Élan™SC400 Microcontroller Register Set

Reference Manual

Rev. A, December 1996

AMD

TABLE OF CONTENTS

INTRODUCTION

ÉLANSC400 MICROCONTROLLER

The Élan™SC400 microcontroller is the latest in a series of E86™ family microcontrollers using AMD's new Systems-in-Silicon™ design philosophy, which integrates proven x86 CPU cores with a comprehensive set of on-chip peripherals in an advanced 0.35 micron process.

The ÉlanSC400 microcontroller combines a 32-bit, low-voltage Am486® CPU with a complete set of PC/AT-compatible peripherals, along with the power management features required for battery operation. With its low-voltage Am486 CPU core and ultra-small form factor, the ÉlanSC400 microcontroller is highly optimized for mobile computing applications.

PURPOSE OF THIS MANUAL

This manual includes in reference format the complete set of configuration and control registers required to program the ÉlanSC400 microcontroller.

Intended Audience

This reference manual is intended primarily for programmers who are developing code for the ÉlanSC400 microcontroller. Computer software and hardware architects and system engineers who are designing or are considering designing systems based on the ÉlanSC400 microcontroller may also be interested in the information contained in this document. For more information on using the ÉlanSC400 microcontroller, see the *ÉlanSC400 Microcontroller User's Manual* (order #21030).

Overview of This Manual

This manual is organized into the following chapters.

- Chapter 1 contains **an overview of the configuration registers** on the ÉlanSC400 microcontroller.
- Chapter 2 includes descriptions for all of the **direct-mapped registers**.
- Chapter 3 includes descriptions for the **indexed ÉlanSC400 Chip Setup and Control (CSC) registers**
- Chapter 4 includes descriptions for the **indexed Real-Time Clock and CMOS RAM registers**.
- Chapter 5 includes descriptions for the **indexed LCD Graphics Controller registers**.
- Chapter 6 includes descriptions for the **indexed PC Card Controller registers**.

Within each chapter, the registers are listed in ascending hexadecimal order.

RELATED DOCUMENTS

AMD Documentation

The following AMD documents provide additional information about the ÉlanSC400 microcontroller.

- The *ÉlanSC400 Microcontroller Data Sheet* (order #21028) includes complete pin lists, pin state tables, timing and thermal characteristics, and package dimensions for the ÉlanSC400 microcontroller.
- The *ÉlanSC400 Microcontroller User's Manual* (order #21030) provides a functional description of the microcontroller for both hardware and software designers.
- The *Am486 Microprocessor Software User's Manual* (order #18497) includes the Am486 microprocessor instruction set. Appendices provide useful information about programming the base architecture and system level registers, as well as describing segmentation and paging on the Am486 microprocessor. A glossary of terms is also included. Note that this document describes floating-point features not supported on the ÉlanSC400 microcontroller.

Other documents of interest:

- *Enhanced Am486 Microprocessor Data Sheet* (order #19225)
- *Am486DX/DX2 Microprocessor Hardware Reference Manual* (order #17965). Note that this document describes floating-point features not supported on the ÉlanSC400 microcontroller.

DOCUMENTATION CONVENTIONS

The following table lists the documentation conventions used throughout this manual.

Documentation Conventions Table

Notation	Meaning
Register Descriptions	
Default	Power-on reset value or value after master reset asserted
x in default register value	Non-deterministic or floating; no value is guaranteed
? in default register value	Determined by sources external to the ÉlanSC400 microcontroller
Shading in PC Card index register bit description	Deviates from strict 83865SL compliance
Reference Notation	
CSC index 00h[1]	ÉlanSC400 Chip Setup and Control (CSC) indexed register 00h, bit 1
Graphics index 00h[1]	Graphics controller indexed register 00h, bit 1
PC Card index 00h[1]	PC Card controller indexed register 00h, bit 1
Port 00h[1]	Direct-mapped register 00h, bit 1
RTC index 00h[1]	RTC and configuration RAM indexed register 00h, bit 1
Pin Naming	
/	Two functions available on the pin at the same time
{ }	Pin function during hardware reset
[ ]	Alternative pin function selected by firmware configuration
[[ ]]	Alternative pin function selected by a hardware configuration pin state at power-on reset
$\overline{\text{ROMCS2}}$–$\overline{\text{ROMCS0}}$	All three ROM chip select signals
$\overline{\text{ROMCSx}}$	Any of the three ROM chip select signals
Numbers	
b	Binary number
d	Decimal number Decimal is the default radix
h	Hexadecimal number
x in register address	Any of several legal values; e.g., 3x4h as a graphics index address register can be either 3B4h or 3D4h, depending on the mode selected

Notation	Meaning
[X–Y, Z]	The bit field that consists of bits X through Y, and the bit field consisting of the single bit Z. Example: Use CSC index 52h[5–3,1]
General	
field	Bit field in a register (one or more consecutive and related bits)
can	It is possible to perform an action if properly configured
will	A certain action is going to occur
XMI	SMI or NMI
Set 29h[1]	Write bit 1 of index 29h to 1. ***Note:*** *The applicable indexed register space will either be obvious from the surrounding text, or will be stated explicitly. For example, RTC index 0h[1] would be a reference to index 1 in Real-time Clock indexed register space.*
Clear 29h[1]	Write bit 1 of index 29h to 0. ***Note:*** *The applicable indexed register space will either be obvious from the surrounding text, or will be stated explicitly. For example, RTC index 0h[1] would be a reference to index 1 in Real-time Clock indexed register space.*

CHAPTER

1 OVERVIEW

This chapter provides an overview of the different types of configuration registers that are documented in this manual.

1.1 CONFIGURATION REGISTERS

Configuration registers are used to read back status or to control various aspects of the ÉlanSC400 microcontroller's on-board cores or peripherals. The internal configuration registers on the ÉlanSC400 microcontroller fall into one of five categories:

- Direct-mapped PC/AT-compatible I/O registers
- ÉlanSC400 Chip Setup and Control (CSC) indexed registers
- Real-Time Clock (RTC) and CMOS RAM indexed registers
- LCD graphics controller indexed registers
- PC Card controller indexed registers

1.1.1 Direct-Mapped Configuration Registers

Direct-mapped PC/AT-compatible I/O registers include those for the typical PC/AT cores, such as the DMA controllers, programmable interval timer, prioritized interrupt controllers, parallel port, and serial port. The registers in this group include the industry-standard list of registers for IBM PC/AT-compatible computers which have been implemented in the ÉlanSC400. A summary listing of these standard I/O port addresses is shown in Table 1-1.

Table 1-1 Internal I/O Port Address Map Summary

Internal I/O Device	I/O Address Range
Slave DMA (DMA1)	0000–000Fh
Master Programmable Interrupt Controller (PIC)	0020–0021h
CSC Index, Data	0022h, 0023h
Programmable Interval Timer (PIT)	0040–0043h
Keyboard	0060h, 0064h
System Control Port B/NMI Status	0061h
RTC Index, Data	0070h, 0071h
General 8x Registers	0080h, 0084–0086h, 0088h, 008C–008Fh
DMA Page Registers	0081–0083h, 0087h, 0089–008Bh
System Control Port A	0092h
Slave PIC	00A0–00A1h
Master DMA (DMA0)	00C0–00DEh (even addresses only)
Alternate A20 Gate Control	00EEh
Alternate CPU Reset Control	00EFh
Parallel Port LPT2	0278–027Fh
Serial Port COM2	02F8–02FFh
Parallel Port LPT1	0378–037Fh
MDA Graphics Index, Data	03B4h, 03B5h
CGA Graphics Index, Data	03D4h, 03D5h
PC Card Index, Data	03E0h, 03E1h
Serial Port COM1	03F8–03FFh

Note: *DMA Page register extension bits are found in Chip Setup and Control (CSC) indexed registers D9h and DAh.*

1.1.2 Indexed Configuration Registers

Four additional groups of configuration registers are indirectly accessible to the programmer by using pairs of direct-mapped I/O ports. The four additional groups of registers available on the ÉlanSC400 microcontroller are illustrated in Figure 1-1.

All of the registers accessed through this mechanism are referred to as “indexed.” Indexing uses direct-mapped I/O index and data ports to expand the I/O space for reading and writing internal system registers.

- An I/O write to one of the index registers latches the index number of the register to be indirectly accessed.
- A subsequent I/O write to the corresponding data port will write the register indexed by the index register. Similarly, an I/O read from data port will read the register indexed by the index register.
- A read from an index register provides the last index value written to that internal index latch.

An example of using indexing to access the ÉlanSC400 Chip Setup and Control (CSC) registers is shown in Figure 1-2.

Figure 1-1 Indexed Configuration Register Space

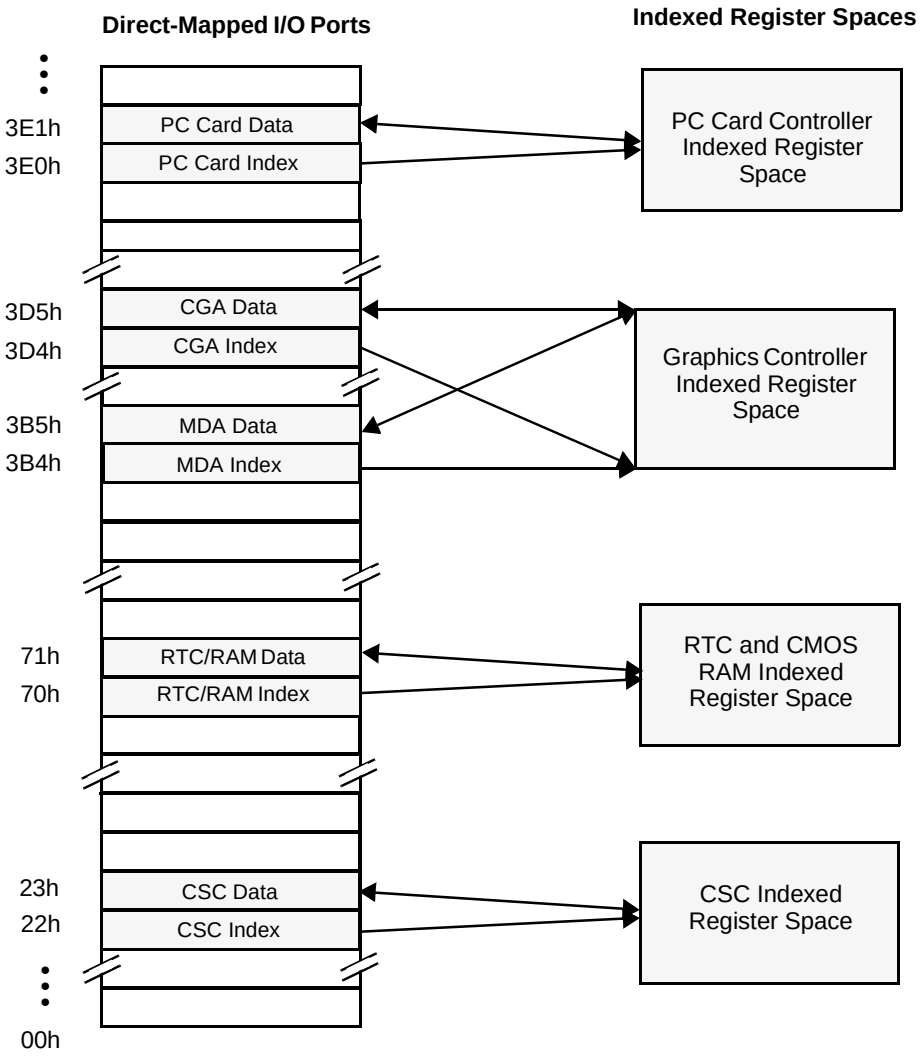
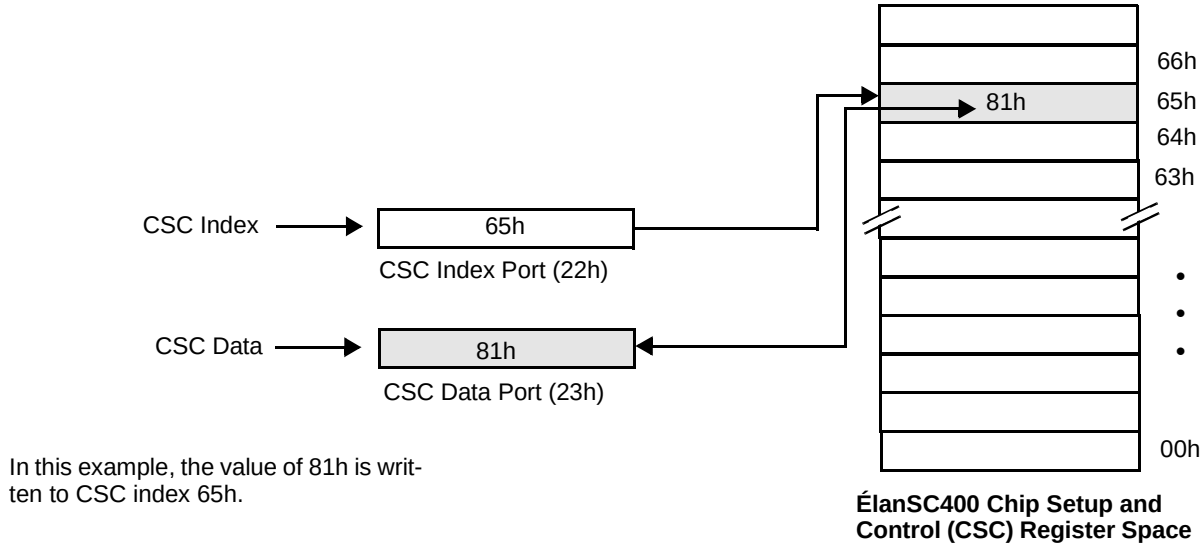


Figure 1-2 Using the Index and Data I/O Ports to Access CSC Register Space



In this example, the value of 81h is written to CSC index 65h.

1.1.2.1 ÉlanSC400 Chip Setup and Control (CSC) Indexed Registers

ÉlanSC400 Chip Setup and Control (CSC) registers are defined as ÉlanSC400 microcontroller-specific registers which support features beyond standard PC/AT compatibility requirements (i.e., all memory controller and power management registers are CSC indexed registers). These registers are accessed through an indexing scheme to limit the number of direct-mapped I/O ports required.

To access the CSC registers, an I/O write to I/O port 0022h is first performed. The data written is the index of the CSC register. This I/O write is followed by an I/O read or write to port 0023h to access the data from the selected register.

The ÉlanSC400 microcontroller does not implement any locking mechanism for CSC register access. Also, back-to-back access of I/O address 0022h/0023h is not required for access to the CSC indexed registers. For example, the following code fragment:

```
mov al, 90h; force an SMI
out 22h, al
mov al, 1
out 23h, al
   :
   :
```

has the same result as this code fragment:

```
mov AX, 0190h; force an SMI.
out 22h, AX
```

1.1.2.2 RTC and CMOS RAM Indexed Registers

Real-Time Clock and CMOS RAM indexed registers are accessed using I/O ports 70h (index) and 71h (data). These registers function as setup, control, and status for the RTC, as well as user CMOS RAM locations.

1.1.2.3 Graphics Controller Indexed Registers

Graphics controller indexed registers are accessed using I/O ports 3D4h (index) and 3D5h (data) for CGA mode and I/O ports 3B4h (index) and 3B5h (data) for MDA mode. These registers function as setup, control, and status for the LCD graphics controller.

1.1.2.4 PC Card Controller Indexed Registers

PC Card controller indexed registers are accessed using I/O ports 3E0h (index) and 3E1h (data). These registers function as setup, control, and status for the PC Card controller.

CHAPTER

2 PC/AT-COMPATIBLE DIRECT-MAPPED REGISTERS

2.1 OVERVIEW

This chapter describes the direct-mapped registers on the ÉlanSC400 microcontroller. These registers include those for the typical PC/AT cores, such as the DMA controllers, programmable interval timer, programmable interrupt controllers, parallel port, and serial port. The registers in this group include those PC/AT compatible I/O ports that have been implemented in the ÉlanSC400 microcontroller. They are listed in hexadecimal order in Table 2-1.

The registers in this chapter are all addressed directly; no indexing is required. Other controls that relate to the PC/AT legacy core functions can be found in the Chip Setup and Control (CSC) registers found in Chapter 3.

Table 2-1 PC/AT-Compatible Direct-Mapped Register Map

Register Name	I/O (Port) Address	Page Number
Slave DMA Channel 0 Memory Address Register	0000h	page 2-6
Slave DMA Channel 0 Transfer Count Register	0001h	page 2-7
Slave DMA Channel 1 Memory Address Register	0002h	page 2-8
Slave DMA Channel 1 Transfer Count Register	0003h	page 2-9
Slave DMA Channel 2 Memory Address Register	0004h	page 2-10
Slave DMA Channel 2 Transfer Count Register	0005h	page 2-11
Slave DMA Channel 3 Memory Address Register	0006h	page 2-12
Slave DMA Channel 3 Transfer Count Register	0007h	page 2-13
Slave DMA Status Register for Channels 0–3	0008h	page 2-14
Slave DMA Control Register for Channels 0–3	0008h	page 2-15
Slave Software DRQ(n) Request Register	0009h	page 2-17
Slave DMA Mask Register Channels 0–3	000Ah	page 2-18
Slave DMA Mode Register Channels 0–3	000Bh	page 2-19
Slave DMA Clear Byte Pointer Register	000Ch	page 2-20
Slave DMA Controller Reset Register	000Dh	page 2-21
Slave DMA Controller Temporary Register	000Dh	page 2-22
Slave DMA Reset Mask Register	000Eh	page 2-23
Slave DMA General Mask Register	000Fh	page 2-24
Master 8259 Interrupt Request Register	0020h	page 2-25
Master 8259 In-Service Register	0020h	page 2-26
Master 8259 Initialization Control Word 1 Register	0020h	page 2-27

Register Name	I/O (Port) Address	Page Number
Master 8259 Operation Control Word 2 Register	0020h	page 2-28
Master 8259 Operation Control Word 3 Register	0020h	page 2-29
Master 8259 Initialization Control Word 2 Register	0021h	page 2-30
Master 8259 Initialization Control Word 3 Register	0021h	page 2-31
Master 8259 Initialization Control Word 1 Register	0021h	page 2-27
Master 8259 Interrupt Mask Register (also known as Master 8259 Operation Control Word 1 Register)	0021h	page 2-33
ÉlanSC400 Microcontroller Chip Setup and Control (CSC) Index Register	0022h	page 2-34
ÉlanSC400 Microcontroller Chip Setup and Control (CSC) Data Port	0023h	page 2-35
Programmable Interval Timer #1 Channel 0 Count Register	0040h	page 2-36
Programmable Interval Timer #1 Channel 1 Count Register	0041h	page 2-37
Programmable Interval Timer #1 Channel 2 Count Register	0042h	page 2-38
Programmable Interval Timer #1 Status Byte Format	0040–0042h	page 2-39
Programmable Interval Timer #1 Mode Control Register (Mode selection)	0043h	page 2-41
Programmable Interval Timer #1 Mode Control Register (Counter latch)	0043h	page 2-43
Programmable Interval Timer #1 Mode Control Register (Read-back)	0043h	page 2-44
Keyboard/Mouse Interface Output Buffer	0060h	page 2-45
PC/AT Keyboard Interface Data Register	0060h	page 2-46
XT Keyboard Data Register	0060h	page 2-47
System Control Port B/ NMI Status Register	0061h	page 2-48
Keyboard/Mouse Interface Status Register	0064h	page 2-49
Keyboard/Mouse Interface Command Register	0064h	page 2-51
RTC/CMOS RAM Index Register	0070h	page 2-52
RTC/CMOS RAM Data Port	0071h	page 2-53
General Register	0080h	page 2-54
DMA Channel 2 Page Register	0081h	page 2-55
DMA Channel 3 Page Register	0082h	page 2-56
DMA Channel 1 Page Register	0083h	page 2-57
General Registers	0084–0086h	page 2-58
DMA Channel 0 Page Register	0087h	page 2-61
General Register	0088h	page 2-62
DMA Channel 6 Page Register	0089h	page 2-63

Register Name	I/O (Port) Address	Page Number
DMA Channel 7 Page Register	008Ah	page 2-64
DMA Channel 5 Page Register	008Bh	page 2-65
General Registers	008C–008Fh	pages 2-66–2-69
System Control Port A Register (PS/2 compatibility port)	0092h	page 2-70
Slave 8259 Interrupt Request Register	00A0h	page 2-71
Slave 8259 In-Service Register	00A0h	page 2-72
Slave 8259 Initialization Control Word 1 Register	00A0h	page 2-73
Slave 8259 Operation Control Word 2 Register	00A0h	page 2-75
Slave 8259 Operation Control Word 3 Register	00A0h	page 2-76
Slave 8259 Initialization Control Word 2 Register	00A1h	page 2-77
Slave 8259 Initialization Control Word 3 Register	00A1h	page 2-76
Slave 8259 Initialization Control Word 4 Register	00A1h	page 2-79
Slave 8259 Interrupt Mask Register (AKA Operation Control Word 1)	00A1h	page 2-80
Master DMA Channel 4 Memory Address Register	00C0h	page 2-81
Master DMA Channel 4 Transfer Count Register	00C2h	page 2-82
Master DMA Channel 5 Memory Address Register	00C4h	page 2-83
Master DMA Channel 5 Transfer Count Register	00C6h	page 2-84
Master DMA Channel 6 Memory Address Register	00C8h	page 2-85
Master DMA Channel 6 Transfer Count Register	00CAh	page 2-86
Master DMA Channel 7 Memory Address Register	00CCh	page 2-87
Master DMA Channel 7 Transfer Count Register	00CEh	page 2-88
Master DMA Status Register for Channels 4–7	00D0h	page 2-89
Master DMA Control Register for Channels 4–7	00D0h	page 2-90
Master Software DRQ(n) Request Register	00D2h	page 2-92
Master DMA Mask Register Channels 4–7	00D4h	page 2-93
Master DMA Mode Register Channels 4–7	00D6h	page 2-94
Master DMA Clear Byte Pointer Register	00D8h	page 2-95
Master DMA Controller Reset Register	00DAh	page 2-96
Master DMA Controller Temporary Register	00DAh	page 2-97
Master DMA Reset Mask Register	00DCh	page 2-98
Master DMA General Mask Register	00DEh	page 2-99
Alternate Gate A20 Control Port	00EEh	page 2-100
Alternate CPU Reset Control Port	00EFh	page 2-101
Parallel Port 2 Data Register	0278h	page 2-102
Parallel Port 2 Status Register(PC/AT Compatible mode)	0279h	page 2-103

Register Name	I/O (Port) Address	Page Number
Parallel Port 2 Status Register (Bidirectional mode)	0279h	page 2-104
Parallel Port 2 Status Register (EPP mode)	0279h	page 2-105
Parallel Port 2 Control Register	027Ah	page 2-106
Parallel Port 2 EPP Address Register	027Bh	page 2-107
Parallel Port 2 EPP 32-bit Data Register	027C–027Fh	page 2-108
COM2 Transmit Holding Register (When 02FB[7] = 0, (COM2 DLAB=0)	02F8h	page 2-109
COM2 Receive Buffer Register (When 02FB[7] = 0, COM2 DLAB = 0)	02F8h	page 2-110
COM2 Baud Clock Divisor Latch LSB (When 02FB[7] = 1, COM2 DLAB =1)	02F8h	page 2-111
COM2 Baud Clock Divisor Latch MSB (When 02FB[7] = 1, COM2 DLAB =1)	02F9h	page 2-112
COM2 Interrupt Enable Register (When 02FB[7] = 0, COM2 DLAB = 0)	02F9h	page 2-113
COM2 Interrupt ID Register	02FAh	page 2-114
COM2 FIFO Control Register	02FAh	page 2-116
COM2 Line Control Register	02FBh	page 2-117
COM2 Modem Control Register	02FCh	page 2-118
COM2 Line Status Register	02FDh	page 2-119
COM2 Modem Status Register	02FEh	page 2-121
COM2 Scratch Pad Register	02FFh	page 2-122
Parallel Port 1 Data Register	0378h	page 2-123
Parallel Port 1 Status Register (PC/AT Compatible mode)	0379h	page 2-124
Parallel Port 1 Status Register (Bidirectional) mode)	0379h	page 2-125
Parallel Port 1 Status Register (EPP mode)	0379h	page 2-126
Parallel Port 1 Control Register	037Ah	page 2-127
Parallel Port 1 EPP Address Register	037Bh	page 2-128
Parallel Port 1 EPP 32-bit Data Register	037C–037Fh	page 2-129
MDA/HGA Index Register	03B4h	page 2-130
MDA/HGA Data Port	03B5h	page 2-131
MDA/HGA Mode Control Register	03B8h	page 2-132
MDA/HGA Status Register	03BAh	page 2-133
HGA Configuration Register	3BFh	page 2-134
CGA Index Register	03D4h	page 2-135
CGA Data Port	03D5h	page 2-136
CGA Mode Control Register	03D8h	page 2-137
CGA Color Select Register	03D9h	page 2-138

Register Name	I/O (Port) Address	Page Number
CGA Status Register	03DAh	page 2-139
Primary 82365-Compatible PC Card Controller Index Register	03E0h	page 2-140
Primary 82365-Compatible PC Card Controller Data Port	03E1h	page 2-141
COM1 Transmit Holding Register (When 03FB[7] = 0, (COM1 DLAB =0)	03F8h	page 2-142
COM1 Receive Buffer Register (When 03FB[7] = 0, COM1 DLAB = 0)	03F8h	page 2-143
COM1 Baud Clock Divisor Latch LSB (When 03FB[7] =1, COM1 DLAB =1)	03F8h	page 2-144
COM1 Baud Clock Divisor Latch MSB (When 03FB[7] = 1, COM1 DLAB = 1)	03F9h	page 2-145
COM1 Interrupt Enable Register (When 03FB[7] = 0, COM1 DLAB = 0)	03F9h	page 2-146
COM1 Interrupt ID Register	03FAh	page 2-147
COM1 FIFO Control Register	03FAh	page 2-148
COM1 Line Control Register	03FBh	page 2-149
COM1 Modem Control Register	03FCh	page 2-150
COM1 Line Status Register	03FDh	page 2-151
COM1 Modem Status Register	03FDh	page 2-153
COM1 Scratch Pad Register	03FFh	page 2-154

2.2 REGISTER DESCRIPTIONS

Each direct-mapped PC/AT Compatible register is described on the following pages. Additional information about using these registers to program the ÉlanSC400 microcontroller can be found in the *ÉlanSC400 User's Manual* (order #21030).

	7	6	5	4	3	2	1	0
Bit	DMA0MAR[15–0]							
Default	x	x	x	x	x	x	x	x
R/W	R/W							

Bit	Name	Function
7–0	DMA0MAR[15–0]	**Lower 16 Bits of DMA Channel 0 Memory Address** This register is written/read via two successive I/O accesses to/from this port (always low byte followed by high byte immediately following a reset of the byte pointer at direct-mapped I/O address 0Ch). Use in conjunction with DMA Page Register for Channel 0 (I/O 87h) to form 24-bit memory address.

Programming Notes

The ÉlanSC400 microcontroller is capable of performing 26-bit DMA accesses using the extended DMA page registers that reside in the indexed ÉlanSC400 microcontroller registers D9h and DAh.

	7	6	5	4	3	2	1	0
Bit	DMA0TC[15–0]							
Default	x	x	x	x	x	x	x	x
R/W	R/W							

Bit	Name	Function
7–0	DMA0TC[15–0]	**DMA Channel 0 Transfer Count (16-Bit Register)** This register is written/read via two successive I/O accesses to/from this port (always low byte followed by high byte immediately following a reset of the byte pointer at direct-mapped I/O address 0Ch). The actual number of transfers will be one more than specified by this register.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	DMA1MAR[15–0]							
Default	x	x	x	x	x	x	x	x
R/W	R/W							

Bit	Name	Function
7–0	DMA1MAR[15–0]	**Lower 16 Bits of DMA Channel 1 Memory Address** This register is written/read via two successive I/O accesses to/from this port (always low byte followed by high byte immediately following a reset of the byte pointer at direct-mapped I/O address 0Ch). Use in conjunction with DMA Page Register for Channel 1 (I/O 83h) to form 24-bit memory address.

