	111	EDI or DI or BH

Table 5-6: One-Byte Addressing Modes by mod and r/m Fields

mod	r/m	Addressing Mode
00	000	DS:[EAX]
	001	DS:[ECX]
	010	DS:[EDX]
	011	DS:[EBX]
	100	escape to 2-byte format
	101	DS:disp32 (Absolute within Segment)
	110	DS:[ESI]
	111	DS:[EDI]
01	000	DS:disp8[EAX]
	001	DS:disp8[ECX]
	010	DS:disp8[EDX]
	011	DS:disp8[EBX]
	100	escape to 2-byte format
	101	SS:disp8[EBP]
	110	DS:disp8[ESI]
	111	DS:disp8[EDI]
10	000	DS:disp32[EAX]
	001	DS:disp32[ECX]
	010	DS:disp32[EDX]
	011	DS:disp32[EBX]
	100	escape to 2-byte format
	101	SS:disp32[EBP]
	110	DS:disp32[ESI]
	111	DS:disp32[EDI]
The following modes reference registers rather than memory. The same encodings apply in the r/m field (with mod = 11) or in the reg field. The register, or portion of the register, is determined by the width of the operand: 32, 16 or 8 bits as determined by other fields of the opcode and/or the Default Data Size attribute of the code segment being executed.		
11	000	EAX or AX or AL
	001	ECX or CX or CL
	010	EDX or DX or DL
	011	EBX or BX or BL
	100	ESP or SP or AH
	101	EBP or BP or CH
	110	ESI or SI or DH
	111	EDI or DI or BH

Figure 5-14 Two-Byte Format



Certain codes appearing in the mod and r/m field indicate that a second byte, the **SIB** (Scaled Index Byte), is present. This byte encodes the rest of the possible addressing expressions, most of which incorporate a Base expression (Table 5.7) to which is appended an Indexing expression (Table 5.8). The General Purpose Register selected as the Index can be scaled by a factor of 1, 2, 4 or 8 before use.

Table 5-7: 32-bit Two-Byte Addressing Modes by mod and base Fields

mod	base	Base Mode Expression
00	000	DS:[EAX]
	001	DS:[ECX]
	010	DS:[EDX]
	011	DS:[EBX]
	100	SS:[ESP]
	101	DS:disp32 (Absolute within Segment)
	110	DS:[ESI]
	111	DS:[EDI]
01	000	DS:disp8[EAX]
	001	DS:disp8[ECX]
	010	DS:disp8[EDX]
	011	DS:disp8[EBX]
	100	SS:disp8[ESP]
	101	SS:disp8[EBP]
	110	DS:disp8[ESI]
	111	DS:disp8[EDI]
10	000	DS:disp32[EAX]
	001	DS:disp32[ECX]
	010	DS:disp32[EDX]
	011	DS:disp32[EBX]
	100	SS:disp32[ESP]
	101	SS:disp32[EBP]
	110	DS:disp32[ESI]
	111	DS:disp32[EDI]

Table 5-8: Indexing Expressions and Encodings

idx	Indexing Expression Appended			
	scale = 00	scale = 01	scale = 10	scale = 11
000	[EAX*1]	[EAX*2]	[EAX*4]	[EAX*8]
001	[ECX*1]	[ECX*2]	[ECX*4]	[ECX*8]
010	[EDX*1]	[EDX*2]	[EDX*4]	[EDX*8]
011	[EBX*1]	[EBX*2]	[EBX*4]	[EBX*8]
100	No index used.			
101	[EBP*1]	[EBP*2]	[EBPX*4]	[EBP*8]
110	[ESI*1]	[ESI*2]	[ESI*4]	[ESI*8]
111	[EDI*1]	[EDI*2]	[EDI*4]	[EDI*8]

5.5 Initial Reset State of Core

The state of the NS486 differs significantly from the documented '486 reset state. First of all, NS486 starts up in Protected Mode instead of Real Mode (which does not exist).

Initial register states are as follows:

Generally, all shadow registers are set to - Base = 0,
Limit = FFFFFFFF, USE32 Mode for code

- EIP = FFFFFFFF0h
- EDX = 00000007h (microcode revision level)
- IDTR, GDTR, TR, LDTR are undefined
- CS, SS, DS, ES, FS, GS are undefined
- Other register contents are undefined.

5.6 Instruction Set Summary

NS486 core processor instructions can be divided into three general classes of operation: Integer, Multi-Segment, and Operating System.

5.6.1 Integer Instructions

Arithmetic

AAA - ASCII Adjust after Addition
 AAD - ASCII Adjust before Division
 AAM - ASCII Adjust after Multiplication
 AAS - ASCII Adjust after Subtraction
 ADC - Add Integers with Carry
 ADD - Add Integers
 BSWAP - Byte Swap
 CBW - Convert Byte to Word
 CWDE - Convert Word to Doubleword

Extended

CMP - Compare
 CWD - Convert Word to Doubleword
 CDQ - Convert Doubleword to Quad-word
 DAA - Decimal Adjust after Addition
 DAS - Decimal Adjust after Subtraction
 DEC - Decrement
 DIV - Unsigned Integer Divide
 IDIV - Signed Integer Divide
 IMUL - Signed Integer Multiply
 INC - Increment
 MUL - Unsigned Integer Multiply
 NEG - Negate (subtract from zero)
 SBB - Subtract Integers with Borrow
 SUB - Subtract Integers

Bit Manipulation

BSF - Bit Scan Forward
 BSR - Bit Scan Reverse
 BT - Bit Test
 BTC - Bit Test and Complement
 BTR - Bit Test and Reset
 BTS - Bit Test and Set

Data Movement

CMPXCHG - Compare and Exchange
 IN - Input from a Port

Conditional Assignment

SETB/SETNAE/SETC - Set byte below
 SETBE/SETNA - Set byte below or equal
 SETE/SETNA - Set byte equal
 SETL/SETNGE - Set byte less
 SETLE/SETNG - Set byte less or equal
 SETNB/SETAE/SETNC - Set byte not below
 SETNBE/SETA - Set byte not below or equal
 SETNE/SETNZ - Set byte not equal
 SETNL/SETGE - Set byte not less
 SETNLE/SETG - Set byte not less or equal
 SETNO - Set byte no overflow
 SETNP/SETPO - Set byte not parity
 SETNS - Set byte not sign
 SETO - Set byte overflow
 SETP/SETPE - Set byte parity
 SETS - Set byte sign

Control Transfer

CALL - Call Procedure
 JB/JNAE/JC - Jump below
 JBE/JNA - Jump below or equal
 JCXZ/JECXZ - Jump CX zero/Jump ECX zero
 JE/JZ - Jump equal
 JL/JNGE - Jump less
 JLE/JNG - Jump less or equal
 JMP - Jump
 JNB/JAE/JNC - Jump not below
 JNBE/JA - Jump not below or equal
 JNE/JNZ - Jump not equal
 JNL/JGE - Jump not less
 JNLE/JG - Jump not less or equal
 JNO - Jump no overflow
 JNP/JPO - Jump not parity
 JNS - Jump not sign
 JO - Jump overflow
 JP/JPE - Jump parity
 JS - Jump sign
 LOOP - Loop While ECX Not Zero
 LOOPE - Loop While Equal
 LOOPZ - Loop While Zero
 LOOPNE - Loop While Not Equal
 LOOPNZ - Loop While Not Zero
 RET - Return from Procedure

LEA - Load Effective Address
 MOV - Move
 MOVSX - Move with Sign Extension

MOVZX - Move with Zero Extension
 OUT - Write to a Port
 POP - Pop off Stack
 POPA/POPAD - Pop All off Stack
 PUSH - Push Onto Stack
 PUSHA/PUSHAD - Push All Onto Stack
 XADD - Exchange and Add
 XCHG - Exchange

Flag Control

CLC - Clear the Carry Flag
 CLD - Clear the direction flag
 CLI - Clear the Interrupt Flag
 CMC - Complement the Carry Flag
 LAHF - Load Flags into AH Register
 POPF/POPFD - Pop From Stack into Flags
 PUSHF/PUSHFD - Push Flags onto Stack
 SAHF - Store AH Register into Flags
 STC - Set Carry Flag
 STD - Set Direction Flag
 STI - Set Interrupt Flag

High-Level Language Support

BOUND - Check Array Index Against Bounds
 ENTER - Enter Procedure
 LEAVE - Leave Procedure

Logic

AND - Boolean AND operation
 NOT - Bit inversion
 OR - Boolean Or operation
 XOR - Boolean Exclusive OR
 RCL - Rotate Through Carry Left
 RCR - Rotate Through Carry Right
 ROL - Rotate Left
 ROR - Rotate Right
 SAL - Shift Arithmetic Left
 SHL - Shift Logical Left
 SHR - Shift Logical Right
 SAR - Shift Arithmetic Right
 SHLD - Shift Left Double
 SHRD - Shift Right Double
 TEST - Test
 XOR - Exclusive OR

String Manipulation

CMPS/CMPSB/CMPSW/CMPSD - Compare String
 INS/INSB/INSW/INSD - Input String

LODS/LODSB/LODSW/LODSD - Load String
 MOVS/MOVSb/MOVSW/MOVSd - Move String
 OUTS/OUTSB/OUTSW/OUTSD - Output String
 REP - Repeat While ECX Not Zero
 REPE/REPZ - Repeat While Equal/Zero
 REPNE/REPNZ - Repeat While Not Equal/Zero
 SCAS/SCASB/SCASW/SCASD - Scan String
 STOS/STOSB/STOSW/STOSD - Store String
 XLAT/XLATB - Table Lookup Translation

Miscellaneous

LOCK - Bus Lock
 NOP - No Operation

5.6.2 Multiple-Segment Instructions

CALL - Call procedure
 INT - Call to Interrupt Procedure
 INTO - On Overflow Call Interrupt Procedure
 IRET - Return from Interrupt
 JMP - Jump
 LDS - Load Pointer to DS
 LES - Load Pointer to ES
 LFS - Load Pointer to FS
 LGS - Load Pointer to GS
 LSS - Load Pointer to SS
 MOV - Move to/from Segment Register
 POP - Pop off Stack into Segment Register
 PUSH - Push onto Stack
 RET - Return

5.6.3 Operating System Instructions

ARPL - Adjust Requested Privilege Level
CLTS - Clear the Task-Switched Flag
HLT - Halt
INVD - Invalidate Cache
LAR - Load Access Rights
LGDT - Load Global Descriptor Table
LIDT - Load Interrupt Descriptor Table
LLDT - Load Local Descriptor Table
LMSW - Load MACHine Status Word
LSL - Load Segment Limit
LTR - Load Task Register
MOV - Move to/from Special Register
SGDT - Store Global Descriptor Table
SIDT - Store Interrupt Descriptor Table
SLDT - Store Local Descriptor Table
SMSW - Store Machine Status Word
STR - Store Task Register
VERR - Verify Segment for Reading
VERW - Verify Segment for Writing

All NS486 instructions operate on zero to three operands. Two operand instructions permit the specification of an explicit source and destination as part of the instruction. Two operand instructions can be organized into eight groups as defined by their operand types:

Register to Register
Register to Memory
Memory to Register
Memory to Memory
Register to I/O
I/O to Register
Immediate Data to Register
Immediate Data to Memory.

An operand can be held in the instruction itself (immediate), in a register, in an I/O port or in memory. An immediate operand is prefetched as part of the instruction opcode. Operand lengths of 8, 16 and 32 bits are supported.

5.7 I/O Addressing

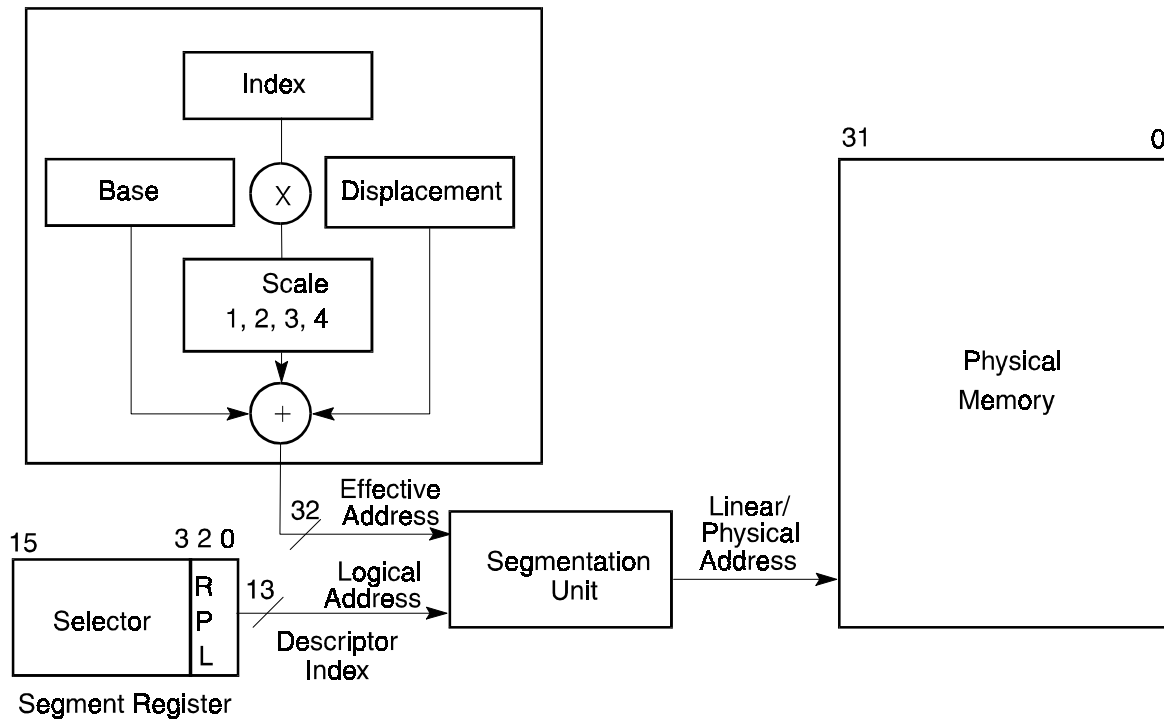
The NS486 processor accesses I/O devices as 8-bit, 16-bit, or 32-bit ports in the I/O address space. The accessible I/O address space is 64 Kbytes. All access to internal NS486 peripherals and external peripheral devices are made with IN and OUT instructions.

5.8 Memory Addressing

NS486 physical memory addressing is computed in two parts. An Offset Mechanism produces the offset or effective address, and the Selector Mechanism produces the base address. The offset and base address

are added together to produce the linear address. Since paging is not used, the linear address is the same as the physical address.

Figure 5-15 Memory Addressing



6.0 Test and Development Support

6.1 Modes

There is one pin allocated for testing and six test mode select pins. If the **TEST** signal is asserted, the test mode pins are sampled at power on time to determine which mode to use.

6.1.1 Chip Test Modes

There are two types of test modes available in the chip. The first is production test modes, used by National Semiconductor for production testing of the chip. These modes allow for high speed, low cost production testing with a high fault test coverage. The **TEST** signal should not be asserted by the end application for the production test modes.

The second type is board manufacturing test modes. These modes are designed to be used by a manufacturer in testing the assembled board, primarily checking for chip-board contact problems.

6.1.2 Board Test Modes

6.1.2.1 AND All Inputs (Testmode 58)

This mode performs a logical AND on all input (and I/O) pins. The result of the logical AND is placed out on the **SYSCLK** pin. This allows a board manufacturer to drive all inputs high, and verify no open connections. The manufacturer can then sequentially drive a logic low to each pin, and watching the output can verify that no input pins are shorted together.

6.1.2.2 Drive All Outputs High (Testmode 59)

This mode drives all output signals to a logic 1.

6.1.2.3 Drive All Outputs Low (Testmode 60)

This mode drives all output signals to a logic 0.

6.1.2.4 Tri-State All Outputs (Testmode 61)

This mode tri-states all output signals.

6.1.2.5 Toggle All Outputs (Testmode 63)

This mode places opposite logic states on adjacent output (and I/O) pins. This allows a board manufacturer to check adjacent pins for shorts and/or opens. The output pin states can be toggled to their opposite state by driving the input pin **TEST** high and low.

6.1.3 Setting up for Test Mode

Test mode is selected by programming the desired values on the IRQ 0-5 pins. If the **TEST** pin is held low when power is applied to the NS486SXF, the part will enter the test mode selected by the IRQ pins. Test mode can only be exited by removing power.

The table below shows the required state of the IRQ pins for the desired test mode.

Test Mode	IRQ 5	IRQ 4	IRQ 3	IRQ 2	IRQ 1	IRQ 0
58	1	1	1	0	1	0
59	1	1	1	0	1	1
60	1	1	1	1	0	0
61	1	1	1	1	0	1
63	1	1	1	1	1	1

Table 1: Pin Functions in Test Mode

Pin #	Pin Name	Test Mode 58	Test Mode 59	Test Mode 60	Test Mode 61	Test Mode 63
1	DP1	X	X	X	Z	X
2	SD15	A	1	0	Z	A
3	SD14	A	1	0	Z	\bar{A}
4	V _{SS}					
5	SD13	A	1	0	Z	A
6	SD12	A	1	0	Z	\bar{A}
7	V _{DD}					
8	SD11	A	1	0	Z	A
9	SD10	A	1	0	Z	\bar{A}
10	SD9	A	1	0	Z	A
11	SD8	A	1	0	Z	\bar{A}
12	DP0	X	X	X	Z	X
13	SD7	A	1	0	Z	A
14	V _{SS}					
15	SD6	A	1	0	Z	\bar{A}
16	SD5	A	1	0	Z	A
Pin #	Pin Name	Test Mode 58	Test Mode 59	Test Mode 60	Test Mode 61	Test Mode 63
17	V _{DD}					
18	SD4	A	1	0	Z	\bar{A}
19	SD3	A	1	0	Z	A
20	SD2	A	1	0	Z	\bar{A}
21	SD1	A	1	0	Z	A
22	SD0	A	1	0	Z	\bar{A}
23	WE	X	X	X	Z	X
24	V _{SS}					
25	CASH1	X	X	X	Z	X
26	CASH0	X	X	X	Z	X
27	V _{DD}					
28	CASL1	X	X	X	Z	X
29	CASL0	X	X	X	Z	X
30	RAST	X	X	X	Z	X
31	RAS0	X	X	X	Z	X
32	DRQ3	A	L	L	L	L
Pin #	Pin Name	Test Mode 58	Test Mode 59	Test Mode 60	Test Mode 61	Test Mode 63
34	DRQ4	A	L	L	L	L
35	DACK4	X	1	0	Z	\bar{A}
36	DRQ2	A	L	L	L	L
37	DACK2	X	1	0	Z	A
38	DRQ0	A	L	L	L	L
39	DACK0	X	1	0	Z	\bar{A}
40	TC/EOP	A	1	0	Z	A
41	SO/DCD	A	1	0	Z	\bar{A}
42	SI/CTS	A	1	0	Z	A
43	SCLK/RT	A	1	0	Z	\bar{A}
44	V _{SS}					
45	OSCX1	C	C	C	C	C
46	OSCX2	X	X	X	X	X
47	V _{DD}					
48	CLF	A	1	0	Z	A
49	CL1	A	1	0	Z	\bar{A}