Programming Notes

The ÉlanSC400 microcontroller is capable of performing 26-bit DMA accesses using the extended DMA page registers that reside in the indexed ÉlanSC400 microcontroller registers D9h and DAh.

	7	6	5	4	3	2	1	0
Bit	DMA1TC[15–0]							
Default	x	x	x	x	x	x	x	x
R/W	R/W							

Bit	Name	Function
7–0	DMA1TC[15–0]	**DMA Channel 1 Transfer Count (16-Bit Register)** This register is written/read via two successive I/O accesses to/from this port (always low byte followed by high byte immediately following a reset of the byte pointer at direct-mapped I/O address 0Ch.The actual number of transfers will be one more than specified by this register.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	DMA2MAR[15–0]							
Default	x	x	x	x	x	x	x	x
R/W	R/W							

Bit	Name	Function
7–0	DMA2MAR[15–0]	**Lower 16 Bits of DMA Channel 2 Memory Address** This register is written/read via two successive I/O accesses to/from this port (always low byte followed by high byte immediately following a reset of the byte pointer at direct-mapped I/O address 0Ch). Use in conjunction with DMA Page Register for Channel 2 (I/O 81h) to form 24-bit memory address.

Programming Notes

The ÉlanSC400 microcontroller is capable of performing 26-bit DMA accesses using the extended DMA page registers that reside in the indexed ÉlanSC400 microcontroller registers D9h and DAh.

	7	6	5	4	3	2	1	0
Bit	DMA2TC[15–0]							
Default	x	x	x	x	x	x	x	x
R/W	R/W							

Bit	Name	Function
7–0	DMA2TC[15–0]	**DMA Channel 2 Transfer Count (16-Bit Register)** This register is written/read via two successive I/O accesses to/from this port (always low byte followed by high byte immediately following a reset of the byte pointer at direct-mapped I/O address 0Ch). The actual number of transfers will be one more than specified by this register.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	DMA3MAR[15–0]							
Default	x	x	x	x	x	x	x	x
R/W	R/W							

Bit	Name	R/W	Function
7–0	DMA3MAR[15–0]	R/W	**Lower 16 Bits of DMA Channel 3 Memory Address** This register is written/read via two successive I/O accesses to/from this port (always low byte followed by high byte immediately following a reset of the byte pointer at direct-mapped I/O address 0Ch). Use in conjunction with DMA Page Register for Channel 3 (I/O 82h) to form 24-bit memory address.

Programming Notes

The ÉlanSC400 microcontroller is capable of performing 26-bit DMA accesses using the extended DMA page registers that reside in the indexed ÉlanSC400 microcontroller registers D9h and DAh.

	7	6	5	4	3	2	1	0
Bit	DMA3TC[15–0]							
Default	x	x	x	x	x	x	x	x
R/W	R/W							

Bit	Name	Function
7–0	DMA3TC[15–0]	**DMA Channel 3 Transfer Count (16-Bit Register)** This register is written/read as two successive bytes. The actual number of transfers will be one more than specified by this register.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	DMAR3	DMAR2	DMAR1	DMAR0	TC3	TC2	TC1	TC0
Default	0	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R	R

Bit	Name	Function
7	DMAR3	**Channel 3 DMA Request** 0 = Channel 3 DMA request not pending 1 = Channel 3 DMA request pending
6	DMAR2	**Channel 2 DMA Request** 0 = Channel 2 DMA request not pending 1 = Channel 2 DMA request pending
5	DMAR1	**Channel 1 DMA Request** 0 = Channel 1 DMA request not pending 1 = Channel 1 DMA request pending
4	DMAR0	**Channel 0 DMA Request** 0 = Channel 0 DMA request not pending 1 = Channel 0 DMA request pending
3	TC3	**Channel 3 Terminal Count** 0 = Channel 3 terminal count not detected 1 = Channel 3 terminal count detected
2	TC2	**Channel 2 Terminal Count** 0 = Channel 2 terminal count not detected 1 = Channel 2 terminal count detected
1	TC1	**Channel 1 Terminal Count** 0 = Channel 1 terminal count not detected 1 = Channel 1 terminal count detected
0	TC0	**Channel 0 Terminal Count** 0 = Channel 0 terminal count not detected 1 = Channel 0 terminal count detected

Programming Notes

Bits 3–0 of this register are read/reset. Any read from this direct-mapped port clears bits 3–0.

	7	6	5	4	3	2	1	0
Bit	DAKSEN	DRQSEN	WRTSEL	PRITYPE	COMPTIM	ENADMA	ADRHEN	MEM2MEM
Default	0	0	0	0	0	0	0	0
R/W	W	W	W	W	W	W	W	W

Bit	Name	Function
7	DAKSEN	**$\overline{\text{DACK}}$(n) Sense** This bit controls the polarity of all $\overline{\text{DACK}}$ outputs from the slave DMA controller: 0 = Asserted Low 1 = Asserted High System logic external to the DMA controller expects the DMA controller to drive active Low $\overline{\text{DACK}}$ outputs. This bit must be written to ‘0b’ for proper system operation.
6	DRQSEN	**DREQ(n) Sense** This bit controls the polarity of all DREQ inputs to the slave DMA controller: 0 = Asserted High 1 = Asserted Low System logic external to the DMA controller expects the DMA controller to respond to active High DREQ inputs. This bit must be written to ‘0b’ for proper system operation.
5	WRTSEL	**Write Selection Control** 0 = Late write selection Any DMA channel that is routed to the ÉlanSC400 IrDA controller must have this bit cleared. 1 = Extended (early) write selection Enabling this feature will result in timing changes on the ISA bus that can violate the ISA specification.
4	PRITYPE	**Priority Type** 0 = Fixed priority 1 = Rotating priority
3	COMPTIM	**Compressed Timing** 0 = Normal timing 1 = Compressed timing Enabling this feature will result in timing changes on the ISA bus that can violate the ISA specification.
2	ENADMA	**Enable DMA Controller** 0 = Enabled 1 = DMA requests are ignored but DMA registers are available to the CPU The DMA controller should be disabled prior to programming it in order to prevent unintended transfers from occurring during the DMA controller programming operation. If an I/O DMA initiator asserts DREQ while the DMA controller is disable via this bit, abnormal system operation can occur.

Bit	Name	Function
1	ADRHEN	**Enable Channel 0 Address Hold Control** IF bit 0 = 1, then: 0 = Disabled, Channel 0 memory address changes for each memory-to-memory transfer 1 = Enabled, Channel 0 memory address does not change for each memory-to-memory transfer (not supported by the system) ELSE this bit does nothing. Since bit 0 should always be written to 0, this bit will be a don't care after the DMA controller has been initialized.
0	MEM2MEM	**Enable Memory-to-Memory Transfer** 0 = Disabled 1 = Enabled (not supported by the system) Memory-to-memory DMA support is not provided in a PC/AT Compatible system. This bit should always be written to 0.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved					REQDMA	REQSEL1 REQSEL0	
Default	x	x	x	x	x	x	x	x
R/W						W	W	

Bit	Name	Function
7–3	Reserved	**Reserved** Software should write these bits to 0.
2	REQDMA	**Software DMA Request** 0 = Reset request bit for channel selected by bits 1–0 1 = Set request bit for channel selected by bits 1–0
1–0	REQSEL1 REQSEL0	**DMA Channel Select** Bits 1–0 select which DMA channel will latch bit 2 internally to assert or deassert a DMA request via software: 0 0 = Set/reset DMA Channel 0 internal DMA request per the REQDMA bit 0 1 = Set/reset DMA Channel 1 internal DMA request per the REQDMA bit 1 0 = Set/reset DMA Channel 2 internal DMA request per the REQDMA bit 1 1 = Set/reset DMA Channel 3 internal DMA request per the REQDMA bit

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved					CHMASK	MSKSEL1 MSKSEL0	
Default	x	x	x	x	x	x	x	x
R/W						W	W	

Bit	Name	Function
7–3	Reserved	**Reserved** Software should write these bits to 0.
2	CHMASK	**DMA Channel Mask** 0 = Clear mask bit for channel selected by bits 1–0 1 = Set mask bit for channel selected by bits 1–0
1–0	MSKSEL1 MSKSEL0	**DMA Channel Mask Select** Bits 1–0 select which DMA channel will latch bit 2 internally to mask or unmask the DRQ signal into the specified DMA channel: 0 0 = Mask/unmask DMA Channel 0 mask per the CHMASK bit 0 1 = Mask/unmask DMA Channel 1 mask per the CHMASK bit 1 0 = Mask/unmask DMA Channel 2 mask per the CHMASK bit 1 1 = Mask/unmask DMA Channel 3 mask per the CHMASK bit

Programming Notes

The same DMA channel masks can be controlled via DMA registers 0Ah, 0Eh, and 0Fh.

	7	6	5	4	3	2	1	0
Bit	TRNMOD		ADDDEC	AINIT	OPSEL1 OPSEL0		MODSEL1 MODSEL0	
Default	x	x	x	x	x	x	x	x
R/W	W		W	W	W		W	

Bit	Name	Function
7–6	TRNMOD	**Transfer Mode** 0 0 = Demand transfer mode 0 1 = Single transfer mode 1 0 = Block transfer mode 1 1 = Cascade mode
5	ADDDEC	**Address Decrement** 0 = Increment the DMA memory address after each transfer 1 = Decrement the DMA memory address after each transfer
4	AINIT	**Automatic Initialization Control** If enabled, the base address and transfer count registers are restored to the values they contained prior to performing the last DMA transfer. The channel selected by bits 0–1 of this register is then ready to perform another DMA transfer without processor intervention as soon as the next DRQ is detected. 0 = Automatic initialization disabled 1 = Automatic initialization enabled
3–2	OPSEL1 OPSEL0	**Operation Select** 0 0 = Verify mode DMA controller acts normally except that no I/O or memory commands are generated, and no data is transferred 0 1 = Write transfer Data will be transferred from a DMA-capable I/O device into system memory 1 0 = Read transfer Data will be transferred from system memory to a DMA-capable I/O device 1 1 = Reserved
1–0	MODSEL1 MODSEL0	**DMA Channel Select** Bits 7–2 of this register are latched internally for each channel. Bits 0–1 determine which of the channels will be programmed by the write to port 0Bh as follows: 0 0 = Select Channel 0 0 1 = Select Channel 1 1 0 = Select Channel 2 1 1 = Select Channel 3

Programming Notes

	7	6	5	4	3	2	1	0
Bit	SLAVE_CPB							
Default	x	x	x	x	x	x	x	x
R/W	W							

Bit	Name	Function
7–0	SLAVE_CPB	**Slave DMA Clear Byte Pointer** The DMA Controller contains some 16-bit registers which are accessed by writing or reading consecutive 8-bit values to the same direct-mapped I/O location. A single byte pointer is used across the slave DMA Controller to determine which byte will be accessed. Any access to one of these 16-bit registers toggles the byte pointer between the low and high bytes. A write of any data to I/O address 0Ch clears this pointer so that the next access will be to the low byte.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	SLAVE_RST							
Default	0	0	0	0	0	0	0	0
R/W	W							

Bit	Name	Function
7–0	SLAVE_RST	**Slave DMA Controller Reset** A write of any data to this address resets the DMA Controller to the same state as a hardware reset.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	SLAVE_TMP							
Default	0	0	0	0	0	0	0	0
R/W	R							

Bit	Name	Function
7–0	SLAVE_TMP	**Slave DMA Controller Temporary Register** Used as a temporary storage buffer when doing memory-to-memory DMA. Memory-to-memory DMA transfer is not supported, and this register was included for compatibility reasons only.

Programming Notes

The same DMA channel masks can be controlled via DMA registers 0Ah, 0Eh, and 0Fh.

	7	6	5	4	3	2	1	0
Bit	SLAVE_MSK_RST							
Default	x	x	x	x	x	x	x	x
R/W	W							

Bit	Name	Function
7–0	SLAVE_MSK_RST	**Slave DMA Reset Mask** Writing any data to this I/O address resets the Mask Register, thereby activating the four slave DMA channels.

Programming Notes

The same DMA channel mask can be controlled via DMA register 0Ah, 0Eh, and 0Fh.

	7	6	5	4	3	2	1	0
Bit	Reserved				DIS3	DIS2	DIS1	DIS0
Default	0	0	0	0	1	1	1	1
R/W					W	W	W	W

Bit	Name	Function
7–4	Reserved	**Reserved** Software should write these bits to 0.
3	DIS3	**DMA Channel 3 Mask** 0 = Enable DMA Channel 3 for servicing DMA requests 1 = Disable DMA Channel 3 from servicing DMA requests
2	DIS2	**DMA Channel 2 Mask** 0 = Enable DMA Channel 2 for servicing DMA requests 1 = Disable DMA Channel 2 from servicing DMA requests
1	DIS1	**DMA Channel 1 Mask** 0 = Enable DMA Channel 1 for servicing DMA requests 1 = Disable DMA Channel 1 from servicing DMA requests
0	DIS0	**DMA Channel 0 Mask** 0 = Enable DMA Channel 0 for servicing DMA requests 1 = Disable DMA Channel 0 from servicing DMA requests

Programming Notes

The same DMA channel masks can be controlled via DMA registers 0Ah, 0Eh, and 0Fh.

	7	6	5	4	3	2	1	0
Bit	IR7	IR6	IR5	IR4	IR3	IR2	IR1	IR0
Default	x	x	x	x	x	x	x	x
R/W	R	R	R	R	R	R	R	R

Bit	Name	Function
7	IR7	**Interrupt Request 7** 0 = IRQ7 input to the Master 8259 is not asserted 1 = IRQ7 is asserted
6	IR6	**Interrupt Request 6** 0 = IRQ6 input to the Master 8259 is not asserted 1 = IRQ6 is asserted
5	IR5	**Interrupt Request 5** 0 = IRQ5 input to the Master 8259 is not asserted 1 = IRQ5 is asserted
4	IR4	**Interrupt Request 4** 0 = IRQ4 input to the Master 8259 is not asserted 1 = IRQ4 is asserted
3	IR3	**Interrupt Request 3** 0 = IRQ3 input to the Master 8259 is not asserted 1 = IRQ3 is asserted
2	IR2	**Interrupt Request 2** 0 = IRQ2 input to the Master 8259 is not asserted 1 = IRQ2 is asserted
1	IR1	**Interrupt Request 1** 0 = IRQ1 input to the Master 8259 is not asserted 1 = IRQ1 is asserted
0	IR0	**Interrupt Request 0** 0 = IRQ0 input to the Master 8259 is not asserted 1 = IRQ0 is asserted

Programming Notes

This register provides a real-time status of the IRQ (interrupt request) inputs to the Master 8259. The Master Interrupt Request register (IRR) is accessed by first writing a value of 0Ah to port 20h followed by a read-back from port 20h.

Since the Slave 8259 cascades into Channel 2 of the Master 8259, IR2 is a real-time status indication that one of the slave IRQ inputs is asserted.

	7	6	5	4	3	2	1	0
Bit	IS7	IS6	IS5	IS4	IS3	IS2	IS1	IS0
Default	x	x	x	x	x	x	x	x
R/W	R	R	R	R	R	R	R	R

Bit	Name	Function
7	IS7	**IRQ7 In-Service** 0 = IRQ7 is not being serviced 1 = IRQ7 is being serviced
6	IS6	**IRQ6 In-Service** 0 = IRQ6 is not being serviced 1 = IRQ6 is being serviced
5	IS5	**IRQ5 In-Service** 0 = IRQ5 is not being serviced 1 = IRQ5 is being serviced
4	IS4	**IRQ4 In-Service** 0 = IRQ4 is not being serviced 1 = IRQ4 is being serviced
3	IS3	**IRQ3 In-Service** 0 = IRQ3 is not being serviced 1 = IRQ3 is being serviced
2	IS2	**IRQ2 In-Service** 0 = IRQ2 is not being serviced 1 = IRQ2 is being serviced
1	IS1	**IRQ1 In-Service** 0 = IRQ1 is not being serviced 1 = IRQ1 is being serviced
0	IS0	**IRQ0 In-Service** 0 = IRQ0 is not being serviced 1 = IRQ0 is being serviced

Programming Notes

The Master In-Service register (ISR) is accessed by first writing a value of 0Bh to port 20h followed by a read-back from port 20h.

Since the Slave 8259 cascades into Channel 2 of the Master 8259, IS2 will be asserted if any slave IRQ level is asserted.

	7	6	5	4	3	2	1	0
Bit	Reserved			SLCT_ICW1	LTIM	ADI	SNGL	IC4
Default	x	x	x	x	x	x	x	x
R/W	W			W	W	W	W	W

Bit	Name	Function
7–5	Reserved	**Reserved** Must all be written to 0.
4	SLCT_ICW1	**Select ICW1** Must be written to 1 to access ICW1 (D4 = 1 means the IOW to port 20h is Initialization Control Word 1).
3	LTIM	**Level-Triggered Interrupt Mode** 0 = Edge-sensitive IRQ detection 1 = Level-sensitive IRQ detection
2	ADI	**Address Interval (has no effect when the 8259 is used in x86 mode)** 0 = Interrupt vectors are separated by 8 locations 1 = Interrupt vectors are separated by 4 locations In the ÉlanSC400 microcontroller design, this PC/AT Compatible bit is internally fixed to ‘1b’.
1	SNGL	**Single 8259** If this bit is set, then the internal register pointer will skip ICW3, and point to ICW4 if ICW4 was selected to be programmed via bit 0 of this register. See the explanation for bit 0 of this register: 0 = Cascade mode, ICW3 will be expected 1 = Single 8259 in the system, ICW3 will not be expected In the ÉlanSC400 microcontroller design, this PC/AT Compatible bit is internally fixed to ‘0b’.
0	IC4	**Initialization Control Word 4** ICW4 is required. The 8259’s initialization control word (ICW) registers 1–4 are programmed in sequence. Writing to port 20h with bit 4 = 1 causes the internal ICW1 Register to be written, and resets the PIC’s internal state machine and ICW register pointer. Master ICW2–4 are programmed via port 21h. Each time 21h is written to (following ICW1), the register pointer points to the next internal ICW register. ICW1 and ICW2 must always be programmed, but ICW3 and ICW4 need only be programmed under certain circumstances. 0 = Initialization Control Word 4 is cleared by writing Initialization Control Word 1, ICW4 will not be expected by the PIC 1 = ICW4 is not cleared by this write to Initialization Control Word 1, and software is expected to initialize ICW4 This bit determines whether ICW4 is required to be explicitly programmed, or whether a value of 00h will serve for ICW4. If ICW4 is selected to be programmed, the register pointer will point to ICW4 after ICW3, and the PIC will expect software to provide an initialization value for it. In the ÉlanSC400 microcontroller design, this PC/AT Compatible bit is internally fixed to ‘1b’.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	R SL EOI			SLCT_ICW1	IS_OCW3	LS[2–0]		
Default	x	x	x	x	x	x	x	x
R/W	W			W	W	W		

Bit	Name	R/W	Function
7–5	R SL EOI	W	**IRQ EOI and Priority Rotation Controls** 0 0 0 = Rotate in auto EOI mode (clear) 0 0 1 = Non-specific EOI 0 1 0 = No operation 0 1 1 = Specific EOI 1 0 0 = Rotate in auto EOI mode (set) 1 0 1 = Rotate on non-specific EOI command 1 1 0 = Set priority command 1 1 1 = Rotate on specific EOI command
4	SLCT_ICW1	W	**Select Initialization Control Word 1** Software must write this bit to 0 to access OCW2 or OCW3.
3	IS_OCW3	W	**Access is OCW3** An I/O write to port 20h with this bit cleared indicates that OCW2 is to be accessed.
2–0	LS[2–0]	W	**Specific EOI Level Select** Interrupt level which is acted upon when the SL bit = ‘1b’ (see bits 7–5 of this register): 0 0 0 = IRQ0 0 0 1 = IRQ1 0 1 0 = IRQ2 0 1 1 = IRQ3 1 0 0 = IRQ4 1 0 1 = IRQ5 1 1 0 = IRQ6 1 1 1 = IRQ7

Programming Notes

I/O writes to port 20h access different PIC registers based on bits 4–3 of the data that is written. See the following table:

Bit 4	Bit 3	Register Accessed
0	0	OCW2
0	1	OCW3
1	x	ICW1

	7	6	5	4	3	2	1	0
Bit	Reserved	ESMM SMM		SLCT_ICW1	IS_OCW3	P	RR RIS	
Default	x	x	x	x	x	x	x	x
R/W		W		W	W	W	W	

Bit	Name	Function
7	Reserved	**Reserved** Software should write this bit to 0.
6–5	ESMM SMM	**Special Mask Mode** 0 x = No operation 1 0 = Reset special mask 1 1 = Set special mask In the ÉlanSC400 microcontroller design, the ESMM bit is internally fixed to ‘1b’.
4	SLCT_ICW1	**Initialization Control Word 1 Select** Software should write this bit to 0 to access OCW2 or OCW3.
3	IS_OCW3	**Access is OCW3** An I/O write to port 20h with this bit set indicates that OCW3 is to be accessed.
2	P	**PIC Poll Command** A system design can choose to use the PIC in a non-interrupting mode. In this case, the interrupt controller can be polled for the status of pending interrupts. In order to support this PC/AT incompatible mode of operation, the PIC supports a special poll command which is invoked by writing OCW3 with this bit set. 0 = Not poll command 1 = Poll command
1–0	RR RIS	**Status Register Select** 0 0 = No change from last state 0 1 = No change from last state 1 0 = Next port 20h read will return Interrupt Request Register (IRR) 1 1 = Next port 20h read will return In-Service Register (ISR)

Programming Notes

I/O writes to port 20h access different PIC registers based on bits 4–3 of the data that is written. See the following table:

Bit 4	Bit 3	Register Accessed
0	0	OCW2
0	1	OCW3
1	x	ICW1

	7	6	5	4	3	2	1	0
Bit	T7–T3					A10–A8		
Default	x	x	x	x	x	x	x	x
R/W	W					W		

Bit	Name	Function
7–3	T7–T3	**Base Interrupt Vector Number** Bits 7–3 of base interrupt vector number for this PIC. Bits 2–0 are concatenated by PIC based on IRQ level. For example, these bits will be programmed to 00001b for the master PIC (IRQ0 generates int 8), and 01110b for the slave PIC (IRQ8 corresponds to int 70h) in a PC/AT-compatible system.
2–0	A10–A8	**A10–A8 of Interrupt Vector** Always = 0 in a PC/AT-compatible system.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	S7	S6	S5	S4	S3	S2	S1	S0
Default	x	x	x	x	x	x	x	x
R/W	W	W	W	W	W	W	W	W

Bit	Name	Function
7	S7	**Channel 7 Slave Cascade Select** 0 = I/O device attached to IRQ7 input 1 = IRQ7 input used for slave cascading
6	S6	**Channel 6 Slave Cascade Select** 0 = I/O device attached to IRQ6 input 1 = IRQ6 input used for slave cascading
5	S5	**Channel 5 Slave Cascade Select** 0 = I/O device attached to IRQ5 input 1 = IRQ5 input used for slave cascading
4	S4	**Channel 4 Slave Cascade Select** 0 = I/O device attached to IRQ4 input 1 = IRQ4 input used for slave cascading
3	S3	**Channel 3 Slave Cascade Select** 0 = I/O device attached to IRQ3 input 1 = IRQ3 input used for slave cascading
2	S2	**Channel 2 Slave Cascade Select** 0 = I/O device attached to IRQ2 input 1 = IRQ2 input used for slave cascading
1	S1	**Channel 1 Slave Cascade Select** 0 = I/O device attached to IRQ1 input 1 = IRQ1 input used for slave cascading In the ÉlanSC400 microcontroller design, this PC/AT Compatible bit is internally fixed to ‘1b’.
0	S0	**Channel 0 Slave Cascade Select** 0 = I/O device attached to IRQ0 input 1 = IRQ0 input used for slave cascading

Programming Notes

Bits 7–3 and 1–0 of ICW3 are internally fixed to ‘0b’ in this design. Bit 2 is internally fixed to ‘1b’.

	7	6	5	4	3	2	1	0
Bit	Reserved			SFNM	BUF M/S		AEOI	PM
Default	x	x	x	x	x	x	x	x
R/W	W			W	W		W	W

Bit	Name	Function
7–5	Reserved	**Reserved** Software should write these bits to 0.
4	SFNM	**Special Fully Nested Mode Enable** 0 = Normal nested mode 1 = Special fully nested mode
3–2	BUF M/S	**Buffered Mode and Master/Slave Select** 0 x = Non-buffered mode 1 0 = Buffered mode/slave 1 1 = Buffered mode/master In the ÉlanSC400 microcontroller design, these bits are internally fixed to ‘00b’.
1	AEOI	**Automatic EOI Mode** 0 = Normal EOI Interrupt handler must send an End of Interrupt command to the PIC(s) 1 = Auto EOI EOI is automatically performed after the second $\overline{\text{INTA}}$ signal from the CPU
0	PM	**Microprocessor Mode** 0 = 8080/8085 mode 1 = 8086 mode In the ÉlanSC400 microcontroller design, this PC/AT Compatible bit is internally fixed to ‘1b’.

Programming Notes

Master 8259 Interrupt Mask Register (also known as Operation Control Word 1)

I/O Address 0021h

	7	6	5	4	3	2	1	0
Bit	IM7	IM6	IM5	IM4	IM3	IM2	IM1	IM0
Default	x	x	x	x	x	x	x	x
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	IM7	**IRQ7 Mask** 0 = Unmask IRQ7 1 = Mask IRQ7
6	IM6	**IRQ6 Mask** 0 = Unmask IRQ6 1 = Mask IRQ6
5	IM5	**IRQ5 Mask** 0 = Unmask IRQ5 1 = Mask IRQ5
4	IM4	**IRQ4 Mask** 0 = Unmask IRQ4 1 = Mask IRQ4
3	IM3	**IRQ3 Mask** 0 = Unmask IRQ3 1 = Mask IRQ3
2	IM2	**IRQ2 Mask** 0 = Unmask IRQ2 1 = Mask IRQ2
1	IM1	**IRQ1 Mask** 0 = Unmask IRQ1 1 = Mask IRQ1
0	IM0	**IRQ0 Mask** 0 = Unmask IRQ0 1 = Mask IRQ0

Programming Notes

	7	6	5	4	3	2	1	0
Bit	CSC_INDEX[7–0]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	CSC_INDEX[7–0]	**ÉlanSC400 Microcontroller Chip Setup and Control (CSC) Index Register** Specify an 8-bit CSC indexed configuration register address via this port. Data can then be read from or written to the specified configuration register via direct-mapped port 23h.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	CSC_DATA[7–0]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	CSC_DATA[7–0]	**ÉlanSC400 Microcontroller Chip Setup and Control (CSC) Data Port** Via this port, data can be read or written to the configuration register that is specified using direct-mapped port 22h.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	CH0_COUNT[15–0]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	CH0_COUNT[15–0]	**16-bit Counter for Programmable Interval Timer #1, Channel 0** Access either the counter high byte only, low byte only, or low byte followed by high byte as defined by bits 5–4 of direct-mapped register 43h for either read or write operations. Immediately following a counter latch command (or a read-back command in which the LCNT bit = ‘0b’, and the CNT0 bit = ‘1b’), the latched count can be read from this register. After the 1 or 2 bytes of latched count are read, subsequent reads return the unlatched count.

Programming Notes

If a read-back command is issued in which LSTAT = ‘0b’, and the CNT0 bit = ‘1b’, a status byte for this channel, as defined by the Programmable Interval Timer #1 Status Register (see below), will be read back. If a read-back command is issued in which LSTAT = ‘0b’, LCNT = ‘0b’, and the CNT0 bit = ‘1b’, the first read from this register will return the status byte, and the second/third bytes will return the 1 or 2 (low/high) latched count byte(s).

When set up for either BCD or 16-bit binary count operation, the maximum count for Channel 0 is achieved by writing the internal counting element associated with this register to 0000h/d. See direct-mapped register 43h for more detail.

	7	6	5	4	3	2	1	0
Bit	CH1_COUNT[15–0]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	CH1_COUNT[15–0]	**16-bit Counter for Programmable Interval Timer #1, Channel 1** Access either the counter high byte only, low byte only, or low byte followed by high byte as defined by bits 5–4 of direct-mapped register 43h for either read or write operations. Immediately following a counter latch command (or a read-back command in which the LCNT bit = ‘0b’, and the CNT1 bit = ‘1b’), the latched count can be read from this register. After the 1 or 2 bytes of latched count are read, subsequent reads return the unlatched count.

Programming Notes

If a read-back command is issued in which LSTAT = ‘0b’, and the CNT1 bit = ‘1b’, a status byte for this channel, as defined by the Programmable Interval Timer #1 Status Register (see below), will be read back. If a read-back command is issued in which LSTAT = ‘0b’, LCNT = ‘0b’, and the CNT1 bit = ‘1b’, the first read from this register will return the status byte, and the second/third bytes will return the 1 or 2 (low/high) latched count byte(s).

When set up for either BCD or 16-bit binary count operation, the maximum count for Channel 1 is achieved by writing the internal counting element associated with this register to 0000h/d. See direct-mapped register 43h for more detail.

	7	6	5	4	3	2	1	0
Bit	CH2_COUNT[15–0]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	CH2_COUNT[15–0]	**16-bit Counter for Programmable Interval Timer #1, Channel 2** Access either the counter high byte only, low byte only, or low byte followed by high byte as defined by bits 5–4 of direct-mapped register 43h for either read or write operations. Immediately following a counter latch command (or a read-back command in which the LCNT bit = ‘0b’, and the CNT2 bit = ‘1b’), the latched count can be read from this register. After the 1 or 2 bytes of latched count are read, subsequent reads return the unlatched count.

Programming Notes

If a read-back command is issued in which LSTAT = ‘0b’, and the CNT2 bit = ‘1b’, a status byte for this channel, as defined by the Programmable Interval Timer #1 Status Register (see below), will be read back. If a read-back command is issued in which LSTAT = ‘0b’, LCNT = ‘0b’, and the CNT2 bit = ‘1b’, the first read from this register will return the status byte, and the second/third bytes will return the 1 or 2 (low/high) latched count byte(s).

When set up for either BCD or 16-bit binary count operation, the maximum count for Channel 2 is achieved by writing the internal counting element associated with this register to 0000h/d. See direct-mapped register 43h for more detail.

	7	6	5	4	3	2	1	0
Bit	OUTPUT	NULLCNT	RW1 RW0		M2 M1 M0			BCD
Default	0	0	0	0	0	0	0	0
R/W	R	R	R		R			R

Bit	Name	Function
7	OUTPUT	**Output Pin State** Output signal for the current timer channel. Each timer channel has an output pin that is driven high or low based on the current mode. See the brief mode descriptions for bits 3–1 of this register for more detail. 0 = Current state of OUT(x) signal = logic 0 1 = Current state of OUT(x) signal = logic 1
6	NULLCNT	**Null Count** When programming a new count value into one of the timers, the new value does not take effect until it has actually been transferred to the counting element, which can take some time. Thus, when attempting to read back a count value, this bit indicates whether the value read back is valid or not. 1 = Null count, read back of the counter will be invalid 0 = Counter is available for reading
5–4	RW1 RW0	**Counter Read/Write Operation Control or Counter Latch Command** Reflects the last bit setting that was programmed into this field for this counter channel via the Programmable Interval Timer #1 Mode Control Register. See the Programmable Interval Timer #1 Mode Control Register on page 2-41 for more information. 0 0 = Counter Latch Command 0 1 = Read/Write LSB only 1 0 = Read/Write MSB only 1 1 = Read/Write LSB first followed by MSB
3–1	M2 M1 M0	**Counter Mode Status** Reflects the last counter mode setting for counters 0, 1, or 2 that was programmed into this channel via the Programmable Interval Timer #1 Mode Control Register. See the Programmable Interval Timer #1 Mode Control Register on page 2-41 for more information. Gate = High unless noted 0 0 0 = Mode 0: Interrupt on terminal count 0 0 1 = Mode 1: Hardware retriggerable one-shot 0 1 0 = Mode 2: Rate generator 0 1 1 = Mode 3: Square wave generator 1 0 0 = Mode 4: Software retriggerable strobe 1 0 1 = Mode 5: Hardware retriggerable strobe 1 1 0 = Alias for mode 2 1 1 1 = Alias for mode 3 See the table below for more detail on this register.
0	BCD	**Binary Coded Decimal Select Status** Reflects the last BCD setting for counters 0, 1, or 2 that was programmed into this channel via the Programmable Interval Timer #1 Mode Control Register. See the Programmable Interval Timer #1 Mode Control Register on page 2-41 for more information. 0 = 16-bit binary counter with a range of 0–FFFFh 1 = BCD counter with a range of 0–9999d

Programming Notes

These three registers (at direct-mapped I/O addresses 40h, 41h, and 42h) are separately available only as a result of sending a read-back command with the LSTAT bit = ‘0b’, and the appropriate CNTX bit set (where X is a counter from 0–2). For more detail, see the Read-Back Command Register on page 2-44. Note also that modes 1 and 5 require a rising edge on the gate input for each timer channel. Only Timer 2 has a gate control (see direct-mapped port 61h[0]) so only Channel 2 is capable of running all modes. The gate controls for Timers 0 and 1 are fixed internally to ‘1b’, so they are only capable of operation in modes 0, 2, 3 and 4.

Counter Mode Status Bits 3–1

Settings	Mode Name	Description
0 0 0 = Mode 0	Interrupt on terminal count	When the counter is programmed, the Counter Output Signal transitions to 0. When counter reaches 0, the Counter Output signal transitions to 1 until another count is written.
0 0 1 = Mode 1	Hardware retriggerable one-shot	When mode/counter are programmed, the Counter Output signal transitions to 1. When gate goes 0 transitions to 1, OUT transitions to 0 until the count reaches 0, then OUT transitions to 1 until the next low-high transition on gate.
0 1 0 = Mode 2	Rate generator	Each time the count transitions to 1, OUT transitions to 0, and remains there for 1 cycle of the input clock, and then OUT transitions to 1. The count is automatically reloaded, and the process repeats. Timer 0 uses this mode by default in the PC/AT.
0 1 1 = Mode 3	Square wave generator	When the count is loaded, OUT transitions to 1. When 1/2 of the count has expired, OUT transitions to 0. When count transitions to 0, OUT transitions to 1, and count is automatically reloaded. In the PC/AT, Timers 1 and 2 use this mode by default to drive DRAM refresh and the speaker respectively.
1 0 0 = Mode 4	Software retriggerable strobe	When the count is loaded, OUT transitions to 1. When counter = 0, OUT transitions to 0 for 1 clock and then OUT transitions to 1.
1 0 1 = Mode 5	Hardware retriggerable strobe	The Counter Output signal behaves just like mode 4 except that the triggering is done by a low to high transition on the gate input. A trigger seen during the count reloads the count to the initial value and then counting continues.
1 1 0 = Alias for mode 2		
1 1 1 = Alias for mode 3		

	7	6	5	4	3	2	1	0
Bit	SC1 SC0		RW1 RW0		M2 M1 M0			BCD
Default	0	0	0	0	0	0	0	0
R/W	W		W		W			W

Bit	Name	Function
7–6	SC1 SC0	**Channel Select or Read-back Command** When this register is written to with bits 7–6 = ‘11b’, this register is redefined (for the duration of the current I/O write) as the Read-back Command Register which is described as a separate register below. When this register is written to with bits 7–6 ≠ ‘11b’, and bits 5–4 ≠ ‘00b’, these bits specify to which of the three on-chip counters the settings in bits 5–0 apply: 0 0 = Select Channel 0 0 1 = Select Channel 1 1 0 = Select Channel 2 1 1 = Read-back Command A read-back command is a higher priority command than the counter latch command. If bits 5–4 are written to ‘00b’, and bits 7–6 are written to ‘11b’, only the read-back command will be recognized as command bits, and bits 5–4 will take on the meanings as described by the Read-back Command Register below.
5–4	RW1 RW0	**Counter Read/Write Operation Control or Counter Latch Command** When this register is written to with bits 5–4 = ‘00b’, and bits 7–6 ≠ ‘11b’, this register is redefined (for the duration of the current I/O write) as the Counter Latch Command Register which is described as a separate register below. When this register is written to with bits 7–6 ≠ ‘11b’, and bits 5–4 ≠ ‘00b’, these bits define what can be written to the Channel X Count Register, where X is selected by bits 7–6 of this register. Note that if only 8 bits of a count register will be accessed (i.e., bits 5–4 = ‘01b’ or ‘10b’), a subsequent write to the count register (as indicated by bits 7–6) will automatically clear the 8 bits of the associated internal 16-bit counting element which were not explicitly written to. 0 0 = Counter Latch Command 0 1 = Read/Write counter bits [7–0] 1 0 = Read/Write counter bits [15–8] 1 1 = Read/Write counter bits [7–0] followed immediately by [15–8] A counter latch command does not stop a counter from running, but rather takes a snapshot of the current value. When a counter’s current value is required, a counter latch command or a read-back command should be used to obtain it. Once the count has been latched, further latch commands are ignored until all latched count data is read back from the associated count register. Also note that the 8254 Programmable Interval Timer does not provide any way to read back the original count programmed into any of the three count registers.

Bit	Name	Function
3–1	M2 M1 M0	**Counter Mode** When this register is written to with bits 7–6 ≠ ‘11b’, and bits 5–4 ≠ ‘00b’, these bits control the counter operation: 0 0 0 = Mode 0: Interrupt on terminal count 0 0 1 = Mode 1: Hardware retriggerable one-shot 0 1 0 = Mode 2: Rate generator 0 1 1 = Mode 3: Square wave generator 1 0 0 = Mode 4: Software retriggerable strobe 1 0 1 = Mode 5: Hardware retriggerable strobe 1 1 0 = Alias for mode 2 1 1 1 = Alias for mode 3
0	BCD	**Binary Coded Decimal Select** When this register is written to with bits 7–6 ≠ ‘11b’, and bits 5–4 ≠ ‘00b’, this bit controls whether the counter indicated by bits 7–6 of this register will count in binary with a range of 0–FFFFh or in binary coded decimal (BCD) with a range of 0–9999d. 0 = 16-bit binary counter 1 = BCD counter

Programming Notes

Writing to this register to change the mode for a particular counter resets the control logic and resets the associated counter value and OUT pin.

Programmable Interval Timer #1 Counter Latch Command Register

I/O Address 0043h

	7	6	5	4	3	2	1	0
Bit	SC1 SC0		RW1 RW0		Reserved			
Default	0	0	0	0	0	0	0	0
R/W	W		W					

Bit	Name	Function
7–6	SC1 SC0	**Counter Select** Specify which of the three counter elements to latch for read back from the associated count register. The counter latch command is a subset of the read-back command since only one channel can have its counter latched per counter latch command: 0 0 = Select counter 0 0 1 = Select counter 1 1 0 = Select counter 2 1 1 = N/A
5–4	RW1 RW0	**Counter Command** 0 0 = Counter Latch Command See Programmable Interval Timer #1 Mode Control Register on page 2-41 for more detail.
3–0	Reserved	**Reserved** Software should write these bits to 0.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	SC1 SC0		LCNT	LSTAT	CNT2	CNT1	CNT0	Reserved
Default	0	0	0	0	0	0	0	0
	W		W	W	W	W	W	W

Bit	Name	Function
7–6	SC1 SC0	**Counter Select/Read-back Command** 1 1 = Read-back Command Values as selected by bits 5–1 of this register are available to be read back from direct-mapped ports 40h–42h immediately following completion of the I/O write that implements this Read-back Command. Latched counts will be read back based on the current mode for each counter (bits 7–0, bits 15–8, or bits 7–0 followed by bits 15–8). For the format of the returned status byte, see the Programmable Interval Timer#1 Status Register bit descriptions above. See the Programmable Interval Timer #1 Mode Control Register on page 2-41 for more detail. If both LSTAT and LCNT = ‘0b’, the status byte will be available at the respective count register (direct-mapped 40h, 41h, and 42h) first. When this byte has been read, the latched count byte(s) will be available, and will read back low order byte and then high order byte (if set up to read back all 16-bits of count via the Programmable Interval Timer #1 Mode Control Register).
5	LCNT	**Latch Count (low true)** 0 = Latch count for counters selected via bits 3–1 1 = Do not latch count for counters selected via bits 3–1
4	LSTAT	**Latch Status (low true)** 0 = Latch status for counters selected via bits 3–1 1 = Do not latch status for counters selected via bits 3–1
3	CNT2	**Select Counter 2** 0 = Counter 2 not selected for operations specified by bits 5–4 1 = Counter 2 selected for operations specified by bits 5–4
2	CNT1	**Select Counter 1** 0 = Counter 1 not selected for operations specified by bits 5–4 1 = Counter 1 selected for operations specified by bits 5–4
1	CNT0	**Select Counter 0** 0 = Counter 0 not selected for operations specified by bits 5–4 1 = Counter 0 selected for operations specified by bits 5–4
0	Reserved	**Reserved** Software should write this bit to 0.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	SCP_OB[7–0]							
Default	0	0	0	0	0	0	0	0
R/W	R							

Bit	Name	Function
7–0	SCP_OB[7–0]	**System Control Processor Output Buffer** Data is read back from the System Control Processor (SCP) through this port. A read from this port simultaneously clears bit 0 of port 64h and the keyboard IRQ signal to the PIC (Master 8259, Channel 1). This data can either be keyboard scan codes, mouse coordinate data, SCP response codes, or keyboard return codes.

Programming Notes

This port is typically inside the external processor that is used to implement the SCP (System Control Processor) on a PC/AT Compatible system. Although an entire SCP is not implemented on-board in this design, some SCP registers are available to support the ÉlanSC400 microcontroller's capability of SCP emulation using a matrix keyboard. This PC/AT SCP emulation support register is only available to be read from this location when the ÉlanSC400 microcontroller's SCP emulation feature is enabled via CSC index C1h[3–2].

	7	6	5	4	3	2	1	0
Bit	SCP_DATA[7–0]							
Default	0	0	0	0	0	0	0	0
R/W	W							

Bit	Name	Function
7–0	SCP_DATA[7–0]	**System Control Processor Data** When the PC/XT keyboard interface is disabled via indexed CSC register C1h[4], writes to this port are interpreted as PC/AT SCP data or parameters when they follow SCP (System Control Processor) commands to port 64h that expect parameters. If a previous SCP command that requires parameters has not been written to port 64h previously, writes to this port are serialized by the SCP and sent to the keyboard. Writing to this port sets bit 1, clears bit 3 of port 64h. Writes to this port have no effect if the PC/XT keyboard interface is enabled.

Programming Notes

This port is typically inside the external processor that is used to implement the SCP (System Control Processor) on a PC/AT Compatible system. Although an entire SCP is not implemented on-board in this design, some SCP registers are available to support the ÉlanSC400 microcontroller's capability of SCP emulation using a matrix keyboard. This PC/AT SCP emulation support register is only available to be read from this location when the ÉlanSC400 microcontroller's SCP emulation feature is enabled via CSC index register C1h[3–2].

	7	6	5	4	3	2	1	0
Bit	XT_DATA[7–0]							
Default	0	0	0	0	0	0	0	0
R/W	R							

Bit	Name	Function
7–0	XT_DATA[7–0]	**XT Keyboard Controller Data** When the XT keyboard controller is enabled via indexed ÉlanSC400 microcontroller register C1h[4], scan codes from an XT keyboard are read from this register. The act of reading data from this register clears the IRQ1 that can be enabled to occur when data is received into the register from the XT keyboard interface. While the XT keyboard function is enabled, this register is cleared by setting bit 7 of port 61h. The XT Keyboard Data Register is disabled if the XT keyboard interface is disabled. In that case, the I/O location 60h is used for the AT keyboard interface.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	PERR	IOCHCK	T2OUT	RFD	Reserved	Reserved	SPKD	T2G
Default	0	0	x	0	0	0	0	0
R/W	R/W	R/W	R	R	R/W	R/W	R/W	R/W

Bit	Name	Function
7	PERR	**PC/AT Parity Error Indicator** Not supported (always reads back '0b').
6	IOCHCK	**PC/AT Channel Check Indicator** Not supported (always reads back '0b').
5	T2OUT	**Timer 2 (Speaker) Output Pin State** This status bit directly reflects the state of the output signal of Channel 2 of the on-board 8254 Programmable Interval Timer and is sampled before the gate controlled by bit 1 of this register. 0 = T2 output is low 1 = T2 output is high
4	RFD	**DRAM Refresh Indicator** This bit changes state each time a refresh is detected. On the original PC/AT, the Programmable Interval Timer Channel 1 output pin was used to generate the refresh signal. As an alternative in this design, the doubled 32 KHz clock can be used as the input to a divider to obtain the refresh clock. This bit tracks the current refresh source, regardless of the refresh generation source.
3	Reserved	**Reserved** On the original PC/AT, this bit was used to enable I/O channel check. Although this read/write register bit has been implemented on this design, it does not control anything. For software using this register to remain PC/AT Compatible, read/modify/write operations should preserve this bit.
2	Reserved	**Reserved** On the original PC/AT, this bit was used to enable RAM parity check. Although this read/write register bit has been implemented on this design, it does not control anything. For software using this register to remain PC/AT Compatible, read/modify/write operations should preserve this bit.
1	SPKD	**Speaker Data Enable** This bit controls a signal which is ANDed with the output from Timer 2. The AND gate output is then used to drive the system speaker. 0 = Do not propagate speaker data 1 = Propagate speaker data
0	T2G	**Timer 2 GATE Input Control** This bit drives the GATE input signal for Channel 2 of the 8254 Programmable Interval Timer which can be used to control the Timer Channel 2 operation depending on the current timer mode. See the Programmable Interval Timer Status Register description at direct-mapped 40–42h for more detail on the function of the gate input. 0 = Programmable Interval Timer GATE input is deasserted 1 = Programmable Interval Timer GATE input is asserted

Programming Notes

	7	6	5	4	3	2	1	0
Bit	KEYPAR	KEYRTO	KEYTTO	KEYENB	KEYCMD	KEYSYS	KEYIBF	KEYOBF
Default	0	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R	R

Bit	Name	Function
7	KEYPAR	**Keyboard Data Transmission Parity Error** 0 = No keyboard serial data parity error occurred 1 = Parity error occurred in the serial data transmission between the processor within the keyboard and the SCP There is no real SCP on-board the ÉlanSC400 microcontroller. This bit is provided for software compatibility purposes only. SCP emulation software can control this bit via the Keyboard Status Register Write Register at ÉlanSC400 microcontroller configuration index register C5h.
6	KEYRTO	**Keyboard Data Receive Time-out** 0 = No receive time-out 1 = Receive time-out A transmission started by the keyboard did not complete. There is no real SCP on-board the ÉlanSC400 microcontroller. This bit is provided for software compatibility purposes only. SCP emulation software can control this bit via the Keyboard Status Register Write Register at ÉlanSC400 microcontroller configuration index register C5h.
5	KEYTTO	**Transmit Time-out/Mouse Output Buffer Full** When CSC index C0h[0] is set, this bit is a PC/2 mouse-compatible mouse output buffer full flay. When CSC index C0h[0] is cleared, this bit becomes an emulated keyboard time-out bit. See CSC indexes C0h and C5h for more detail. 0 = No keyboard to SCP communications time-out or no mouse data available 1 = Keyboard to SCP communications time-out or mouse data available When Configuration Index Register C0h[0] = 0 (Transmit Time-Out), this bit indicates that a transmission started by the keyboard was not properly completed; the transmit byte was not clocked out in the time limit. This bit is set/cleared by AT SCP emulation software via a CSC index register. When Configuration Index Register C0h[0] = 1 (Mouse Output Buffer Full), this bit is automatically set when AT SCP emulation software places a byte in the SCP output buffer using the Mouse Output Buffer Extended Register. It is cleared when AT SCP emulation software places a byte in the output buffer using the Output Buffer Extended Register. In this mode, this bit is set/cleared automatically by writing to CSC index registers, not by a direct register bit manipulation by software. This bit works in conjunction with bit 0 of this register. When bits 0 and 5 are set, then mouse device data is in the output buffer. If bit 0 is set and bit 5 is cleared then keyboard or command controller (SCP) response data is in the buffer.
4	KEYENB	**Keyboard Password Protection Status** 0 = Password protected 1 = Not protected There is no real SCP on-board the ÉlanSC400 microcontroller. This bit is provided for software compatibility purposes only. SCP emulation software can control this bit via the Keyboard Status Register Write Register at chip configuration index register C5h.

3	KEYCMD	**Command/Data** 0 = Between the keyboard command and data ports, the data port at direct mapped 60h that was last written to 1 = Between the keyboard command and data ports, the command port at direct-mapped 64h that was last written to
2	KEYSYS	**System Flag Bit** In a PC/AT-compatible system, this bit reads back the state of the system flag bit (bit 2) of the SCP's command byte. This SCP command byte is stored in location 0 of the on-board SCP RAM of a normal PC/AT system. This bit would normally be cleared by the SCP upon system power-up (cold boot) and would remain that way until the system BIOS enabled the keyboard interface. At this time, the result of the SCP's on-board diagnostic would be written to the system flag bit (1 = pass, 0 = fail). Since BIOS must always enable the keyboard interface before the system flag bit can be set, BIOS would read this bit after a reset to know whether the reset was a cold reset, or if the reset was initiated by software. 0 = A keyboard diagnostic pass indication has not been posted since SCP/system cold power-up reset 1 = A keyboard diagnostic pass has been posted since SCP/system cold power-up reset There is no real SCP or SCP RAM on-board the ÉlanSC400 microcontroller. This bit is provided for software compatibility purposes only. SCP emulation software can control this bit via the Keyboard Status Register Write Register at chip configuration index register C5h.
1	KEYIBF	**Input Buffer Full (Keyboard Input Buffer Status)** 0 = Buffer empty (SCP able to receive next command/data byte) 1 = Buffer full (SCP has not read the last command/data byte)
0	KEYOBF	**Output Buffer Full (Keyboard Output Buffer Status)** 0 = SCP (System Control Processor) has no data available for the system 1 = SCP has data available for read-back by the system When this bit is active, either the keyboard IRQ1 signal to the PIC (master 8259, Channel 1), or the mouse IRQ12 signal to the PIC (slave 8259, Channel 4) is automatically asserted based upon whether the data in the output buffer was from the keyboard or from the mouse. If bit 5 of this register has been defined to be mouse output buffer full, and bits 0 and 5 of this register are set, IRQ12 is being asserted to the slave PIC. If bit 5 is not defined to be mouse output buffer full, then when bit 0 is set IRQ1 is being asserted to the master PIC. Either IRQ1 or IRQ12 are cleared by reading port 60h.

Programming Notes

This port is typically inside the external processor that is used to implement the SCP (System Control Processor) on a PC/AT Compatible system. Although an entire SCP is not implemented on-board in this design, some SCP registers are available to support the ÉlanSC400 microcontroller's capability of SCP emulation using a matrix keyboard. This PC/AT SCP Emulation Support Register is only available to be read from this location when the ÉlanSC400 microcontroller's SCP emulation feature is enabled via CSC index register C1h[3–2].

	7	6	5	4	3	2	1	0
Bit	KEY_CMD[7–0]							
Default	0	0	0	0	0	0	0	0
R/W	W							

Bit	Name	Function
7–0	KEY_CMD[7–0]	**Keyboard Command Register** Writes to this port are interpreted as keyboard or mouse commands. Some commands require parameters which would be written to port 60h. Writing to this port sets bits 1 and 3 of port 64h.

Programming Notes

This port is typically inside the external processor that is used to implement the SCP (System Control Processor) on a PC/AT Compatible system. Although an entire SCP is not implemented on-board in this design, some SCP registers are available to support the ÉlanSC400 microcontroller's capability of SCP emulation using a matrix keyboard. This PC/AT SCP emulation support register is only available to be read from this location when the ÉlanSC400 microcontroller's SCP emulation feature is enabled via CSC index register C1h[3–2].

	7	6	5	4	3	2	1	0
Bit	Reserved	CMOSIDX						
Default	–	x	x	x	0	0	0	0
R/W		W						

Bit	Name	Function
7	Reserved	**Reserved**
6–0	CMOSIDX	**CMOS RAM Index** CMOS RAM index to read/write.

Programming Notes

Bit 7 of this register is the master NMI gate control in a typical PC/AT Compatible system. For various reason, this bit has been made to reside at CSC index 9Dh[2] on the ÉlanSC400 microcontroller. Compatibility issues are minimized since the ÉlanSC400 microcontroller does not support generation of either of the legacy NMI sources (channel check or parity error).

	7	6	5	4	3	2	1	0
Bit	CMOSDATA							
Default	x	x	x	x	x	x	x	x
R/W	W							

Bit	Name	Function
7–0	CMOSDATA	**RTC/CMOS Data Port** Data to be written or read.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	PORT80[7–0]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	PORT80[7–0]	**General Purpose R/W Register**

Programming Notes

In the original PC/AT, this register would have been DMA Channel 4 Page Register, but DMA Channel 4 was used for the cascade function, so this register was not used by the DMA subsystem. The I/O address 80h, however, was used to send BIOS Power-on Self Test (POST) codes to the ISA bus where a special card (port 80h card) could display the progress/error codes from BIOS. Because of this legacy use by BIOS, I/O writes to this internal ÉlanSC400 microcontroller register automatically go to the ISA bus as well as to an internal storage element for this register. I/O reads from port 80h come from the internal register only.

Access to this register is not affected by the DMA disable bits in CSC index D0h.

	7	6	5	4	3	2	1	0
Bit	DMA2MAR[23–16]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	DMA2MAR[23–16]	**DMA Channel 2 Memory Address Bits [A23–A16]**

Programming Notes

The ÉlanSC400 microcontroller is capable of performing 26-bit DMA accesses using the extended DMA page registers that reside in CSC index registers D9h and DAh.

Access to this register is not affected by the DMA disable bits in CSC index D0h.

	7	6	5	4	3	2	1	0
Bit	DMA3MAR[23–16]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	DMA3MAR[23–16]	**DMA Channel 3 Memory Address Bits [A23–A16]**

Programming Notes

The ÉlanSC400 microcontroller is capable of performing 26-bit DMA accesses using the extended DMA page registers that reside in CSC index registers D9h and DAh.

Access to this register is not affected by the DMA disable bits in CSC index D0h.

	7	6	5	4	3	2	1	0
Bit	DMA1MAR[23–16]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	DMA1MAR[23–16]	**DMA Channel 1 Memory Address Bits [A23–A16]**

Programming Notes

The ÉlanSC400 microcontroller is capable of performing 26-bit DMA accesses using the extended DMA page registers that reside in CSC index registers D9h and DAh.

Access to this register is not affected by the DMA disable bits in CSC index D0h.

	7	6	5	4	3	2	1	0
Bit	PORT84[7–0]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	PORT84[7–0]	**General Purpose R/W Register**

Programming Notes

Access to this register is not affected by the DMA disable bits in CSC index D0h.

Writes to this port go both to an internal register and externally to the VL or ISA bus. Reads from this port produce an internal cycle only and return the contents of the internal register.