Table 1: Pin Functions in Test Mode

51	LCD3		Pin #	68	CD_RST		Pin Name		Test Mode 58	A	Test Mode 59	A	Test Mode 60	0	Test Mode 61	Z	Test Mode 63	A
52	LCD2		Pin #	69	V _{CC_SEL}		Pin Name		Test Mode 58	A	Test Mode 59	A	Test Mode 60	0	Test Mode 61	Z	Test Mode 63	A
53	LCD1		Pin #	70	V _{PP_SEL1}		Pin Name		Test Mode 58	A	Test Mode 59	A	Test Mode 60	0	Test Mode 61	Z	Test Mode 63	A
54	LCD0		Pin #	71	V _{PP_SEL2}		Pin Name		Test Mode 58	A	Test Mode 59	A	Test Mode 60	0	Test Mode 61	Z	Test Mode 63	A
55	T0		Pin #	72	GPI		Pin Name		Test Mode 58	A	Test Mode 59	A	Test Mode 60	0	Test Mode 61	Z	Test Mode 63	A
56	T1		Pin #	73	BVD1/STSCNG		Pin Name		Test Mode 58	A	Test Mode 59	L	Test Mode 60	L	Test Mode 61	L	Test Mode 63	A
57	TX		Pin #	74	BVD2/SPKR		Pin Name		Test Mode 58	X	Test Mode 59	L	Test Mode 60	L	Test Mode 61	L	Test Mode 63	A
58	RX		Pin #	75	CDT		Pin Name		Test Mode 58	A	Test Mode 59	H	Test Mode 60	H	Test Mode 61	H	Test Mode 63	A
59	UCLK		Pin #	76	CD2		Pin Name		Test Mode 58	A	Test Mode 59	H	Test Mode 60	H	Test Mode 61	H	Test Mode 63	A
60	PWGOOD		Pin #	77	DIR		Pin Name		Test Mode 58	A	Test Mode 59	A	Test Mode 60	A	Test Mode 61	A	Test Mode 63	A
61	V _{SS}		Pin #	78	ENABLE		Pin Name		Test Mode 58		Test Mode 59	A	Test Mode 60	0	Test Mode 61	Z	Test Mode 63	A
62	RTCX1		Pin #	79	REG		Pin Name		Test Mode 58	H	Test Mode 59	H	Test Mode 60	H	Test Mode 61	H	Test Mode 63	A
63	V _{DD}		Pin #	80	RESERVED		Pin Name		Test Mode 58		Test Mode 59		Test Mode 60		Test Mode 61		Test Mode 63	
64	RTCX2		Pin #	81	PD7		Pin Name		Test Mode 58	X	Test Mode 59	X	Test Mode 60	X	Test Mode 61	X	Test Mode 63	
65	V _{BAT}		Pin #	82	PD6		Pin Name		Test Mode 58		Test Mode 59		Test Mode 60		Test Mode 61		Test Mode 63	
66	TEST		Pin #	83	PD5		Pin Name		Test Mode 58	A	Test Mode 59	A	Test Mode 60	A	Test Mode 61	A	Test Mode 63	A
67	IOIS16/WP		Pin #	84	V _{SS}		Pin Name		Test Mode 58	A	Test Mode 59		Test Mode 60		Test Mode 61		Test Mode 63	
			Pin #	85	PD4		Pin Name		Test Mode 58	A	Test Mode 59	A	Test Mode 60	0	Test Mode 61	Z	Test Mode 63	A
			Pin #	86	PD3		Pin Name		Test Mode 58	A	Test Mode 59	A	Test Mode 60	0	Test Mode 61	Z	Test Mode 63	A
			Pin #	87	V _{DD}		Pin Name		Test Mode 58		Test Mode 59		Test Mode 60		Test Mode 61		Test Mode 63	
			Pin #	88	PD2		Pin Name		Test Mode 58	A	Test Mode 59	A	Test Mode 60	0	Test Mode 61	Z	Test Mode 63	A
			Pin #	89	PD1		Pin Name		Test Mode 58	A	Test Mode 59	A	Test Mode 60	0	Test Mode 61	Z	Test Mode 63	A
			Pin #	90	PD0		Pin Name		Test Mode 58	A	Test Mode 59	A	Test Mode 60	0	Test Mode 61	Z	Test Mode 63	A
			Pin #	91	SLIN		Pin Name		Test Mode 58	A	Test Mode 59	A	Test Mode 60	0	Test Mode 61	Z	Test Mode 63	A
			Pin #	92	STB		Pin Name		Test Mode 58	A	Test Mode 59	A	Test Mode 60	0	Test Mode 61	Z	Test Mode 63	A
			Pin #	93	AFD		Pin Name		Test Mode 58	A	Test Mode 59	A	Test Mode 60	0	Test Mode 61	Z	Test Mode 63	A
			Pin #	94	INIT		Pin Name		Test Mode 58	A	Test Mode 59	A	Test Mode 60	0	Test Mode 61	Z	Test Mode 63	A
			Pin #	95	ACK		Pin Name		Test Mode 58	A	Test Mode 59	A	Test Mode 60	0	Test Mode 61	Z	Test Mode 63	A
			Pin #	96	PE		Pin Name		Test Mode 58	A	Test Mode 59	A	Test Mode 60	0	Test Mode 61	Z	Test Mode 63	A
			Pin #	97	SLCT		Pin Name		Test Mode 58	A	Test Mode 59	A	Test Mode 60	0	Test Mode 61	Z	Test Mode 63	A
			Pin #	98	V _{SS}		Pin Name		Test Mode 58		Test Mode 59		Test Mode 60		Test Mode 61		Test Mode 63	
			Pin #	99	ERR		Pin Name		Test Mode 58	A	Test Mode 59	A	Test Mode 60	0	Test Mode 61	Z	Test Mode 63	A
			Pin #	100	BUSY		Pin Name		Test Mode 58	A	Test Mode 59	A	Test Mode 60	0	Test Mode 61	Z	Test Mode 63	A
			Pin #	101	V _{DD}		Pin Name		Test Mode 58		Test Mode 59		Test Mode 60		Test Mode 61		Test Mode 63	

Table 1: Pin Functions in Test Mode

Pin #	102	103	104	105	106	107	108	109	110	111	112	113	114	115	116	117	118
Pin Name	EREQ/CS6/RTS	EACK/CS7/DSR	DRV/CS8/DTR	NMI	INTA	IRQ5	IRQ4	IRQ3	IRQ2	IRQ1	IRQ0	CS5	CS4	CS3	CS2	CS1	CS0
Test Mode 58	X	A	X	A	X	A	A	A	A	A	A	A	A	A	A	A	A
Test Mode 59	1	1	1	L	1	A	A	A	A	A	A	1	1	1	1	1	1
Test Mode 60	0	0	0	L	0	A	A	A	A	A	A	0	0	0	0	0	0
Test Mode 61	Z	Z	Z	L	Z	A	A	A	A	A	A	Z	Z	Z	Z	Z	Z
Test Mode 63	A	A	A	L	A	A	A	A	A	A	A	A	A	A	A	A	A
Pin #	119	120	121	122	123	124	125	126	127	128	129	130	131	132	133	134	135
Pin Name	RESET	RESET	SYSCLK	CS16	RDY	TOR	TOW	MEMR	MEMW	V _{SS}	BHE	SA25	V _{DD}	SA24	SA23	SA22	SA21
Test Mode 58	A	A	A	A	A	A	A	A	A		A	A		A	A	A	A
Test Mode 59	1	1	1	1	H	1	1	1	1		1	1		1	1	1	1
Test Mode 60	0	0	0	0	H	0	0	0	0		0	0		0	0	0	0
Test Mode 61	Z	Z	Z	Z	H	Z	Z	Z	Z		Z	Z		Z	Z	Z	Z
Test Mode 63	A	A	A	A	H	A	A	A	A		A	A		A	A	A	A
Pin #	136	137	138	139	140	141	142	143	144	145	146	147	148	149	150	151	152
Pin Name	SA20	SA19	SA18	SA17	SA16	SA15	SA14	SA13	V _{SS}	SA12	SA11	V _{DD}	SA10	SA9	SA8	SA7	SA6
Test Mode 58	A	A	A	A	A	A	A	A		A	A		A	A	A	A	A
Test Mode 59	1	1	1	1	1	1	1	1		1	1		1	1	1	1	1
Test Mode 60	0	0	0	0	0	0	0	0		0	0		0	0	0	0	0
Test Mode 61	Z	Z	Z	Z	Z	Z	Z	Z		Z	Z		Z	Z	Z	Z	Z
Test Mode 63	A	A	A	A	Z	Z	Z	Z		Z	Z		Z	Z	Z	Z	Z

Symbols:

- * Two modes
- A Active (\bar{A} = opposite of A state)
- C Clock (\bar{C} = opposite of C state)
- H Tied High
- L Tied Low
- X Don't Care
- Z Tri-State
- 0 Output is Low
- 1 Output is High





-  Input
-  Power
-  I/O
-  Output

Table 1: Pin Functions in Test Mode

Test Mode 63								
Test Mode 61								
Test Mode 60								
Test Mode 59								
Test Mode 58								
Pin Name								
Pin #								
Test Mode 63								
Test Mode 61								
Test Mode 60								
Test Mode 59								
Test Mode 58								
Pin Name								
Pin #								
Test Mode 63	\bar{A}		A	\bar{A}		A	\bar{A}	A
Test Mode 61	Z		Z	Z		Z	Z	Z
Test Mode 60	0		0	0		0	0	0
Test Mode 59	1		1	1		1	1	1
Test Mode 58	A		A	A		A	A	A
Pin Name	SA5	V _{SS}	SA4	SA3	V _{DD}	SA2	SA1	SA0
Pin #	153	154	155	156	157	158	159	160

6.2 ICE Considerations

If it is intended that an ICE-Probe be used in development with the NS486SXF, there are a few considerations for the board designer.

Conservative design practice will produce more reliable ICE operation. The following recommendations are not absolute requirements, only more conservative approaches: The designer might consider using external Oscillator modules rather than a crystal for the main clock. The PWGOOD signal to the NS486SXF could be de-bounced for more deterministic reset. The direct loading of the NS486SXF should be minimized, perhaps by limiting the number of DRAMs, avoiding SIMMs or using only one bank on the development board itself. Also, design such that only the DRAM and one set of buffers load directly connect to the NS486SXF.

To simplify logic, the ICE address space is mapped into the normal NS486SXF address space. The address range from 0x80000000 through 0x83FFFFFF should not be used by the target design to allow for the ICE mapping.

The SA[0] and the $\overline{\text{TEST}}$ lines will be asserted low by the ICE to put the installed NS486SXF on the board into TRISTATE mode, so take care to pullup both the lines properly in your design (a 10k ohm resistor is recommended).

The designer should plan for the orientation of the NS486SXF chip such that clearance is provided for the ICE cables. If you look at the NS486SXF from the top, with pin 1 at the upper left corner, the ICE cables will go straight down.

The board should provide clearance for a test clip at a minimum (an example test clip drawing is shown below). If board space is not a problem, the designer might consider putting a ring of four 40-pin headers around the NS486SXF so that the ICE pod can connect directly to the board without a test-clip.

The drawing for the ITT Pomona Electronics 160-pin QFP test clip is shown below:

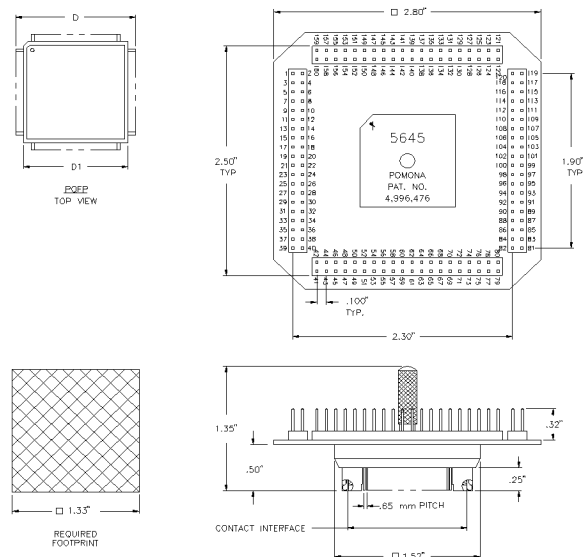


Figure 6-1 160-pin Test Clip (Pomona 5645)

7.0 Device Specifications

7.1 DC Electrical Specifications 5V ± 5%

7.1.1 Recommended Operating Conditions

Characteristic	Condition	Symbol	Min	Typ	Max	Unit
Supply Voltage		V_{DD}	4.75	5.0	5.25	V
Operating Temperature		T_A	0		+70	°C
ESD Tolerance	$C_{ZAP} = 100$ pF $R_{ZAP} = 1.5$ k Ω (Note 1)		2000			V

7.1.2 Absolute Maximum Ratings (Notes 2 and 3)

Characteristic	Condition	Symbol	Min	Max	Unit
Supply Voltage		V_{DD}, V_{DDA}	-0.5	7.0	V
Input Voltage		V_I	-0.5	$V_{DD} + 0.5$	V
Output Voltage		V_O	-0.5	$V_{DD} + 0.5$	V
Storage Temperature		T_{STG}	-65	+165	°C
Lead Temperature Soldering (10 sec.)		T_L		+260	°C

7.1.3 Capacitance: $T_A = 25^\circ\text{C}$, $f = 1$ MHz

Characteristics	Symbol	Min	Typ	Max	Unit
Input Pin Capacitance	C_{IN}		5	7	pF
Clock Input Capacitance	C_{IN1}		8	10	pF
I/O Pin Capacitance	C_{IO}		10	12	pF
Output Pin Capacitance	C_O		6	8	pF

7.0 Device Specifications

7.1.4 DC Characteristics (Under Recommended Operating Conditions)

Parameter	Symbol	Condition	Min	Typ	Max	Unit
Input High Voltage	V_{IH}		2.0		V_{DD}	V
Input Low Voltage	V_{IL}		-0.5		0.8	V
V_{DD} Average Supply Current	I_{CC}	$V_{IL} = 0.5V$ $V_{IH} = 2.4V$ No Load		--	--	mA

Note 1: Value based on test complying with NSC SOP5-028 human body model ESD testing using the ETS-910 tester.

Note 2: Absolute Maximum Ratings are those values beyond which damage to the device may occur.

Note 3: Unless otherwise specified all voltages are referenced to ground.

7.1.4.1 External Bus

Parameter	Symbol	Condition	Min	Max	Unit	Notes
Output High Voltage	V_{OH}	$I_{OH} = -6mA$ (Nch Quiet-drive) or $I_{OH} = -24mA$ (High-drive) on: SA12-1, DP1-0, SD15-0 $I_{OH} = -12mA$ on: SA0, SA25-13 [SA0 - min. 10Kohm pullup]	2.4		V	Max load on SA12-1 is 100pF, and SD0-15 is 50pF
Output Low Voltage	V_{OL}	$I_{OL} = 20mA$ on: SA12-1, DP1-0, SD15-0 $I_{OL} = 12mA$ on: SA0, SA25-13, BFE		0.4	V	

7.1.4.2 DMA Control Unit

Parameter	Symbol	Condition	Min	Max	Unit	Notes
Output High Voltage	V_{OH}	$I_{OH} = -6mA$ on: TC/EOP $I_{OH} = -4mA$ on: $\overline{DACK4}$, $\overline{DACK3}$, $\overline{DACK2}$, $\overline{DACK0}$	2.4		V	
Output Low Voltage	V_{OL}	$I_{OL} = 6mA$ on: TC/EOP $I_{OL} = 4mA$ on: $\overline{DACK4}$, $\overline{DACK3}$, $\overline{DACK2}$, $\overline{DACK0}$		0.4	V	

7.1.4.3 DRAM Control Unit

Parameter	Symbol	Condition	Min	Max	Unit	Notes
Output High Voltage	V_{OH}	$I_{OH} = -6mA$ (Nch Quiet-drive) or $I_{OH} = -24mA$ (High-drive) on: RAS0-1, CASH0-1, CASL0-1, WE	2.4		V	Max load on RAS1-0, CASH1-0, & CASL1-0 is 63pF.
Output Low Voltage	V_{OL}	$I_{OL} = 20mA$ on: RAS1-0, CASH1-0, CASL1-0, WE		0.4	V	Max load on WE is 100pF.

7.1.4.4 Auxiliary Processor Interface

Parameter	Symbol	Condition	Min	Max	Unit	Notes
Output High Voltage	V_{OH}	$I_{OH} = -6mA$ on: EACK $I_{OH} = -4mA$ on: DRV, EREQ	2.4		V	
Output Low Voltage	V_{OL}	$I_{OL} = 6mA$ on: EACK $I_{OL} = 4mA$ on: DRV, EREQ		0.4	V	

7.1.4.5 HP-SIR/UART

Parameter	Symbol	Condition	Min	Max	Unit	Notes
Output High Voltage	V_{OH}	$I_{OH} = -100\mu A$ $I_{OH} = -6mA$ on: Tx, UCLK, Rx	$V_{CC} - 0.2$ 2.4		V V	
Output Low Voltage	V_{OL}	$I_{OL} = 100\mu A$ $I_{OL} = 6mA$ on: Tx, UCLK, Rx		0.2 0.4	V V	

7.1.4.6 External Bus Control

Parameter	Symbol	Condition	Min	Max	Unit	Notes
Output High Voltage	V_{OH}	$I_{OH} = -12mA$ on: IOR, IOW, MEMR, MEMW RESET, RESET, CS16, BHE [CS16 - min. 10Kohm pullup]	2.4		V	
Output Low Voltage	V_{OL}	$I_{OL} = 12mA$ on: IOR, IOW, MEMR, MEMW RESET, RESET, CS16, BHE		0.4	V	

7.1.4.7 Oscillator (CPUX1/CLK)

Parameter	Symbol	Condition	Min	Max	Unit	Notes
Output High Voltage	V_{OH}	$I_{OH} = -12mA$ on: SYSCLK	2.4		V	
Output Low Voltage	V_{OL}	$I_{OL} = 12mA$ on: SYSCLK		0.4	V	
OSCX1 Input High Voltage	V_{IH}		2.4			OSCX2 is the output
OSCX2 Input Low Voltage	V_{IL}			0.4	V	

7.1.4.8 LCD Interface

Parameter	Symbol	Condition	Min	Max	Unit	Notes
Output High Voltage	V_{OH}	$I_{OH} = -2.6mA$ on: LCD[3:0], CL1, CL2, CLF	$V_{CC} - 0.8$		V	CMOS Level
Output High Voltage	V_{OH}	$I_{OH} = -6mA$ on: LCD[3:0], CL1, CL2, CLF	2.4		V	Ttl Level
Output Low Voltage	V_{OL}	$I_{OL} = 6mA$ on: LCD[3:0], CL1, CL2, CLF		0.4	V	

7.0 Device Specifications

7.1.4.9 Real Time Clock (RTCX1/CLK)

Parameter	Symbol	Condition	Min	Max	Unit	Notes
RTCX1 Input High Voltage	V_{IH}		2.0			RTCX2 is the output
RTCX1 Input Low Voltage	V_{IL}			0.4	V	
Battery Voltage	V_{BAT}		2.4		V	Lithium Battery
Battery Current	I_{BAT}	$V_{BAT} = 3.0\text{ V}$		3	μA	

7.1.4.10 PCMCIA (RIO8-15)

Parameter	Symbol	Condition	Min	Max	Unit	Notes
Output High Voltage	V_{OH}	$I_{OH} = -12\text{mA}$ on: V_{CC_SEL} , V_{PP_SEL1} , V_{PP_SEL2} $I_{OH} = -6\text{mA}$ on: DIR, ENABLE, REG, CD_RST, BUSY_LED, GPI, RSV (RSV is a reserved usage pin)	2.4		V	Power switch 1 card at a 50pF load
Output Low Voltage	V_{OL}	$I_{OL} = 12\text{mA}$ on: V_{CC_SEL} , V_{PP_SEL} , V_{PP_SEL2} $I_{OL} = 6\text{mA}$ on: DIR, ENABLE, REG, CD_RST, BUSY_LED, GPI, RSV (RSV is a reserved usage pin)		0.4	V	

7.1.4.11 IEEE-1284 Port (ECP Mode) & (RIO16-31)

Parameter	Symbol	Condition	Min	Max	Unit	Notes
High-level output current (Note 4)	I_{CH}	$V_{OH} = 2.4\text{V}$ on: PD[7:0], SLIN, STB, AFD, PE, INIT, ACK, SLCT ERR, BUSY	14		mA	
Low-level output current	I_{CL}	$V_{OL} = 0.4\text{V}$ on: PD[7:0], SLIN, STB, AFD, PE, INIT, ACK, SLCT ERR, BUSY	14		mA	

Note 4: When ECP mode 0, or ECP mode 2 and bit 1 of PCR is 0 for the parallel port, are selected, pins AFD, INIT, SLIN, and STB are open drain supports. 4.7 K Ω resistors should be used. The ECP I/Os have over-voltage protection against being backdriven by higher external voltages when the I/Os are TRISTATED. The I/Os also isolate the NS486SXF power-rail from external voltages when the chip is powered down. The maximum power-down leakage is 1mA to ground.

7.1.4.12 Timer

Parameter	Symbol	Condition	Min	Max	Unit	Notes
Output High Voltage	V_{OH}	$I_{OH} = -6\text{mA}$ on: T0, T1	2.4		V	
Output Low Voltage	V_{OL}	$I_{OL} = 6\text{mA}$ on: T0, T1		0.4	V	

7.1.4.13 General Purpose Chip Selects

Parameter	Symbol	Condition	Min	Max	Unit	Notes
Output High Voltage	V_{OH}	$I_{OH} = -6\text{mA}$ on: $\overline{CS5-0}$	2.4		V	
Output Low Voltage	V_{OL}	$I_{OL} = 6\text{mA}$ on: $\overline{CS5-0}$		0.4	V	

7.1.4.14 Interrupt Controller

Parameter	Symbol	Condition	Min	Max	Unit	Notes
Output High Voltage	V_{OH}	$I_{OH} = -12\text{mA}$ on: \overline{INTA}	2.4		V	
Output Low Voltage	V_{OL}	$I_{OL} = 12\text{mA}$ on: \overline{INTA}		0.4	V	

7.1.4.15 3-Wire I/O (& Access.bus)

Parameter	Symbol	Condition	Min	Max	Unit	Notes
Output High Voltage	V_{OH}	$I_{OH} = -12\text{mA}$ on: SO, SI, SCLK	2.4		V	
Output Low Voltage	V_{OL}	$I_{OL} = 12\text{mA}$ on: SO, SI, SCLK		0.4	V	

7.2 General AC Specifications

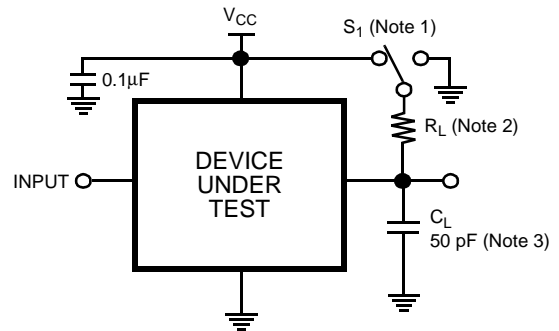
AC TEST CONDITIONS

Note 1: $S_1 = V_{CC}$ for t_{PZL} , and t_{PLZ} measurements
 $S_1 = GND$ for t_{PZH} , and t_{PHZ} measurements
 $S_1 = \text{Open}$ for push pull outputs