	7	6	5	4	3	2	1	0
Bit	PORT85[7–0]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	PORT85[7–0]	**General Purpose R/W Register**

Programming Notes

Access to this register is not affected by the DMA disable bits in CSC index D0h.

Writes to this port go both to an internal register and externally to the VL or ISA bus. Reads from this port produce an internal cycle only and return the contents of the internal register.

	7	6	5	4	3	2	1	0
Bit	PORT86[7–0]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	PORT86[7–0]	**General Purpose R/W Register**

Programming Notes

Access to this register is not affected by the DMA disable bits in CSC index D0h.

Writes to this port go both to an internal register and externally to the VL or ISA bus. Reads from this port produce an internal cycle only and return the contents of the internal register.

	7	6	5	4	3	2	1	0
Bit	DMA0MAR[23–16]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	DMA0MAR[23–16]	**DMA Channel 0 Memory Address Bits [A23–A16]**

Programming Notes

The ÉlanSC400 microcontroller is capable of performing 26-bit DMA accesses using the extended DMA page registers that reside in CSC index registers D9h and DAh.

Access to this register is not affected by the DMA disable bits in CSC index D0h.

Writes to this port go both to an internal register and externally to the VL or ISA bus. Reads from this port produce an internal cycle only and return the contents of the internal register.

	7	6	5	4	3	2	1	0
Bit	PORT88[7–0]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	PORT88[7–0]	**General Purpose R/W Register**

Programming Notes

Access to this register is not affected by the DMA disable bits in CSC index D0h.

Writes to this port go both to an internal register and externally to the VL or ISA bus. Reads from this port produce an internal cycle only and return the contents of the internal register.

	7	6	5	4	3	2	1	0
Bit	DMA6MAR[23–17]							Reserved
Default	0	0	0	0	0	0	0	0
R/W	R/W							R/W

Bit	Name	Function
7–1	DMA6MAR[23–17]	**DMA Channel 6 Memory Address Bits [A23–A17] (16-bit channel)**
0	Reserved	**Reserved** Not used for DMA memory address generation. Software should write this bit to 0.

Programming Notes

The ÉlanSC400 microcontroller is capable of performing 26-bit DMA accesses using the extended DMA page registers that reside in CSC index registers D9h and DAh.

Access to this register is not affected by the DMA disable bits in CSC index D0h.

	7	6	5	4	3	2	1	0
Bit	DMA7MAR[23–17]							Reserved
Default	0	0	0	0	0	0	0	0
R/W	R/W							R/W

Bit	Name	Function
7–1	DMA7MAR[23–17]	**DMA Channel 7 Memory Address Bits [A23–A17] (16-bit channel)**
0	Reserved	**Reserved** Not used for DMA memory address generation. Software should write this bit to 0.

Programming Notes

The ÉlanSC400 microcontroller is capable of performing 26-bit DMA accesses using the extended DMA page registers that reside in CSC index registers D9h and DAh.

Access to this register is not affected by the DMA disable bits in CSC index D0h.

	7	6	5	4	3	2	1	0
Bit	DMA5MAR[23–17]							Reserved
Default	0	0	0	0	0	0	0	0
R/W	R/W							R/W

Bit	Name	Function
7–1	DMA5MAR[23–17]	**DMA Channel 5 Memory Address Bits [A23–A17] (16-bit channel)**
0	Reserved	**Reserved** Not used for DMA memory address generation. Software should write this bit to 0.

Programming Notes

Access to this register is not affected by the DMA disable bits in CSC index D0h.

	7	6	5	4	3	2	1	0
Bit	PORT8C[7–0]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	PORT8C[7–0]	**General Purpose R/W Register**

Programming Notes

Access to this register is not affected by the DMA disable bits in CSC index D0h.

Writes to this port go both to an internal register and externally to the VL or ISA bus. Reads from this port produce an internal cycle only and return the contents of the internal register.

	7	6	5	4	3	2	1	0
Bit	PORT8D[7–0]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	PORT8D[7–0]	**General Purpose R/W Register**

Programming Notes

Access to this register is not affected by the DMA disable bits in CSC index D0h.

Writes to this port go both to an internal register and externally to the VL or ISA bus. Reads from this port produce an internal cycle only and return the contents of the internal register.

	7	6	5	4	3	2	1	0
Bit	PORT8E[7–0]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	PORT8E[7–0]	**General Purpose R/W Register**

Programming Notes

Access to this register is not affected by the DMA disable bits in CSC index D0h.

Writes to this port go both to an internal register and externally to the VL or ISA bus. Reads from this port produce an internal cycle only and return the contents of the internal register.

	7	6	5	4	3	2	1	0
Bit	PORT8F[7–0]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	PORT8F[7–0]	**General Purpose R/W Register**

Programming Notes

This general register is slightly different from the other direct-mapped general registers. All I/O accesses to port 8Fh go to the internal register only. Therefore, no VL or ISA bus cycles are generated on I/O writes to this port.

	7	6	5	4	3	2	1	0
Bit	Reserved						AGA20	ACPURESET
Default	0	0	0	0	0	0	0	0
R/W							R/W	R/W

Bit	Name	R/W	Function
7–2	Reserved		**Reserved** Software should write these bits to 0.
1	AGA20	R/W	**Alternate A20 Gate Control** This bit can be used to cause the same type of masking of the CPU A20 signal that was historically performed by an external SCP (System Control Processor) in a PC/AT Compatible system, but much faster: 0 = Deasserts the forcing of the propagation of the A20 signal via this particular control 1 = Forces the A20 signal to propagate For software compatibility and other reasons, there are several sources of GateA20 control. These controls are effectively ORed together with the output of the OR gate driving the Enhanced Am486 microprocessor A20M input. Therefore, A20 will propagate if any of the independent sources are forcing A20 to propagate.
0	ACPURESET	R/W	**Alternate CPU Core Reset Control** Changing this bit from ‘0b’ to ‘1b’ will pulse the CPU SRESET signal. This will cause the same type of CPU core reset to occur that was historically performed by an external SCP (System Control Processor) in a PC/AT Compatible system, but much faster. The 486 cache state, SMBASE, and ÉlanSC400 microcontroller indexed or direct-mapped registers will not be affected as a result of this soft reset. Following the reset, this bit will remain set until software clears it. This feature can be used by the BIOS as an indication that port 92h was used to generate the reset. 0 = Do not generate a soft reset to the CPU core 1 = Pulse the SRESET CPU signal

Programming Notes

	7	6	5	4	3	2	1	0
Bit	IR15	IR14	IR13	IR12	IR11	IR10	IR9	IR8
Default	x	x	x	x	x	x	x	x
R/W	R	R	R	R	R	R	R	R

Bit	Name	Function
7	IR15	**Interrupt Request 15** 0 = IRQ15 is not asserted 1 = IRQ15 is asserted
6	IR14	**Interrupt Request 14** 0 = IRQ14 is not asserted 1 = IRQ14 is asserted
5	IR13	**Interrupt Request 13** 0 = IRQ13 is not asserted 1 = IRQ13 is asserted
4	IR12	**Interrupt Request 12** 0 = IRQ12 is not asserted 1 = IRQ12 is asserted
3	IR11	**Interrupt Request 11** 0 = IRQ11 is not asserted 1 = IRQ11 is asserted
2	IR10	**Interrupt Request 10** 0 = IRQ10 is not asserted 1 = IRQ10 is asserted
1	IR9	**Interrupt Request 9** 0 = IRQ9 is not asserted 1 = IRQ9 is asserted
0	IR8	**Interrupt Request 8** 0 = IRQ8 is not asserted 1 = IRQ8 is asserted

Programming Notes

This register provides a real-time status of the IRQ inputs to the Slave 8259. It is accessed by first writing a value of 0Ah to port A0h followed by a read-back of port A0h.

	7	6	5	4	3	2	1	0
Bit	IS15	IS14	IS13	IS12	IS11	IS10	IS9	IS8
Default	x	x	x	x	x	x	x	x
R/W	R	R	R	R	R	R	R	R

Bit	Name	Function
7	IS15	**IRQ15 In-Service** 0 = IRQ15 is not being serviced 1 = IRQ15 is being serviced
6	IS14	**IRQ14 In-Service** 0 = IRQ14 is not being serviced 1 = IRQ14 is being serviced
5	IS13	**IRQ13 In-Service** 0 = IRQ13 is not being serviced 1 = IRQ13 is being serviced
4	IS12	**IRQ12 In-Service** 0 = IRQ12 is not being serviced 1 = IRQ12 is being serviced
3	IS11	**IRQ11 In-Service** 0 = IRQ11 is not being serviced 1 = IRQ11 is being serviced
2	IS10	**IRQ10 In-Service** 0 = IRQ10 is not being serviced 1 = IRQ10 is being serviced
1	IS9	**IRQ9 In-Service** 0 = IRQ9 is not being serviced 1 = IRQ9 is being serviced
0	IS8	**IRQ8 In-Service** 0 = IRQ8 is not being serviced 1 = IRQ8 is being serviced

Programming Notes

The Slave In-Service Register (ISR) is accessed by writing a value of 0Bh to port A0h followed by a read-back from port A0h.

	7	6	5	4	3	2	1	0
Bit	Reserved			SLCT_ICW1	LTIM	ADI	SNGL	IC4
Default	x	x	x	x	x	x	x	x
R/W	W			W	W	W	W	W

Bit	Name	Function
7–5	Reserved	**Reserved** Must all be written to 0.
4	SLCT_ICW1	**Initialization Control Word 1 Select** Must be written to 1 to access ICW1 (D4 = 1 means the IOW to port A0h is Initialization Control Word 1).
3	LTIM	**Level Triggered Interrupt Mode** 0 = Edge sensitive IRQ detection 1 = Level sensitive IRQ detection
2	ADI	**Address Interval** Has no effect when the 8259 is used in x86 mode: 0 = Interrupt vectors are separated by 8 locations 1 = Interrupt vectors are separated by 4 locations In the ÉlanSC400 microcontroller design, this PC/AT Compatible bit is internally fixed to ‘1b’.
1	SNGL	**Single 8259** If this bit is set, the internal register pointer will skip ICW3 and point to ICW4, if ICW4 was selected to be programmed via bit 0 of this register. See the explanation for bit 0 of this register: 0 = Cascade mode, ICW3 will be expected 1 = Single 8259 in the system, ICW3 will not be expected In the ÉlanSC400 microcontroller design, this PC/AT Compatible bit is internally fixed to ‘0b’.
0	IC4	**Initialization Control Word 4** ICW4 is required. The 8259’s initialization control word (ICW) Registers 1–4 are programmed in sequence. Writing to port A0h with bit 4 = 1 causes the internal ICW1 Register to be written, and resets the PIC’s internal state machine and ICW register pointer. Slave ICW2–4 are programmed via port A1. Each time A1h is written to (following ICW1), the register pointer points to the next internal ICW register. ICW1 and ICW2 must always be programmed, but ICW3 and ICW4 need only be programmed under certain circumstances. This bit determines whether ICW4 is required to be explicitly programmed, or whether a value of 00h will serve for ICW4. If ICW4 is selected to be programmed, the register pointer will point to ICW4 after ICW3, and the PIC will expect software to provide an initialization value for it. 0 = Initialization Control Word 4 is cleared by writing Initialization Control Word 1, ICW4 will not be expected by the PIC 1 = ICW4 is not cleared by this write to Initialization Control Word 1, and software is expected to initialize ICW4 In the ÉlanSC400 microcontroller, this bit is internally fixed to ‘1b’.

Programming Notes

I/O writes to port A0h access different slave PIC registers based on bits 4–3 of the data that is written. See the following table:

Bit 4	Bit 3	Register Accessed
0	0	OCW2
0	1	OCW3
1	x	ICW1

	7	6	5	4	3	2	1	0
Bit	R SL EOI			SLCT_ICW1	IS_OCW3	LS[2–0]		
Default	x	x	x	x	x	x	x	x
R/W	W			W	W	W		

Bit	Name	Function
7–5	R SL EOI	**IRQ EOI and Priority Rotation Controls** 0 0 0 = Rotate in auto EOI mode (clear) 0 0 1 = Non-specific EOI 0 1 0 = No operation 0 1 1 = Specific EOI 1 0 0 = Rotate in auto EOI mode (set) 1 0 1 = Rotate on non-specific EOI command 1 1 0 = Set priority command 1 1 1 = Rotate on specific EOI command
4	SLCT_ICW1	**Initialization Control Word 1 Select** Software should write this bit to 0 to access OCW2 or OCW3.
3	IS_OCW3	**Access is OCW3** An I/O write to port A0h with this bit set indicates that OCW3 is to be accessed.
2–0	LS[2–0]	**Specific EOI Level Select** Interrupt level which is acted upon when the SL bit = ‘1b’ (see bits 7–5] below: 0 0 0 = IRQ8 0 0 1 = IRQ9 0 1 0 = IRQ10 0 1 1 = IRQ11 1 0 0 = IRQ12 1 0 1 = IRQ13 1 1 0 = IRQ14 1 1 1 = IRQ15

Programming Notes

I/O writes to port A0h access different slave PIC registers based on bits 4–3 of the data that is written. See the following table:

Bit 4	Bit 3	Register Accessed
0	0	OCW2
0	1	OCW3
1	x	ICW1

	7	6	5	4	3	2	1	0
Bit	Reserved	ESMM SMM		SLCT_ICW1	IS_OCW3	P	RR RIS	
Default	x	x	x	x	x	x	x	x
R/W		W		W	W	W	W	

Bit	Name	Function
7	Reserved	**Reserved** Software should write this bit to 0.
6–5	ESMM SMM	**Special Mask Mode** 0 x = No operation 1 0 = Reset special mask 1 1 = Set special mask In the ÉlanSC400 microcontroller design, the ESMM bit is internally fixed to ‘1b’.
4	SLCT_ICW1	**Initialization Control Word 1 Select** Software must write this bit to 0 to access OCW2 or OCW3.
3	IS_OCW3	**Access is OCW3** An I/O write to port A0h with this bit set indicates that OCW3 is to be accessed.
2	P	**PIC Poll Command** A system design can choose to use the PIC in a non-interrupting mode. In this case, the interrupt controller can be polled for the status of pending interrupts. In order to support this PC/AT incompatible mode of operation, the PIC supports a special poll command which is invoked by writing OCW3 with this bit set. 0 = Not poll command 1 = Poll command
1–0	RR RIS	**Status Register Select** 0 0 = No change from last state 0 1 = No change from last state 1 0 = Next port A0h read will return Interrupt Request Register (IRR) 1 1 = Next port A0h read will return In-Service Register (ISR)

Programming Notes

I/O writes to port A0h access different slave PIC registers based on bits 4–3 of the data that is written. See the following table:

Bit 4	Bit 3	Register Accessed
0	0	OCW2
0	1	OCW3
1	x	ICW1

	7	6	5	4	3	2	1	0
Bit	T7–T3					A10–A8		
Default	x	x	x	x	x	x	x	x
R/W	W					W		

Bit	Name	Function
7–3	T7–T3	**Bits 7–3 of Base Interrupt Vector Number for this PIC** Bits 2–0 are concatenated by PIC based on IRQ level. For example, these bits will be programmed to 00001b for the master PIC (IRQ0 generates int 8), and 01110b for the slave PIC (IRQ8 corresponds to int 70h) in a PC/AT Compatible system.
2–0	A10–A8	**A10–A8 of Interrupt Vector** Software should write these to 0 in a PC/AT-compatible system.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved					ID2–ID0		
Default	x	x	x	x	x	x	x	x
R/W	W							

Bit	Name	Function
7–3	Reserved	**Reserved** Software should write these bits to 0.
2–0	ID2–ID0	**Slave PIC ID 2–0** Program these bits with the binary slave 8259 ID (000b–111b) so that it responds to the proper address on the cascade bus for a PC/AT Compatible system. In this design, these bits are internally fixed to ‘010b’.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved			SFNM	BUF M/S		AEOI	PM
Default	x	x	x	x	x	x	x	x
R/W	W			W	W		W	W

Bit	Name	Function
7–5	Reserved	**Reserved** Software should write these bits to 0.
4	SFNM	**Special Fully Nested Mode Enable** 0 = Normal nested mode 1 = Special fully nested mode In the ÉlanSC400 microcontroller design, this PC/AT-compatible bit is internally fixed to ‘0b’.
3–2	BUF M/S	**Buffered Mode and Master/Slave Select** 0 x = Non buffered mode 1 0 = Buffered mode/slave 1 1 = Buffered mode/master In the ÉlanSC400 microcontroller, these PC/AT-compatible bits are internally fixed to ‘00b’.
1	AEOI	**Automatic EOI Mode** 0 = Normal EOI Interrupt handler must send an End of Interrupt command to the PIC(s) 1 = Auto EOI EOI is automatically performed after the second $\overline{\text{INTA}}$ signal from the CPU In the ÉlanSC400 microcontroller design, this PC/AT-compatible bit is internally fixed to ‘0b’.
0	PM	**Microprocessor Mode** 0 = 8080/8085 mode 1 = 8086 mode In the ÉlanSC400 microcontroller design, this PC/AT-compatible bit is internally fixed to ‘1b’.

Programming Notes

Slave 8259 Interrupt Mask Register (also known as Operation Control Word 1)

I/O Address 00A1h

	7	6	5	4	3	2	1	0
Bit	IM15	IM14	IM13	IM12	IM11	IM10	IM9	IM8
Default	x	x	x	x	x	x	x	x
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	IM15	**IRQ15 Mask** 0 = Unmask IRQ15 1 = Mask IRQ15
6	IM14	**IRQ14 Mask** 0 = Unmask IRQ14 1 = Mask IRQ14
5	IM13	**IRQ13 Mask** 0 = Unmask IRQ13 1 = Mask IRQ13
4	IM12	**IRQ12 Mask** 0 = Unmask IRQ12 1 = Mask IRQ12
3	IM11	**IRQ11 Mask** 0 = Unmask IRQ11 1 = Mask IRQ11
2	IM10	**IRQ10 Mask** 0 = Unmask IRQ10 1 = Mask IRQ10
1	IM9	**IRQ9 Mask** 0 = Unmask IRQ9 1 = Mask IRQ9
0	IM8	**IRQ8 Mask** 0 = Unmask IRQ8 1 = Mask IRQ8

Programming Notes

	7	6	5	4	3	2	1	0
Bit	DMA4MAR[16–1]							
Default	x	x	x	x	x	x	x	x
R/W	R/W							

Bit	Name	Function
7–0	DMA4MAR[16–1]	**DMA Channel 4 Memory Address** Generically speaking, this register holds the lower 16 bits of memory address for DMA Channel 4. Since the DREQ and DACK signals for this DMA channel are used to cascade to the slave DMA controller in PC/AT architecture, DMA Channel 4 is not used directly for DMA transfers, and this register has no real function. It is merely documented for completeness. For the same reason, there is no corresponding page register for DMA Channel 4.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	DMA4TC[15–0]							
Default	x	x	x	x	x	x	x	x
R/W	R/W							

Bit	Name	Function
7–0	DMA4TC[15–0]	**DMA Channel 4 Transfer Count (16-Bit Register)** Generically speaking, this register holds the 16-bit transfer count for DMA Channel 4. This register is written/read as two successive bytes for DMA Channel 4. Since the DREQ and DACK signals for this DMA channel are used to cascade to the slave DMA controller in PC/AT architecture, DMA Channel 4 is not used directly for DMA transfers, and this register has no real function. It is merely documented for completeness.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	DMA5MAR[16–1]							
Default	x	x	x	x	x	x	x	x
R/W	R/W							

Bit	Name	Function
7–0	DMA5MAR[16–1]	**Lower 16-bits of DMA Channel 5 Memory Address** This register is written/read via two successive I/O accesses to/from this port (always low byte followed by high byte immediately following a reset of the byte pointer at direct-mapped I/O address D8h). When set up for PC/AT compatibility (16-bit DMA transfers on the master DMA controller), this register holds address bits [16–1]. Address bit 0 is always ‘0b’ for these word-aligned transfers. Because of this, software must load this register with the desired memory address divided by 2. Use in conjunction with DMA Page Register for Channel 5 (I/O 8Bh) to form 24-bit memory address.

Programming Notes

The ÉlanSC400 microcontroller is capable of performing 26-bit DMA accesses using the extended DMA page registers that reside in CSC index registers D9h and DAh.

	7	6	5	4	3	2	1	0
Bit	DMA5TC[15–0]							
Default	x	x	x	x	x	x	x	x
	R/W							

Bit	Name	Function
7–0	DMA5TC[15–0]	**DMA Channel 5 Transfer Count (16-Bit Register)** This register is written/read via two successive I/O accesses to/from this port (always low byte followed by high byte immediately following a reset of the byte pointer at direct-mapped I/O address D8h). The actual number of transfers will be one more than specified by this register. When set up for PC/AT compatibility (16-bit DMA transfers on the master DMA controller), each transfer will be one word. Thus, a transfer count of FFFFh will result in a transfer of 128 Kbytes.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	DMA6MAR[16–1]							
Default	x	x	x	x	x	x	x	x
R/W	R/W							

Bit	Name	Function
7–0	DMA6MAR[16–1]	**Lower 16-bits of DMA Channel 6 Memory Address** This register is written/read via two successive I/O accesses to/from this port (always low byte followed by high byte immediately following a reset of the byte pointer at direct-mapped I/O address D8h). When set up for PC/AT compatibility (16-bit DMA transfers on the master DMA controller), this register hold address bits [16–1]. Address bit 0 is always ‘0b’ for these word-aligned transfers. Because of this, software must load this register with the desired memory address divided by 2. Use in conjunction with DMA Page Register for Channel 6 (I/O 89h) to form 24-bit memory address.

Programming Notes

The ÉlanSC400 microcontroller is capable of performing 26-bit DMA accesses using the extended DMA page registers that reside in CSC index registers D9h and DAh.

	7	6	5	4	3	2	1	0
Bit	DMA6TC[15–0]							
Default	x	x	x	x	x	x	x	x
R/W	R/W							

Bit	Name	Function
7–0	DMA6TC[15–0]	**DMA Channel 6 Transfer Count (16-Bit Register)** This register is written/read via two successive I/O accesses to/from this port (always low byte followed by high byte immediately following a reset of the byte pointer at direct-mapped I/O address D8h). The actual number of transfers will be one more than specified by this register. When set up for PC/AT compatibility (16-bit DMA transfers on the master DMA controller), each transfer will be one word. Thus, a transfer count of FFFFh will result in a transfer of 128 Kbytes.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	DMA7MAR[16–1]							
Default	x	x	x	x	x	x	x	x
R/W	R/W							

Bit	Name	Function
7–0	DMA7MAR[16–1]	**Lower 16-bits of DMA Channel 7 Memory Address** This register is written/read via two successive I/O accesses to/from this port (always low byte followed by high byte immediately following a reset of the byte pointer at direct-mapped I/O address D8h). When set up for PC/AT compatibility (16-bit DMA transfers on the master DMA controller), this register hold address bits [16–1]. Address bit 0 is always ‘0b’ for these word-aligned transfers. Because of this, software must load this register with the desired memory address divided by 2. Use in conjunction with DMA Page Register for Channel 7 (I/O 8Ah) to form 24-bit memory address.

Programming Notes

The ÉlanSC400 microcontroller is capable of performing 26-bit DMA accesses using the extended DMA page registers that reside in CSC index registers D9h and DAh.

	7	6	5	4	3	2	1	0
Bit	DMA7TC[15–0]							
Default	x	x	x	x	x	x	x	x
R/W	R/W							

Bit	Name	Function
7–0	DMA7TC[15–0]	**DMA Channel 7 Transfer Count (16-Bit Register)** This register is written/read as two successive bytes. The actual number of transfers will be one more than specified by this register. When set up for PC/AT compatibility (16-bit DMA transfers on the master DMA controller), each transfer will be one word. Thus, a transfer count of FFFFh will result in a transfer of 128 Kbytes.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	DMAR7	DMAR6	DMAR5	DMAR4	TC7	TC6	TC5	TC4
Default	0	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R	R

Bit	Name	Function
7	DMAR7	**Channel 7 DMA Request** 0 = Not pending 1 = Pending
6	DMAR6	**Channel 6 DMA Request** 0 = Not pending 1 = Pending
5	DMAR5	**Channel 5 DMA Request** 0 = Not pending 1 = Pending
4	DMAR4	**Channel 4 DMA Request** 0 = Not pending 1 = Pending
3	TC7	**Channel 7 Terminal Count** 0 = Not detected 1 = Detected
2	TC6	**Channel 6 Terminal Count** 0 = Not detected 1 = Detected
1	TC5	**Channel 5 Terminal Count** 0 = Not detected 1 = Detected
0	TC4	**Channel 4 Terminal Count** 0 = Not detected 1 = Detected

Programming Notes

Bits 3–0 of this register are read/reset. Any read from this direct mapped port clears all of these bits.

	7	6	5	4	3	2	1	0
Bit	DAKSEN	DRQSEN	WRTSEL	PRITYPE	COMPTIM	ENADMA	ADRHEN	MEM2MEM
Default	0	0	0	0	0	0	0	0
R/W	W	W	W	W	W	W	W	W

Bit	Name	Function
7	DAKSEN	**$\overline{\text{DACK}}$(n) Sense** This bit controls the polarity of all $\overline{\text{DACK}}$ outputs from the master DMA controller: 0 = Asserted Low 1 = Asserted high System logic external to the DMA controller expects the DMA controller to drive active Low $\overline{\text{DACK}}$ outputs. This bit must be written to ‘0b’ for proper system operation.
6	DRQSEN	**DREQ(n) Sense** This bit controls the polarity of all DREQ inputs to the master DMA controller: 0 = Asserted High 1 = Asserted Low System logic external to the DMA controller expects the DMA controller to respond to active High DREQ inputs. This bit must be written to ‘0b’ for proper system operation.
5	WRTSEL	**Write Selection Control** 0 = Late write selection 1 = Extended (early) write selection Enabling this feature will result in timing changes on the ISA bus that can violate the ISA specification.
4	PRITYPE	**Priority Type** 0 = Fixed priority 1 = Rotating priority
3	COMPTIM	**Compressed Timing** 0 = Normal timing 1 = Compressed timing Enabling this feature will result in timing changes on the ISA bus that can violate the ISA specification.
2	ENADMA	**Enable DMA Controller** 0 = Enabled 1 = DMA requests are ignored but DMA registers are available to the CPU The DMA controller must be disabled prior to programming it in order to prevent unintended transfers from occurring during the DMA controller programming operation. If an I/O DMA initiator asserts DREQ while the DMA controller is disable via this bit, abnormal system operation can occur.

Bit	Name	Function
1	ADRHEN	**Enable Channel 4 Address Hold Control (not supported)** IF bit 0 = 1, then: 0 = Disabled, Channel 0 memory address changes for each memory-to-memory transfer 1 = Enabled, Channel 0 memory address does not change for each memory-to-memory transfer (not supported by the system) ELSE this bit does nothing. Since bit 0 should always be written to 0, this bit will be a don't care after the DMA controller has been initialized.
0	MEM2MEM	**Enable Memory-to-Memory Transfer** 0 = Disabled 1 = Enabled (not supported by the system) Memory-to-memory DMA support is not provided in an AT compatible system. This bit should always be written to 0.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved					REQDMA	REQSEL1 REQSEL0	
Default	0	0	0	0	0	0	0	0
R/W						W	W	

Bit	Name	Function
7–3	Reserved	**Reserved** Software should write these bits to 0.
2	REQDMA	**DMA Request** 0 = Reset request bit for channel selected by bits 1–0 1 = Set request bit for channel selected by bits 1–0
1–0	REQSEL1 REQSEL0	**DMA Channel Select** Bits 1–0 select which DMA channel will latch bit 2 internally to assert or deassert a DMA request via software: 0 0 = Mask/unmask DMA Channel 4 mask per the REQDMA bit 0 1 = Mask/unmask DMA Channel 5 mask per the REQDMA bit 1 0 = Mask/unmask DMA Channel 6 mask per the REQDMA bit 1 1 = Mask/unmask DMA Channel 7 mask per the REQDMA bit

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved					CHMASK	MSKSEL1 MSKSEL0	
Default	0	0	0	0	0	0	0	0
R/W						W	W	

Bit	Name	Function
7–3	Reserved	**Reserved** Software should write these bits to 0.
2	CHMASK	**DMA Channel Mask** 0 = Clear mask bit for channel selected by bits 1–0 1 = Set mask bit for channel selected by bits 1–0
1–0	MSKSEL1 MSKSEL0	**DMA Channel Mask Select** Bits 1–0 select which DMA channel will latch bit 2 internally to mask or unmask the DRQ signal into the specified DMA channel: 0 0 = Set/reset DMA Channel 4 mask per the CHMASK bit 0 1 = Set/reset DMA Channel 5 mask per the CHMASK bit 1 0 = Set/reset DMA Channel 6 mask per the CHMASK bit 1 1 = Set/reset DMA Channel 7 mask per the CHMASK bit

Programming Notes

The same DMA channel masks can be controlled via DMA registers D4h, DCh, and DEh.

	7	6	5	4	3	2	1	0
Bit	TRNMOD		ADDDEC	AINIT	OPSEL1 OPSEL0		MODSEL1 MODSEL0	
Default	?	?	?	?	?	?	?	?
R/W	W		W	W	W		W	

Bit	Name	Function
7–6	TRNMOD	**Transfer Mode** 0 0 = Demand transfer mode 0 1 = Single transfer mode 1 0 = Block transfer mode 1 1 = Cascade mode
5	ADDDEC	**DMA Address Mode Decrement** 0 = Increment the DMA memory address after each transfer 1 = Decrement the DMA memory address after each transfer
4	AINIT	**Automatic Initialization Control** If enabled, the base address and transfer count registers are restored to the values they contained prior to performing the last DMA transfer. The channel selected by bits 0–1 of this register is then ready to perform another DMA transfer without processor intervention as soon as the next DRQ is detected. 0 = Auto initialization disabled 1 = Auto initialization enabled
3–2	OPSEL1 OPSEL0	**Operation Select** 0 0 = Verify mode DMA controller acts normally except that no I/O or memory commands are generated, and no data is transferred 0 1 = Write transfer Data will be transferred from a DMA-capable I/O device into system memory 1 0 = Read transfer Data will be transferred from system memory to a DMA-capable I/O device 1 1 = Reserved
1–0	MODSEL1 MODSEL0	**DMA Channel Select** Bits 7–2 of this register are latched internally for each channel. Bits 0–1 determine which of the channels will be programmed by the write to port D6h as follows: 0 0 = Select Channel 4 0 1 = Select Channel 5 1 0 = Select Channel 6 1 1 = Select Channel 7

Programming Notes

	7	6	5	4	3	2	1	0
Bit	MASTR_CBP							
Default	0	0	0	0	0	0	0	0
R/W	W							

Bit	Name	Function
7–0	MASTR_CBP	**Master DMA Clear Byte Pointer** The DMA Controller contains some 16-bit registers which are accessed by writing or reading consecutive 8-bit values to the same direct-mapped I/O location. A single byte pointer is used across the master DMA controller to determine which byte will be accessed. Any access to one of these 16-bit registers toggles the byte pointer between the low and high bytes. A write of any data to I/O address D8h clears this pointer so that the next access will be to the low byte.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	MASTR_RST							
Default	0	0	0	0	0	0	0	0
R/W	W							

Bit	Name	Function
7–0	MASTR_RST	**Master DMA Controller Reset** Writing data of any value to this address resets the DMA Controller to the same state as a hardware reset.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	MASTR_TMP							
Default	0	0	0	0	0	0	0	0
R/W	R							

Bit	Name	Function
7–0	MASTR_TMP	**Master DMA Controller Temporary Register** Used as a temporary storage buffer when doing memory-to-memory DMA. Note that the memory-to-memory DMA transfer is not supported. This register is included for compatibility purposes only.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	MSTR_MSK_RST							
Default	0	0	0	0	0	0	0	0
R/W	W							

Bit	Name	Function
7–0	MSTR_MSK_RST	**Master DMA Reset Mask** Writing any data to this address resets the Mask Register, thereby activating the four Master DMA channels.

Programming Notes

The same DMA channel masks can be controlled via DMA registers D4h, DCh, and DEh.

	7	6	5	4	3	2	1	0
Bit	Reserved				DIS7	DIS6	DIS5	DIS4
Default	0	0	0	0	1	1	1	1
R/W					W	W	W	W

Bit	Name	Function
7–4	Reserved	**Reserved** Software should write these bits to 0.
3	DIS7	**Disable DMA Channel 7 Mask** 0 = Enable DMA Channel 7 for servicing DMA requests 1 = Disable DMA Channel 7 from servicing DMA requests
2	DIS6	**Disable DMA Channel 6 Mask** 0 = Enable DMA Channel 6 for servicing DMA requests 1 = Disable DMA Channel 6 from servicing DMA requests
1	DIS5	**Disable DMA Channel 5 Mask** 0 = Enable DMA Channel 5 for servicing DMA requests 1 = Disable DMA Channel 5 from servicing DMA requests
0	DIS4	**Disable DMA Channel 4 Mask** 0 = Enable DMA Channel 4 for servicing DMA requests, also enables DMA cascading to the slave DMA controller on this channel 1 = Disable DMA Channel 4 from servicing DMA requests, also enables DMA cascading to the slave DMA controller on this channel

Programming Notes

The same DMA channel masks can be controlled via DMA registers D4h, DCh, and DEh.

	7	6	5	4	3	2	1	0
Bit	ALT_GA20							
Default	1	1	1	1	1	1	1	1
R/W	R/W							

Bit	Name	Function
7–0	ALT_GA20	**Alternate GateA20 Control** This register can be used to cause the same type of masking of the CPU A20 signal that was historically performed by an external SCP (System Control Processor) in a PC/AT Compatible system, but much faster. This control defaults to *not* forcing the propagation of A20: Dummy Read = Returns FFh, and forces the A20 signal to propagate. Dummy Write = Deasserts the forcing of the propagation of the A20 signal via this particular control, data value written is N/A.

Programming Notes

Bits 7–0: For software compatibility and other reasons, there are several sources of GateA20 control. These controls are effectively ORed together with the output of the OR gate driving the Enhanced Am486 microprocessor A20M pin. Therefore, A20 will propagate if ANY of the independent sources are forcing A20 to propagate.

	7	6	5	4	3	2	1	0
Bit	ALT_CPU_RST							
Default	0	0	0	0	0	0	0	0
R/W	R							

Bit	Name	Function
7–0	ALT_CPU_RST	**Alternate CPU Reset** Reading this register performs a soft reset on the CPU. This is the same type of CPU core reset that was typically generated via the external SCP on a PC/AT-compatible system. The 486 cache state, SMBASE, and ÉlanSC400 microcontroller indexed or direct mapped registers are not directly affected by this soft reset. Instruction execution will begin at FFFF0h as a result of this reset.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	PAR2DATA[7–0]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	PAR2DATA[7–0]	**Parallel Port 2 Data** Data read from or written to the parallel port data port. Write data to this port when the DIR bit of the Parallel Control Register is cleared. Read data from this port when the DIR bit is set. When in PC/AT Compatible mode, the data read back is the last byte written. When in Bidirectional mode, the data read back is that which is being driven by the remote device.

Programming Notes

There is only one parallel port on the ÉlanSC400 microcontroller, but it can be configured to have a base address of either 378h or 278h (LPT1/LPT2).

	7	6	5	4	3	2	1	0
Bit	BUSY	ACK	PE	SLCT	ERROR	Reserved		
Default	–	–	–	–	–	–	–	–
R/W	R	R	R	R	R			

Bit	Name	Function
7	BUSY	**Printer Busy** This bit is the inverse of the (BUSY/$\overline{\text{WAIT}}$) pin (active Low): 0 = Printer busy 1 = Printer is ready
6	ACK	**Printer Acknowledge** The printer pulses this line Low when it has received a byte of data. This bit follows the state of the $\overline{\text{ACK}}$ pin (active Low): 0 = Printer acknowledge 1 = No printer acknowledge
5	PE	**Paper End** This bit follows the state of the PE pin: 0 = Printer has paper 1 = Paper end (out of paper)
4	SLCT	**Printer Selected** This bit follows the state of the SLCT pin: 0 = Printer not selected 1 = Printer selected
3	ERROR	**Printer Error** This bit follows the state of $\overline{\text{ERROR}}$ pin (active Low): 0 = Printer error 1 = No printer error
2–0	Reserved	**Reserved**

Programming Notes

The default value for this register depends on the mode. For PC/AT Compatible mode, the default value for this register is ?????xxxb. For EPP mode, the default value is ?????xx0b.

	7	6	5	4	3	2	1	0
Bit	BUSY	ACK	PE	SLCT	ERROR	Reserved		
Default	–	–	–	–	–	–	–	–
R/W	R	R	R	R	R			

Bit	Name	Function
7	BUSY	**Printer Busy** This bit is the inverse of the (BUSY/$\overline{\text{WAIT}}$) pin (active Low): 0 = Printer busy 1 = Printer ready
6	ACK	**Printer Acknowledge** The printer pulses this line Low when it has received a byte of data. This bit follows the state of the $\overline{\text{ACK}}$ pin (active Low): 0 = Printer acknowledge 1 = No printer acknowledge
5	PE	**Paper End** This bit follows the state of the PE pin: 0 = Printer has paper 1 = Paper end (out of paper)
4	SLCT	**Printer Selected** This bit follows the state of the SLCT pin: 0 = Printer not selected 1 = Printer selected
3	ERROR	**Printer Error** This bit follows the state of $\overline{\text{ERROR}}$ pin (active Low): 0 = Printer error 1 = No printer error
2–0	Reserved	**Reserved**

Programming Notes

The default value for this register depends on the mode. For PC/AT Compatible mode, the default value for this register is ?????xxxb. For EPP mode, the default value is ?????xx0b.

	7	6	5	4	3	2	1	0
Bit	BUSY	ACK	PE	SLCT	ERROR	Reserved		EPP_TIMEO
Default	0	0	0	0	0	0	0	0
R/W	R	R	R	R	R			R

Bit	Name	Function
7	BUSY	**Printer Busy** This bit is the inverse of the (BUSY/$\overline{\text{WAIT}}$) pin (active Low): 0 = Printer busy 1 = Printer ready
6	ACK	**Printer Acknowledge** The printer pulses this line Low when it has received a byte of data. This bit follows the state of the $\overline{\text{ACK}}$ pin (active Low): 0 = Printer acknowledge 1 = No printer acknowledge
5	PE	**Paper End** This bit follows the state of the PE pin: 0 = Printer has paper 1 = Paper end (out of paper)
4	SLCT	**Printer Selected** This bit follows the state of the SLCT pin: 0 = Printer not selected 1 = Printer selected
3	ERROR	**Printer Error** This bit follows the state of $\overline{\text{ERROR}}$ pin (active Low): 0 = Printer error 1 = No printer error
2–1	Reserved	**Reserved**
0	EPP_TIMEO	**EPP Time-out Status** 0 = No time-out 1 = EPP cycle time-out occurred This bit is set if a time-out occurred only when EPP mode is enabled (bits 1–0 of the Parallel Port Configuration Register in the CSC indexed address space) = ‘01b’. This bit is reset when either the status register is read, or when EPP mode is enabled. An EPP time-out occurs when the BUSY pin remains inactive (low) for greater than 10s after either $\overline{\text{SLCTIN}}$ or $\overline{\text{AFDT}}$ go active Low in EPP mode.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved		DIR	IRQEN	SLCTIN	INIT	AUTOFDXT	STROBE
Default	0	0	0	0	0	0	0	0
R/W			R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7–6	Reserved	**Reserved** Software should write these bits to 0.
5	DIR	**Bidirectional Parallel Port Data Direction** When the parallel port is set to operate in either Bidirectional mode or EPP mode, this bit controls the data direction between host and peripheral. For a parallel port data write operation where either Bidirectional mode or EPP mode is selected: 0 = Data written to PD7–PD0 1 = Data written is latched only For a parallel port data read operation where either Bidirectional mode or EPP mode is selected: 0 = Internal data register read 1 = Data read from PD7–PD0 This bit is undefined when neither Bidirectional mode nor EPP mode is selected.
4	IRQEN	**Printer IRQ Enable** Clearing this bit clears any pending interrupts: 0 = Disable printer IRQ 1 = Enable printer IRQ
3	SLCTIN	**Select Printer Signal Control** This bit is the inverse of the $\overline{\text{SLCTIN}}$ pin. 0 = The $\overline{\text{SLCTIN}}$ pin is a logic 1 1 = The $\overline{\text{SLCTIN}}$ pin a logic 0
2	INIT	**Printer Reset Signal Control** 0 = Hold printer in reset 1 = Release printer from reset, this bit follows the $\overline{\text{INIT}}$ pin (active Low)
1	AUTOFDXT	**Auto Line Feed Signal Control** This bit is the inverse of the $\overline{\text{AFDT}}$ pin: 0 = $\overline{\text{AFDT}}$ pin is a logic 1 1 = $\overline{\text{AFDT}}$ pin is a logic 0 When connected to a printer, setting this bit causes the printer to automatically insert a line feed when it sees a carriage return (ASCII 13) character.
0	STROBE	**Printer Port Strobe Signal Control** This bit is the inverse of the $\overline{\text{STRB}}$ pin: 0 = $\overline{\text{STRB}}$ pin not active 1 = $\overline{\text{STRB}}$ pin active

Programming Notes

	7	6	5	4	3	2	1	0
Bit	EPP2ADDR[7–0]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	EPP2ADDR[7–0]	**Parallel Port 2 EPP Address** When this port is written to, the value written appears on the parallel port data lines. In addition, an automatic address strobe is generated to allow the peripheral EPP device to latch the address.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	EPP2DATA[31–0]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	R/W	Function
31–0	EPP2DATA[31–0]	R/W	**Parallel Port 2 EPP 32-bit Data** A 32-bit I/O write to 27Ch will cause four back-to-back 8-bit bus cycles to occur to the four EPP Data Registers 27Ch–27Fh. Four bytes of data will be presented to the parallel port data lines in succession starting with the data written to 27Ch. An EPP data strobe will be automatically generated for each of the bus cycles.

Programming Notes

Bits 31–0: A 16-bit I/O write to 27Ch will cause two back-to-back 8-bit bus cycles to occur to the EPP Data Registers 27Ch and 27Dh. Two bytes of data will be presented to the parallel port data lines in succession starting with the data written to 27Ch. An EPP data strobe will be automatically generated for each of the bus cycles.

An 8-bit I/O write to 27Ch will cause a single 8-bit bus cycle to occur to the EPP Data Register at 27Ch. The single byte of data will be presented to the parallel port data lines along with an automatically generated EPP data strobe.

In common practice, all write accesses to the parallel port 2 EPP Data Register (x8, x16, or x32 I/O instructions) should be directed to port 27Ch.

	7	6	5	4	3	2	1	0
Bit	COM2THR[7–0]							
Default	0	0	0	0	0	0	0	0
R/W	W							

Bit	Name	Function
7–0	COM2THR[7–0]	**COM2 Transmit Holding Register** When COM2 DLAB = 0, and the transmitter is not busy, write the byte to transmit to this register.

Programming Notes

There is only one serial port on the ÉlanSC400 microcontroller, but it can be configured to have a base address of either 3F8h or 2F8h (COM1/COM2).

	7	6	5	4	3	2	1	0
Bit	COM2RBR[7–0]							
Default	0	0	0	0	0	0	0	0
R/W	R							

Bit	Name	Function
7–0	COM2RBR[7–0]	**COM2 Receive Buffer** When COM2 DLAB = 0, and a received byte is available, read received byte from this register.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	COM2_DIV[7–0]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	COM2_DIV[7–0]	**COM2 Baud Clock Divisor Latch** When COM2 DLAB = 1, this register holds the least significant byte of a 16-bit baud rate clock divisor that is used to generate the 16x baud clock.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	COM2_DIV[15–8]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	COM2_DIV[15–8]	**COM2 Baud Clock Divisor Latch** When COM2 DLAB = 1, this register holds the most significant byte of a 16-bit baud rate clock divisor that is used to generate the 16x baud clock.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved				EMSI	ERLSI	ETHREI	ERDAI
Default	0	0	0	0	0	0	0	0
R/W					R/W	R/W	R/W	R/W

Bit	Name	Function
7–4	Reserved	**Reserved** Software should write these bits to 0.
3	EMSI	**Enable Modem Status Interrupt** 0 = Disable modem status interrupt 1 = Enable modem status interrupt
2	ERLSI	**Enable Receiver Line Status Interrupt** 0 = Disable receiver line status interrupt 1 = Enable receiver line status interrupt
1	ETHREI	**Enable Transmitter Holding Register Empty Interrupt** 0 = Disable transmitter holding register empty interrupt 1 = Enable transmitter holding register empty interrupt
0	ERDAI	**Enable Received Data Available Interrupt** 0 = Disable data available interrupt in 16450-compatible mode; in 16550-compatible mode, this bit also disables time-out interrupts 1 = Enable received data available interrupt in 16450-compatible mode; in 16550-compatible mode, this bit also enables time-out interrupts More detail on time-out interrupts can be found in the Interrupt Identification Register (I/O address 2FAh/3FAh) description.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	FIFO1 FIFO0		Reserved		ID2 ID1 ID0			IP
Default	0	0	0	0	1	1	1	1
R/W	R				R			R

Bit	Name	Function	Priority
7–6	FIFO1 FIFO0	**FIFO Feature Presence Indication** FIFO is only present when 16550-compatible mode is enabled): 0 0 = No significance 0 1 = No significance 1 0 = 16450-compatible mode is enabled 1 1 = 16550-compatible mode is enabled	
5–4	Reserved	**Reserved** These bits always read back ‘00b’.	
3–1	ID2 ID1 ID0	**Interrupt Identification Bit Field** 0 0 0 = Modem status	Fourth (Lowest)
		0 0 1 = Transmitter Holding Register Empty/Transmit FIFO Empty (16550-compatible mode)	Third
		0 1 0 = Received Data Available/Receiver FIFO trigger (16550-compatible mode)	Second
		0 1 1 = Receive Line Status	First (Highest)
		1 0 0 = Not used	
		1 0 1 = Not used	
		1 1 0 = FIFO time-out	Second
		1 1 1 = Not used	
		In 16450-compatible mode, bit 3 always reads back ‘0b’. See the Modem Status Register (MSR) and Line Status Register (LSR) for more detail on interrupt events.	
		If two interrupt sources are pending simultaneously, only the highest priority interrupt (as defined by the rightmost column for bits 3–1) will be indicated by bits 3–1 of the Interrupt Identification Register (IIR). When the interrupt source is cleared, a subsequent read from the Interrupt Identification Register (IIR) will return the next highest priority interrupt source. Bits 3–1 have no meaning if bit 0 = ‘1b’.	
		A FIFO time-out occurs when the receive FIFO is not empty, and more than four continuous character-times have transpired without more data being placed into or read out of the receive FIFO. Reading a character from the receive FIFO clears the time-out interrupt.	

Bit	Name	Function	Priority
0	IP	**Serial Port Interrupt Pending (Active Low)** 0 = Interrupt pending 1 = No interrupt pending	

Programming Notes

	7	6	5	4	3	2	1	0
Bit	RFRT[1–0]		Reserved		Reserved	TFCLR	RFCLR	FIFOEN
Default	0	0	0	0	0	0	0	0
R/W	W				W	W	W	W

Bit	Name	R/W	Function
7–6	RFRT[1–0]	W	**Receiver FIFO Register Trigger Bits** When in 16550-compatible mode, this bit field specifies the trigger level at which the Interrupt Identification Register will report that a received data available interrupt is pending. If received data available interrupts are enabled in the IER, the system will be interrupted when the receive FIFO fills to the trigger level per the following table: 0 0 = 1 byte 0 1 = 4 bytes 1 0 = 8 bytes 1 1 = 14 bytes When the data in the receive FIFO falls below this trigger level, the interrupt will be cleared.
5–4	Reserved		**Reserved** Software should write these bits to 0.
3	Reserved	W	**Reserved** Software should write this bit to 1.
2	TFCLR	W	**Transmitter FIFO Buffer Clear** Writing a ‘1b’ to this bit position clears the transmit FIFO and resets the transmit FIFO counter logic. It does not clear the Transmitter Shift Register. This bit is cleared by either a read of the Interrupt ID Register, or by a write to the Transmit Holding Register.
1	RFCLR	W	**Receiver FIFO Buffer Clear** Writing a ‘1b’ to this bit position clears the receive FIFO and resets the receive FIFO counter logic. It does not clear the Receiver Shift Register. This bit is self-clearing, and does not need to be reset to 0 under software control.
0	FIFOEN	W	**FIFO Enable (16550-Compatible Mode Enable)** 0 = Causes UART to enter 16450-compatible mode Disables accesses to receive and transmit FIFOs and all FIFO control bits, except this bit. 1 = Causes UART to enter 16550-compatible mode Enables receive and transmit FIFOs and enables accesses to other FIFO control bits. The main difference between a 16450-compatible mode and 16550-compatible mode is the addition of transmit and receive FIFOs in the 16550-compatible mode This bit must be ‘1b’ when other FIFO control register bits are written to or they will not be programmed. Any mode switch will clear both FIFOs. The FIFOs must be enabled for the IrDA interface to operate in High-Speed IrDA mode.

Programming Notes

The contents of this write-only register can be read back via the Chip Setup and Control (CSC) indexed register D3h. The on-board UART can be mapped to only one of COM1 or COM2 at any time, and the Shadow register at CSC index D3h serves either configuration.

	7	6	5	4	3	2	1	0
Bit	DLAB	SB	SP	EPS	PE	STP	WLB0 WLB1	
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

Bit	Name	Function
7	DLAB	**Divisor Latch Access Bit** Set this bit to gain access to the baud rate divisor latches for this COM port. Clear this bit to mux in the Transmit Holding Register and Receive Buffer Register at port xF8h, and the Interrupt Enable Register at port xF9h.
6	SB	**Set Break Enable** Setting this bit causes a break condition to be transmitted to the receiving UART. 0 = Disable set break 1 = Force serial output to spacing state (logic '0') regardless of other transmitter activity The break control acts on the SOUT pin only and has no other effect on the transmitter logic.
5	SP	**Stick Parity Enable** Forces the parity bit to always be 0 or 1 versus dynamically changing it so that there are an odd or even number of 1 bits in the transmitted data. If this bit is 0, then normal parity is used if parity is enabled. If bits 5,4, and 3 = 1, the parity bit is generated/checked as a logical 1. If bits 5 and 3 = 1 and bit 4 = 0, the parity bit is generated/checked as a logical 0.
4	EPS	**Even Parity Select** Parity must be enabled via bit 3 for this bit to have meaning: 0 = Odd parity The parity bit will be manipulated to force an odd number of 1 bits in the transmitted data and the same condition will be checked for in the received data. 1 = Even parity The parity bit will be manipulated to force an even number of 1 bits in the transmitted data and the same condition will be checked for in the received data. Start/stop bits are not included in the parity generation/checking scheme.
3	PE	**Asynchronous Data Parity Enable** Causes a parity bit to be generated in the transmitted data and to be checked in the received data. The parity bit is located between the last data word bit and the first stop bit in the bit stream.
2	STP	**Stop Bits** Based on character length from bits 1–0 above. If character length = 5: 0 = 1 stop bit 1 = 1.5 stop bits If character length = 6, 7, or 8: 0 = 1 stop bit 1 = 2 stop bits
1–0	WLB0 WLB1	**Transmit/Receive Character Length** 0 0 = 5 bits 0 1 = 6 bits 1 0 = 7 bits 1 1 = 8 bits

	7	6	5	4	3	2	1	0
Bit	Reserved			LOOP	OUT2	OUT1	RTS	DTR
Default	0	0	0	0	0	0	0	0
R/W				R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7–5	Reserved	**Reserved** Software should write these bits to 0.
4	LOOP	**Loopback (Diagnostic Mode) Enable** 1 = Enabled The following internal connections are made by setting this diagnostic bit: $\overline{RTS}$ is internally connected to $\overline{CTS}$ $\overline{DTR}$ is internally connected to $\overline{DSR}$ $\overline{OUT1}$ is internally connected to $\overline{RI}$ $\overline{OUT2}$ is internally connected to $\overline{DCD}$ 0 = Loopback mode is disabled In addition, the SOUT bit is driven High, and the SIN input line is blocked. The Transmit Shift Out Register is directly connected to the receive shift in register. In addition, the $\overline{DTR}$, $\overline{RTS}$, $\overline{OUT1}$ and $\overline{OUT2}$ signals are forced inactive. Modem control interrupts can be forced by enabling the appropriate bit in the IER, and asserting one of the bits 3–0 in loopback mode.
3	OUT2	**Enable COM2 Interrupts** This bit controls the $\overline{OUT2}$ signal which is used as a master enable for COM2 interrupts. If this bit is not set, no COM port 2 IRQs will be felt at the Programmable Interrupt Controller (PIC).
2	OUT1	**$\overline{OUT1}$ Control** This bit controls the $\overline{OUT1}$ signal which is not tied to anything. It is provided for PC/AT compatibility and can be used as part of the loopback diagnostics. Other than that, it has no effect on system operation and can be used as a scratch pad during normal system operation.
1	RTS	**Request to Send** 0 = Deassert the Request To Send signal ($\overline{RTS}$) 1 = Assert the Request To Send signal ($\overline{RTS}$)
0	DTR	**Data Terminal Ready** 0 = Deassert the Data Terminal Ready signal ($\overline{DTR}$) 1 = Assert the Data Terminal Ready signal ($\overline{DTR}$)

Programming Notes

	7	6	5	4	3	2	1	0
Bit	16550_ERR	TEMT	THRE	BI	FE	PE	OE	DR
Default	0	1	1	0	0	0	0	0
R/W	R	R	R	R	R	R		R

Bit	Name	Function
7	16550_ERR	**16550-Mode Error** In 16450-compatible mode, this bit reads back ‘0b’. In 16550-compatible mode, when this bit is set it indicates at least one parity error, framing error, or break condition is present in the receive FIFO. This bit is cleared by a read from the Line Status Register if there are no subsequent errors in the FIFO.
6	TEMT	**Transmitter Empty Indicator** In 16450-compatible mode, this bit is set when both the Transmit Holding Register and the Transmit Shift Register are empty. In 16550-compatible mode, this bit is set when both the Transmit Shift Register and transmit FIFO are empty.
5	THRE	**Transmitter Holding Register (16450-Compatible Mode) or Transmit FIFO Empty (16550-Compatible Mode)** In 16450-compatible mode, this bit indicates that the Transmit Holding Register is ready to accept a new character. This bit is automatically reset by a write to the Transmit Holding Register when in 16450-compatible mode. In 16550-compatible mode, this bit indicates that the transmit FIFO is completely empty. When in 16550-compatible mode, this interrupt will be cleared when either the transmit FIFO is written to, or when the Interrupt Identification Register is read. This bit can be used to generate an interrupt if programmed to do so via the Interrupt Enable Register.
4	BI	**Break Indicator** In 16450-compatible mode, this bit is set when the UART detects that the sending UART is transmitting a break condition for a period longer than the time it takes to receive start, data, parity, and stop bits. In 16550-compatible mode, this bit is set when an entire word (start, data, parity, stop) that has been received with break indication present into the receive FIFO is at the top of the FIFO. Only one break indication will be loaded into the FIFO regardless of the duration of the break condition. A new character will not be loaded into the FIFO until the next valid start bit is detected. This latched status bit is automatically cleared by a read from the Line Status Register.
3	FE	**Framing Error** In 16450-compatible mode, this bit is set to indicate that a received character did not have a valid stop bit. In 16550-compatible mode, this bit is set when a character that has been received with a framing error into the receive FIFO is at the top of the FIFO. This latched status bit is automatically cleared by a read from the Line Status Register.
2	PE	**Parity Error** In 16450-compatible mode, this bit is set upon receipt of data with incorrect parity. In 16550-compatible mode, this bit is set when a character that has been received with bad parity into the receive FIFO is at the top of the FIFO. This latched status bit is automatically cleared by a read from the Line Status Register.