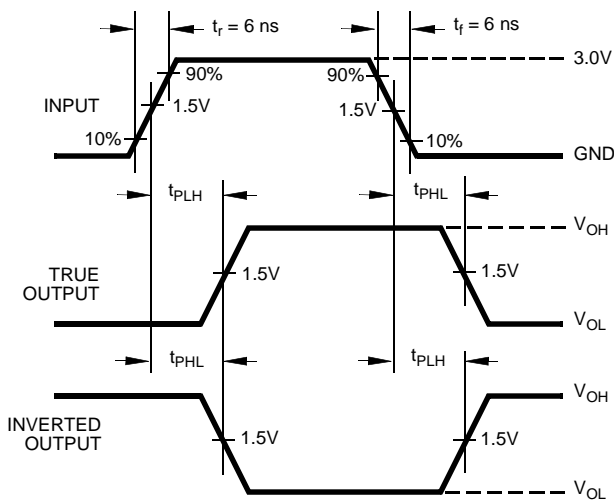
Note 2: $R_L = 1.1k$

Note 3: C_L includes scope and jig capacitance

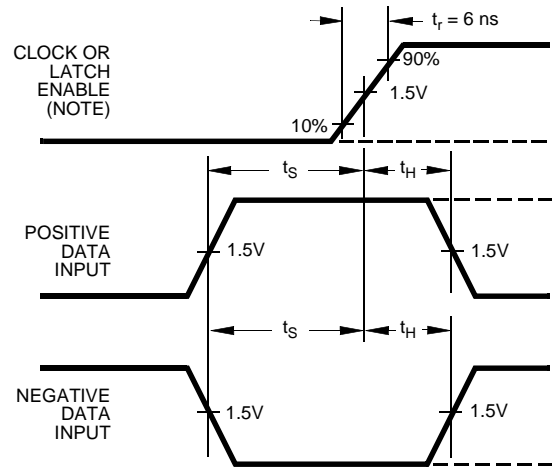
Test Circuit for Output Tests



Propagation Delay Waveforms

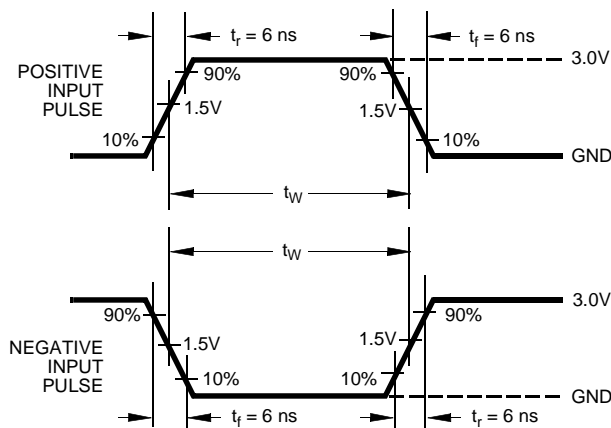


Setup and Hold Time Waveforms



Note: Waveform for negative edge sensitive circuits will be invert

Input Pulse Width Waveforms Except for Clock Pins



TRI-STATE Output Enable and Disable Waveforms

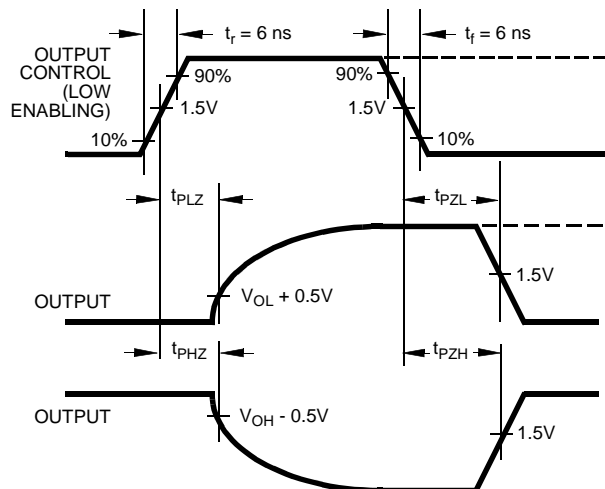


Figure 7-1 Switching Characteristic Measurement Waveforms

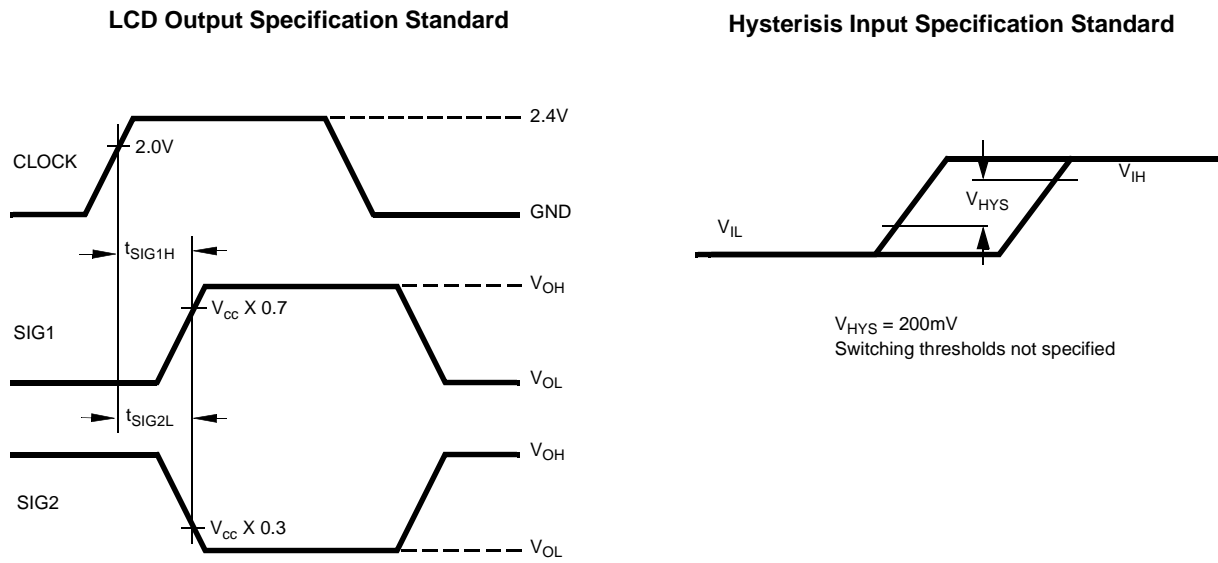


Figure 7-2 More Switching Specifications

7.2.1 Power Ramp Times

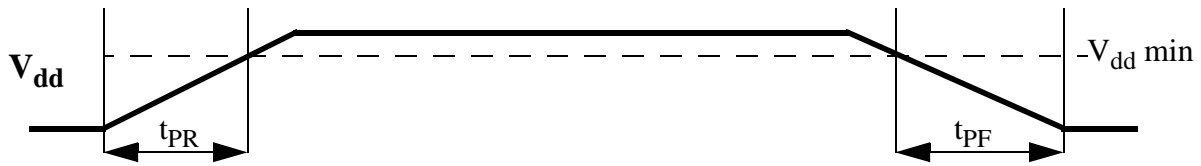


Figure 7-3 Power Supply Rise and Fall

Table 7-1: V_{dd} Rise and Fall Times

Symbol	Parameter	Min	Max	Unit
t _{PF}	V _{dd} falling time from 4.5V to 0 Volts	5		ms
t _{PR}	V _{dd} rising time from 0V to 4.5 Volts	5		ms

Note: The rising/falling rate is assumed linear.

7.2.2 PWRGOOD and Power Rampdown Timing

Table 7-2: V_{dd} Rampdown vs. PWRGOOD

Symbol	Parameter	Min	Max	Unit
t _{VPG}	V _{dd} (4.5V) to PWRGOOD high	1		μs
t _{PGV}	PWRGOOD falling to V _{dd} (4.5V)	1		μs

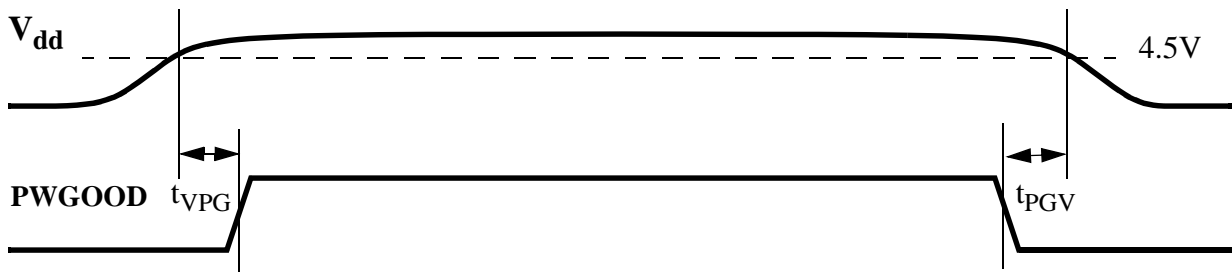


Figure 7-4 PWRGOOD in relation to V_{dd}

Note: The rising/falling rate is assumed linear.

7.3 AC Switching Specifications

The following pages list some of the preliminary AC Specifications for the NS486SXF. All parameters are listed in alphabetical order according to their Symbol.

The Tables consist of the following:

Parameter - A short description of the specification being documented.

Symbol - A quick reference between the timing diagram and the Table entries.

Formula - An equation, which in addition to the Minimum and Maximum Specifications can be used to determine the actual timing provided at any operating frequency.

Min. - Minimum Specification when added to the value produced by the formula.

Max. - Maximum Specification when added to the value produced by the formula.

How to calculate the actual specification at a given frequency:

In the formula column, one will see many formulae, which contain the variable T. The T represents one period (or one T-state) of the CPU Clock. So if the CPU is running at 25 MHz, T is equivalent to 40 nsec; similarly if the CPU is running at 20 MHz, T is equivalent to 50 nsec.

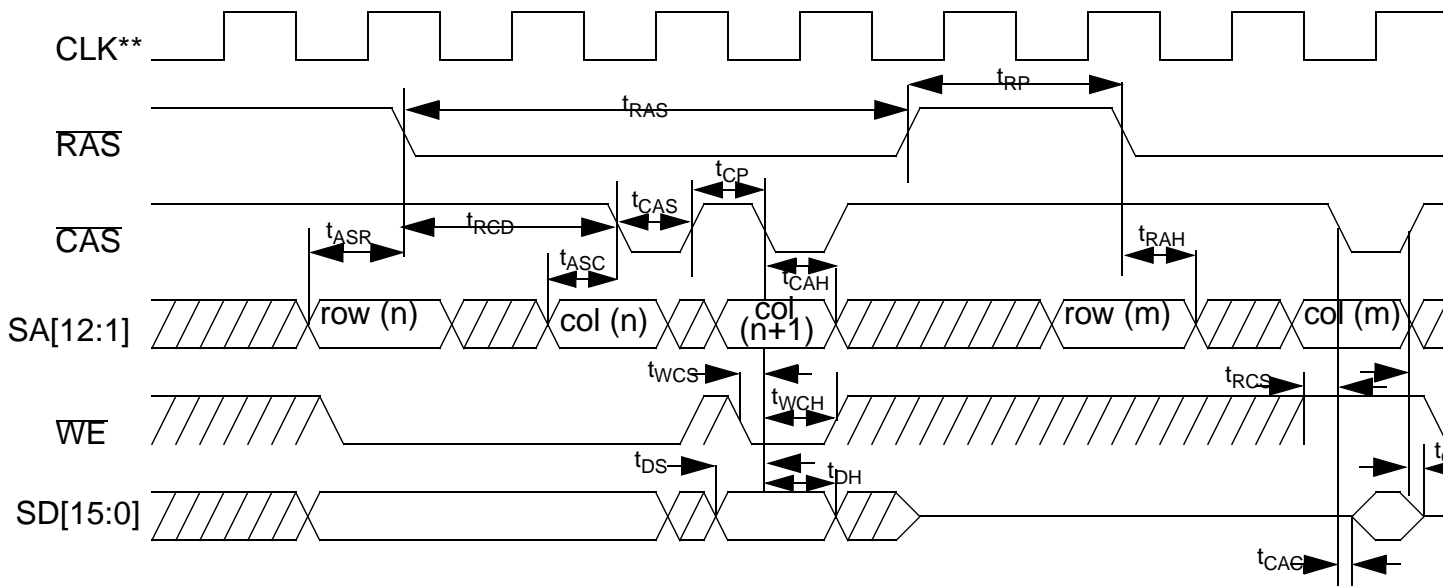
EXAMPLE: Calculate the minimum guaranteed Column Address Setup Time

$$\begin{array}{rcl}
 \text{At 25 MHz: Formula + Min. Spec.} & = & \\
 (0.5T) + (-20 \text{ nsec}) & = & \\
 0.5(40 \text{ nsec}) + (-20 \text{ nsec}) & = & \\
 20 \text{ nsec} - 20 \text{ nsec} & = & 0 \text{ nsec}
 \end{array}$$

$$\begin{array}{rcl}
 \text{At 20 MHz: Formula + Min. Spec.} & = & \\
 (0.5T) + (-20 \text{ nsec}) & = & \\
 0.5(50 \text{ nsec}) + (-20 \text{ nsec}) & = & \\
 25 \text{ nsec} - 20 \text{ nsec} & = & 5 \text{ nsec}
 \end{array}$$

As the frequency varies, so will many of the specifications. One should always calculate the specification based on the CPU's operating frequency.

7.3.1 DRAM Interface Timing Specification.



**The CLK signal is only included as a reference; no specifications are guarantee to this signal.

Figure 7-5 DRAM Timing Diagram

Table 7-3: 4 Cycle Page Miss Preliminary Specifications

Parameter	Symbol	Formula	Min	Max
Column Address Setup Time	t _{ASC}	0.5T +	-20	
Row Address Setup Time	t _{ASR}	0.5T +	-20	
Access Time From CAS	t _{CAC}	0.5T +		-5
Column Address Hold Time	t _{CAH}	0.5T +	-5	
CAS Pulse Width	t _{CAS}	0.5T +	0	10
Page Mode CAS Precharge	t _{CP}	0.5T +	-10	
Write Data Hold Time	t _{DH}	0.5T +	-5	
Write Data Setup Time	t _{DS}	0.5T +	-20	
Read Data Valid Hold Time	t _{OFF}		0	
RAS Pulse Width	t _{RAS}	2.5T +	-15	Progr'm'ble
Row Address Hold Time	t _{RAH}	0.5T +	-10	
RAS to CAS Delay Time	t _{RCD}	1.5T +	-20	

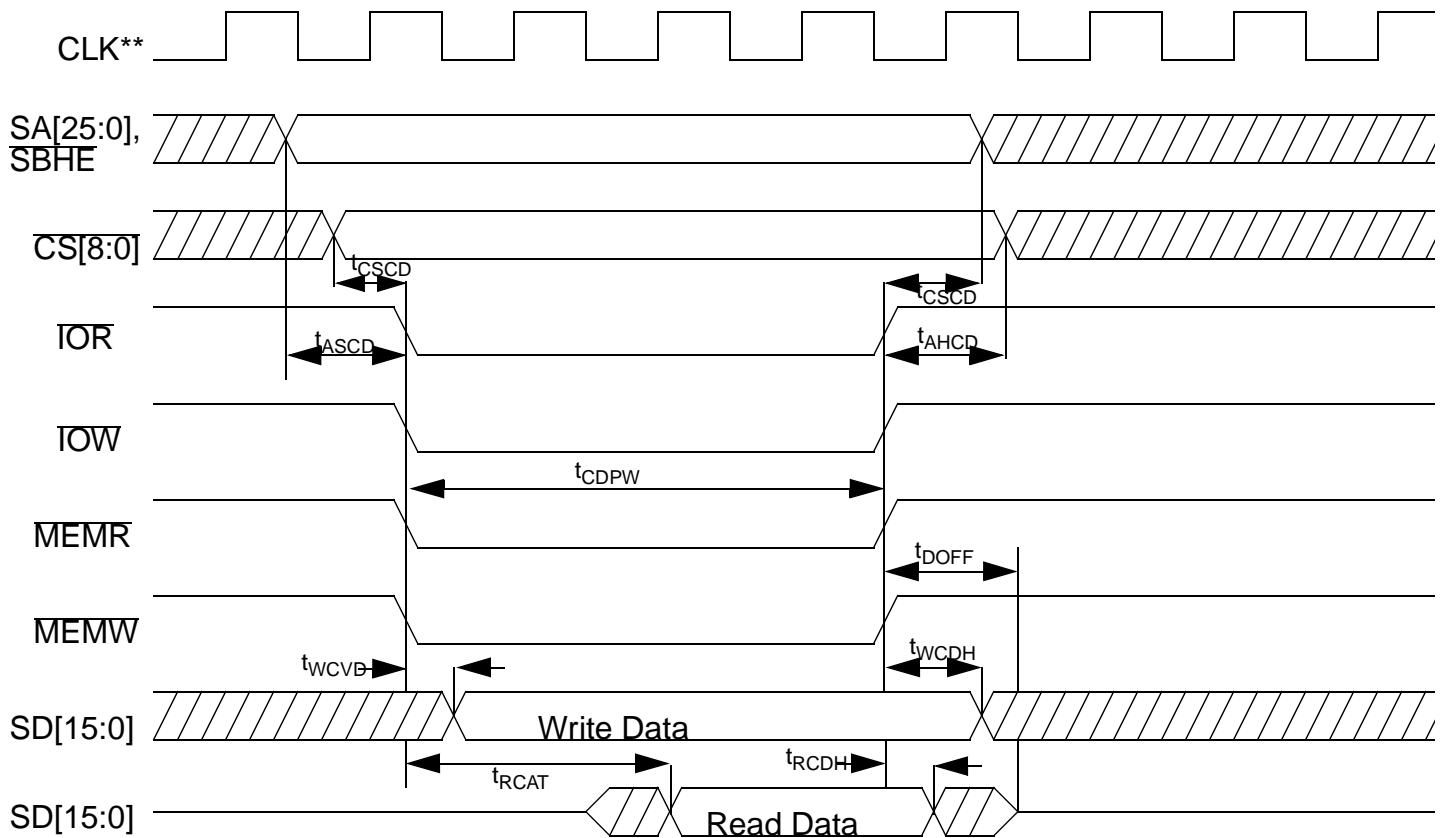
Table 7-3: 4 Cycle Page Miss Preliminary Specifications

Parameter	Symbol	Formula	Min	Max
Read Command Hold Time	t_{RCH}		0	
Read Command Setup Time	t_{RCS}	$0.5T +$	-20	
RAS Precharge Time	t_{RP}	$1.5T +$	-10	
Write Command Hold Time	t_{WCH}	$0.5T +$	-5	
Write Command Setup Time	t_{WCS}	$0.5T +$	-20	

Table 7-4: 3 Cycle Miss Preliminary Specifications

Parameter	Symbol	Formula	Min	Max
Column Address Setup Time	t_{ASC}	$0.5T +$	-20	
Row Address Setup Time	t_{ASR}	$0.5T +$	-20	
Access Time From \overline{CAS}	t_{CAC}	$0.5T +$		-5
Column Address Hold Time	t_{CAH}	$0.5T +$	-5	
\overline{CAS} Pulse Width	t_{CAS}	$0.5T +$	0	10
Page Mode \overline{CAS} Precharge	t_{CP}	$0.5T +$	-10	
Write Data Hold Time	t_{DH}	$0.5T +$	-5	
Write Data Setup Time	t_{DS}	$0.5T +$	-20	
Read Data Valid Hold Time	t_{OFF}		0	
RAS Pulse Width	t_{RAS}	$2.0T +$	-15	PROG
Row Address Hold Time	t_{RAH}	$0.5T +$	-10	
RAS to \overline{CAS} Delay Time	t_{RCD}	$1.0T +$	-20	
Read Command Hold Time	t_{RCH}		0	
Read Command Setup Time	t_{RCS}	$0.5T +$	-20	
RAS Precharge Time	t_{RP}	$1.0T +$	0	
Write Command Hold Time	t_{WCH}	$0.5T +$	-5	
Write Command Setup Time	t_{WCS}	$0.5T +$	-20	

7.3.2 ISA-like Bus Cycles Timing Specification



**The CLK signal is only included as a reference; no specifications are guarantee to this signal.

Figure 7-6 ISA-like Bus Timing Diagram

Table 7-5: No Command Delay ISA-like Bus Specifications

Parameter	Symbol	Formula	Min	Max
Address Hold Time from \overline{CMD}	t_{AHCD}	$1.0T +$	-20	
Address Setup Time to \overline{CMD}	t_{ASCd}	$1.0T +$	-20	
Command Pulse Width	t_{CDPW}	$1.0T + (Wait)T +$	-10	
Chip Select Hold Time from \overline{CMD}	t_{CHCD}	$1.0T +$	-25	
Chip Select Setup Time to \overline{CMD}	t_{CSCd}	$1.0T +$	-40	
Read Data TRI-STATE	t_{DOFF}	$1.0T +$		-25
Read \overline{CMD} Data Access Time	t_{RCAT}	$1.0T + (Wait)T +$		-30
Read \overline{CMD} Data Hold Time	t_{RCDH}		0	

Table 7-5: No Command Delay ISA-like Bus Specifications

Parameter	Symbol	Formula	Min	Max
Write $\overline{\text{CMD}}$ Data Hold Time	t_{WCDH}	$1.0T +$	-25	
Write $\overline{\text{CMD}}$ to Valid Data	t_{WCVD}			5
Write Command Setup Time	t_{WCS}	$0.5T +$	-20	

NOTE: The value of (Wait) in the above formulae, is the number of programmed wait states associated with that access cycle (default value is 7, but may be programmed to 0-7).

Table 7-6: One Programmed Command Delay ISA-like Bus Specifications

Parameter	Symbol	Formula	Min	Max
Address Hold Time from $\overline{\text{CMD}}$	t_{AHCD}	$1.0T +$	-20	
Address Setup Time to $\overline{\text{CMD}}$	t_{ASCD}	$2.0T +$	-20	
Command Pulse Width	t_{CDPW}	$1.0T + (\text{Wait})T +$	-10	
Chip Select Hold Time from $\overline{\text{CMD}}$	t_{CHCD}	$1.0T +$	-25	
Chip Select Setup Time to $\overline{\text{CMD}}$	t_{CSCD}	$2.0T +$	-40	
Read Data TRI-STATE	t_{DOFF}	$1.0T +$		-25
Read $\overline{\text{CMD}}$ Data Access Time	t_{RCAT}	$1.0T + (\text{Wait})T +$		-30
Read $\overline{\text{CMD}}$ Data Hold Time	t_{RCDH}		0	
Write $\overline{\text{CMD}}$ Data Hold Time	t_{WCDH}	$1.0T +$	-25	
Write Valid Data to CMD ^{NOTE_2}	t_{WCVD}	$1.0T +$	-5	
Write Command Setup Time	t_{WCS}	$0.5T +$	-20	

NOTE: The value of (Wait) in the above formulae, is the number of programmed wait states associated with that access cycle (default value is 7, but may be programmed to 0-7).

NOTE_2: For this case Valid Write Data Sets-up to the leading edge of the Command Strobe.

7.3.3 Ready Feedback Timing Specifications

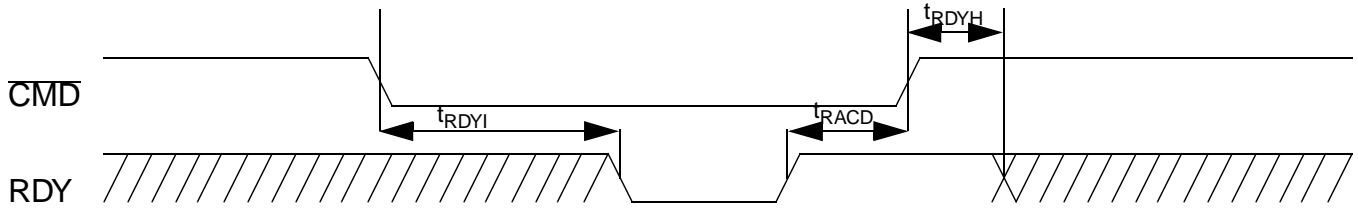


Figure 7-7 Ready Feedback Timing Diagram

Table 7-7: Ready Signal Timing Specifications

Parameter	Symbol	Formula	Min	Max
RDY Active to $\overline{\text{CMD}}$ rising	t_{RACD}	$(E_RDY)T +$	0	
RDY Hold Time from $\overline{\text{CMD}}$	t_{RDYH}		0	
$\overline{\text{CMD}}$ to RDY Inactive Feedback	t_{RDYI}	$1.0T + (\text{Wait})T +$		-30

NOTE: The value of (Wait) in the above formulae, is the number of programmed wait states associated with that access cycle (default value is 7, but may be programmed to 0-7). The value of (E_RDY) in the above formulae, is the number of programmed extended ready states associated with every access cycle (default number is 2, but may be programmed to 0-2)

7.3.4 OSCX1 AC Specification

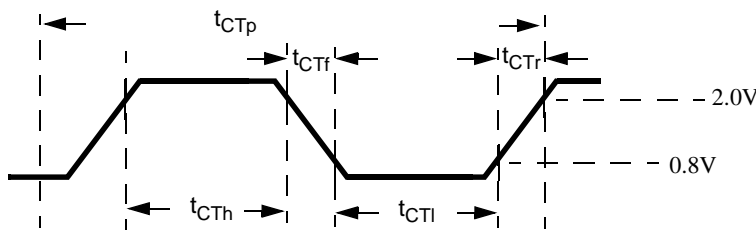


Figure 7-8 TTL Clock Input Timing Diagram

Table 7-8: TTL Clock Input Specification

Symbol	Description	Min	Max	Unit
t_{CTp}	CTTL Clock period	40	870	ns
t_{CTh}	CTTL high time (Note)	$(0.5 \times t_{\text{CTp}}) - 4$		ns
t_{CTI}	CTTL low time (Note)	$(0.5 \times t_{\text{CTp}}) - 4$		ns
t_{CTr}	CTTL rise time		4	ns
t_{CTf}	CTTL fall time		4	ns

Note: Except for the cycle in which the core frequency is changed. In this cycle, t_{CTh} and t_{CTI} relate to different t_{CTp} cycles.

7.3.5 Peripheral Timing Specifications

7.3.5.1 DMA Controller

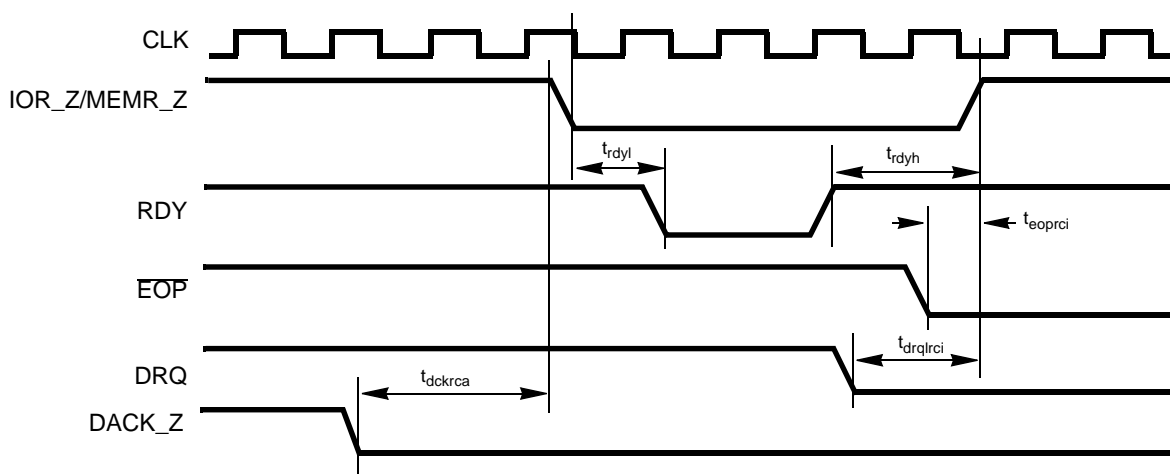


Figure 7-9 DMA Controller Read Timing Diagram

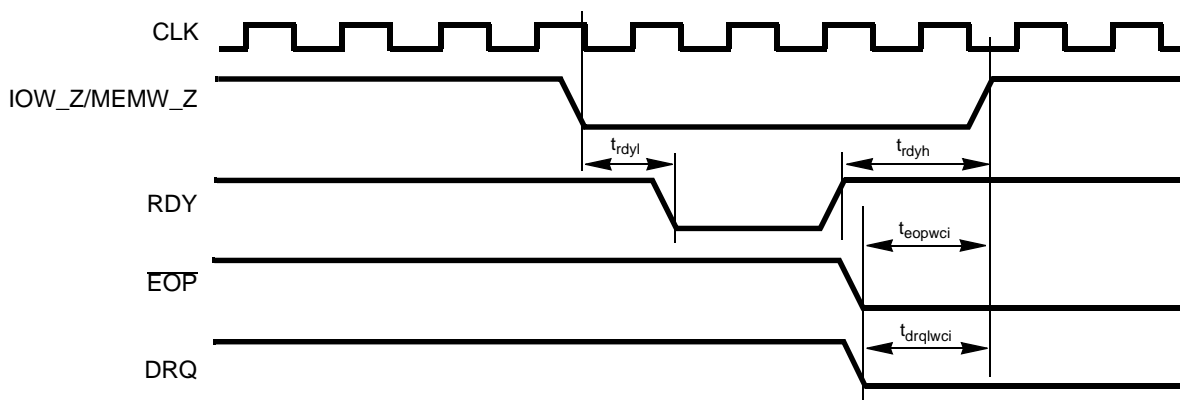


Figure 7-10 DMA Controller Write Timing Diagram

Table 7-9: DMA Controller Specifications

Symbol	Parameter	Min	Typ	Max
t_{rdyL}	RDY inactive low setup to CMD active	-15 ns		
t_{rdyH}	RDY active high setup to CMD inactive			$2T + 15$ ns
t_{eoprci}		45 ns		
t_{eopwci}		$2T + 5$ ns		
$t_{drqlwci}$		$2T + 36$ ns		
$t_{drqlrci}$		$T + 40$ ns		
t_{dckrca}		$2T + 2.2$ ns		

7.3.5.2 PIC AC Specs

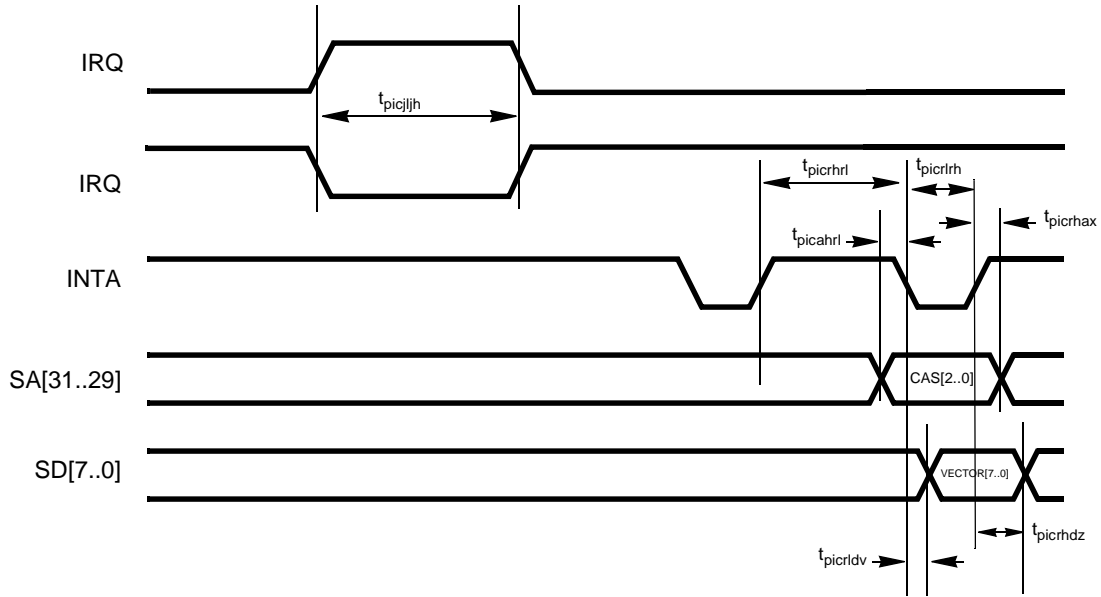


Figure 7-11 PIC Timing Diagram

Table 7-10: PIC Timing Specifications

Symbol	Parameter	Min	Typ	Max
t_{picjfh}		100		
$t_{picahrl}$		0		
$t_{picrlrh}$		235		
$t_{picrhax}$		0		
t_{picldv}				200
t_{picrdz}		10		
t_{picrhl}		100		

7.3.5.3 Parallel Port

Table 7-11: Parallel Port Compatibility Mode Handshake Timing Values

Symbol	Measured At	Measured From	Measured to	Value (min/max)	Compliance
T_{ready}	Host Output	Busy V_{IL}	\overline{Strobe} V_{OH}	0 min.	Compatible hosts
$T_{setup(host)}$	Host Output	Data stable	\overline{Strobe} V_{OH}	750 ns min.	Compatible hosts
$T_{setup(peripheral)}$	Peripheral input	Data stable	\overline{Strobe} V_{IH}	500 ns max.*	Compatible peripherals
$T_{strobe(host)}$	Host output	\overline{Strobe} V_{OL}	$\overline{Strobe} > V_{OL}$	750 ns min 500 μ s max	Compatible hosts
$T_{strobe(peripheral)}$	Peripheral input	\overline{Strobe} V_{IL}	$\overline{Strobe} > V_{IL}$	500 ns max	Compatible peripherals
$T_{hold(host)}$	Host output	\overline{Strobe} V_{OH}	Data or \overline{AutoFd} change	750 ns min	Compatible hosts
$T_{hold(peripheral)}$	Peripheral input	\overline{Strobe} V_{IL}	Data or \overline{AutoFd} change	500 ns max	Compatible peripherals
T_{busy}	Peripheral output	\overline{Strobe} V_{IL}	Busy V_{OH}	500 ns max	Compliant peripherals
T_{reply}	Peripheral output	\overline{Strobe} V_{IL}	\overline{Ack} V_{OH}	0 min	Compatible peripherals
T_{ack}	Peripheral output	\overline{Ack} V_{OL}	\overline{Ack} V_{OL}	500 ns min 10 μ s max	Compatible peripherals
T_{nbusy}	Peripheral output	\overline{Ack} V_{OH}	Busy V_{OH}	0 min**	Compliant peripherals
T_{next}	Host output	\overline{Ack} V_{IL}	\overline{Strobe} V_{OH}	0 min	Compliant hosts

Notes:

- For more information on the history of Centronics Standard Parallel and PC-Compatible Parallel Interfaces, see annex Ca and in particular C.6.2 for Busy-to-Ack timing variations.
 - V_{IL} is the low-level voltage input
 V_{OL} is the low-level voltage output
 V_{OH} is the high-level voltage output
 V_{IH} is the high-level voltage input
- * The maximum value stated for peripherals in this table are referenced to the peripheral. For example, the peripheral cannot require more than 500 ns data setup time.
- ** Recognize that complementary signal changes may have overlapping signal transitions. The zero minimum value cannot be guaranteed.

Table 7-12: Parallel Port IEEE 1284 Mode Handshake Timing Values

Symbol	Description	Min	Max
T_H	Host response time	0	1.0s
T_{∞}	Infinite response time	0	Infinite
T_L	Peripheral response time	0	35 ms
T_R	Peripheral response time (ECP Mode only)	0	—
T_S	Host recovery time (ECP Mode only)	35 ms	—
T_P	Minimum setup or pulse width	0.5 μ s	—
T_D	Minimum data setup time (ECP/EPP Modes only)	0	—
T_{ES}	Short response time (EPP Mode only)	0	125 ms
T_{EL}	Long response time (EPP Mode only)	0	10 μ s
T_{ER}	Termination pulse width (EPP Mode only)	50 μ s	Infinite

7.3.5.4 PCMCIA Controller

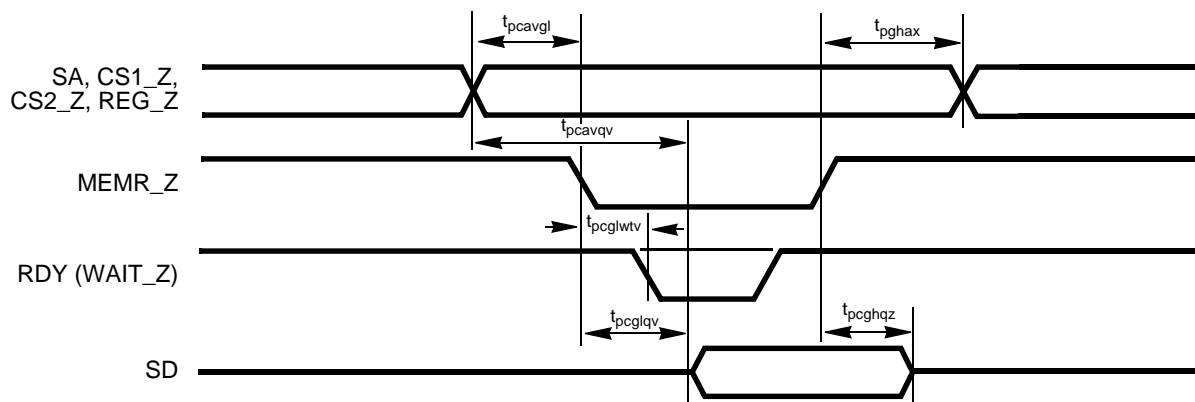


Figure 7-12 Memory Read Timing

Table 7-13: PCMCIA Memory Read Timing Specifications

Symbol	Parameter	Condition	Min	Typ	Max
t_{pcavgl}			50 ns		
t_{pghax}			20 ns		
$t_{pcglwtv}$					35 ns
t_{pcavqv}					250 ns
t_{pcghqz}			0 ns		
t_{pcglqv}					125 ns

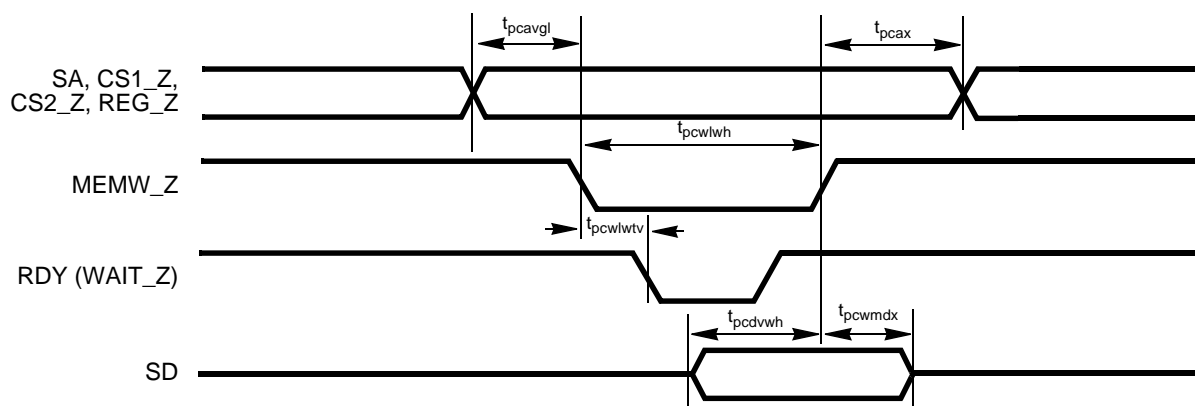


Figure 7-13 Memory Write Timing Diagram

Table 7-14: Memory Write Timing

Symbol	Parameter	Condition	Min	Typ	Max
t_{pcavgl}			50 ns		
t_{pcax}			20 ns		

Table 7-14: Memory Write Timing

Symbol	Parameter	Condition	Min	Typ	Max
t_{pcwlwh}			60 ns + (tsysclk) • (number of waitstates)		
t_{pcdvwh}			100 ns		
t_{pcwmdx}			30 ns		
t_{pcwltv}					35 ns

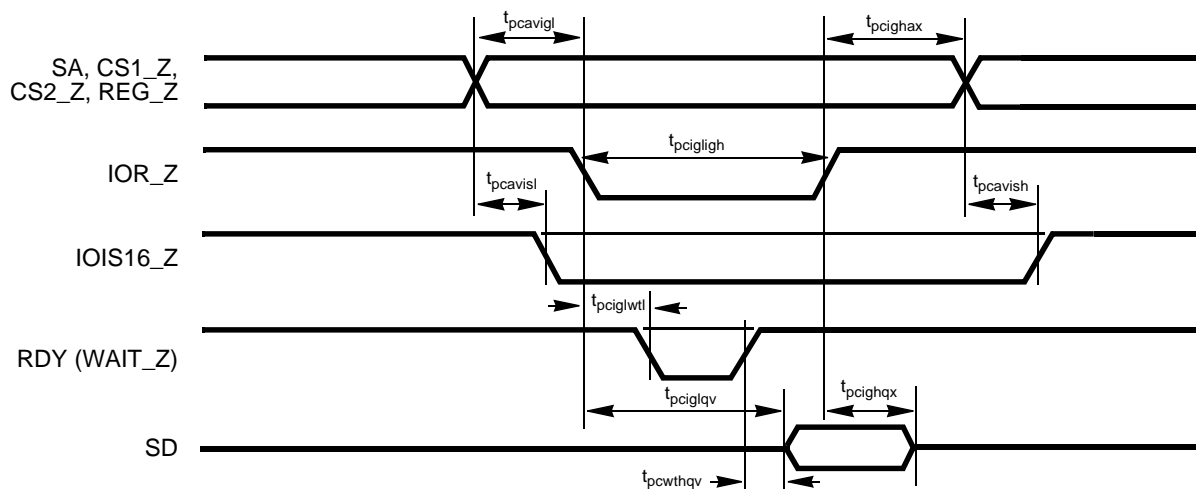


Figure 7-14 I/O Read Timing

Table 7-15: PCMCIA I/O Read Specifications

Symbol	Parameter	Condition	Min	Typ	Max
$t_{pcavigl}$			100 ns		
$t_{pcighax}$			20 ns		
$t_{pcigligh}$			180 ns		
$t_{pcavisl}$					35 ns
$t_{pcavish}$					35 ns
$t_{pciglwtl}$					35 ns
$t_{pciglqv}$					120 ns
$t_{pcighqx}$			0 ns		
$t_{pcwthqv}$					35 ns

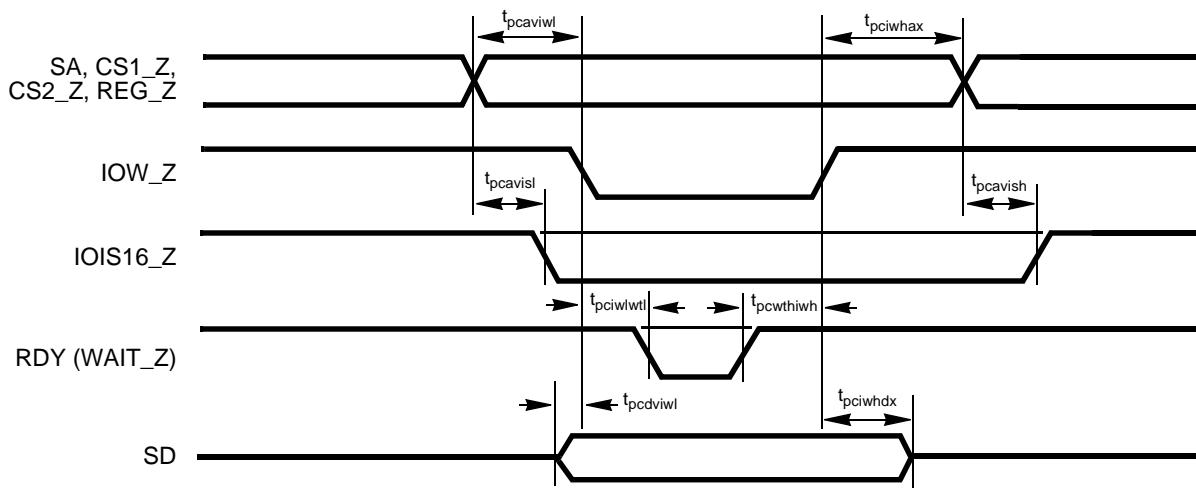


Figure 7-15 I/O Write Timing Diagram

Table 7-16: PCMCIA I/O Write Specifications

Symbol	Parameter	Condition	Min	Typ	Max
t_{pcvawl}			100 ns		
$t_{pciwhax}$			20 ns		
t_{pcvawl}					35 ns
t_{pcvish}					35 ns
t_{pcvish}					35 ns
$t_{pciwltl}$					35 ns
t_{pcdvwl}			80 ns		
$t_{pcwthwh}$			0 ns		
$t_{pciwhdx}$			30 ns		