Bit	Name	Function
1	OE	**Overrun Error** In 16450-compatible mode, this bit is set if a new character is received into the receiver buffer before a previous character was read, thus resulting in lost data. In 16550-compatible mode, this bit is set if a new character is completely received into the Shift Register when the FIFO is already 100% full. Data in the FIFO is not overwritten by this overrun. The data in the Shift Register will be lost when the next character is received. This latched status bit is automatically cleared by a read from the Line Status Register.
0	DR	**Data Ready** In 16450-compatible mode, when this bit is set, a character has been received and placed in the Receiver Buffer Register. This bit is automatically cleared by reading the Receiver Buffer Register. In 16550-compatible mode, when this bit is set, a character has been received and placed in the receive FIFO. This bit is automatically cleared by reading the Receiver FIFO.

Programming Notes

When a receiver line status interrupt is enabled and detected, bits 1 through 4 above will indicate the reason for the interrupt. The status bits are valid even when the receiver line status interrupt is not enabled.

	7	6	5	4	3	2	1	0
Bit	DCD	RI	DSR	CTS	DDCD	TERI	DDSR	DCTS
Default	x	x	x	x	0	0	0	0
R/W	R	R	R	R	R	R	R	R

Bit	Name	Function
7	DCD	**Data Carrier Detect** 0 =DCD input signal is deasserted 1 = DCD input signal is asserted If in loopback mode, this bit tracks bit 3 of the Modem Control Register.
6	RI	**Ring Indicator** 0 =RI input signal is deasserted 1 = RI input signal is asserted If in loopback mode, this bit tracks bit 2 of the Modem Control Register.
5	DSR	**Data Set Ready** 0 =DSR input signal is deasserted 1 = DSR input signal is asserted If in loopback mode, this bit tracks bit 0 of the Modem Control Register.
4	CTS	**Clear To Send** 0 =CTS input signal is deasserted 1 = CTS input signal is asserted If in loopback mode, this bit tracks bit 1 of the Modem Control Register.
3	DDCD	**Delta Data Carrier Detect** 0 = Indicates that the DCD signal has not changed since the Modem Status Register was last read 1 = Indicates that the $\overline{\text{DCD}}$ signal changed since the Modem Status Register MSR) was last read
2	TERI	**Trailing Edge Ring Indicator** 0 = Indicates that the $\overline{\text{RI}}$ signal has not changed from an active to an inactive state since the Modem Status Register was last read 1 = Indicates that the $\overline{\text{RI}}$ signal changed from an active to an inactive state since the Modem Status Register (MSR) was last read
1	DDSR	**Delta Data Set Ready** 0 = Indicates that the $\overline{\text{DSR}}$ signal has not changed since the Modem Status Register was last read 1 = Indicates that the $\overline{\text{DSR}}$ signal changed since the Modem Status Register (MSR) was last read
0	DCTS	**Delta Clear To Send** 0 = Indicates that the $\overline{\text{CTS}}$ signal has not changed since the Modem Status Register was last read 1 = Indicates that the $\overline{\text{CTS}}$ signal changed since the Modem Status Register (MSR) was last read

Programming Notes

Bits 7–4: Real-time status indicators for the indicated inputs to the UART.

Bits 3–0: Latched status bits that will generate an interrupt if modem status interrupts are unmasked in the Interrupt Enable Register (IER). Reading the Modem Status Register (MSR) clears these bits and their associated interrupt.

	7	6	5	4	3	2	1	0
Bit	SCRATCH2[7–0]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	SCRATCH2[7–0]	**Scratch Bit** General-purpose I/O location, not required for serial data transfer.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	PAR1_DATA[7–0]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	PAR1_DATA[7–0]	**Parallel Port 1 Data** Data read from or written to the parallel port data register. Write data to this register when the DIR bit of the Parallel Control Register is cleared. Read data from this register when the DIR bit is set. When in PC/AT Compatible mode, the data read back is the last byte written. When in Bidirectional mode, the data read back is that which is being driven by the remote device.

Programming Notes

There is only one parallel port on the ÉlanSC400 microcontroller, but it can be configured to have a base address of either 378h or 278h (LPT1/LPT2).

	7	6	5	4	3	2	1	0
Bit	BUSY	ACK	PE	SLCT	ERROR	Reserved		
Default	0	0	0	0	0	0	0	0
R/W	R	R	R	R	R			

Bit	Name	Function
7	BUSY	**Printer Busy** This bit is the inverse of the (BUSY/$\overline{\text{WAIT}}$) pin (active Low): 0 = Printer busy 1 = Printer ready
6	ACK	**Printer Acknowledge** The printer pulses this line Low when it has received a byte of data. This bit follows the state of the $\overline{\text{ACK}}$ pin (active Low): 0 = Printer acknowledge 1 = No printer acknowledge
5	PE	**Paper End** This bit follows the state of the PE pin: 0 = Printer has paper 1 = Paper end (out of paper)
4	SLCT	**Printer Selected** This bit follows the state of the SLCT pin: 0 = Printer not selected 1 = Printer selected
3	ERROR	**Printer Error** This bit follows the state of $\overline{\text{ERROR}}$ pin (active Low): 0 = Printer error 1 = No printer error
2–0	Reserved	**Reserved**

Programming Notes

	7	6	5	4	3	2	1	0
Bit	BUSY	ACK	PE	SLCT	ERROR	Reserved		
Default	–	–	–	–	–	–	–	–
R/W	R	R	R	R	R			

Bit	Name	Function
7	BUSY	**Printer Busy** This bit is the inverse of the (BUSY/$\overline{\text{WAIT}}$) pin (active Low): 0 = Printer busy 1 = Printer ready
6	ACK	**Printer Acknowledge** The printer pulses this line Low when it has received a byte of data. This bit follows the state of the $\overline{\text{ACK}}$ pin (active Low): 0 = Printer acknowledge 1 = No printer acknowledge
5	PE	**Paper End** This bit follows the state of the PE pin: 0 = Printer has paper 1 = Paper end (out of paper)
4	SLCT	**Printer Selected** This bit follows the state of the SLCT pin: 0 = Printer not selected 1 = Printer selected
3	ERROR	**Printer Error** This bit follows the state of $\overline{\text{ERROR}}$ pin (active Low): 0 = Printer error 1 = No printer error
2–0	Reserved	**Reserved**

Programming Notes

The default value for this register depends on the mode. For PC/AT Compatible mode, the default value for this register is ?????xxxb. For EPP mode, the default value is ?????xx0b.

	7	6	5	4	3	2	1	0
Bit	BUSY	ACK	PE	SLCT	ERROR	Reserved		EPP_TIMEO
Default	–	–	–	–	–	–	–	–
R/W	R	R	R	R	R			R

Bit	Name	Function
7	BUSY	**Printer Busy** This bit is the inverse of the (BUSY/$\overline{\text{WAIT}}$) pin (active Low): 0 = Printer busy 1 = Printer ready
6	ACK	**Printer Acknowledge** The printer pulses this line Low when it has received a byte of data. This bit follows the state of the $\overline{\text{ACK}}$ pin (active Low): 0 = Printer acknowledge 1 = No printer acknowledge
5	PE	**Paper End** This bit follows the state of the PE pin: 0 = Printer has paper 1 = Paper end (out of paper)
4	SLCT	**Printer Selected** This bit follows the state of the SLCT pin: 0 = Printer not selected 1 = Printer selected
3	ERROR	**Printer Error** This bit follows the state of $\overline{\text{ERROR}}$ pin (active Low): 0 = Printer error 1 = No printer error
2–1	Reserved	**Reserved**
0	EPP_TIMEO	**EPP Time-out Status** 0 = No time-out 1 = EPP cycle time-out occurred This bit is set if a time-out occurred only when EPP mode is enabled (bits 1–0 of the Parallel Port Configuration Register in the ÉlanSC400 microcontroller-specific indexed address space) = ‘01b’. This bit is reset when either the status register is read, or when EPP mode is enabled. An EPP time-out occurs when the BUSY pin remains inactive (Low) for greater than 10µs after either $\overline{\text{SLCTIN}}$ or $\overline{\text{AFDT}}$ go active (Low) in EPP mode.

Programming Notes

The default value for this register depends on the mode. For PC/AT Compatible mode, the default value for this register is ?????xxxb. For EPP mode, the default value is ?????xx0b.

	7	6	5	4	3	2	1	0
Bit	Reserved		DIR	IRQEN	SLCTIN	INIT	AUTOFDXT	STROBE
Default	0	0	0	0	0	0	0	0
R/W			R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7–6	Reserved	**Reserved** Software should write these bits to 0.
5	DIR	**Bidirectional Parallel Port Data Direction** When the parallel port is set to operate in either Bidirectional mode or EPP mode, this bit controls the data direction between host and peripheral. For a parallel port data write operation where either Bidirectional mode or EPP mode is selected: 0 = Data written to PD7–PD0 1 = Data written is latched only For a parallel port data read operation where either Bidirectional mode or EPP mode is selected: 0 =Internal data register read 1 = Data read from PD7–PD0] This bit is undefined when neither Bidirectional mode nor EPP mode is selected.
4	IRQEN	**Printer IRQ Enable** Clearing this bit clears any pending interrupts. 0 = Disable printer IRQ 1 = Enable printer IRQ
3	SLCTIN	**Select Printer Signal Control** This bit is the inverse of the $\overline{\text{SLCTIN}}$ pin. 0 = The $\overline{\text{SLCTIN}}$ pin is a logic 1 1 = The $\overline{\text{SLCTIN}}$ pin a logic 0
2	INIT	**Printer Reset Signal Control** 0 = Hold printer in reset 1 = Release printer from reset, this bit follows the $\overline{\text{INIT}}$ pin (active Low)
1	AUTOFDXT	**Auto Line Feed Signal Control** This bit is the inverse of the $\overline{\text{AFDT}}$ pin: 0 = $\overline{\text{AFDT}}$ pin is a logic 1 1 = $\overline{\text{AFDT}}$ pin is a logic 0 When connected to a printer, setting this bit causes the printer to automatically insert a line feed when it sees a carriage return (ASCII 13) characters.
0	STROBE	**Printer Port Strobe Signal Control** This bit is the inverse of the $\overline{\text{STRB}}$ pin: 0 = $\overline{\text{STRB}}$ pin not active 1 = $\overline{\text{STRB}}$ pin active

Programming Notes

	7	6	5	4	3	2	1	0
Bit	EPP1_ADDR[7–0]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	EPP1_ADDR[7–0]	**Parallel Port 1 EPP Address** When this port is written to, the value written appears on the parallel port data lines. In addition, an automatic address strobe is generated to allow the peripheral EPP device to latch the address.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	EPP1_DATA[31–0]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
31–0	EPP1_DATA[31–0]	**Parallel Port 1 EPP 32-bit Data** A 32-bit I/O write to 37Ch will cause four back-to-back 8-bit bus cycles to occur to the four EPP Data Registers 37Ch–37Fh. Four bytes of data will be presented to the parallel port data lines in succession starting with the data written to 37Ch. An EPP data strobe will be automatically generated for each of the bus cycles.

Programming Notes

Bits 31–0: A 16-bit I/O write to 37Ch will cause two back-to-back 8-bit bus cycles to occur to the EPP Data Registers 37Ch and 37Dh. Two bytes of data will be presented to the parallel port data lines in succession starting with the data written to 37Ch. An EPP data strobe will be automatically generated for each of the bus cycles.

An 8-bit I/O write to 37Ch will cause a single 8-bit bus cycle to occur to the EPP Data Register at 37Ch. The single byte of data will be presented to the parallel port data lines along with an automatically generated EPP data strobe.

In common practice, all write accesses to the parallel port 1 EPP Data Register (x8, x16, or x32 I/O instructions) should be directed to port 37Ch.

	7	6	5	4	3	2	1	0
Bit	MDA_INDX[7–0]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	MDA_INDX[7–0]	**MDA Index** Graphics Controller Index Register when operating in Monochrome Display Adapter/Hercules Display Adapter (MDA/HGA) modes.

Programming Notes

This register is visible only when the Comp_Mod bit in the Internal Graphics Control Register A (CSC index DDh[0]) is set (MDA compatibility).

	7	6	5	4	3	2	1	0
Bit	MDA_DATA[7–0]							
Default	x	x	x	x	x	x	x	x
R/W	R/W							

Bit	Name	Function
7–0	MDA_DATA[7–0]	**MDA Data** Graphics Controller Data Register when operating in Monochrome Display Adapter/Hercules Display Adapter (MDA/HGA) modes.

Programming Notes

This register is visible only when the Comp_Mod bit in the Internal Graphics Control Register A (CSC index DDh[0]) is set (MDA compatibility). In addition, if the HGA extensions to MDA are required, CRTC index 52h[4] must be set. The B0000h and B8000h pages specified in the description for bit 7 assume that the internal graphics controller has had its display buffer base address set to reside at B0000h. Generally speaking, setting bit 7 of this register when in HGA mode will functionally add 32 K to the shared memory base address (must be configured via graphics index registers 4Dh/4Eh).

	7	6	5	4	3	2	1	0
Bit	HGA_PG_SEL	Reserved	TXTBLNK	Reserved	VIDCON	Reserved	HGA_GR_EN	TXTCON
Default	0	0	0	0	0	0	0	0
R/W	R/W		R/W		R/W		R/W	R/W

Bit	Name	Function
7	HGA_PG_SEL	**HGA Page Select (HGA Mode Only)** 0 = Use B0000h as the start of the display buffer 1 = Use B8000h as the start of the display buffer
6	Reserved	**Reserved** Software should write this bit to 0.
5	TXTBLNK	**Text Blink Control** 0 = Blinking attribute disabled 1 = Blinking attribute enabled (attribute byte bit 7 will control blinking)
4	Reserved	**Reserved** Software should write this bit to 0.
3	VIDCON	**Video Blanking Control** 0 = Video is blanked 1 = Video is not blanked
2	Reserved	**Reserved** Software should write this bit to 0.
1	HGA_GR_EN	**HGA Graphics Enable (HGA Mode Only)**
0	TXTCON	**MDA/HGA Column Select** This bit is provided for compatibility purposes only. It does not actually set the column width. This must be done via the CRTC index registers. The MDA/HGA legacy values for this bit are: 0 = 40x25 text mode 1 = 80x25 text mode

Programming Notes

This register is visible only when the Comp_Mod bit in the Internal Graphics Control Register A (chip configuration register DDh[0]) is set (MDA compatibility). In addition, if the HGA extensions to MDA are required, CRTC index 52h[4] must be set. Also note that the B0000h and B8000h pages specified in the description for bit 7 assume that the internal graphics controller has had its display buffer base address set up to reside at B0000h. Generally speaking, setting bit 7 of this register when in HGA mode will functionally add 32 Kbytes to the shared memory base address (must be configured via graphics index registers 4Dh/4Eh).

	7	6	5	4	3	2	1	0
Bit	Reserved				VERTRET	Reserved		DMSTAT
Default	1	1	1	1	x	0	0	x
R/W	R				R	R		R

Bit	Name	Function
7–4	Reserved	**Reserved** Reads back ‘1111b’.
3	VERTRET	**Vertical Retrace Status (simulated vertical sync)** 0 = Raster is not in vertical retrace 1 = Raster is in vertical retrace
2–1	Reserved	**Reserved**
0	DMSTAT	**Display-Memory Access Status (Simulated Horizontal Sync)** This bit is provided for software compatibility purposes. The legacy meanings for this bit are as shown below. In actuality, screen sparkle will never occur. 0 = CPU access to video buffer RAM will cause screen sparkle 1 = CPU can access video buffer RAM without causing screen sparkle

Programming Notes

This register is visible only when the Comp_Mod bit in the Internal Graphics Control Register A (CSC index DDh[0]) is set (MDA compatibility).

	7	6	5	4	3	2	1	0
Bit	Reserved						PG_MEM1_EN	HGA_GR_EN
Default	x	x	x	x	x	x	0	0
R/W							W	W

Bit	Name	Function
7–2	Reserved	**Reserved** Software should write these bits to 0.
1	PG_MEM1_EN	**Enable Memory Page 1** 0 = Prevent setting of the Page Select bit (bit 7 of the MDA/HGA Mode Control Register) 1 = Allow setting of the Page Select bit The read-back for the write-only PG_MEM1_EN bit is in graphics index 52h[6].
0	HGA_GR_EN	**Enable HGA Graphics Mode** 0 = Lock the chip in Hercules text mode 1 = Permit entry to Hercules Graphics mode The read-back for the write-only HGA_GR_EN bit is in graphics index 52h[5].

Programming Notes

This register is visible only when the Comp_Mod bit in the Internal Graphics Control Register A (CSC index DDh[0]) is set (MDA compatibility) and the Extended Feature Control Register (graphics index 52h) bit 4 is set, thus enabling HGA mode. Also, bits 1–0 can be read back via the internal graphics controller's Extended Feature Control Register at graphics index 52h in 3x4h/3x5h (internal graphics controller indexed register space).

	7	6	5	4	3	2	1	0
Bit	CGA_INDX[7–0]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	CGA_INDX[7–0]	**CGA Index** Graphics Controller Index Register when operating in Color Graphics Adapter mode.

Programming Notes

This register is visible only when the Comp_Mod bit in the Internal Graphics Control Register A (CSC index DDh[0]) is cleared (CGA mode).

	7	6	5	4	3	2	1	0
Bit	CGA_DATA[7–0]							
Default	x	x	x	x	x	x	x	x
R/W	R/W							

Bit	Name	Function
7–0	CGA_DATA[7–0]	**CGA Data Port** Graphics Controller Data Register when operating in Color Graphics Adapter mode.

Programming Notes

This register is visible only when the Comp_Mod bit in the Internal Graphics Control Register A (CSC index DDh[0]) is cleared (CGA mode).

	7	6	5	4	3	2	1	0
Bit	Reserved		TXTBLNK	GRPCON	VIDCON	COLBUR	TGCON	TXTCON
Default	0	0	0	0	0	0	0	0
R/W			R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7–6	Reserved	**Reserved** Software should write these bits to 0.
5	TXTBLNK	**Text Attribute Control** 0 = Blinking attribute disabled 1 = Blinking attribute enabled (attribute byte bit 7 will control blinking)
4	GRPCON	**CGA Graphics Control** 0 = All other modes 1 = 1bpp high-resolution APA graphics mode
3	VIDCON	**Video Blanking Control** 0 = Video is blanked 1 = Video is not blanked
2	COLBUR	**Color Burst Select (Affects Only Text Modes)** 0 = Use color text attributes 1 = Normal graphics panels will use black and white text attributes. Color STN panels will use monochrome gray scales
1	TGCON	**Text/Graphics Control** 0 = Text mode 1 = Graphics mode
0	TXTCON	**CGA Column Select** This bit is provided for compatibility purposes only. It does not actually set the column width. This must be done via the CRTC index registers. The CGA legacy values for this bit are: 0 = 40x25 text mode 1 = 80x25 text mode

Programming Notes

Bits 3–0 of this register select the border color for text mode, the background color (C0 = C1 = 0) for 320x200 graphics modes, and the foreground color (C0 = 1) for 640x200 graphics modes. This register is visible only when the Comp_Mod bit in the Internal Graphics Control Register A (CSC index DDh[0]) is cleared (CGA mode).

	7	6	5	4	3	2	1	0
Bit	Reserved		ALTPAL	ALTBAK	SBINT	SBRED	SBGREEN	SBBLUE
Default	0	0	0	0	0	0	0	0
R/W			R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7–6	Reserved	**Reserved** Software should write these bits to 0.
5	ALTPAL	**Select Alternate Palette** Selects an alternate palette for the 320-column graphics mode display: 0 = Green, red, yellow palette 1 = Cyan, magenta, white palette
4	ALTBAK	**Select Alternate Background** 0 = No intense foreground or background colors 1 = Selects intense colors in graphics, and intense background colors in text mode
3	SBINT	**Select Intense Border/Background** 0 = Border/foreground not intense 1 = Intense border in text mode, intense background in 320x200 graphics, intense foreground in 640x200 graphics
2	SBRED	**Select Red Border/Background** 0 = No red component 1 = Red border in text mode, red background in 320x200 graphics, red foreground in 640x200 graphics
1	SBGREEN	**Select Green Border/Background** 0 = No green component 1 = Green border in text mode, green background in 320x200 graphics, green foreground in 640x200 graphics
0	SBBLUE	**Select Blue Border/Background** 0 = No blue component 1 = Blue border text mode, blue background in 320x200 graphics, blue foreground in 640x200 graphics

Programming Notes

This register is visible only when the Comp_Mod bit in the Internal Graphics Control Register A (CSC index DDh[0]) is cleared (CGA mode).

	7	6	5	4	3	2	1	0
Bit	Reserved				VERTRET	LPENSW	LPENSTAT	DMSTAT
Default	1	1	1	1	x	1	x	x
R/W	R				R	R	R	R

Bit	Name	Function
7–4	Reserved	**Reserved** Reads back ‘1111b’.
3	VERTRET	**Vertical Retrace Status (Simulated Vertical Sync)** 0 = Raster is not in vertical retrace 1 = Raster is in vertical retrace
2	LPENSW	**Light Pen Switch** Always reads back logic 1.
1	LPENSTAT	**Light Pen Status** This bit will read 1 if I/O address 3DC was the most recent previously accessed, or zero if I/O address 3DB was the most recent previously accessed.
0	DMSTAT	**Display-Memory Access Status (Simulated Horizontal Sync)** This bit is provided for software compatibility purposes. The legacy meanings for this bit are as shown below. In actuality, screen sparkle will never occur. 0 = CPU access to video buffer RAM will cause screen sparkle 1 = CPU can access video buffer RAM without causing screen sparkle

Programming Notes

This register is visible only when the Comp_Mod bit in the Internal Graphics Control Register A (CSC index DDh[0]) is cleared.

	7	6	5	4	3	2	1	0
Bit	PCM_IDX[7–0]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	PCM_IDX[7–0]	**PC Card Controller Index Register** Write the offset of the 82365-compatible index register to access to this port.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	PCM_DATA[7–0]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	PCM_DATA[7–0]	**PC Card Data** Access 82365-compatible indexed registers via this port.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	COM1THR[7–0]							
Default	0	0	0	0	0	0	0	0
R/W	W							

Bit	Name	Function
7–0	COM1THR[7–0]	**COM1 Transmit Holding Register** When COM1 DLAB = 0, and the transmitter is not busy, write the byte to transmit to this register.

Programming Notes

There is only one serial port on the ÉlanSC400 microcontroller, but it can be configured to have a base address of either 3F8h or 2F8h (COM1/COM2).

COM1 Receive Buffer Register — I/O Address 03F8h

	7	6	5	4	3	2	1	0
Bit	COM1RBR[7–0]							
Default	0	0	0	0	0	0	0	0
R/W	R							

Bit	Name	Function
7–0	COM1RBR[7–0]	**COM1 Receive Buffer** When COM1 DLAB = 0, and a received byte is available, read received byte from this register.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	COM1_DIV[7–0]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	COM1_DIV[7–0]	**COM1 Baud Block Divisor Latch** When COM1 DLAB = 1, this register holds the least significant byte of a 16-bit baud rate clock divisor that is used to generate the 16x baud clock.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	COM1_DIV[15–8]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	COM1_DIV[15–8]	**COM1 Baud Clock Divisor Latch** When COM1 DLAB = 1, this register holds the most significant byte of a 16-bit baud rate clock divisor that is used to generate the 16x baud clock.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved				EMSI	ERLSI	ETHREI	ERDAI
Default	0	0	0	0	0	0	0	0
R/W					R/W	R/W	R/W	R/W

Bit	Name	Function
7–4	Reserved	**Reserved** Software should write these bits to 0.
3	EMSI	**Enable Modem Status Interrupt** 0 = Disable modem status interrupt 1 = Enable modem status interrupt
2	ERLSI	**Enable Receiver Line Status Interrupt** 0 = Disable receiver line status interrupt 1 = Enable receiver line status interrupt
1	ETHREI	**Enable Transmitter Holding Register Empty Interrupt** 0 = Disable transmitter holding register empty interrupt 1 = Enable transmitter holding register empty interrupt
0	ERDAI	**Enable Received Data Available Interrupt** 0 = Disable data available/16550 time-out interrupt 1 = Enable received data available interrupt in 16450-compatible mode; in 16550-compatible mode, this bit also enables time-out interrupts More detail on time-out interrupts can be found in the Interrupt Identification Register description.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	FIFO1 FIFO0		Reserved		ID2 ID1 ID0			IP
Default	0	0	0	0	1	1	1	1
R/W	R				R			R

Bit	Name	Function	Priority
7–6	FIFO1 FIFO0	**FIFO Feature Presence Indication** FIFO is only present when 16550-compatible mode is enabled): 0 0 = No significance 0 1 = No significance 1 0 = 16450-compatible mode is enabled 1 1 = 16550-compatible mode is enabled	
5–4	Reserved	**Reserved** These bits always read back ‘00b’.	
3–1	ID2 ID1 ID0	Interrupt Identification Bit Field 0 0 0 = Modem status	Fourth (Lowest)
		0 0 1 = Transmitter Holding Register Empty/Transmit FIFO Empty (16550-compatible mode)	Third
		0 1 0 = Received Data Available/Receiver FIFO trigger (16550-compatible mode)	Second
		0 1 1 = Receive Line Status	First (Highest)
		1 0 0 = Not used	
		1 0 1 = Not used	
		1 1 0 = FIFO time-out	Second
		1 1 1 = Not used	
		In 16450-compatible mode, bit 3 always reads back ‘0b’. See the Modem Status Register (MSR) and Line Status Register (LSR) for more detail on interrupt events.	
		If two interrupt sources are pending simultaneously, only the highest priority interrupt (as defined by the rightmost column for bits 3–1) will be indicated by bits 3–1 of the Interrupt Identification Register (IIR). When the interrupt source is cleared, a subsequent read from the Interrupt Identification Register will return the next highest priority interrupt source. Bits 3–1 have no meaning if bit 0 = ‘1b’.	
		A FIFO time-out occurs when the receive FIFO is not empty, and more than four continuous character-times have transpired without more data being placed into or read out of the receive FIFO. Reading a character from the receive FIFO clears the time-out interrupt.	
0	IP	**Serial Port Interrupt Pending (Active Low)** 0 = Interrupt pending	
		1 = No interrupt pending	

Programming Notes

	7	6	5	4	3	2	1	0
Bit	RFRT[1–0]		Reserved		Reserved	TFCLR	RFCLR	FIFOEN
Default	0	0	0	0	0	0	0	0
R/W	W				W	W	W	W

Bit	Name	Function
7–6	RFRT[1–0]	**Receiver FIFO Register Trigger Bits** When in 16550-compatible mode, this bit field specifies the trigger level at which the Interrupt Identification Register will report that a received data available interrupt is pending. If received data available interrupts are enabled in the IER, the system will be interrupted when the receive FIFO fills to the trigger level per the following table: 0 0 = 1 byte 0 1 = 4 bytes 1 0 = 8 bytes 1 1 = 14 bytes When the data in the receive FIFO falls below this trigger level, the interrupt will be cleared.
5–4	Reserved	**Reserved** Software should write these bits to 0.
3	Reserved	**Reserved** Software should write this bit to 1.
2	TFCLR	**Transmitter FIFO Buffer Clear** Writing a ‘1b’ to this bit position clears the transmit FIFO, and resets the transmit FIFO counter logic. It does not clear the Transmitter Shift Register. This bit is cleared by either a read of the Interrupt ID Register, or by a write to the Transmit Holding Register.
1	RFCLR	**Receiver FIFO Buffer Clear** Writing a ‘1b’ to this bit position clears the receive FIFO, and resets the receive FIFO counter logic. It does not clear the Receiver Shift Register. This bit is self clearing, and does not need to be reset to 0 under software control.
0	FIFOEN	**FIFO Enable (16550-Compatible Mode Enable)** 0 = Causes UART to enter 16450-compatible mode Disables accesses to receive and transmit FIFOs, and all FIFO control bits except this bit. 1 = Causes UART to enter 16550-compatible mode Enables receive and transmit FIFOs, and enables accesses to other FIFO control bits. The main difference between 16450-compatible mode and 16550-compatible mode is the addition of transmit and receive FIFOs in 16550-compatible mode. This bit must be ‘1b’ when other FIFO control register bits are written to or they will not be programmed. Also note that any mode switch will clear both FIFOs.