7.3.5.5 MICROWIRE (3-Wire) & Access.bus

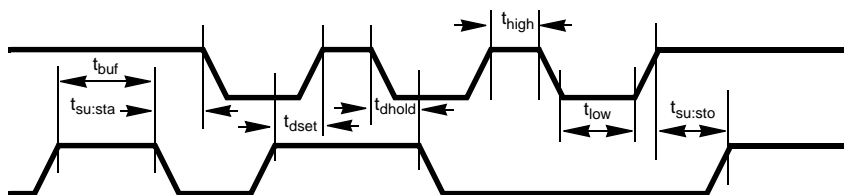


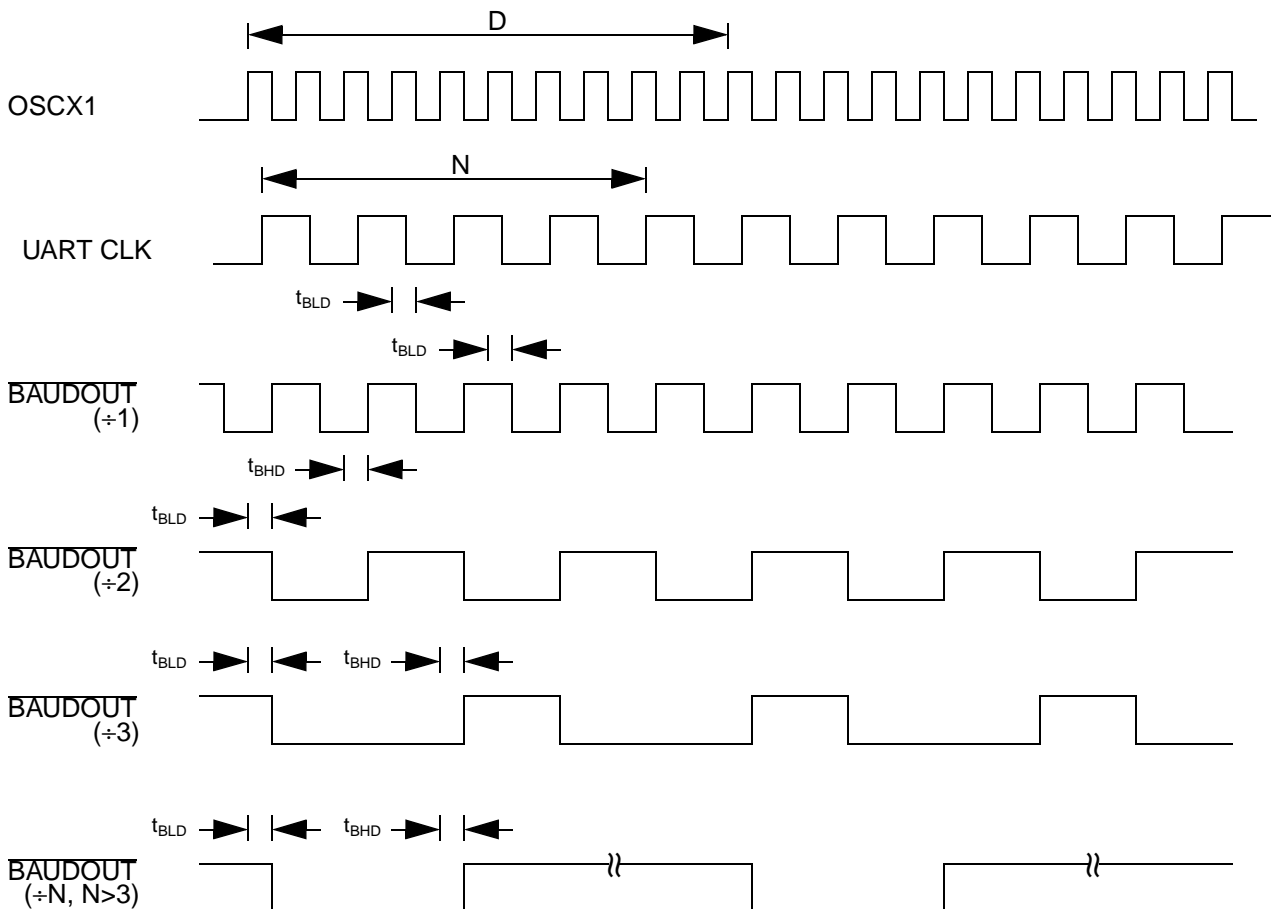
Figure 7-16 Access.bus Timing Diagram

Table 7-17: Access.Bus Timing Specifications

Symbol	Parameter	Formula	Min	Max
f_{sclk}	SCLK clock frequency			100 KHz
t_{buf}	Bus free time between STOP and START condition		4.7 us	
t_{low}	Low period of the SCLK clock		4.7 us	
t_{high}	High period of the SCLK clock		4.0 us	
t_{dhold}	Data hold time		250	
t_{dset}	Data setup time		250	
$t_{su:sto}$	Setup time for STOP condition		4.0 us	
$t_{su:sta}$	Hold time for START condition		4.7 us	

7.3.5.6 FIFO UART

Symbol	Parameter	Conditions	Min	Max	Units
D	Osc Clock Divider		1	63	Clks
N	Baud Divisor		1	65535	Clks
t_{BHD}	Baud Output Positive Edge Delay			56	ns
t_{BLD}	Baud Output Negative Edge Delay			56	ns



Symbol	Parameter	Conditions	Min	Max	Units
t_{IRTXW}	IRTX Pulse Width		1.6 μ s	3/16	BAUD OUT Cycles
t_{IRRXW}	IRRX Pulse Width		1.6 μ s	6/16	BAUD OUT Cycles

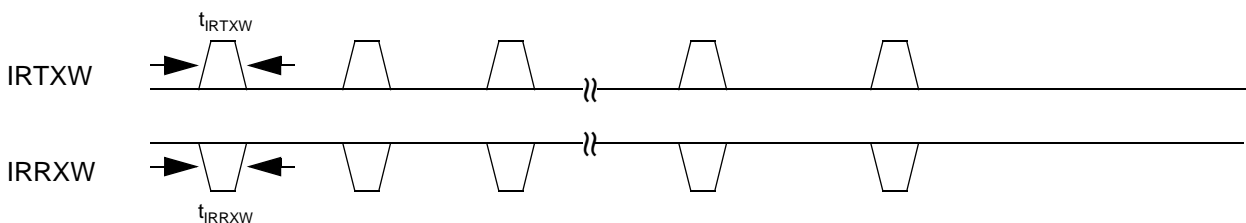


Figure 7-17 UART Baud Rate and Infrared Clocks

Symbol	Parameter	Min	Max	Units
t_{SINT}	Delay from Stop Bit to Set Interrupt		2	BAUDOUT Cycles
t_{STI}	Delay from Start Bit to IRQ		8	BAUDOUT Cycles
t_{SI}	Delay from Initial Write to IRQ	16	24	BAUDOUT Cycles
t_{IRS}	Delay from IRQ Reset to Tx Start	8	24	BAUDOUT Cycles
t_{MDO}	Delay from Write to Output		40	ns
t_{RIM}	Delay to Reset IRQA from Read		78	ns
t_{SIM}	Delay to Set IRQ from Modem Input		40	ns

$$\text{BAUDOUT Cycle} = \frac{\text{Input Clock Frequency}}{16 \times \text{Baudrate Divisor}}$$

$$\text{Input Clock Frequency} = \frac{\text{OSCX1 Frequency}}{\text{UART Clock Divisor}}$$

Registers: Divisor Latch holds Baudrate Divisor
EF70 holds UART Clock Divisor

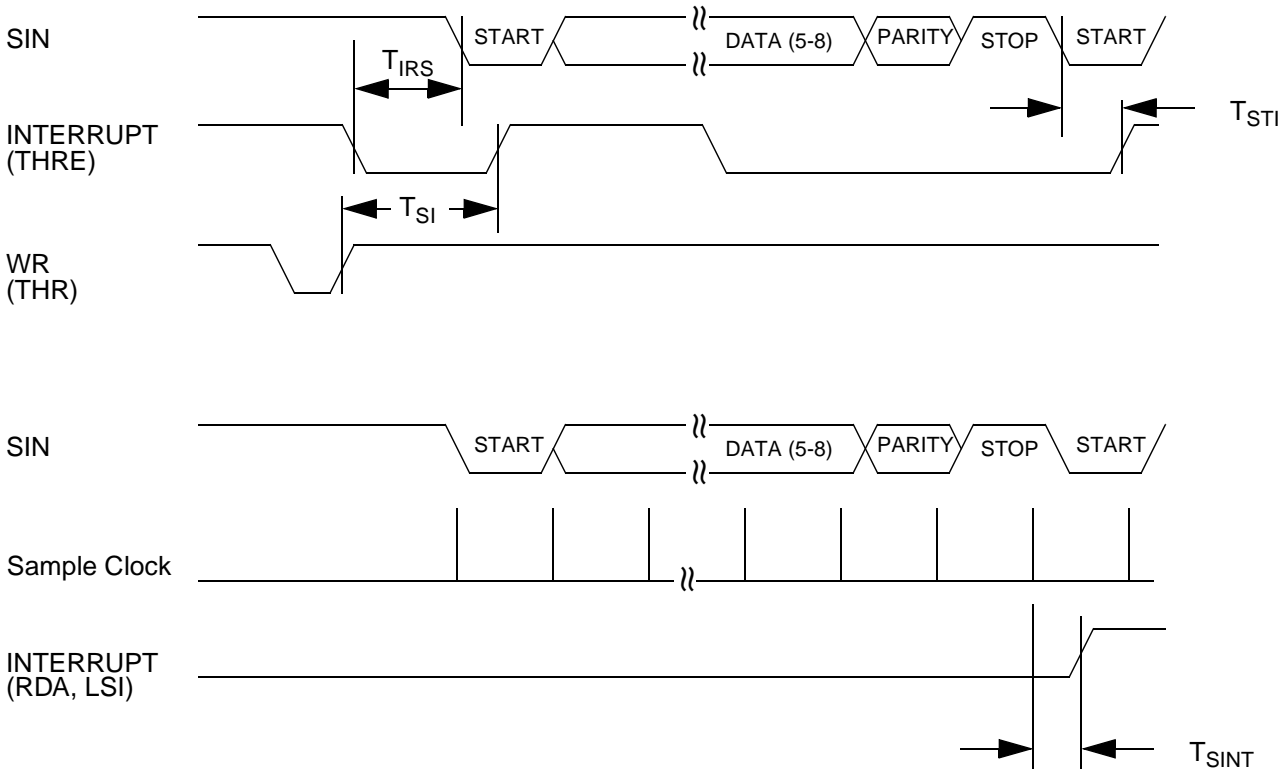


Figure 7-18 UART IRQ Timing

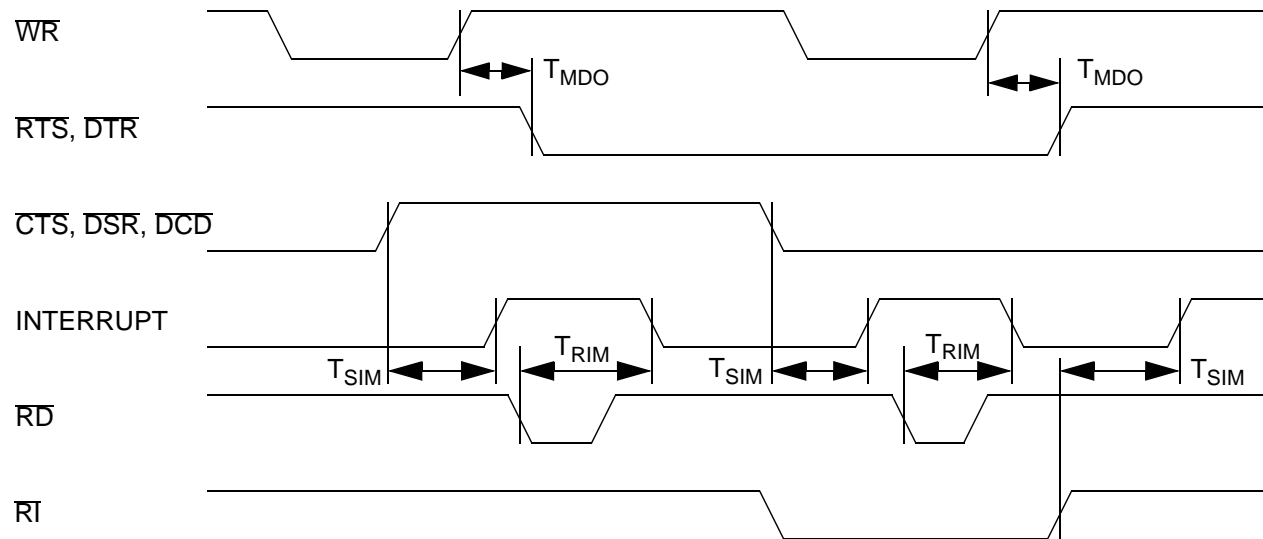


Figure 7-19 UART Modem Control Timing

7.3.5.7 LCD Controller

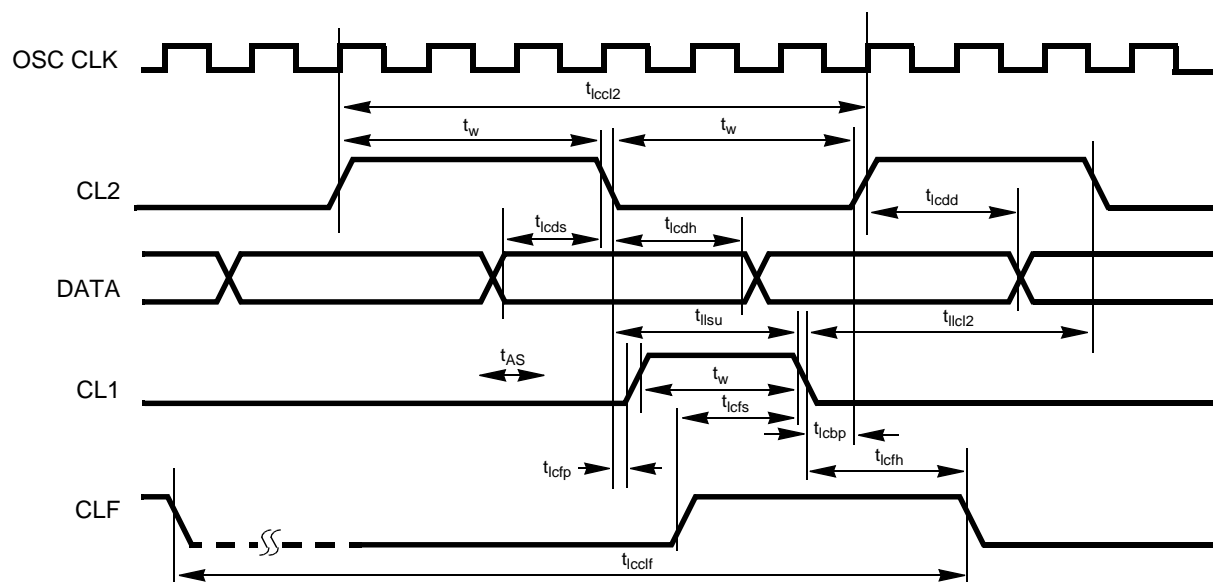


Figure 7-20 LCD Controller Timing Diagram

Table 7-18: LCD Controller Timing Specifications

Symbol	Parameter	Condition	Min	Typ	Max
t_{lccif}	Frame period	Programmable		14.3ms	
t_{lcc2}	Dot clock period	Programmable	Oscx1 *6		
t_{ics}	LCD data setup, CL2 fall		$(t_{lcc2})/2-50ns$		
t_{ich}	LCD data hold, CL2 fall		$(t_{lcc2})/2-50ns$		
t_{icdd}	LCD data delay, CL2 rise		$(t_{lcc2})/2-50ns$		50ns
t_{icsp}	CL2 falling to CL1 rising		$(t_{lcc2})/2-50ns$		
t_{icbp}	CL1 falling to CL2 rising		$(t_{lcc2})/2-50ns$		
t_{icsf}	CLF setup to CL1 fall		t_{lcc2}		
t_{ichf}	CLF hold from CL1 fall		$(t_{lcc2})/2-20ns$		
t_{ilsu}	CL1 load setup time		$(t_{lcc2})/2-50ns$		
t_{lic2}	CL1 falling to CL2 falling		$(t_{lcc2})/2-50ns$		
t_w	Pulse width		$(t_{lcc2})/2-50ns$		

7.3.5.8 Supported Testmodes

Symbol	Parameter	Conditions	Min	Max	Units
t_{AND}	AND Function Result Delay			1	ms
t_{HILO}	HI/LO Function Drive Delay			1	ms
t_{TRI}	Tri-State Outputs Delay			1	ms
t_{TOG}	Toggle Function Delay			1	ms

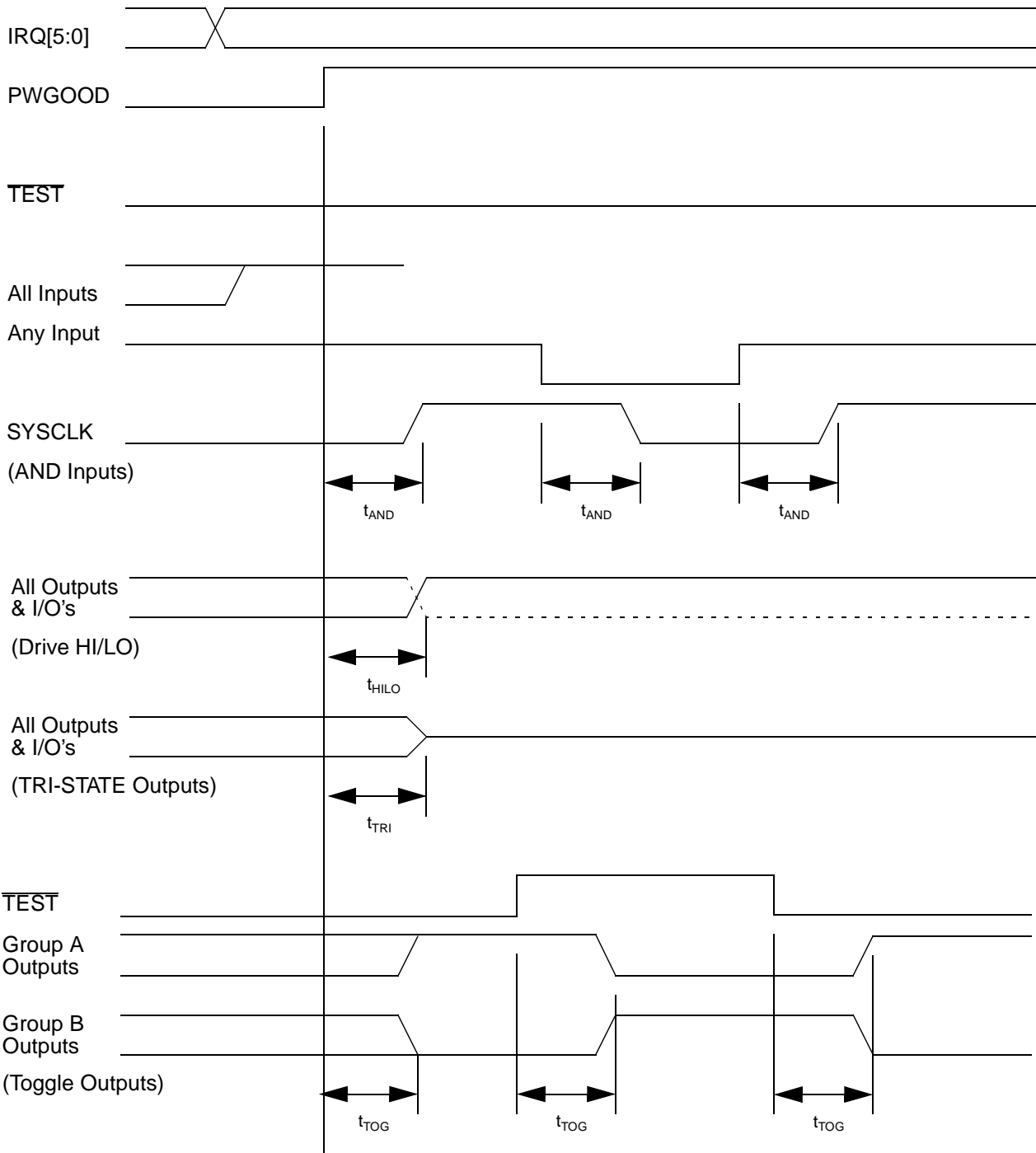
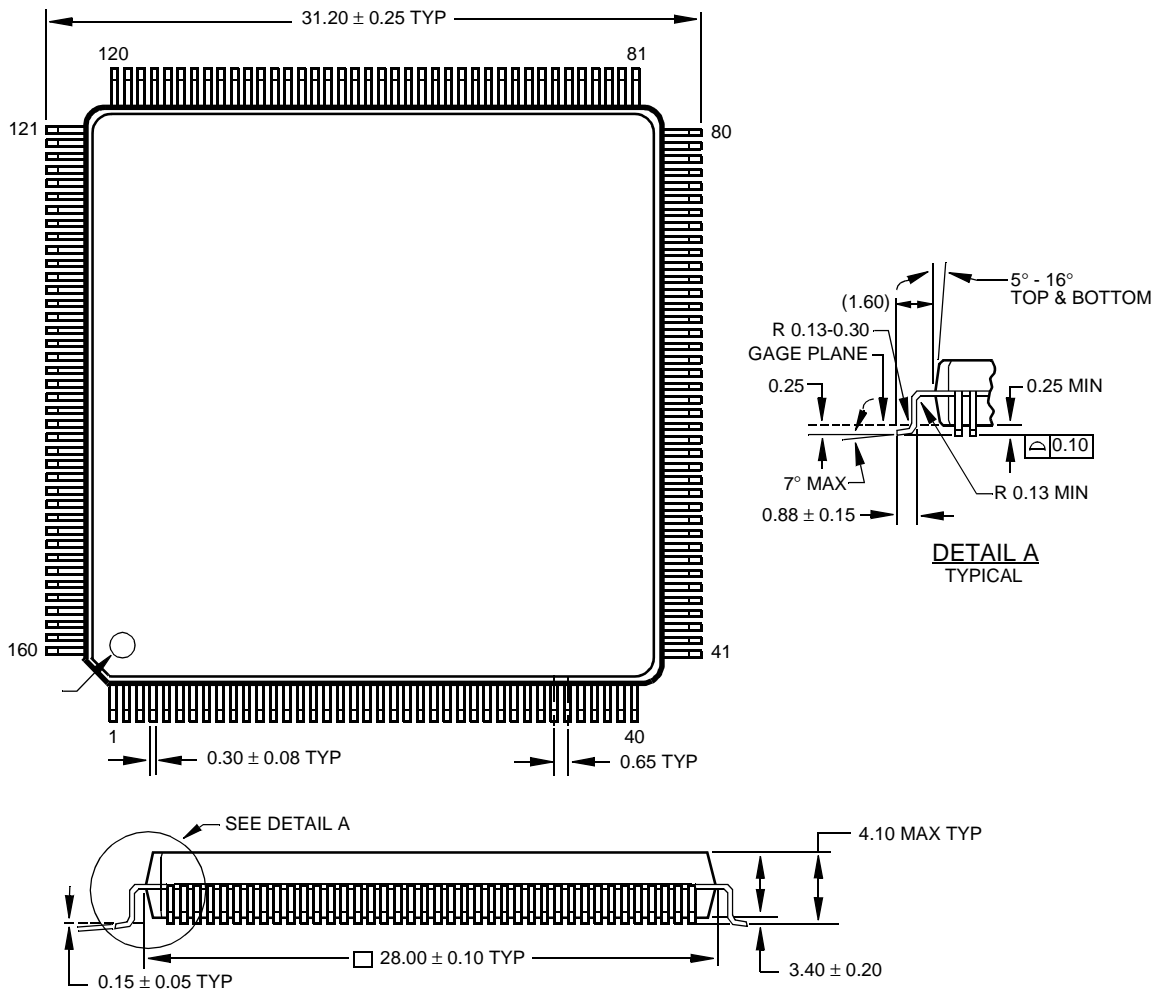


Figure 7-21 Testmode Timing Diagram

7.4 Physical Description

The NS486SXF is provided in a 160 lead, 28X28 mm, PQFP package.