Programming Notes

The contents of this write-only register can be read back via the shadow register at CSC index D3h. Also note that the on board UART can be mapped to only one of COM1 or COM2 at any time, and CSC index D3h serves either configuration.

	7	6	5	4	3	2	1	0
Bit	DLAB	SB	SP	EPS	PE	STP	WLB0 WLB1	
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	DLAB	**Divisor Latch Access Bit** Set this bit to gain access to the baud rate divisor latches for this COM port. Clear this bit to mux in the Transmit Holding Register and Receive Buffer Register at port xF8h, and the Interrupt Enable Register at port xF9h.
6	SB	**Set Break Enable** Setting this bit causes a break condition to be transmitted to the receiving UART. 0 = Disable set break 1 = Force serial output to spacing state (logic ‘0’) regardless of other transmitter activity The break control acts on the SOUT pin only, and has no other effect on the transmitter logic.
5	SP	**Stick Parity Enable** Forces the parity bit to always be 0 or 1 versus dynamically changing it so that there are an odd or even number of 1 bits in the transmitted data. If this bit is 0, then normal parity is used if parity is enabled. If bits 5,4, and 3 = 1, the parity bit is generated/checked as a logical 1. If bits 5 and 3 = 1 and bit 4 = 0, the parity bit is generated/checked as a logical 0.
4	EPS	**Even Parity Select** Parity must be enabled via bit 3 for this bit to have meaning: 0 = Odd parity, the parity bit will be manipulated to force an odd number of 1 bits in the transmitted data, and the same condition will be checked for in the received data. 1 = Even parity, the parity bit will be manipulated to force an even number of 1 bits in the transmitted data, and the same condition will be checked for in the received data. Start/stop bits are not included in the parity generation/checking scheme.
3	PE	**Asynchronous Data Parity Enable** Causes a parity bit to be generated in the transmitted data, and to be checked in the received data. The parity bit is located between the last data word bit and the first stop bit in the bit stream.
2	STP	**Stop Bits** Based on character length from bits 1–0 above. If character length = 5: 0 = 1 stop bit 1 = 1.5 stop bits If character length = 6, 7, or 8: 0 = 1 stop bit 1 = 2 stop bits
1–0	WLB0 WLB1	**Transmit/Receive Character Length** 0 0 = 5 bits 0 1 = 6 bits 1 0 = 7 bits 1 1 = 8 bits

	7	6	5	4	3	2	1	0
Bit	Reserved			LOOP	OUT2	OUT1	RTS	DTR
Default	0	0	0	0	0	0	0	0
R/W				R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7–5	Reserved	**Reserved** Software should write these bits to 0.
4	LOOP	**Loopback (Diagnostic Mode) Enable** 1 = Enabled The following internal connections are made by setting this diagnostic bit: $\overline{RTS}$ is internally connected to $\overline{CTS}$ $\overline{DTR}$ is internally connected to $\overline{DSR}$ $\overline{OUT1}$ is internally connected to $\overline{RI}$ $\overline{OUT2}$ is internally connected to $\overline{DCD}$ 0 = Loopback mode is disabled In addition, the SOUT bit is driven High, and the SIN input line is blocked. The Transmit Shift Out Register is directly connected to the receive shift in register. In addition, the $\overline{DTR}$, $\overline{RTS}$, $\overline{OUT1}$ and $\overline{OUT2}$ signals are forced inactive. Modem control interrupts can be forced by enabling the appropriate bit in the IER, and asserting one of the bits 3–0 in loopback mode.
3	OUT2	**Enable COM2 Interrupts** This bit controls the internal $\overline{OUT2}$ signal which is used internally as a master enable for COM1 interrupts. If this bit is not set, no COM port 1 IRQs will be felt at the Programmable Interrupt Controller (PIC).
2	OUT1	**$\overline{OUT1}$ Control** This bit controls the internal $\overline{OUT1}$ signal which is not tied to anything internally. It is provided for PC/AT compatibility, and can be used as part of the loopback diagnostics. Other than that, it has no effect on system operation, and can be used for a scratch pad during normal system operation.
1	RTS	**Request to Send** 0 = Deassert the Request To Send signal ($\overline{RTS}$) 1 = Assert the Request To Send signal ($\overline{RTS}$)
0	DTR	**Data Terminal Ready** 0 = Deassert the Data Terminal Ready signal ($\overline{DTR}$) 1 = Assert the Data Terminal Ready signal ($\overline{DTR}$)

Programming Notes

	7	6	5	4	3	2	1	0
Bit	16550_ERR	TEMT	THRE	BI	FE	PE	OE	DR
Default	0	1	1	0	0	0	0	0
R/W	R	R	R	R	R	R	R	R

Bit	Name	Function
7	16550_ERR	**16550-Mode Error** In 16450-compatible mode, this bit reads back ‘0b’. In 16550-compatible mode, when this bit is set it indicates at least one parity error, framing error, or break condition is present in the receive FIFO. This bit is cleared by a read from the Line Status Register if there are no subsequent errors in the FIFO.
6	TEMT	**Transmitter Empty Indicator** In 16450-compatible mode, this bit is set when both the Transmit Holding Register and the Transmit Shift Register are empty. In 16550-compatible mode, this bit is set when both the Transmit Shift Register and transmit FIFO are empty.
5	THRE	**Transmitter Holding Register (16450-Compatible Mode) or Transmit FIFO Empty (16550-Compatible Mode)** In 16450-compatible mode, this bit indicates that the Transmit Holding Register is ready to accept a new character. This bit is automatically reset by a write to the Transmit Holding Register when in 16450-compatible mode. In 16550-compatible mode, this bit indicates that the transmit FIFO is completely empty. When in 16550-compatible mode this interrupt will be cleared when either the transmit FIFO is written to, or when the Interrupt Identification Register is read. This bit can be used to generate an interrupt if programmed to do so via the Interrupt Enable Register.
4	BI	**Break Indicator** In 16450-compatible mode, this bit is set when the UART detects that the sending UART is transmitting a break condition for a period longer than the time it takes to receive start, data, parity, and stop bits. In 16550-compatible mode, this bit is set when an entire word (start, data, parity, stop) that has been received with break indication present into the receive FIFO is at the top of the FIFO. Only one break indication will be loaded into the FIFO regardless of the duration of the break condition. A new character will not be loaded into the FIFO until the next valid start bit is detected. This latched status bit is automatically cleared by a read from the Line Status Register.
3	FE	**Framing Error** In 16450-compatible mode, this bit is set to indicate that a received character did not have a valid stop bit. In 16550-compatible mode, this bit is set when a character that has been received with a framing error into the receive FIFO is at the top of the FIFO. This latched status bit is automatically cleared by a read from the Line Status Register.
2	PE	**Parity Error** In 16450-compatible mode, this bit is set upon receipt of data with incorrect parity. In 16550-compatible mode, this bit is set when a character that has been received with bad parity into the receive FIFO is at the top of the FIFO. This latched status bit is automatically cleared by a read from the Line Status Register.

Bit	Name	Function
1	OE	**Overrun Error** In 16450-compatible mode, this bit is set if a new character is received into the receiver buffer before a previous character was read, thus resulting in lost data. In 16550-compatible mode, this bit is set if a new character is completely received into the Shift Register when the FIFO is already 100% full. Data in the FIFO is not overwritten by this overrun. The data in the Shift Register will be lost when the next character is received. This latched status bit is automatically cleared by a read from the Line Status Register.
0	DR	**Data Ready** In 16450-compatible mode, when this bit is set, a character has been received and placed in the Receiver Buffer Register. This bit is automatically cleared by reading the Receiver Buffer Register. In 16550-compatible mode, when this bit is set, a character has been received and placed in the receive FIFO. This bit is automatically cleared by reading the Receiver FIFO.

Programming Notes

When a receiver line status interrupt is enabled and detected, bits 1 through 4 above will indicate the reason for the interrupt. The status bits are valid even when the receiver line status interrupt is not enabled.

	7	6	5	4	3	2	1	0
Bit	DCD	RI	DSR	CTS	DDCD	TERI	DDSR	DCTS
Default	x	x	x	x	0	0	0	0
R/W	R	R	R	R	R	R	R	R

Bit	Name	Function
7	DCD	**Data Carrier Detect** 0 = $\overline{DCD}$ input signal is deasserted 1 = $\overline{DCD}$ input signal is asserted If in loopback mode, this bit tracks bit 3 of the Modem Control Register.
6	RI	**Ring Indicator** 0 = $\overline{RI}$ input signal is deasserted 1 = $\overline{RI}$ input signal is asserted If in loopback mode, this bit tracks bit 2 of the Modem Control Register.
5	DSR	**Data Set Ready** 0 = $\overline{DSR}$ input signal is deasserted 1 = $\overline{DSR}$ input signal is asserted If in loopback mode, this bit tracks bit 0 of the Modem Control Register.
4	CTS	**Clear To Send** 0 = $\overline{CTS}$ input signal is deasserted 1 = $\overline{CTS}$ input signal is asserted If in loopback mode, this bit tracks bit 1 of the Modem Control Register.
3	DDCD	**Delta Data Carrier Detect** 0 = Indicates that the DCD signal has not changed since the Modem Status Register was last read 1= Indicates that the $\overline{DCD}$ signal changed since the Modem Status Register was last read
2	TERI	**Trailing Edge Ring Indicator** 0 = Indicates that the $\overline{RI}$ signal has not changed from an active to an inactive state since the Modem Status Register was last read 1= Indicates that the $\overline{RI}$ signal changed from an active to an inactive state since the Modem Status Register was last read
1	DDSR	**Delta Data Set Ready** 0 = Indicates that the $\overline{DSR}$ signal has not changed since the Modem Status Register was last read 1 = Indicates that the $\overline{DSR}$ signal changed since the Modem Status Register was last read
0	DCTS	**Delta Clear To Send** 0 = Indicates that the $\overline{CTS}$ signal has not changed since the Modem Status Register was last read 1 = Indicates that the $\overline{CTS}$ signal changed since the Modem Status Register was last read

Programming Notes

Bits 3–0 are latched status bits that will generate an interrupt if modem status interrupts are unmasked in the Interrupt Enable Register. Reading the Modem Status Register clears these bits, and their associated interrupt. Bits 7–4 are real time status indicators for the indicated inputs to the UART.

	7	6	5	4	3	2	1	0
Bit	SCRATCH2[7–0]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	SCRATCH2[7–0]	**Scratch Bit** General-purpose I/O location, not required for serial data transfer.

Programming Notes

CHAPTER

3 CHIP SETUP AND CONTROL (CSC) INDEXED REGISTERS

3.1 OVERVIEW

This chapter describes the Chip Setup and Control (CSC) indexed registers available on the ÉlanSC400 microcontroller. Chip Setup and Control indexed registers are defined as ÉlanSC400 microcontroller-specific registers beyond standard PC/AT compatibility requirements. For example, all memory controller and power management registers are included in this group. Also included in this group are all registers that provide extended control features for the on-board peripherals. They are grouped logically by function in hexadecimal order, as listed in Table 3-1.

In addition to the CSC indexed registers, the RTC, graphics controller, and PC Card controller also have their own dedicated groups of indexed registers accessible through different index and data ports. These dedicated register groups are described in following chapters of this manual.

The ÉlanSC400 Chip Setup and Control (CSC) indexed registers are accessed using a two-step process:

- An I/O write to I/O address 22h is first performed. The data written is the index of the CSC register to be accessed.
- This I/O write is followed by an I/O read or write to address 23h. This access causes data to be read from or written to the addressed configuration register.

The ÉlanSC400 microcontroller does not implement any locking mechanism for extended register access. Also, back-to-back access of I/O address 22h/23h is not required for access to the extended index registers. The following code fragment:

```
mov al, 90h; force an SMI
out 22h, al
mov al, 1
out 23h, al
   :
   :
```

has the same result as this code fragment:

```
mov AX, 0190h; force an SMI.
out 22h, AX
```

The Chip Setup and Control indexed registers are typically accessed at CPU speeds unless they are being echoed to the ISA bus for debug.

Table 3-1 Chip Setup and Control (CSC) Index Register Map

Register Name	I/O (Port) Address	Index	Page Number
ÉlanSC400 Microcontroller Chip Setup and Control (CSC) Index Register	0022h		page 3-8

Register Name	I/O (Port) Address	Index	Page Number
ÉlanSC400 Microcontroller Chip Setup and Control (CSC) Data Port	0023h		page 3-9
DRAM Setup and Configuration Group		**00–07h**	
DRAM Bank 0 Configuration Register		00h	page 3-10
DRAM Bank 1 Configuration Register		01h	page 3-11
DRAM Bank 2 Configuration Register		02h	page 3-12
DRAM Bank 3 Configuration Register		03h	page 3-13
DRAM Control Register		04h	page 3-14
DRAM Refresh Control Register		05h	page 3-16
Drive Strength Control Register A		06h	page 3-17
Drive Strength Control Register B		07h	page 3-18
Reserved		08–0Fh	
Cache Control Registers		**10–14h**	
Non-Cacheable Window 0 Address Register		10h	page 3-19
Non-Cacheable Window 0 Address/Attributes/SMM Register		11h	page 3-20
Non-Cacheable Window 1 Address Register		12h	page 3-21
Non-Cacheable Window 1 Address/Attributes Register		13h	page 3-22
Cache and VL Miscellaneous Register		14h	page 3-23
Reserved		15–1Fh	
ROM Configuration, Setup, and Control Group		**20–28h**	
Pin Strap Status Register		20h	page 3-25
Linear $\overline{\text{ROMCS0}}$/Shadow Register		21h	page 3-26
Linear $\overline{\text{ROMCS0}}$ Attributes Register		22h	page 3-28
$\overline{\text{ROMCS0}}$ Configuration Register A		23h	page 3-29
$\overline{\text{ROMCS0}}$ Configuration Register B		24h	page 3-31
$\overline{\text{ROMCS1}}$ Configuration Register A		25h	page 3-32
$\overline{\text{ROMCS1}}$ Configuration Register B		26h	page 3-34
$\overline{\text{ROMCS2}}$ Configuration Register A		27h	page 3-35
$\overline{\text{ROMCS2}}$ Configuration Register B		28h	page 3-37
Reserved		29–2Fh	
Memory Management System (MMS)		**30–35h**	
MMS Window C–F Attributes Register		30h	page 3-38
MMS Window C–F Device Select Register		31h	page 3-39
MMS Window A Destination Register		32h	page 3-40

Register Name	I/O (Port) Address	Index	Page Number
MMS Window A Destination/Attributes Register		33h	page 3-41
MMS Window B Destination Register		34h	page 3-42
MMS Window B Destination/Attributes Register		35h	page 3-43
Reserved		36–37h	
GPIO Pin Muxing and Termination Group		**38–3Eh**	
Pin Mux Register A		38h	page 3-44
Pin Mux Register B		39h	page 3-45
Pin Mux Register C		3Ah	page 3-46
GPIO Termination Control Register A		3Bh	page 3-47
GPIO Termination Control Register B		3Ch	page 3-48
GPIO Termination Control Register C		3Dh	page 3-49
GPIO Termination Control Register D		3Eh	page 3-50
Reserved		3Fh	
PMU Mode Control and Status Group		**40–45h**	
PMU Force Mode Register		40h	page 3-51
PMU Present and Last Mode Register		41h	page 3-53
Hyper/High-Speed Mode Timers Register		42h	page 3-54
Low-Speed/Standby Mode Timers Register		43h	page 3-55
Suspend/Temporary Low-Speed Mode Timers Register		44h	page 3-56
Wake-Up Pause/High-Speed Clock Timers Register		45h	page 3-57
Reserved		46–4Fh	
PMU Wake-up Control and Status Group		**50–5Bh**	
SUS_RES Pin Configuration Register		50h	page 3-58
Reserved		51h	
Wake-Up Source Enable Register A		52h	page 3-59
Wake-Up Source Enable Register B		53h	page 3-60
Wake-Up Source Enable Register C		54h	page 3-61
Wake-Up Source Enable Register D		55h	page 3-62
Wake-Up Source Status Register A		56h	page 3-63
Wake-Up Source Status Register B		57h	page 3-64
Wake-Up Source Status Register C		58h	page 3-65
Wake-Up Source Status Register D		59h	page 3-66
GPIO as a Wake-Up or Activity Source Status Register A		5Ah	page 3-67

Register Name	I/O (Port) Address	Index	Page Number
GPIO as a Wake-Up or Activity Source Status Register B		5Bh	page 3-68
Reserved		5C–5Fh	
PMU Activity Control and Status Group		**60–6Dh**	
GP_CS Activity Enable Register		60h	page 3-69
GP_CS Activity Status Register		61h	page 3-70
Activity Source Enable Register A		62h	page 3-71
Activity Source Enable Register B		63h	page 3-72
Activity Source Enable Register C		64h	page 3-73
Activity Source Enable Register D		65h	page 3-74
Activity Source Status Register A		66h	page 3-75
Activity Source Status Register B		67h	page 3-76
Activity Source Status Register C		68h	page 3-77
Activity Source Status Register D		69h	page 3-78
Activity Classification Register A		6Ah	page 3-79
Activity Classification Register B		6Bh	page 3-80
Activity Classification Register C		6Ch	page 3-81
Activity Classification Register D		6Dh	page 3-82
Reserved		6E–6Fh	
Battery Level BL Pin Control and Status Group		**70–72h**	
Battery/AC Pin Configuration Register A		70h	page 3-83
Battery/AC Pin Configuration Register B		71h	page 3-85
Battery/AC Pin State Register		72h	page 3-86
Reserved		73–7Fh	
Clock Control and Status Group		**80–83h**	
CPU Clock Speed Register		80h	page 3-87
CPU Clock Auto Slowdown Register		81h	page 3-88
Clock Control Register		82h	page 3-90
CLK_IO Pin Output Clock Select Register		83h	page 3-91
Reserved		84–87h	
Factory Level Debug Group		**88–8Fh**	
Factory Debug Register A		88h	page 3-92
Factory Debug Register B		89h	page 3-93
SMI/NMI Generation and Status Group		**90–9Dh**	
Miscellaneous SMI/NMI Enable Register		90h	page 3-94
PC Card and Keyboard SMI/NMI Enable Register		91h	page 3-95

Register Name	I/O (Port) Address	Index	Page Number
Mode Timer SMI/NMI Enable Register		92h	page 3-96
Battery Low and ACIN SMI/NMI Enable Register		93h	page 3-97
Miscellaneous SMI/NMI Status Register A		94h	page 3-99
PC Card and Keyboard SMI/NMI Status Register		95h	page 3-100
Mode Timer SMI/NMI Status Register		96h	page 3-101
Battery Low and ACIN SMI/NMI Status Register		97h	page 3-102
SMI/NMI Select Register		98h	page 3-104
I/O Access SMI Enable Register A		99h	page 3-105
I/O Access SMI Enable Register B		9Ah	page 3-106
I/O Access SMI Status Register A		9Bh	page 3-107
I/O Access SMI Status Register B		9Ch	page 3-108
XMI Control Register		9Dh	page 3-109
Reserved		9E–9Fh	
GPIO Pin Control, Status, and Muxing Group		**A0–BDh**	
GPIO_CS Function Select Register A		A0h	page 3-110
GPIO_CS Function Select Register B		A1h	page 3-111
GPIO_CS Function Select Register C		A2h	page 3-112
GPIO_CS Function Select Register D		A3h	page 3-113
GPIO Function Select Register E		A4h	page 3-114
GPIO Function Select Register F		A5h	page 3-115
GPIO Read-Back/Write Register A		A6h	page 3-116
GPIO Read-Back/Write Register B		A7h	page 3-117
GPIO Read-Back/Write Register C		A8h	page 3-118
GPIO Read-Back/Write Register D		A9h	page 3-119
GPIO_PMUA Mode Change Register		AAh	page 3-120
GPIO_PMUB Mode Change Register		ABh	page 3-122
GPIO_PMUC Mode Change Register		ACh	page 3-124
GPIO_PMUD Mode Change Register		ADh	page 3-126
GPIO_PMU to GPIO_CS Map Register A		AEh	page 3-128
GPIO_PMU to GPIO_CS Map Register B		AFh	page 3-129
GPIO_XMI to GPIO_CS Map Register		B0h	page 3-130
Standard Decode To GPIO_CS Map Register		B1h	page 3-131
GP_CS to GPIO_CS Map Register A		B2h	page 3-132
GP_CS to GPIO_CS Map Register B		B3h	page 3-133
GP_CSA I/O Address Decode Register		B4h	page 3-134
GP_CSA I/O Address Decode and Mask Register		B5h	page 3-135

Register Name	I/O (Port) Address	Index	Page Number
GP_CSB I/O Address Decode Register		B6h	page 3-136
GP_CSB I/O Address Decode and Mask Register		B7h	page 3-137
GP_CSA/B I/O Command Qualification Register		B8h	page 3-138
GP_CSC Memory Address Decode Register		B9h	page 3-140
GP_CSC Memory Address Decode and Mask Register		BAh	page 3-141
GP_CSD Memory Address Decode Register		BBh	page 3-142
GP_CSD Memory Address Decode and Mask Register		BCh	page 3-143
GP_CSC/D Memory Command Qualification Register		BDh	page 3-144
Reserved		BE–BFh	
Matrix Keyboard/XT Keyboard Group		**C0–CAh**	
Keyboard Configuration Register A		C0h	page 3-146
Keyboard Configuration Register B		C1h	page 3-149
Keyboard Input Buffer Read-Back Register		C2h	page 3-151
Keyboard Output Buffer Write Register		C3h	page 3-152
Mouse Output Buffer Write Register		C4h	page 3-153
Keyboard Status Register Write Register		C5h	page 3-154
Keyboard Timer Register		C6h	page 3-155
Keyboard Column Register		C7h	page 3-156
Keyboard Row Register A		C8h	page 3-158
Keyboard Row Register B		C9h	page 3-160
Keyboard Column Termination Register		CAh	page 3-162
Reserved		CB–CFh	
PC/AT-Related Extended Feature Group		**D0–DEh**	
Internal I/O Device Disable/Echo ZBUS Configuration Register		D0h	page 3-164
Parallel/Serial Port Configuration Register		D1h	page 3-167
Parallel Port Configuration Register		D2h	page 3-168
UART FIFO Control Register Shadow		D3h	page 3-169
Interrupt Configuration Register A		D4h	page 3-170
Interrupt Configuration Register B		D5h	page 3-171
Interrupt Configuration Register C		D6h	page 3-172
Interrupt Configuration Register D		D7h	page 3-173
Interrupt Configuration Register E		D8h	page 3-174
DMA Channel 0–3 Extended Page Register		D9h	page 3-175

Register Name	I/O (Port) Address	Index	Page Number
DMA Channel 5–7 Extended Page Register		DAh	page 3-176
DMA Resource Channel Map Register A		DBh	page 3-177
DMA Resource Channel Map Register B		DCh	page 3-178
Internal Graphics Control Register A		DDh	page 3-179
Internal Graphics Control Register B		DEh	page 3-180
Reserved		DFh	
ISA Bus Configuration Group		**E0–E5h**	
Write-protected System Memory (DRAM) Window/ Overlapping ISA Window Enable Register		E0h	page 3-181
Overlapping ISA Window Start Address Register		E1h	page 3-182
Overlapping ISA Window Size Register		E2h	page 3-183
Suspend Mode Pin State Register A		E3h	page 3-184
Suspend Mode Pin State Register B		E4h	page 3-185
Suspend Mode Pin State Over-ride Register		E5h	page 3-186
Reserved		E6–E9h	
IrDA Group		**EA–EDh**	
IrDA Control Register		EAh	page 3-188
IrDA Status Register		EBh	page 3-190
IrDA CRC Status Register		ECh	page 3-192
IrDA Own Address Register		EDh	page 3-193
IrDA Frame Length Register A		EEh	page 3-194
IrDA Frame Length Register B		EFh	page 3-195
PC Card Configuration Group		**F0–F2h**	
PC Card Extended Features Register		F0h	page 3-196
PC Card Mode and DMA Control Register		F1h	page 3-198
PC Card Socket A/B Input Termination Control Register		F2h	page 3-200
Reserved		F3–F7h	
Miscellaneous		**FFh**	
ÉlanSC400 Microcontroller Revision ID Register		FFh	page 3-201

3.2 REGISTER DESCRIPTIONS

Each CSC indexed register is described on the following pages. Additional information about using these registers to program the ÉlanSC400 microcontroller can be found in the *ÉlanSC400 User's Manual* (order #21030).