160-Lead Plastic Quad Flatpack JEDEC (VUL)
NS Package Number VUL160

Figure 7-22 Plastic Package Specifications

Deliberately Blank Page

Appendix A: Key Processor Differences from the '486

The NS486SXF's 32-bit processor is instruction set compatible with the standard Intel486 processor with the following exceptions:

- 1) NS486SXF operates in 16-bit and 32-bit protected mode only.

This means that NS486SXF does not support virtual 8086 mode or real mode operation. Note that it also means that upon reset, the NS486SXF starts up in protected mode versus the standard '486's real address mode state. This means that internal register values will be different at reset.

- 2) NS486SXF uses a simple, segmented memory addressing model only.

This means that NS486SXF does not support virtual memory paging and there is no Page Unit or Translation Lookaside Buffer.

- 3) NS486SXF does not support the use of a hardware floating point co-processor.

This means that NS486SXF will execute floating point instructions in software only. There are no Floating Point instructions. All coprocessor ESCAPE instructions will trap instead, allowing software emulation of all Floating-Point instructions. The WAIT instruction will execute as a no-operation instruction; in effect, the emulated co-processor always appears to be finished.

- 4) NS486SXF has a 1 Kbyte on-board instruction cache vs. the '486's 8 Kbyte unified cache.

- 5) NS486SXF does not include the register resources to support those functions that are not implemented.

In some cases, this means some bits in some registers are reserved and not used (the EFLAGS register and Control Register 0). In some other cases, it means that the registers

controlling that resource do not exist (Control Registers 2 and 3).

Other instructions whose purpose is to affect non-existent registers or control bits will act as no-operation instructions. The result of reading a non-existent register or bit is undefined.

The WBINVD (Write Back and Invalidate Data Cache) instruction is not implemented in NS486SXF.

The INVLPG (Invalidate TLB Entry) instruction is not implemented in NS486SXF.

Otherwise, the NS486SXF core processor executes all '486 instructions (appropriate to the protected mode), including debug instructions and provides the same programming model and register set as the standard '486.

At reset, unlike the standard '486, the NS486SXF starts up in protected mode instead of real mode (since it is not implemented in the NS486SXF CPU). All shadow registers are set to zero base and maximum limit values (Base - 0, Limit = FFFFFFFF). The EIP is set to FFFFFFFF0h and the EDX is set to the current microcode revision (00000007h). The processor is in USE32 mode for code.

The processor clock is the crystal frequency divided by two. For example, a 25 MHz NS486SXF runs with a 50 MHz crystal divided by two, or 25 MHz.

Where changes have been made to the standard '486 processor, they are minor and have been made to improve performance, reduce cost and eliminate unneeded resources.

For example, using advanced semiconductor processes allows a greater number of operations to be performed in one clock cycle. This allows the reduction

of the number of instruction pipeline stages from five to only three. The ability to perform multiple operations simultaneously in one clock cycle improves both performance and power efficiency. By eliminating the extra locking and control mechanisms needed for a five-stage pipeline, significant silicon real estate space is saved and power consumption is cut. In addition, the control logic needed for full 8086 compatibility has been eliminated. With the elimination of Virtual Memory (paging), linear addresses are identical with the actual physical addresses. These changes have little or no impact in embedded applications and save significant silicon real estate. These savings translate into reduced power consumption and lower device cost. It also makes NS486SXF ideal for “green” systems and battery-operated systems.

Appendix B: Dealing with Unused Signals on the NS486SXF

This document answers the common question about what to do with unused signals on the 'SXF.

The basic rule followed is: Any signal that is an input should be pulled to its inactive state (or Pulled high if it has no effect if it is not enabled). This is necessary so the input signal does not float to the input threshold (turning on the input's p- and n-channels and getting current flow through both channels). Often this is termed as input leakage current, but it truly is current lost from input flow through switching current.

Please note that recommendations are made in this document based on the power up state of a given signal. Some of the treatments may seem to contradict the functional description of the pin in the pinout description section of the databook. In general those are signals that power-up in a TRI-STATE condition, even though they are declared as outputs in the Databook. In their enabled functionality, they will be outputs, but only after being enabled.

At a minimum, the following signals will usually be used in the system:

- SA[25:0]
- SD[15:0]
- $\overline{\text{SBHE}}$
- $\overline{\text{IOR}}$
- $\overline{\text{IOW}}$
- $\overline{\text{MEMR}}$
- $\overline{\text{MEMW}}$
- $\overline{\text{CS16}}$
- RDY
- $\overline{\text{RAS}}[1:0]$
- $\overline{\text{CASH}}[1:0]$
- $\overline{\text{CASL}}[1:0]$
- $\overline{\text{WE}}$
- Vdd
- Vss
- PWGOOD
- OSCX1 (and OSCX2 if a crystal network is used, but no OSCX2 if OSCX1 is driven by a TTL-oscillator)
- $\overline{\text{CS}}[0]$

No Pullup or Pulldown resistors should be connected to the following signals:

- $\overline{\text{RAS}}[1:0]$
- $\overline{\text{CASH}}[1:0]$
- $\overline{\text{CASL}}[1:0]$
- $\overline{\text{WE}}$
- Vdd
- Vss
- OSCX1
- $\overline{\text{CS}}[0]$

Cascade Master Mode

If Cascade Master Mode is used and an External Master takes control of the ISA-like Bus, then the following signals should have Pullups on them:

- SA[25:0]
- $\overline{\text{IOR}}$
- $\overline{\text{IOW}}$
- $\overline{\text{MEMR}}$
- $\overline{\text{MEMW}}$

Otherwise, these signals do not need Pullup or Pulldown resistors.

Data Bus

The ISA-like data bus (SD[15:0]) should be pulled up with 10k resistors so that it does not float during long periods of inactivity. SD[15:0] defaults to be an input into the NS486SXF and thus will float (if it is not pulled up) during IDLE times.

Strapping Options

$\overline{\text{SBHE}}$ - Sampled at the rising edge of PWGOOD to determine Boot ROM interface size (8-bit if pulled down, 16-bit if pulled up). Choose appropriately.

Pulled High

If the following signals are unused, pulling them high will prevent them from floating to input threshold and resulting in high input leakage.

- $\overline{\text{CS16}}$ - Should always be pulled high.
- RDY - Should always be pulled high.
- $\overline{\text{TEST}}$ - Should always be pulled high.

- $\overline{\text{EACK/CS[7]/DSR}}$ - This signal should be pulled high and should never be enabled as $\overline{\text{CS[7]}}$ or $\overline{\text{DSR}}$ (This is due to a Revision C Bug). It may be used as $\overline{\text{EACK}}$, but in that case a Pullup resistor should not cause any issue.
- $\overline{\text{IRQ[5:0]}}$ - Should always be pulled high.

Tie Low If Unused

- DRQ[4,3,2,0]
- NMI
- Vbat
- RTCX1 (RTCX2 should be a No Connect if the RTC is unused and RTCX1 is tied low.)

Pull High If Unused

- DPH and DPL - These signals will be TRI-STATE if DRAM Parity is not enabled. So to prevent these signals from floating to input threshold and resulting in high input leakage, it is suggested that these signals are pulled high if DRAM Parity is not to be enabled.
- Rx - This will be an input and if not pulled high it may float to input threshold and result in high input leakage.
- T0 - This pin defaults to being an input and pulling it high will prevent it from floating to input threshold and resulting in high input leakage (It may also be configured as an output via Boot software to prevent this situation, if so desired.)
- T1 - This pin defaults to being an input and pulling it high will prevent it from floating to input threshold and resulting in high input leakage (It may also be configured as an output via Boot software to prevent this situation, if so desired.)
- CD_RST and $\overline{\text{REG}}$ - These PCMCIA output signals will be in TRI-STATE after a PWGOOD reset. To prevent the signal from floating to input threshold and resulting in high input leakage, pull them high.
- $\overline{\text{IOIST16/WP}}$, $\overline{\text{BVD1/STSCNG}}$, $\overline{\text{BVD2/SPKR}}$, $\overline{\text{CD1}}$, $\overline{\text{CD2}}$, $\overline{\text{GPI}}$ - All of these signals are PCMCIA inputs and should be pulled high to prevent the signal from floating to input threshold and resulting in high input leakage when they are not being used.
- $\overline{\text{SO/DCD}}$, $\overline{\text{SI/CTS}}$, $\overline{\text{SCLK/RI}}$ - All of these signals are 3-wire/UART signals that power up as

inputs and should be pulled high to prevent the signal from floating to input threshold and resulting in high input leakage when they are not being used.

No Connection When Unused

- UCLK - This pin defaults to being an output (oscillating) upon reset, so it can be left unconnected.
- $\overline{\text{DACK[4,3,2,0]}}$
- TC/ $\overline{\text{EOP}}$ (NOTE: If TC/ $\overline{\text{EOP}}$ is ever used as $\overline{\text{EOP}}$, then it should be pulled high, because it may be driven active-low by an external open-drain driver, that is if the external device adheres to traditional PC implementation of EOP# signals.)
- RESET
- $\overline{\text{RESET}}$
- SYSCLK
- $\overline{\text{EREQ/CS[6]/RTS}}$ - This signal may be configured as either $\overline{\text{EREQ}}$ or $\overline{\text{CS[6]}}$, but should not be configured as $\overline{\text{RTS}}$ due to the fact that would also configure the $\overline{\text{EACK/CS[7]/DSR}}$ pin as $\overline{\text{DSR}}$ and would run into the documented Revision C bug.
- $\overline{\text{DRV/CS[8]/DTR}}$ - This signal may be configured as either $\overline{\text{DRV}}$ or $\overline{\text{CS[8]}}$, but should not be configured as $\overline{\text{DTR}}$ due to the fact that would also configure the $\overline{\text{EACK/CS[7]/DSR}}$ pin as $\overline{\text{DSR}}$ and would run into that documented Revision C bug.
- $\overline{\text{INTA}}$
- LCD[3:0]
- CL2
- CL1
- CLF
- Tx
- $\overline{\text{CS[5:1]}}$
- $\overline{\text{Vcc_SEL}}$
- Vpp_SEL1
- Vpp_SEL2
- DIR
- $\overline{\text{ENABLE}}$
- RSV

Parallel Port Signals

The Parallel Port signals all Power-Up TRI-STATE. So the safest course is to pull them up with a resistor if the parallel port is going to be unused.

- PD[7:0]
- $\overline{\text{SLIN}}$
- $\overline{\text{STB}}$
- $\overline{\text{AFD}}$
- $\overline{\text{INIT}}$
- $\overline{\text{ACK}}$
- $\overline{\text{PE}}$
- SLCT
- $\overline{\text{ERR}}$
- BUSY

Appendix C: Getting Started

There are a number of tasks that the user software must perform upon power-up or reset condition. Basically these are setting up the configuration registers of the various NS486SXF I/O resources (such as the ECP port, the UART, the three-wire serial interface, etc.), setting up the operating parameters of the various NS486SXF functional subsystems (such as the DRAM controller and the DMA controller), and other housekeeping functions for the NS486SXF CPU.

In general, the programmer will want to set up all configuration parameters of a functional unit **before** enabling it.

Note: In particular, the user should take care to power-up and power-down the LCD controller and the display panel in the proper sequence. See the LCD controller section for further information.

A Typical Power-up Sequence

The NS486SXF power-up sequence is invoked by the assertion of the active high hardware PWRGOOD pin, pin 60. The power supply logic should keep this pin low until power is stable and valid voltage levels are present.

(Note that the two reset pins, RESET and $\overline{\text{RESET}}$, pins 119 and 120, are available for use with peripheral circuitry that need active high or active low signals for reset. These are not the NS486SXF processor reset signals.)

Processor Reset

1) at power up the processor will set all shadow registers to Base = 0, Limit = FFFFFFFF. The EIP Register will be set to FFFFFFFF0 h, and the EDX Register will be set to 00000007h (the current microcode revision level). The processor is in USE32 mode for code access.

IDTR, GDTR, TR, LDTR registers are undefined, CD, SS, DS, ES, FS, GS registers are undefined. Other registers are undefined.

Note that these values differ from the standard '486 processor reset values because the NS486SXF starts up in protected mode where the standard '486 starts up in real-address mode.

After the processor housekeeping, the user program is executed. This program should first configure the NS486SXF resources, enable the various internal peripheral controllers and setup the power management structure.

Power Management Setup

NS486SXF power management is strictly software controlled via writes to the Power Management I/O block (Power Management Registers 1-4) in the NS486SXF I/O address map (EF90h - EF93h). A configuration bit should be set for either 3 volt or 5 volt I/O operation (Bit 5 of Power Management Register 4). The default condition is 3 volt operation, and thus this parameter must be set immediately upon startup. Depending upon whether a crystal oscillator circuit or an external clock oscillator is providing the system clock, the appropriate enable/disable bits should be set in Power Management Register 1. (Note: when disabling a crystal oscillator circuit, there will be an approximately 1 msec delay time for the external crystal to restart and stabilized upon enabling.)

Since at power-up all resources are enabled at the maximum clock frequency, all unused NS486SXF resources should be disabled. Next, the clocks to enabled internal resources should be selected (the reference clock for the timer, SYSCLK, PCMCIA, etc.) using Power Management Register 3. In addition, clock power saving modes can be selected using Power Management Register 1. This allows `cpu_clock` division by 1, 4, 8, 16, 32 or 64.

At this point, the user software can set an appropriate power saving mode. (Power-save, Crystal Oscillator Circuit Disable, Power down, CPU Idle, individual enable/disable of peripheral clocks, etc.)

If after setup of power saving modes, the CPU should go into a power saving idle mode, note that DMA transfers can still take place since the BIU provides the necessary handshaking signals when the CPU is idle.

DRAM Sequence

Since it is likely that the user software will want to operate out of RAM rather quickly, (especially since the display data for the LCD controller is usually stored in RAM), the next function to be performed at start up is the initialization of the DRAM controller. This requires setting bits in the DRAM Control Registers that set the memory page size, the amount and location of the second bank of DRAM memory, whether

parity is being used or not, and if so if a NMI is being generated on a parity error condition, whether a 3 CPU clock cycle on paging or a 4 clock cycle is being used. The configuration parameters must be programmed into the DRAM Control Registers before enabling the DRAM controller (DRAM Control Register, I/O address EF80-81h, bit 0 =1).

At system reset the DRAM Controller is in an idle state. Following reset, the DRAM controller will wait for 100 μ sec (with a 66 MHz oscillator clock) or until the DRAM Enable bit [DRAM Control Register, address EF80-81h, bit 0] is programmed to a 1, whichever occurs later. After the 100 μ sec delay, and with the DRAM Enable bit set, the DRAM Controller will initiate eight CAS-before-RAS refresh cycles. At the end of the eighth refresh cycle, the DRAM Initialization Complete bit [DRAM Status Register, address EF86-87, bit 0] is set to a 1, indicating that the DRAM may now be read and written. The user program can check this bit to see when the DRAM is ready. Note that if during operation, the DRAM Enable bit [DRAM Control Register, address EF80-81h, bit 0] is set to 0, the DRAM controller will not stop immediately. It will wait until the end of the next refresh cycle. Subsequent re-enabling is immediate.

The LCD Controller

At this point, other peripherals and internal functional blocks can be enabled and configured for normal operation. One peripheral does require special handling - the LCD controller. Because of the nature of the LCD panels, care must be taken in applying the high voltages used in the display panels themselves.

Applying Power:

It is very important that power be applied to the LCD display in proper sequence, otherwise damage can result. To prevent damage to the LCD panels, the external DC power supplied to the LCD Display (V_{EE}) must be disabled before the LCD Controller's clock is disabled.

The power-up sequence is as follows:

- 1) Configure the LCD control registers.
- 2) Apply V_{DD} to the display.

- 3) Enable the LCD clocks from the power management registers - this must be done within 20 msec. of applying V_{DD} .
- 4) Enable the LCD controller.
- 5) Within 20 msec. max after applying the LCD clocks, apply V_{EE} (22V/-26V) to the display.

The power-down sequence is as follows:

- 1) Remove V_{EE} from the display.
- 2) Disable the LCD controller.
- 3) Within 20 msec. of removing V_{EE} , disable the LCD clocks.
- 4) Within 20 msec. of removing the LCD clocks, remove V_{DD} from the display.

Never disable the LCD clocks when the LCD is enabled.

This sequence is shown in the following figure.

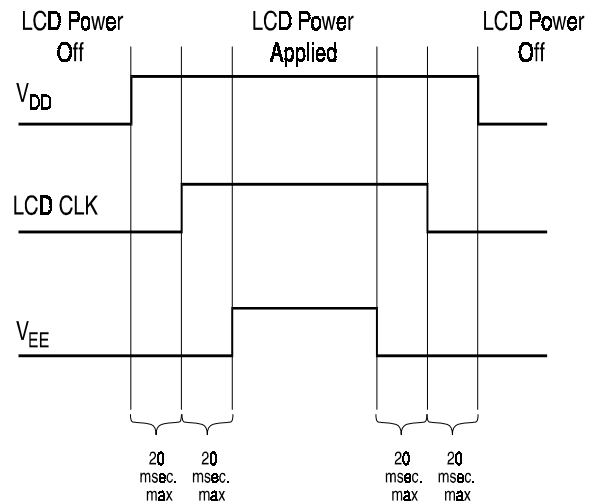


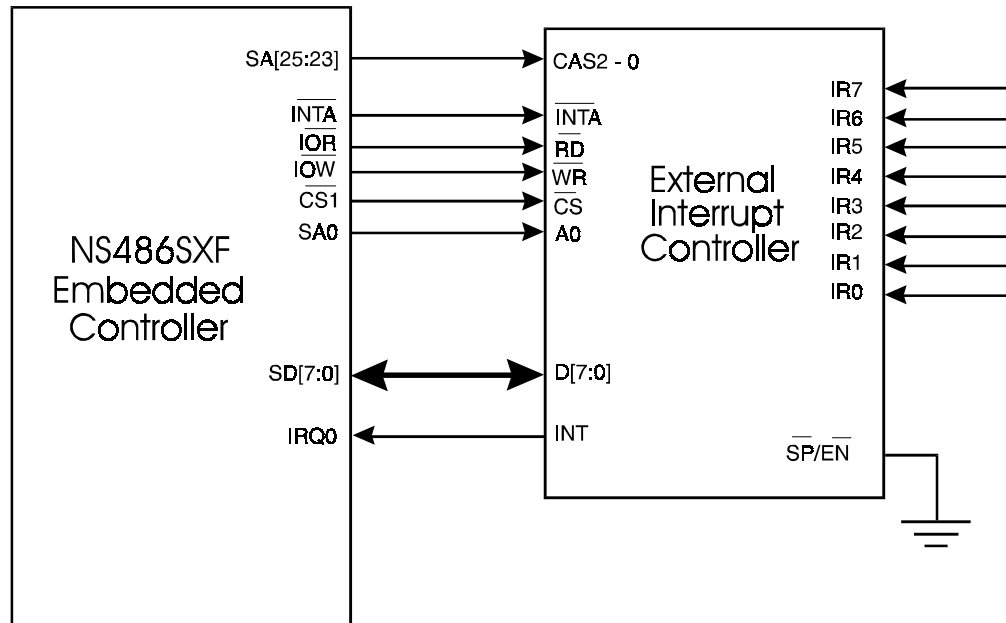
Figure C-1 LCD Power Application Sequencing

This Page Intentionally Left Blank.

Appendix D: External Interrupt Controller

Connecting a cascaded external interrupt controller to NS486SXF is shown in the following diagram.

Figure D-1 External Interrupt Controller Connection



Note: This diagram shows one method to connect a cascaded external interrupt controller up to the NS486SXF. However, any of the IRQ[n] pins can be used (as long as that pin is multiplexed to the NS486SXF's internal master interrupt controller) and any one of CS8-CS1 signals. IRQ0 and CS1 are provided as examples only.

For a system that uses data buffering, the buffers will obviously need to be enabled for reading during an interrupt acknowledge cycle for an external slave. One way of achieving this is to put the external interrupt controller into slave mode through software, and use the SP/EN and INTA signals to enable the system data read buffers.

This Page Intentionally Left Blank.

Appendix E: Instruction Set

The following table summarizes the instruction set of the NS486SXF. Except as noted here, all documented 486 instructions are implemented in a fully binary-compatible manner. The differences between the NS486SXF and 486 instruction sets are as follows:

- There are no Floating Point instructions. All coprocessor ESCAPE instructions will trap instead, allowing software emulation of all Floating-Point instructions. The WAIT instruction will execute as a no-operation instruction; in effect, the emulated co-processor always appears to be finished.
- WBINVD (Write Back and Invalidate Data Cache) is not implemented in NS486SXF.
- INVLPG (Invalidate TLB Entry) is not implemented in NS486SXF.
- Other instructions whose purpose is to affect non-existent registers or control bits will act as no-operation instructions. The result of reading a non-existent register or bit is undefined.

The Clock Count column expresses the execution time in units of the CPU clock. In the NS486SXF, this is the crystal frequency divided by two. For example, a “25-MHz” NS486SXF device runs at full speed with a 50 MHz crystal, and the Clock Count column counts cycles of the 25 MHz (divided) clock.

The following assumptions are made in the Clock Count column:

1. Each instruction is assumed to be fully available in the prefetch buffer before execution.
2. All data accesses are assumed to be best-case: page-mode hits on one of the two banks of DRAM.
3. Except where noted, all memory accesses are assumed to be bytes or aligned words. There is a penalty of one clock for a misaligned word or an aligned double word, and a penalty of two clocks for a misaligned double word.
4. For general (r/m) operands, it is assumed that only one General Register is involved in the address calculation. If two General Registers (a base register and an index register) are used, add one clock.
5. In branching instructions, the target instruction is assumed to be in the instruction cache.
6. It is assumed that no exceptions are detected during instruction execution.
7. The bus is assumed to be available whenever it is used.

Figure E-1 NS486SXF Core Processor Instruction Set Summary

Instruction	Format	Clock Count	'486 Count	Notes
INTEGER OPERATIONS				
MOV = Move				
reg1 to reg2	1 0 0 0 1 0 0 w 1 1 reg1 reg2	1	1	
reg2 to reg1	1 0 0 0 1 0 1 w 1 1 reg1 reg2	1	1	
Memory to reg	1 0 0 0 1 0 1 w mod reg r/m	1	1	
reg to Memory	1 0 0 0 1 0 0 w mod reg r/m	1	1	
Immediate to reg	1 1 0 0 0 1 1 w 1 1 0 0 0 reg	1	1	immediate data
or:	1 0 1 1 w reg	1	1	immediate data
Immediate to Memory	1 1 0 0 0 1 1 w mod 0 0 0 r/m	1	1	disp, immediate
Memory to Accumulator	1 0 1 0 0 0 0 w full displacement	1	1	
Accumulator to Memory	1 0 1 0 0 0 1 w full displacement	1	1	
MOVSX/MOVZX = Move with Sign/Zero Extension				
reg2 to reg1	0 0 0 0 1 1 1 1 1 1 0 1 1 z 1 1 w 1 1 reg1 reg2	1	3	
Memory to reg	0 0 0 0 1 1 1 1 1 1 0 1 1 z 1 1 w mod reg r/m	3	3	
Instruction	z			
MOVZX	0			
MOVSX	1			
PUSH = Push				
reg	1 1 1 1 1 1 1 1 1 1 1 1 1 0 reg	1	4	
or:	0 1 0 1 0 reg	1	1	
Memory	1 1 1 1 1 1 1 1 1 1 1 1 0 r/m	3	4	
Immediate	0 1 1 0 1 0 s 0 immediate data	1	1	
PUSHA = Push All				
	0 1 1 0 0 0 0 0	8	11	

Figure E-1 NS486SXF Core Processor Instruction Set Summary

Instruction	Format	Clock Count	'486 Count	Notes
POP = Pop				
reg	1 0 0 0 1 1 1 1 1 1 0 0 0 reg	1	4	
or:	0 1 0 1 1 reg	1	1	
Memory	1 0 0 0 1 1 1 1 mod 0 0 0 r/m	5	5	
POPA = Pop All				
	0 1 1 0 0 0 0 1	8	9	
XCHG = Exchange				
reg1 with reg2	1 0 0 0 0 1 1 w 1 1 reg1 reg2	3	3	
Accumulator with reg	1 0 0 1 0 reg	3	3	
Memory with reg	1 0 0 0 0 1 1 w mod reg r/m	3	5	
NOP = No Operation				
	1 0 0 1 0 0 0 0	1	1	
LEA = Load EA to Register				
no index register	1 0 0 0 1 1 0 1 mod reg r/m	2	1	
with index register		3	2	

Figure E-1 NS486SXF Core Processor Instruction Set Summary

Instruction	Format	Clock Count	'486 Count	Notes
ALU2 Instructions: ADD, ADC, AND, OR, SUB, SBB, XOR				
reg1 to reg2	00TTTT00w 11 reg1 reg2	1	1	
reg2 to reg1	00TTTT01w 11 reg1 reg2	1	1	
Memory to reg	00TTTT01w mod reg r/m	3	2	
reg to Memory	00TTTT00w mod reg r/m	3	3	
Immediate to reg	100000sw 11 TTT reg	1	1	immediate data
Immediate to Accumulator	00TTTT10w	1	1	immediate data
Immediate to Memory	100000sw mod TTT r/m	3	3	immediate data
Instruction	TTT			
ADD = Add	000			
ADC = Add with Carry	010			
AND = Logical And	100			
OR = Logical Or	001			
SUB = Subtract	101			
SBB = Subtract with Borrow	011			
XOR = Logical Exclusive Or	110			
Increment / Decrement Instructions				
reg	1111111w 11 TTT reg	1	1	
or:	01 TTT reg	1	1	
Memory	1111111w mod TTT r/m	3	3	
Instruction	TTT			
INC = Increment	000			
DEC = Decrement	001			
ALU1 Instructions: NOT, NEG				
reg	1111011w 11 TTT reg	1	1	
Memory	1111011w mod TTT r/m	3	3	
Instruction	TTT			
NOT = Logical Complement	010			
NEG = Negate	011			

Figure E-1 NS486SXF Core Processor Instruction Set Summary

Instruction	Format	Clock Count	'486 Count	Notes
CMP = Compare				
reg1 to reg2	0011100w 11 reg1 reg2	1	1	
reg2 to reg1	0011101w 11 reg1 reg2	1	1	
Memory to reg	0011101w mod reg r/m	3	2	
reg to Memory	0011100w mod reg r/m	3	2	
Immediate to reg	100000sw 11 111 reg	1	1	immediate data
Immediate to Accumulator	0011110w immediate data	1	1	
Immediate to Memory	100000sw mod 111 r/m	3	2	immediate data
TEST = Logical Compare				
reg1 and reg2	1000010w 11 reg1 reg2	1	1	
Memory and reg	1000010w mod reg r/m	3	2	
Immediate and reg	1111011w 11 000 reg	1	1	immediate data
Immediate and Accumulator	1010100w immediate data	1	1	
Immediate and Memory	1111011w mod 000 r/m	3	2	immediate data
MUL = Multiply (Unsigned)				
Accumulator by reg	1111011w 11 100 reg			
Multiplier = Byte		14	13/18	min/max
Multiplier = Word		21	13/26	
Multiplier = Double Word		37	13/42	
Accumulator by Memory	1111011w mod 100 r/m			
Multiplier = Byte		16	13/18	
Multiplier = Word		23	13/26	
Multiplier = Double Word		39	13/42	

Figure E-1 NS486SXF Core Processor Instruction Set Summary

Instruction	Format	Clock Count	'486 Count	Notes			
IMUL = Integer Multiply (Signed)							
Accumulator by reg	<table border="1"><tr><td>1 1 1 1 0 1 1 w</td><td>1 1 1 0 1 reg2</td></tr></table>	1 1 1 1 0 1 1 w	1 1 1 0 1 reg2				
1 1 1 1 0 1 1 w	1 1 1 0 1 reg2						
Multiplier = Byte		14	13/18	min/max			
Multiplier = Word		24	13/26				
Multiplier = Double Word		36	13/42				
Accumulator by Memory	<table border="1"><tr><td>1 1 1 1 0 1 1 w</td><td>mod 1 0 1 r/m</td></tr></table>	1 1 1 1 0 1 1 w	mod 1 0 1 r/m				
1 1 1 1 0 1 1 w	mod 1 0 1 r/m						
Multiplier = Byte		16	13/18				
Multiplier = Word		26	13/26				
Multiplier = Double Word		42	13/42				
reg1 by reg2	<table border="1"><tr><td>0 0 0 0 1 1 1 1</td><td>1 0 1 0 1 1 1 1</td><td>1 1 reg1 reg2</td></tr></table>	0 0 0 0 1 1 1 1	1 0 1 0 1 1 1 1	1 1 reg1 reg2			
0 0 0 0 1 1 1 1	1 0 1 0 1 1 1 1	1 1 reg1 reg2					
Multiplier = Word		24	13/26				
Multiplier = Double Word		40	13/42				
reg by Memory	<table border="1"><tr><td>0 0 0 0 1 1 1 1</td><td>1 0 1 0 1 1 1 1</td><td>mod reg r/m</td></tr></table>	0 0 0 0 1 1 1 1	1 0 1 0 1 1 1 1	mod reg r/m			
0 0 0 0 1 1 1 1	1 0 1 0 1 1 1 1	mod reg r/m					
Multiplier = Word		24	13/26				
Multiplier = Double Word		40	13/42				
reg1 by Immediate to reg2	<table border="1"><tr><td>0 1 1 0 1 0 s 1</td><td>1 1 reg1 reg2</td></tr></table> immediate data	0 1 1 0 1 0 s 1	1 1 reg1 reg2				
0 1 1 0 1 0 s 1	1 1 reg1 reg2						
Multiplier = Word		24	13/26				
Multiplier = Double Word		40	13/42				
Memory by Immediate to reg	<table border="1"><tr><td>0 1 1 0 1 0 s 1</td><td>mod reg r/m</td></tr></table> immediate data	0 1 1 0 1 0 s 1	mod reg r/m				
0 1 1 0 1 0 s 1	mod reg r/m						
Multiplier = Word		24	13/26				
Multiplier = Double Word		40	13/42				
DIV = Divide (Unsigned)							
Accumulator by reg	<table border="1"><tr><td>1 1 1 1 0 1 1 w</td><td>1 1 1 1 0 reg</td></tr></table>	1 1 1 1 0 1 1 w	1 1 1 1 0 reg				
1 1 1 1 0 1 1 w	1 1 1 1 0 reg						
Divisor = Byte		8/15	16	min/max			
Divisor = Word		8/23	24				
Divisor = Double Word		8/39	40				
Accumulator by Memory	<table border="1"><tr><td>1 1 1 1 0 1 1 w</td><td>mod 1 1 0 r/m</td></tr></table>	1 1 1 1 0 1 1 w	mod 1 1 0 r/m				
1 1 1 1 0 1 1 w	mod 1 1 0 r/m						
Divisor = Byte		8/15	16				
Divisor = Word		8/23	24				
Divisor = Double Word		8/39	40				

Figure E-1 NS486SXF Core Processor Instruction Set Summary

Instruction	Format	Clock Count	'486 Count	Notes
IDIV = Integer Divide (Signed)				
Accumulator by reg	1 1 1 1 0 1 1 w 1 1 1 1 1 reg			
Divisor = Byte		3/23	19	min/max
Divisor = Word		3/31	27	
Divisor = Double Word		3/47	43	
Accumulator by Memory	1 1 1 1 0 1 1 w mod 1 1 1 r/m			
Divisor = Byte		5/25	20	
Divisor = Word		5/33	28	
Divisor = Double Word		5/49	44	
CBW / CWDE = Convert Byte to Word / Convert Word to Double Word				
	1 0 0 1 1 0 0 0	3	3	
CWD / CDQ = Convert Word to Double Word / Convert Double Word to Quad Word				
	1 0 0 1 1 0 0 1	2	3	

Figure E-1 NS486SXF Core Processor Instruction Set Summary

Instruction	Format	Clock Count	'486 Count	Notes
Shift and Rotate Instructions				
Not Through Carry (ROL, ROR, SAL, SAR, SHL, SHR)				
reg by 1	1 1 0 1 0 0 0 w 1 1 T T T reg	1	3	
Memory by 1	1 1 0 1 0 0 0 w mod T T T r/m	3	4	
reg by count in CL	1 1 0 1 0 0 1 w 1 1 T T T reg	1	3	
Memory by count in CL	1 1 0 1 0 0 1 w mod T T T r/m	3	4	
reg by Immediate count	1 1 0 0 0 0 0 w 1 1 T T T reg	immediate 8-bit	1	2
Memory by Immediate count	1 1 0 0 0 0 0 w mod T T T r/m	immediate 8-bit	3	4
Through Carry (RCL, RCR): Times are min/max based on count.				
reg by 1	1 1 0 1 0 0 0 w 1 1 T T T reg	1	3	
Memory by 1	1 1 0 1 0 0 0 w mod T T T r/m	3	4	
reg by count in CL	1 1 0 1 0 0 1 w 1 1 T T T reg	4/4+N	8/30	N = 31max
Memory by count in CL	1 1 0 1 0 0 1 w mod T T T r/m	5/6+N	9/31	
reg by Immediate count	1 1 0 0 0 0 0 w 1 1 T T T reg	immediate 8-bit	4/4+N	8/30
Memory by Immediate count	1 1 0 0 0 0 0 w mod T T T r/m	immediate 8-bit	5/6+N	9/31
Instruction	T T T			
ROL = Rotate Left	0 0 0			
ROR = Rotate Right	0 0 1			
RCL = Rotate w/Carry Left	0 1 0			
RCR = Rotate w/Carry Right	0 1 1			
SHL / SAL = Shift Left	1 0 0			
SHR = Shift Logical Right	1 0 1			
SAR = Shift Arithmetic Right	1 1 1			
Double Precision Shift Instructions (SHLD, SHRD): Times are min/max.				
reg by immediate count (8-bit, not shown)	0 0 0 0 1 1 1 1 1 0 T T T 1 0 0 1 1 reg2 reg1	6/10	2	
Memory by immed. count (8-bit, not shown)	0 0 0 0 1 1 1 1 1 0 T T T 1 0 0 mod reg r/m	7/10	3	
reg by count in CL	0 0 0 0 1 1 1 1 1 0 T T T 1 0 1 1 1 reg2 reg1	5/9	3	
Memory by count in CL	0 0 0 0 1 1 1 1 1 0 T T T 1 0 1 mod reg r/m	6/9	4	
Instruction	T T T			
SHLD = Shift Left Double	1 0 0			
SHRD = Shift Right Double	1 0 1			

Figure E-1 NS486SXF Core Processor Instruction Set Summary

Instruction	Format	Clock Count	'486 Count	Notes
BSWAP = Byte Swap				
	00001111 11001 reg	11	1	
XADD = Exchange and Add				
reg1, reg2	00001111 1100000w 11 reg2 reg1	3	3	
Memory, reg	00001111 1100000w mod reg r/m	5	4	
CMPXCHG = Compare and Exchange: Times are equal/not equal.				
reg1, reg2	00001111 1011000w 11 reg2 reg1	3	6	
Memory, reg	00001111 1011000w mod reg r/m	4	7/10	

Figure E-1 NS486SXF Core Processor Instruction Set Summary

Instruction	Format	Clock Count	'486 Count	Notes
CONTROL TRANSFER OPERATIONS (within segment) (Conditional Branch Times are Taken / Not Taken)				
Jccc = Jump on Condition "ccc"				
8-bit displacement	0 1 1 1 t t t n 8-bit disp.	3 / 1	3 / 1	
full displacement	0 0 0 0 1 1 1 1 1 0 0 0 t t t n full displacement	3 / 1	3 / 1	
SETccc = Set Byte on Condition "ccc": Times are true/false.				
reg	0 0 0 0 1 1 1 1 1 0 0 1 t t t n 1 1 0 0 0 reg	1	4/3	
Memory	0 0 0 0 1 1 1 1 1 0 0 1 t t t n mod 0 0 0 r/m	2	3/4	
ccc	Condition	t t t n		
O	= Overflow	0000		
NO	= No Overflow	0001		
B/NAE	= Below / Not Above or Equal	0010		
NB/AE	= Not Below / Above or Equal	0011		
E/Z	= Equal / Zero	0100		
NE/NZ	= Not Equal / Not Zero	0101		
BE/NA	= Below or Equal / Not Above	0110		
NBE/A	= Not Below or Equal / Above	0111		
S	= Sign	1000		
NS	= Not Sign	1001		
P/PE	= Parity / Parity Even	1010		
NP/PO	= Not Parity / Parity Odd	1011		
L/NGE	= Less / Not Greater or Equal	1100		
NL/GE	= Not Less / Greater or Equal	1101		
LE/NG	= Less or Equal / Not Greater	1110		
NLE/G	= Not Less or Equal / Greater	1111		
LOOP = Loop CX Times: Times are loop/not loop.				
8-bit displacement (only)	1 1 1 0 0 0 1 0 8-bit disp.	5 / 4	7 / 6	
LOOPZ / LOOPE = Loop with Zero / Equal: Times are loop/not loop.				
8-bit displacement (only)	1 1 1 0 0 0 0 1 8-bit disp.	5 / 4	9 / 6	
LOOPNZ / LOOPNE = Loop while Not Zero / Not Equal: Times are loop/not loop.				
8-bit displacement (only)	1 1 1 0 0 0 0 0 8-bit disp.	5 / 4	9 / 6	
JCXZ = Jump on CX zero: Times are true/not true. JECXZ = Jump on ECX zero: Times are true/not true.				
8-bit displacement (only)	1 1 1 0 0 0 1 1 8-bit disp.	5 / 4	8 / 5	
(Note: Address Size prefix differentiates between JCXZ and JCXZE)				

Figure E-1 NS486SXF Core Processor Instruction Set Summary

Instruction	Format	Clock Count	'486 Count	Notes
JMP = Unconditional Jump (within segment)				
Short	1 1 1 0 1 0 1 1 8-bit disp.	3	3	
Direct	1 1 1 0 1 0 0 1 full displacement	3	3	
Register Indirect	1 1 1 1 1 1 1 1 1 1 1 0 0 reg	4	5	
Memory Indirect	1 1 1 1 1 1 1 1 mod 1 0 0 r/m	5	5	
CALL = Call (within segment)				
Direct	1 1 1 0 1 0 0 0 full displacement	3	3	
Register Indirect	1 1 1 1 1 1 1 1 1 1 0 1 0 reg	4	5	
Memory Indirect	1 1 1 1 1 1 1 1 mod 0 1 0 r/m	5	5	
RET = Return from CALL (within segment)				
	1 1 0 0 0 0 1 1	5	5	
Adding 16-bit Immediate to SP	1 1 0 0 0 0 1 0 16-bit disp.	5	5	
ENTER = Enter Procedure				
	1 1 0 0 1 0 0 0 16-bit disp., 8-bit level			
Level = 0		9	14	
Level = 1		14	17	
Level (L) > 1		14 + 4L	17 + 3L	
LEAVE = Leave Procedure				
	1 1 0 0 1 0 0 1	1	5	

Figure E-1 NS486SXF Core Processor Instruction Set Summary

Instruction	Format	Clock Count	'486 Count	Notes
INTERSEGMENT INSTRUCTIONS				
MOV = Move Segment Register				
reg to Segment reg	1 0 0 0 1 1 1 0 1 1 sreg3 reg	1	9	
Memory to Segment reg	1 0 0 0 1 1 1 0 mod sreg3 r/m	1	9	
Segment reg to reg	1 0 0 0 1 1 0 0 1 1 sreg3 reg	5/9	3	
Segment reg to Memory	1 0 0 0 1 1 0 0 mod sreg3 r/m	6/10	3	
PUSH = Push Segment Register				
Segment reg ES, CS, SS or DS	0 0 0 sreg2 1 1 1 0	1	3	
Segment reg FS or GS	0 0 0 0 1 1 1 1 1 0 sreg3 0 0 0	1	3	
POP = Pop Segment Register				
Segment reg ES, CS, SS or DS	0 0 0 sreg2 1 1 1	8/11	9	
Segment reg FS or GS	0 0 0 0 1 1 1 1 1 0 sreg3 0 0 1	8/11	9	
LDS = Load Pointer to DS				
	1 1 0 0 0 1 0 1 mod reg r/m	9/12	12	
LES = Load Pointer to ES				
	1 1 0 0 0 1 0 0 mod reg r/m	9/12	12	
LFS = Load Pointer to FS				
	0 0 0 0 1 1 1 1 1 0 1 1 0 1 0 0 mod reg r/m	9/12	12	
LGS = Load Pointer to GS				
	0 0 0 0 1 1 1 1 1 0 1 1 0 1 0 1 mod reg r/m	9/12	12	
LSS = Load Pointer to SS				
	0 0 0 0 1 1 1 1 1 0 1 1 0 0 1 0 mod reg r/m	9/12	12	

Figure E-1 NS486SXF Core Processor Instruction Set Summary

Instruction	Format	Clock Count	'486 Count	Notes
CALL = Call (Intersegment)				
Direct Intersegment	1 0 0 1 1 0 1 0 unsigned full offset, selector			
to same level		14/15	20	
through Call Gate to same level		21/22	35	
to inner level, no parameters		42/44	69	
to inner level, X parameter (d) words		44+4X	77+4X	
to TSS	286 [to 486] 286 [to 286] 486 [to 486] 486 [to 286]	102/142 96/119 114/149 103/126	199 180 199 180	
through Task Gate	286 [to 486] 286 [to 286] 486 [to 486] 486 [to 286]	111/151 105/128 123/158 112/135	200 181 200 181	
Indirect Intersegment	1 1 1 1 1 1 1 1 mod 0 1 1 r/m			
to same level		14/15	20	
through Call Gate to same level		21/22	35	
to inner level, no parameters		42/44	69	
to inner level, X parameter (d) words		44+4X	77+4X	
to TSS	286 [to 486] 286 [to 286] 486 [to 486] 486 [to 286]	102/142 96/119 114/149 103/126	199 180 199 180	
through Task Gate	286 [to 486] 286 [to 286] 486 [to 486] 486 [to 286]	111/151 105/128 123/158 112/135	200 181 200 181	
RET = Return from CALL (Intersegment)				
	1 1 0 0 1 0 1 1			
to same level		12/13	17	
to outer level		25/27	37	
Adding 16-bit Immediate to SP	1 1 0 0 1 0 1 0 16-bit disp.			
to same level		12/13	18	
to outer level		25/27	36	

Figure E-1 NS486SXF Core Processor Instruction Set Summary

Instruction	Format	Clock Count	'486 Count	Notes
JUMP = Jump (Intersegment).				
Direct Intersegment /	1 1 1 0 1 0 1 0 unsigned full offset, selector			
Indirect Intersegment	1 1 1 1 1 1 1 1 mod 1 0 1 r/m			
to same level		11/12	18/19	
through Call Gate to same level		17/19	31/32	
to TSS	286 [to 486] 286 [to 286] 486 [to 486] 486 [to 286]	108/146 102/123 114/151 108/128	203/204 184/185 203/204 184/185	
through Task Gate	286 [to 486] 286 [to 286] 486 [to 486] 486 [to 286]	119/157 113/134 125/162 119/139	204/205 185/186 204/205 184/185	
BIT MANIPULATION INSTRUCTIONS				
BT / BTS / BTR / BTC = Bit Test / Bit Test and Set / Reset / Complement (Times are BT / BTS / BTR / BTC)				
reg, Immediate (imm. 8-bit data follows)	0 0 0 0 1 1 1 1 1 0 1 1 1 0 1 0 1 1 TTT reg	2 / 5 / 5 / 5	3 / 6 / 6 / 6	
Memory, Immediate (imm. 8-bit data follows)	0 0 0 0 1 1 1 1 1 0 1 1 1 0 1 0 mod TTT r/m	4 / 7 / 7 / 7	3 / 8 / 8 / 8	
reg1, reg2	0 0 0 0 1 1 1 1 1 0 TTT 0 1 1 1 1 reg2 reg1	2 / 5 / 5 / 5	3 / 6 / 6 / 6	
Memory, reg	0 0 0 0 1 1 1 1 1 0 TTT 0 1 1 mod reg r/m	13/15/15/15	8/13/13/13	
Instruction	TTT			
BT = Bit Test	1 0 0			
BTS = Bit Test and Set	1 0 1			
BTR = Bit Test and Reset	1 1 0			
BTC = Bit Test and Complement	1 1 1			
BSF = Bit Scan Forward (Times are Word / Double Word, n = 16 or 32)				
reg1, reg2	0 0 0 0 1 1 1 1 1 0 1 1 1 1 1 0 0 1 1 reg2 reg1	3/5+n	6/42	
Memory, reg	0 0 0 0 1 1 1 1 1 0 1 1 1 1 1 0 0 mod reg r/m	5/6+n	7/43	
BSR = Bit Scan Reverse (Times are Word / Double Word, n = 16 or 32)				
reg1, reg2	0 0 0 0 1 1 1 1 1 0 1 1 1 1 1 0 1 1 1 reg2 reg1	3/7+n	6/103	
Memory, reg	0 0 0 0 1 1 1 1 1 0 1 1 1 1 1 0 1 mod reg r/m	5/9+n	7/104	

Figure E-1 NS486SXF Core Processor Instruction Set Summary

Instruction	Format	Clock Count	'486 Count	Notes
STRING INSTRUCTIONS (Times are Byte-or-Word / Double Word)				
CMPS[B/W/D] = Compare Bytes / Words / Double Words				
	1 0 1 0 0 1 1 w	4	8	
LODS[B/W/D] = Load Byte / Word / Double Word to AL / AX / EAX				
	1 0 1 0 1 1 0 w	1	5	
MOVS[B/W/D] = Move Byte / Word / Double Word				
	1 0 1 0 0 1 0 w	4	7	
SCAS[B/W/D] = Scan Byte / Word / Double Word for match with AL / AX / EAX				
	1 0 1 0 1 1 1 w	3	6	
STOS[B/W/D] = Store Byte / Word / Double Word from AL / AX / EAX				
	1 0 1 0 1 0 1 w	1	5	
XLAT[B] = Translate Byte in AL				
	1 1 0 1 0 1 1 1	3	4	(byte form only)

Figure E-1 NS486SXF Core Processor Instruction Set Summary

Instruction	Format	Clock Count	'486 Count	Notes
REPEATED STRING INSTRUCTIONS				
REPE CMPS[B/W/D] = Compare String Bytes / Words / Double Words: Find Non-Match				
(C = CX or ECX)	11110011 1010011 w			
C = 0		3	5	
C > 0		3+6C	7+7C	
REPNE CMPS[B/W/D] = Compare String Bytes / Words / Double Words: Find Match				
(C = CX or ECX)	11110010 1010011 w			
C = 0		3	5	
C > 0		3+6C	7+7C	
REP LODS[B/W/D] = Load String Bytes / Words / Double Words				
(C = CX or ECX)	11110011 1010110 w			
C = 0		3	5	
C > 0		3+3C	7+4C	
REP MOVS[B/W/D] = Move String Bytes / Words / Double Words				
(C = CX or ECX)	11110011 1010010 w			
C = 0		3	5	
C = 1		8	13	
C > 0		3+5C	12+3C	
REPE SCAS[B/W/D] = Scan String Bytes / Words / Double Words: Find Non-Match with AL / AX / EAX				
(C = CX or ECX)	11110011 1010111 w			
C = 0		3	5	
C > 0		3+5C	7+5C	
REPNE SCAS[B/W/D] = Scan String Bytes / Words / Double Words: Find Match with AL / AX / EAX				
(C = CX or ECX)	11110010 1010111 w			
C = 0		3	5	
C > 0		3+5C	7+5C	

Figure E-1 NS486SXF Core Processor Instruction Set Summary

Instruction	Format	Clock Count	'486 Count	Notes		
REP STOS[B/W/D] = Store String Bytes / Words / Double Words: Fill from AL / AX / EAX						
(C = CX or ECX)	<table border="1" style="display: inline-table; vertical-align: middle;"> <tr> <td style="padding: 2px;">11110011</td> <td style="padding: 2px;">1010101w</td> </tr> </table>	11110011	1010101w			
11110011	1010101w					
C = 0		3	5			
C > 0		3+3C	7+4C			

Figure E-1 NS486SXF Core Processor Instruction Set Summary

Instruction	Format	Clock Count	'486 Count	Notes
FLAG CONTROL INSTRUCTIONS				
CLC = Clear Carry Flag	11111000	1	2	
STC = Set Carry Flag	11111001	1	2	
CMC = Complement Carry Flag	11110101	2	2	
CLD = Clear Direction Flag	11111100	1	2	
STD = Set Direction Flag	11111001	1	2	
CLI = Clear Interrupt Enable Flag	11111010	1	5	
STI = Set Interrupt Enable Flag	11111011	1	5	
LAHF = Load AH from Flags	10011111	1	3	
SAHF = Store AH into Flags	10011110	1	2	
PUSHF = Push Flags	10011100	3	3	
POPF = Pop Flags	11111101	4	6	

Figure E-1 NS486SXF Core Processor Instruction Set Summary

Instruction	Format	Clock Count	'486 Count	Notes
DECIMAL ARITHMETIC INSTRUCTIONS				
AAA = ASCII Adjust for Add				
	00110111	4/5	3	min/max
AAS = ASCII Adjust for Subtract				
	00111111	4/5	3	
AAM = ASCII Adjust for Multiply				
	11010100 00001010	7/15	15	
AAD = ASCII Adjust for Divide				
	11010101 00001010	15	14	
DAA = Decimal Adjust for Add				
	00100111	5/7	2	
DAS = Decimal Adjust for Subtract				
	00101111	5/7	2	

Figure E-1 NS486SXF Core Processor Instruction Set Summary

Instruction	Format	Clock Count	'486 Count	Notes
PROCESSOR CONTROL INSTRUCTIONS				
HLT = Halt				
	11110100	4	4	
MOV = Move To and From Control / Debug Registers				
CR0 from reg	00001111 00100010 11 000 reg	5	17	
reg from CR0	00001111 00100000 11 000 reg	1	4	
DR0—DR3 from reg	00001111 00100011 11 eee reg	4	10	
DR6—DR7 from reg	00001111 00100011 11 eee reg	4	10	
reg from DR0—DR3	00001111 00100001 11 eee reg	3	9	
reg from DR6—DR7	00001111 00100001 11 eee reg	3	9	
CLTS = Clear Task Switched Flag				
	00001111 00000110	5	7	
INVD = Invalidate Cache				
	00001111 00001000	3	4	

Figure E-1 NS486SXF Core Processor Instruction Set Summary

Instruction	Format	Clock Count	'486 Count	Notes
INSTRUCTION PREFIX BYTES				
Address Size Override Prefix	01100111	1	1	
LOCK = Lock Prefix	11110000	1	1	
Operand Size Override Prefix	01100110	1	1	
Segment Override Prefixes				
CS:	00101110	1	1	
DS:	00111110	1	1	
ES:	00100110	1	1	
FS:	01100100	1	1	
GS:	01100101	1	1	
SS:	00110110	1	1	

Figure E-1 NS486SXF Core Processor Instruction Set Summary

Instruction	Format	Clock Count	'486 Count	Notes
PROTECTION CONTROL INSTRUCTIONS				
ARPL = Adjust Requested Privilege Level				
From Registers	0 1 1 0 0 0 1 1 1 1 reg1 reg2	8/10	9	min/max
From Memory	0 1 1 0 0 0 1 1 mod reg r/m	6/8	9	
LAR = Load Access Rights				
From Registers	0 0 0 0 1 1 1 1 0 0 0 0 0 0 1 0 1 1 reg1 reg2	3/5	11	
From Memory	0 0 0 0 1 1 1 1 0 0 0 0 0 0 1 0 mod reg r/m	5/7	11	
LGDT = Load Global Descriptor Table Register				
Table Register	0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 1 mod 0 1 0 r/m	4	12	
LIDT = Load Interrupt Descriptor Table Register				
Table Register	0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 1 mod 0 1 1 r/m	4	12	
LLDT = Load Local Descriptor Table Register				
Table Register from reg	0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 0 1 1 0 1 0 reg	5/8	11	
Table Register from Memory	0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 0 mod 0 1 0 r/m	6/9	11	
LMSW = Load Machine Status Word				
From reg	0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 1 1 1 1 1 0 reg	5	13	
From Memory	0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 1 mod 1 1 0 r/m	7	13	
LSL = Load Segment Limit				
From reg	0 0 0 0 1 1 1 1 0 0 0 0 0 0 1 1 1 1 reg1 reg2	3/7	10	
From Memory	0 0 0 0 1 1 1 1 0 0 0 0 0 0 1 1 mod reg r/m	5/9	10	
LTR = Load Task Register				
From reg	0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 0 1 1 0 1 1 reg	8	20	
From Memory	0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 0 mod 0 1 1 r/m	9	20	

Figure E-1 NS486SXF Core Processor Instruction Set Summary

Instruction	Format	Clock Count	'486 Count	Notes
SGDT = Store Global Descriptor Table Register				
Table Register	00001111 00000001 mod 000 r/m	5	10	
SIDT = Store Interrupt Descriptor Table Register				
Table Register	00001111 00000001 mod 001 r/m	5	10	
SLDT = Store Local Descriptor Table Register				
To reg	00001111 00000000 11 000 reg	1	2	
To Memory	00001111 00000000 mod 000 r/m	1	3	
SMSW = Store Machine Status Word				
From reg	00001111 00000001 11 100 reg	1	2	
From Memory	00001111 00000001 mod 100 r/m	3	3	
STR = Store Task Register				
From reg	00001111 00000000 11 001 reg	1	2	
From Memory	00001111 00000000 mod 001 r/m	1	3	
VERR = Verify Read Access				
From reg	00001111 00000000 11 100 reg	3/7	11	
From Memory	00001111 00000000 mod 100 r/m	5/9	11	
VERW = Verify Write Access				
From reg	00001111 00000000 11 101 reg	3/7	11	
From Memory	00001111 00000000 mod 101 r/m	5/9	11	

Figure E-1 NS486SXF Core Processor Instruction Set Summary

Instruction	Format	Clock Count	'486 Count	Notes		
INTERRUPTS AND INTERRUPTING INSTRUCTIONS						
INT n = Interrupt Type n						
	<table border="1" style="display: inline-table;"> <tr> <td style="padding: 2px;">1 1 0 0 1 1 0 1</td> <td style="padding: 2px;">type</td> </tr> </table>	1 1 0 0 1 1 0 1	type			
1 1 0 0 1 1 0 1	type					
	same level	24/25	44			
	different level	37/39	71			
	286 [to 486]	107/147	199			
	286 [to 286]	102/124	180			
	486 [to 486]	114/154	199			
	486 [to 286]	108/131	180			
INT 3 = Interrupt Type 3						
	<table border="1" style="display: inline-table;"> <tr> <td style="padding: 2px;">1 1 0 0 1 1 0 0</td> </tr> </table>	1 1 0 0 1 1 0 0	INTn	INTn		
1 1 0 0 1 1 0 0						
INTO = Interrupt 4 if Overflow						
	<table border="1" style="display: inline-table;"> <tr> <td style="padding: 2px;">1 1 0 0 1 1 1 0</td> </tr> </table>	1 1 0 0 1 1 1 0				
1 1 0 0 1 1 1 0						
	Interrupt Taken	INTn +1	INTn +2			
	Interrupt Not Taken	3	3			
BOUND = Interrupt 5 if Detect Value Out of Range						
	<table border="1" style="display: inline-table;"> <tr> <td style="padding: 2px;">0 1 1 0 0 0 1 0</td> <td style="padding: 2px;">mod reg r/m</td> </tr> </table>	0 1 1 0 0 0 1 0	mod reg r/m			
0 1 1 0 0 0 1 0	mod reg r/m					
	If in range (no interrupt)	7	7			
	If out of range (interrupt)	INTn +11	INTn +24			
IRET = Interrupt Return						
	<table border="1" style="display: inline-table;"> <tr> <td style="padding: 2px;">1 1 0 0 1 1 1 1</td> </tr> </table>	1 1 0 0 1 1 1 1				
1 1 0 0 1 1 1 1						
	To same level	11/12	20			
	To outer level	30/32	36			
	To nested task (EFLAGS.NT = 1):					
	NS486SXF or 286 TSS to NS486SXF TSS	110/157	194			
	NS486SXF or 286 TSS to 286 TSS	104/134	175			
External Interrupt						
		INTn +6	INTn +11			
NMI = Non-Maskable Interrupt						
		INTn +5	INTn +3			

Figure E-1 NS486SXF Core Processor Instruction Set Summary

Instruction	Format	Clock Count	'486 Count	Notes
I/O INSTRUCTIONS				
(Note: Times are "CPL ≤ IOPL" / "CPL > IOPL")				
IN = Input				
From Fixed Port	1 1 1 0 0 1 0 w port number	6/19-21	9/29	
From Variable Port (number in DX)	1 1 1 0 1 1 0 w	6/19-21	8/28	
OUT = Output				
To Fixed Port	1 1 1 0 0 1 1 w port number	6/19-21	11/31	
To Variable Port (number in DX)	1 1 1 0 1 1 1 w	6/19-21	10/30	
INS[B/W/D] = Input Byte / Word / Double Word String Element from DX Port				
	0 1 1 0 1 1 0 w	9/19-24	10/32	
OUTS[B/W/D] = Output Byte / Word / Double Word String Element to DX Port				
	0 1 1 0 1 1 1 w	9/19-24	10/32	
REP INS[B/W/D] = Input String of Bytes / Words / Double Words from DX Port				
	1 1 1 1 0 0 1 1 0 1 1 0 1 1 0 w	7+6C/ 22+6C	10+8C/ 30+8C	
REP OUTS[B/W/D] = Output String of Bytes / Words / Double Words to DX Port				
	1 1 1 1 0 0 1 1 0 1 1 0 1 1 1 w	7+5C/ 22+5C	11+5C/ 31+5C	

This Page Intentionally Left Blank.

Appendix F: Glossary

Access.bus - a two-wire synchronous serial bus protocol that uses the I²C bus as its hardware base and higher level software to implement multimaster and transmitter/receiver functions between integrated circuits.

Breakpoint - the process of stopping program execution so that a debug process of register examination can take place. The breakpoint location is held in the debug register. When a memory access is made to that location, an NS486SXF processor exception occurs.

Bus - a common data channel between hardware devices. Can be serial (information travels one-bit at a time) or parallel (information travels in groups of bits at one time, each moving along one of a set of parallel paths).

Cache - Cache memory is a small amount of fast local memory used to hold the most likely to be used instructions.

Coprocessor - An auxiliary processor that is closely coupled to the CPU. Often it is a math coprocessor that can extend the processor's instruction set. This feature is not supported by the NS486SXF.

CS - Chip Select. An active low signal that indicates to an offchip peripheral that it is selected for read or write operations.

Current Privilege Level (CPL) - the privilege level of the task currently being executed by the NS486SXF processor. The CPL value is defined by the lowest two bits of the CS segment register.

DACK - DMA Acknowledge. A handshake signal between the NS486SXF DMA controller and a DMA device. When active (low), it signals the DMA controller's readiness to initiate a data transfer with the requesting device.

Debug Registers - Six NS486SXF processor registers used to specify breakpoints and to control debugging operations.

Descriptor Privilege Level (DPL) - a field in the segment descriptor that defines the privilege level for that segment.

DMA - Direct Memory Access. A technique of performing data transfers without the interaction of the processor. Transfers can be from a peripheral to memory, from memory to a peripheral or from memory to memory. The NS486SXF DMA controller takes control of the system bus, performs the data transfer and returns control of the bus to the core processor. NS486SXF supports 7 channels of DMA.

DRQ - DMA Request. A handshake signal between a peripheral and the NS486SXF DMA controller. When active (high) it signals the request for a DMA data transfer.

ECP - Extended Capabilities Port. A recently defined high speed (2 Mbytes/second and up) bidirectional parallel port standard that uses pre-existing Centronics compatible cables.

EEPROM - Electrically Erasable Programmable Read Only Memory. Nonvolatile memory that can be erased and programmed electrically, usually "in-system" (in the socket).

Exception - A control transfer mechanism initiated by either a software trap instruction, or more often, when the processor detects a hardware fault, or breakpoint. If it occurs during an instruction, the instruction cycle is aborted and the NS486SXF processor jumps to an exception processing routine.

Global Descriptor Table (GDT) - An array of 8-byte segment descriptors for programs executing in protected mode. Every NS486SXF program has a GDT because NS486SXF executes only in protected mode.

GDTR - Global Descriptor Table Register - An NS486SXF processor register that points to the location of the Global Descriptor Table.

I²C or Inter-Integrated Circuit bus - a simple two-wire synchronous serial interface used to connect two or more ICs in a system. One wire, SDA, carries the data between components and the other wire, SCL,

carries the clock timing signal. Data can be transmitted at speeds of up to 400 kilobits per second at lengths of up to 25 feet.

Interrupt Descriptor Table (IDT) - An array of 8-byte gate descriptors that can select up to 256 interrupt service routines.

IDTR - Interrupt Descriptor Table Register. An NS486SXF processor register that points to or contains the starting address location of the NS486SXF processor's interrupt descriptor table.

IRQ - Maskable Interrupt Request. An interrupt signal that can be prioritized or masked through software settings.

Local Descriptor Table (LDT) - An array of 8-byte segment descriptors for one NS486SXF program or task.

LDTR - Local Descriptor Table Register. An NS486SXF register that points to the local descriptor table for a given NS486SXF program.

MICROWIRE - A three wire synchronous serial interface for data transfers between ICs.

NMI - Non-maskable Interrupt. A high priority interrupt signal. It cannot be masked (or disabled) through software.

Parity - An extra bit used in RAM memory to maintain data security. If the parity bit does not match after a data read with the parity bit set by the original data write, the parity bit error condition is flagged (sometimes with an NMI).

PCMCIA - Personal Computer Memory Card International Association. An interface card standard that defines interface signals, card sizes and connectors for credit-card sized printed circuit boards. Originally designed for memory card makers attaching additional memory to notebook and laptop computers, it has been extended to support many types of I/O and mass storage systems.

Privilege Level - there are four privilege levels (0 - 3) for each NS486SXF processor task or program.

Segment Descriptor - An 8-byte data entry in the NS486SXF processor segment descriptor table that

includes the definition of the base address, size, type, attribute, and protection for a segment.

Segment Selector - A 16-bit offset value relative to the NS486SXF processor descriptor table (GDT or LDT) that specifies which table and the requested privilege level.

Snooping - A technique used to allow the NS486SXF cache to monitor the processor bus to detect a write to a cached location. Such a write will invalidate the cache entry, and force a memory access on the next reference to that address.

System Service Elements - include all of the programmable non-memory elements required for system level functions in an embedded operating environment. These include the DRAM controller, a DMA controller, programmable interval timers, a protected watchdog timer, a programmable interrupt controller, a real-time clock and calendar, and power management features.

UART - Universal Asynchronous Receiver Transmitter. A parallel-to-serial data converter that takes 8-bits of parallel data from the processor and converts it into a serial stream of data. It also takes an incoming stream of serial data and converts it into parallel data in 8-bit bytes.