	7	6	5	4	3	2	1	0
Bit	CSCIDX[7–0]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	CSCIDX[7–0]	**ÉlanSC400 Microcontroller Chip Setup and Control (CSC) Index Register** Select internal CSC index register to access.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	CSCDATA[7–0]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	CSCDATA[7–0]	**ÉlanSC400 Microcontroller Chip Setup and Control (CSC) Data Port** Read or write to the ÉlanSC400 microcontroller register pointed to by the ÉlanSC400 microcontroller index register.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	BNK_ENBL0	Reserved	PG_TYPE0	ASYM0	WIDTH0	DEPTH0[2–0]		
Default	0	x	0	0	0	0	0	0
R/W	R/W		R/W	R/W	R/W	R/W		

Bit	Name	Function
7	BNK_ENBL0	**Bank 0 Enable** 0 = Bank 0 disabled 1 = Bank 0 enabled If this DRAM bank is disabled, refresh will not be generated to it regardless of the state of the refresh enable bit at CSC index 5h[6].
6	Reserved	**Reserved** During read/modify/write operations, software must preserve this bit.
5	PG_TYPE0	**Bank 0 DRAM Page Type** 0 = Fast page DRAM 1 = Hyper page (Extended Data Out) DRAM
4	ASYM0	**Bank 0 DRAM Symmetry** 0 = Symmetrical addressing 1 = Asymmetrical addressing This bit should only be set when the bank is populated with devices which operate in the asymmetrical address mode. If the bank depth has been programmed to 256 Kbits, bit 4 of this register must be cleared.
3	WIDTH0	**DRAM Bank 0 Data Width** 0 = Bank uses a 16-bit DRAM data width 1 = Bank uses a 32-bit DRAM data width When this bit is set the $\overline{CASL3}$–$\overline{CASL2}$, $\overline{CASH3}$–$\overline{CASH2}$, $\overline{RAS3}$–$\overline{RAS2}$, and MA12 pins will automatically be enabled. When this bit is set, the Keyboard Row signals (KBD_ROW6–KBD_ROW0]) will be unavailable.
2–0	DEPTH0[2–0]	**DRAM Bank 0 Data Depth** This bit field controls the generation of $\overline{RAS}$ and $\overline{CAS}$ based on the depth of the DRAM devices used to populate DRAM bank 0. System boot software should configure this bit field according to the depth of the DRAM devices that are used to populate this bank. The supported device (and therefore bank) depths are: 0 0 0 = 256 Kbits (Asymmetrical addressing is not supported for this depth) 0 0 1 = 512 Kbits 0 1 0 = 1 Mbits 0 1 1 = 2 Mbits 1 0 0 = 4 Mbits 1 0 1 = 8 Mbits 1 1 0 = 16 Mbits 1 1 1 = Reserved

Programming Notes

	7	6	5	4	3	2	1	0
Bit	BNK_ENBL1	Reserved	PG_TYPE1	ASYM1	WIDTH1	DEPTH1[2–0]		
Default	0	x	0	0	0	0	0	0
R/W	R/W		R/W	R/W	R/W	R/W		

Bit	Name	Function
7	BNK_ENBL1	**Bank 1 Enable** 0 = Bank 1 disabled 1 = Bank 1 enabled If this DRAM bank is disabled, refresh will not be generated to it regardless of the state of the refresh enable bit at CSC index 5h[6].
6	Reserved	**Reserved** During read/modify/write operations, software must preserve this bit.
5	PG_TYPE1	**Bank 1 DRAM Page Type** 0 = Fast page DRAM 1 = Hyper page (Extended Data Out) DRAM
4	ASYM1	**Bank 1 DRAM Symmetry** 0 = Symmetrical addressing 1 = Asymmetrical addressing This bit should only be set when the bank is populated with devices which operate in the asymmetrical address mode. If the bank depth has been programmed to 256 Kbits, bit 4 of this register must be cleared.
3	WIDTH1	**DRAM Bank 1 Data Width** 0 = 16-bit DRAM data width 1 = 32-bit DRAM data width When this bit is set the $\overline{CASL3}$–$\overline{CASL2}$, $\overline{CASH3}$–$\overline{CASH2}$, $\overline{RAS3}$–$\overline{RAS2}$, and MA12 pins will automatically be enabled. When this bit is set, the Keyboard Row signals (KBD_ROW6–KBD_ROW0]) will be unavailable.
2–0	DEPTH1[2–0]	**DRAM Bank 1 Data Depth** This bit field controls the generation of $\overline{RAS}$ and $\overline{CAS}$ based on the depth of the DRAM devices used to populate DRAM Bank 1. System boot software should configure this bit field according to the depth of the DRAM devices that are used to populate this bank. The supported device (and therefore bank) depths are: 0 0 0 = 256 Kbits (Asymmetrical addressing is not supported for this depth) 0 0 1 = 512 Kbits 0 1 0 = 1 Mbits 0 1 1 = 2 Mbits 1 0 0 = 4 Mbits 1 0 1 = 8 Mbits 1 1 0 = 16 Mbits 1 1 1 = Reserved

Programming Notes

	7	6	5	4	3	2	1	0
Bit	BNK_ENBL2	Reserved	PG_TYPE2	ASYM2	WIDTH2	DEPTH2[2–0]		
Default	0	x	0	0	0	0	0	0
R/W	R/W		R/W	R/W	R/W	R/W		

Bit	Name	Function
7	BNK_ENBL2	**Bank 2 Enable** 0 = Bank 2 disabled 1 = Bank 2 enabled If this DRAM bank is disabled, refresh will not be generated to it regardless of the state of the refresh enable bit at CSC index 5h[6]. Also note that when this bit is set the $\overline{CASL3}$–$\overline{CASL2}$, $\overline{CASH3}$–$\overline{CASH2}$, $\overline{RAS3}$–$\overline{RAS2}$, and MA12 pins will automatically be enabled, and the Keyboard Row signals (KBD_ROW6–KBD_ROW0) will be unavailable.
6	Reserved	**Reserved** During read/modify/write operations, software must preserve this bit.
5	PG_TYPE2	**Bank 2 DRAM Page Type** 0 = Fast page DRAM 1 = Hyper page (Extended Data Out) DRAM
4	ASYM2	**Bank 2 DRAM Symmetry** 0 = Symmetrical addressing 1 = Asymmetrical addressing This bit should only be set when the bank is populated with devices which operate in the asymmetrical address mode. If the bank depth has been programmed to 256 Kbits, bit 4 of this register must be cleared.
3	WIDTH2	**DRAM Bank 2 Data Width** 0 = 16-bit DRAM data width 1 = 32-bit DRAM data width When this bit is set the $\overline{CASL3}$–$\overline{CASL2}$, $\overline{CASH3}$–$\overline{CASH2}$, $\overline{RAS3}$–$\overline{RAS2}$, and MA12 pins will automatically be enabled. When this bit is set, the Keyboard Row signals (KBD_ROW6–KBD_ROW0]) will be unavailable.
2–0	DEPTH2[2–0]	**DRAM Bank 2 Data Depth** This bit field controls the generation of $\overline{RAS}$ and $\overline{CAS}$ based on the depth of the DRAM devices used to populate DRAM Bank 2. System boot software should configure this bit field according to the depth of the DRAM devices that are used to populate this bank. The supported device (and therefore bank) depths are: 0 0 0 = 256 Kbits (Asymmetrical addressing is not supported for this depth) 0 0 1 = 512 Kbits 0 1 0 = 1 Mbits 0 1 1 = 2 Mbits 1 0 0 = 4 Mbits 1 0 1 = 8 Mbits 1 1 0 = 16 Mbits 1 1 1 = Reserved

Programming Notes

	7	6	5	4	3	2	1	0
Bit	BNK_ENBL3	Reserved	PG_TYPE3	ASYM3	WIDTH3	DEPTH3[2–0]		
Default	0	x	0	0	0	0	0	0
R/W	R/W		R/W	R/W	R/W	R/W		

Bit	Name	Function
7	BNK_ENBL3	**Bank 3 Enable** 0 = Bank 3 disabled 1 = Bank 3 enabled If this DRAM bank is disabled, refresh will not be generated to it regardless of the state of the refresh enable bit at CSC index 5h[6]. When this bit is set, the $\overline{\text{CASL3}}$–$\overline{\text{CASL2}}$, $\overline{\text{CASH3}}$–$\overline{\text{CASH2}}$, $\overline{\text{RAS3}}$–$\overline{\text{RAS2}}$, and MA12 pins will automatically be enabled and the Keyboard Row signals (KBD_ROW6–KBD_ROW0) will be unavailable.
6	Reserved	**Reserved** During read/modify/write operations, software must preserve this bit.
5	PG_TYPE3	**Bank 3 DRAM Page Type** 0 = Fast page DRAM 1 = Hyper page (Extended Data Out) DRAM
4	ASYM3	**Bank 3 DRAM Symmetry** 0 = Symmetrical addressing 1 = Asymmetrical addressing This bit should only be set when the bank is populated with devices which operate in the asymmetrical address mode. If the bank depth has been programmed to 256 Kbits, bit 4 of this register must be cleared.
3	WIDTH3	**DRAM Bank 3 Data Width** 0 = 16-bit DRAM data width 1 = 32-bit DRAM data width When this bit is set, the $\overline{\text{CASL3}}$–$\overline{\text{CASL2}}$, $\overline{\text{CASH3}}$–$\overline{\text{CASH2}}$, $\overline{\text{RAS3}}$–$\overline{\text{RAS2}}$, and MA12 pins will automatically be enabled and the Keyboard Row signals (KBD_ROW6–KBD_ROW0) will be unavailable.
2–0	DEPTH3[2–0]	**DRAM Bank 3 Data Depth** This bit field controls the generation of $\overline{\text{RAS}}$ and $\overline{\text{CAS}}$ based on the depth of the DRAM devices used to populate DRAM Bank 3. System boot software should configure this bit field according to the depth of the DRAM devices that are used to populate this bank. The supported device (and therefore bank) depths are: 0 0 0 = 256 Kbits (Asymmetrical addressing is not supported for this depth) 0 0 1 = 512 Kbits 0 1 0 = 1 Mbits 0 1 1 = 2 Mbits 1 0 0 = 4 Mbits 1 0 1 = 8 Mbits 1 1 0 = 16 Mbits 1 1 1 = Reserved

Programming Notes

	7	6	5	4	3	2	1	0
Bit	EDO_DETECT	TWCS	Reserved	TCAS	TCP	TRCD	INTLV[1–0]	
Default	0	0	x	1	1	1	0	0
R/W	R/W	R/W		R/W	R/W	R/W	R/W	

Bit	Name	Function
7	EDO_DETECT	**EDO DRAM Detection** This bit is used for automatic EDO DRAM detection. Setting this bit causes zeros to be driven onto D15–D0] of the DRAM bus for one clock period (15 ns) to get rid of data held on the bus. In addition, it inhibits $\overline{CAS}$ from being asserted for the following DRAM read.
6	TWCS	**$\overline{MWE}$ Setup Time** Defines $\overline{MWE}$ setup time to $\overline{CAS}$ active (where t = the memory clock period) as follows: 0 = 1t 1 = 2t When this bit is set, the DRAM Controller delays one memory clock period before asserting $\overline{CAS}$ on write page hit cycles. This function is provided to accommodate systems with a heavily loaded $\overline{MWE}$ signal, to guarantee that the TWCS parameter is met on DRAM read followed immediately by a DRAM write, with no arbitration. This will also cause assertion of CAS to be delayed one memory clock period on read page hits, to accommodate the TRCS parameter, for the same reason.
5	Reserved	**Reserved** During read/modify/write operations, software must preserve this bit.
4	TCAS	**$\overline{CAS}$ Pulse Width** Defines $\overline{CAS}$ pulse width (where t = the memory clock period) as follows: For banks configured as fast page DRAM: 0 = 3t minimum for CPU/DMA accesses, and 2t for Graphics controller reads 1 = 4t minimum for CPU/DMA accesses, and 3t for Graphics controller reads For banks configured as Hyper Page (EDO) DRAM: 0 = 2t minimum for all cycles 1 = 3t minimum for all cycles

Bit	Name	Function
3	TCP	**$\overline{\text{CAS}}$ Precharge Delay** Defines $\overline{\text{CAS}}$ precharge delay (where t = the memory clock period) as follows. 0 = 1t 1 = 2t
2	TRCD	**$\overline{\text{RAS}}$ to $\overline{\text{CAS}}$ Delay** Defines $\overline{\text{RAS}}$ to $\overline{\text{CAS}}$ delay (where t = the memory clock period) as follows: 0 = 3t 1 = 4t
1–0	INTLV[1–0]	**DRAM Interleave Control** Defines DRAM interleave options as follows: 0 0 = Banks 0-3 non-interleaved 0 1 = Banks 0-1 two-way interleaved, Banks 2-3 non-interleaved 1 0 = Banks 2-3 two-way interleaved, Banks 0-1 non-interleaved 1 1 = Banks 0-1 two-way interleaved, Banks 2-3 two-way interleaved Only Fast-Page DRAM banks of the same DRAM data bus width (16/32), depth, and symmetry can be interleaved. The DRAM Controller will not permit banks which have been configured as EDO page type to be interleaved, nor will it allow banks having different data widths, depths, or symmetry to be interleaved. Due to the 64 Mbyte maximum DRAM support, interleaving banks programmed as 16 Mbyte x 4 is not supported.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved	RFEN	RTOLEN	RTODIS	REFCLK_SEL	SELF_RFSH	RFSH_SPD	
Default	x	0	0	0	0	0	0	0
		R/W	R/W	R/W	R/W	R/W	R/W	

Bit	Name	Function
7	Reserved	**Reserved** During read/modify/write operations, software must preserve this bit.
6	RFEN	**Refresh Enable** 0 = DRAM refresh is disabled 1 = DRAM refresh is enabled This bit is provided for testing and EDO DRAM detection. Note that if a DRAM bank is disabled, refresh will not be generated to it regardless of the state of this global refresh enable bit.
5	RTOLEN	**$\overline{\text{RAS}}$ Time-Out Value** 0 = $\overline{\text{RAS}}$ time-out occurs after 10 µS 1 = $\overline{\text{RAS}}$ time-out occurs after 100 µS
4	RTODIS	**$\overline{\text{RAS}}$ Time-Out Disable** 0 = $\overline{\text{RAS}}$ time-outs are enabled 1 = $\overline{\text{RAS}}$ time-outs are disabled This bit causes all RAS signals to be precharged after a programmable length of time (see bit 5) to guarantee the DRAM TRAS parameter.
3	REFCLK_SEL	**Refresh Clock Select** This bit determines the input source of the Refresh Timer during all PMU modes except Suspend mode. When this bit is cleared, the 32 KHz clock input is used to generate refresh requests. When this bit is set, the 8254 PIT (Timer 1, counter 1) is used to generate refresh requests. The selected clock will be divided as required for extended refresh rates, according to bits 1–0 of this register.The 32 KHz clock input is always used during Suspend mode, because the 8254 PIT is disabled in this mode.
2	SELF_RFSH	**DRAM Self Refresh** When set, this bit enables the Self Refresh option in the refresh controller. This bit should only be set when the devices which are populated in the DRAM array support this feature.
1–0	RFSH_SPD	**DRAM Refresh Request Speed** Determines DRAM refresh request period as follows: 0 0 = Refresh clock source divided by 1 0 1 = Refresh clock source divided by 2 1 0 = Refresh clock source divided by 4 1 1 = Refresh clock source divided by 8 When the 32 KHz clock is used as the refresh clock source (controlled by bit 3 of this register), both edges (rising and falling) are used. This results in a refresh rate of 64,000 per second. Thus, selecting the divide by 1 option results in a refresh interval of 15.6 mS. For the divide by 2, 4, and 8 selections, the refresh intervals are 31.2 mS, 62.5 mS, and 125 mS respectively. When the Programmable Interval Timer (PIT), Channel 1 is used as the refresh clock source, only a single edge is used. Therefore, the DRAM refresh rate will equal the PIT Channel 1 rate when the divide by 1 option is selected.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	D_HI_DRIVE[1–0]		MA_HI_DRIVE[1–0]		MWE_HI_DRIVE[1–0]		RAS_HI_DRIVE[1–0]	
Default	0	0	0	0	0	0	0	0
R/W	R/W		R/W		R/W		R/W	

Bit	Name	Function
7–6	D_HI_DRIVE[1–0]	**I/O Pad Drive Strength for D15–D0** Selects drive strength of I/O pads for D15–D0 pins, as follows: 0 0 = 24 mA pads selected 0 1 = 12 mA pads selected 1 0 = 18 mA pads selected 1 1 = Pads are three-stated
5–4	MA_HI_DRIVE[1–0]	**I/O Pad Drive Strength for MA12–MA0** Selects drive strength of I/O pads for MA12–MA0 pins, as follows: 0 0 = 24 mA pads selected 0 1 = 12 mA pads selected 1 0 = 18 mA pads selected 1 1 = Pads are three-stated
3–2	MWE_HI_DRIVE[1–0]	**I/O Pad Drive Strength for $\overline{\text{MWE}}$** Selects drive strength of I/O pads for $\overline{\text{MWE}}$ pin, as follows: 0 0 = 24 mA pads selected 0 1 = 12 mA pads selected 1 0 = 18 mA pads selected 1 1 = Pads are three-stated
1–0	RAS_HI_DRIVE[1–0]	**I/O Pad Drive Strength for $\overline{\text{RAS3}}$–$\overline{\text{RAS0}}$** Selects drive strength of I/O pads for $\overline{\text{RAS3}}$–$\overline{\text{RAS0}}$ pins, as follows: 0 0 = 24 mA pads selected 0 1 = 12 mA pads selected 1 0 = 18 mA pads selected 1 1 = Pads are three-stated

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved				SA_HI_DRIVE[1–0]		SD_HI_DRIVE[1–0]	
Default	x	x	x	x	0	0	0	0
R/W					R/W		R/W	

Bit	Name	Function
7–4	Reserved	**Reserved** During read/modify/write operations, software must preserve these bits.
3–2	SA_HI_DRIVE[1–0]	**I/O Pad Drive Strength for SA23–SA0** Selects drive strength of I/O pads for SA23–SA0 pins, as follows: 0 0 = 24 mA pads selected 0 1 = 12 mA pads selected 1 0 = 18 mA pads selected 1 1 = Pads are three-stated
1–0	SD_HI_DRIVE[1–0]	**I/O Pad Drive Strength for SD15–SD0** Selects drive strength of I/O pads for SD15–SD0 pins, as follows: 0 0 = 24 mA pads selected 0 1 = 12 mA pads selected 1 0 = 18 mA pads selected 1 1 = Pads are three-stated

Programming Notes

	7	6	5	4	3	2	1	0
Bit	NCWIN0_START[23–16]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	NCWIN0_START[23–16]	**Start Address Bits SA23–SA16** Contains bits 23–16 of the start address for non-cacheable window 0. There are 1024 possible start addresses which reside on 64 Kbyte boundaries in the destination address space. If all ten bits are cleared, the window start address SA25–SA0 would be 0000000h.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved	CACHE_ SMM_EN	FLUSH_ ENTRY_DIS	NCWIN0_SIZE[2–0]			NCWIN0_START[25–24]	
Default	0	0	0	0	0	0	0	0
R/W		R/W	R/W	R/W			R/W	

Bit	Name	Function
7	Reserved	**Reserved** During read/modify/write operations, software must preserve this bit.
6	CACHE_SMM_EN	**Enable Caching in SMM** 0 = Caching is disabled in SMM 1 = Caching is enabled in SMM
5	FLUSH_ENTRY_DIS	**Disable Auto-Flush on SMM Entry** 0 = Automatic cache flushing upon SMM entry is enabled 1 = Automatic cache flushing upon SMM entry is disabled
4–2	NCWIN0_SIZE[2–0]	**Window Size Selection Bits** Selects one of the five possible window sizes for non-cacheable window 0. 0 0 0 = Window Disabled 0 0 1 = 64 Kbytes 0 1 0 = 128 Kbytes 0 1 1 = 256 Kbytes 1 0 0 = 512 Kbytes 1 0 1 = 1 Mbytes 1 1 0 = Window Disabled 1 1 1 = Window Disabled
1–0	NCWIN0_START[25–24]	**Start Address Bits SA25–SA24** Contains bits 25–24 of the start address for Non-cacheable Window 0. There are 1024 possible start addresses which reside on 64 Kbytes boundaries in the destination address space. If all ten bits are cleared, the window start address SA25–SA0 would be 0000000h.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	NCWIN1_START[23–16]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	NCWIN1_START[23–16]	**Start Address Bits SA23–SA16** Contains bits 23–16 of the start address for Non-cacheable Window 1. There are 1024 possible start addresses which reside on 64 Kbytes boundaries in the destination address space. If all ten bits are cleared, the window start address SA25–SA0 would be 0000000h.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved			NCWIN1_SIZE[2–0]			NCWIN1_START[25–24]	
Default	x	x	x	0	0	0	0	0
R/W				R/W			R/W	

Bit	Name	Function
7–5	Reserved	**Reserved** During read/modify/write operations, software must preserve these bits.
4–2	NCWIN1_SIZE[2–0]	**Window Size Selection Bits** Selects one of the five possible window sizes for Non-Cacheable Window 1. 0 0 0 = Window Disabled 0 0 1 = 64 Kbytes 0 1 0 = 128 Kbytes 0 1 1 = 256 Kbytes 1 0 0 = 512 Kbytes 1 0 1 = 1 Mbytes 1 1 0 = Window Disabled 1 1 1 = Window Disabled
1–0	NCWIN1_START[25–24]	**Start Address Bits SA25–SA24** Contains bits 25–24 of the start address for non-cacheable window 1. There are 1024 possible start addresses which reside on 64 Kbytes boundaries in the destination address space. If all ten bits are cleared, the translated window start address SA25–SA0 would be 0000000h.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	VIDCACHE	SHUTDN_DET	MMU_DLY	VL_RESET	VL_EN	SW_FLUSH	SW_WB	WBACK
Default	0	0	0	1	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R	R	R/W

Bit	Name	Function
7	VIDCACHE	**Graphics Memory Write-through Caching** This bit controls the write-through caching of the graphics memory regions. 0 = Graphics controller memory areas are not write-through cached 1 = Graphics controller memory areas are write-through cached With this bit set while the graphics controller is configured for any CGA compatible text mode, the fixed size (16 Kbytes) refresh and font buffers will be cached. With this bit set in graphics mode, the variable size refresh buffer (determined by the current graphics mode) will be cached. No font buffer is available in graphics mode. The unified memory architecture allows relocation of these cached areas as defined by indexes 4Dh, 4Eh, and 4Fh in the graphics controller indexed address space (3x4h/3x5h). Except as explained below, the region that is cached is the same size as the buffer. For CGA graphics mode, the refresh buffer is 16 Kbytes in size. For paged, non-CGA graphics, although the buffer is 64 Kbytes (with a base address fixed at B8000h in this case), the cached region size depends on the number of bits per pixel that are configured. For 1 BPP, the cached area is 64 Kbytes wide, for 2 or 4 BPP, the cached area is 128 Kbytes wide. For non-paged, non-CGA graphics, the cached region and the buffer sizes are equal. For 1 BPP, the cached area is 64 Kbytes wide, for 2 or 4 BPP, the cached area is 128 Kbytes wide.
6	SHUTDN_DET	**CPU Shut Down Cycle Status Bit** If the CPU core performs a shutdown cycle, this bit will be set. Software should write this bit to '0b' to clear. This bit is automatically cleared when the master reset is asserted.
5	MMU_DLY	**MMU DRAM Access Delay** When the ÉlanSC400 microcontroller is operated at 2.7 volts, a wait state must be added to MMU generated DRAM hits. 0 = Operation at 3.3 volts, no extra wait state required for MMU DRAM hits 1 = Operation at 2.7 volts, extra wait state automatically inserted for MMU DRAM hits
4	VL_RESET	**Vesa Local Bus Reset** 0 = $\overline{\text{VL\_RESET}}$ deasserted 1 = $\overline{\text{VL\_RESET}}$ asserted
3	VL_EN	**Vesa Local Bus Interface Enable** 0 = Disabled 1 = Enabled The VL_RESET bit in this register should be asserted during modifications of VL_EN to ensure that changes in the states of the VL-Bus interface pins do not adversely effect VL-Target device(s).
2	SW_FLUSH	**S/W Flush Status** 0 = Flush special bus cycle not detected 1 = Flush special bus cycle detected This bit is cleared when read. Note that this bit will not be set when the CPU FLUSH signal is asserted. This bit will be set only as a result of detection of the Am486 CPU's bus cycle that accompanies execution of the Am486 CPU's INVD opcode.

Bit	Name	Function
1	SW_WB	**S/W Write-back Status** 0 = Write-back special bus cycle not detected 1 = Write-back special bus cycle detected This bit is cleared when read. Note that this bit will not be set when the CPU FLUSH signal is asserted. This bit will be set only as a result of detection of the Am486 CPU's bus cycle that accompanies execution of the Am486 CPU's WBINVD opcode.
0	WBACK	**CPU Cache Policy Select** 0 = Write-through 1 = Write-back This bit is cleared on CPU reset assertions in addition to system resets. Immediately after clearing this bit, the programmer should perform a WBINVD instruction to maintain cache coherency.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved				ENEXTBUF	PCC_BOOT	ROMCS0CFG[1–0]	
Default	x	x	x	x	?	?	?	?
R/W					R	R/W	R/W	

Bit	Name	Function
7–4	Reserved	**Reserved** During read/modify/write operations, software must preserve these bits.
3	ENEXTBUF	**Enable External Buffer Controls** Determines the availability of the external buffer control signal on external pins: 0 = $\overline{\text{DBUFOE}}$, $\overline{\text{DBUFRDH}}$, and $\overline{\text{DBUFRDL}}$ are not available on the GPIO_CS4, GPIO_CS3, and GPIO_CS2 pins respectively 1 = $\overline{\text{DBUFOE}}$, $\overline{\text{DBUFRDH}}$, and $\overline{\text{DBUFRDL}}$ are available on the GPIO_CS4, GPIO_CS3, and GPIO_CS2 pins respectively This bit is initialized by latching the state of the MA3{CFG3} pin at the deassertion of the $\overline{\text{RESET}}$ pin.
2	PCC_BOOT	**Redirect $\overline{\text{ROMCS0}}$ to PC Card Socket A** Determines which interface will be driven with the $\overline{\text{ROMCS0}}$ (boot vector) decode: 0 =$\overline{\text{ROMCS0}}$ chip select pin 1 =PC Card Socket A This bit is initialized by latching the state of the MA2{CFG2} pin at the deassertion of the $\overline{\text{RESET}}$ pin. Software can write this bit to dynamically control the redirection of $\overline{\text{ROMCS0}}$ cycles between the $\overline{\text{ROMCS0}}$ pin and the PC Card interfaces at run time. The ability to write this bit is provided for test purposes or specialized applications. Improper manipulation of this bit will cause unexpected results.
1–0	ROMCS0CFG[1–0]	**$\overline{\text{ROMCS0}}$ Data Bus Width Status and Control** These bits configure the data bus width for the devices connected to $\overline{\text{ROMCS0}}$ and enable the $\overline{\text{R32BFOE}}$ signal on an external pin:

ROMCS0CFG {CFG1}	ROMCS0CFG {CFG0}	$\overline{\text{ROMCS0}}$ Width	$\overline{\text{R32BFOE}}$ Available
0	0	8-bit	No
0	1	8-bit	No
1	0	16-bit	No
1	1	32-bit	Yes

These bits are initialized by latching the state of the MA0{CFG0} and MA1{CFG1} pins at the deassertion of the $\overline{\text{RESET}}$ pin. In addition, software can dynamically change the width of the $\overline{\text{ROMCS0}}$ interface by writing to these bits. If the 32-bit interface is configured for $\overline{\text{ROMCS0}}$, the $\overline{\text{ROMCS0}}$ interface is always forced into Fast ROM mode. In addition, configuring the $\overline{\text{ROMCS0}}$ interface for 32-bit operation enables the $\overline{\text{R32BFOE}}$ (buffer enable for low word of 32-bit ROM0 data bus) signal to go off the ÉlanSC400 microcontroller and be asserted during accesses to $\overline{\text{ROMCS0}}$.

Caution: The ability to write these bits is provided for test purposes or specialized applications, and special external system logic can be required to support this. Improper manipulation of these bits will cause unexpected results.

	7	6	5	4	3	2	1	0
Bit	BOOT_ CACHE_EN	PCC_ROM1_ EN	SHADOW_EN	ALIASEDBVEC _DIS	BOOTVEC_DIS	CSEG_EN	DSEG_EN	ESEG_EN
Default	x	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	BOOT_CACHE_EN	**Boot ROM Region (3FF0000–3FFFFFFh) Cache Control** 0 = Disable caching of linear accesses to the boot ROM region 1 = Enable caching of linear accesses to the boot ROM region This bit has no effect if the boot ROM is disabled (CSC index 21h[3] = '1b').
6	PCC_ROM1_EN	**Redirect ROMCS1 to PC Card Socket A** When this bit is set, all accesses to ROM1 space get redirected to PC Card Socket A (ROMCS1 is disabled). When this bit is cleared, all accesses to ROMCS1 space result in the generation of ROMCS1.
5	SHADOW_EN	**Enable Shadowing** When this bit is set, linear accesses to any segments that are enabled for linear ROMCS0 decode (see bits 4 and 2–0 of this register) are redirected to DRAM at the same address. When this bit is cleared, linear accesses to any segments that are enable for ROMCS0 decode (see bits 4 and 2–0 of this register) are directed to ROMCS0 at the same address.
4	ALIASEDBVEC_DIS	**Disable 00F0000–00FFFFFh for Linear ROM0 Decode** 0 = Linear accesses to this region cause either ROMCS0 to be generated or a DRAM cycle to be generated based on bit 5 of this register. 1 = Linear accesses to this region will be directed to the ISA bus or to the target device and address pointed to by one of MMS Windows C–F if such window is located in this region of CPU address space If in SMM mode, linear accesses to this region will go to DRAM regardless. Also note that linear accesses to this region generate ROMCS0 by default at power-on reset. This bit functions similarly to bits 2–0 of this register, but the sense of the control bit is inverted.
3	BOOTVEC_DIS	**Disable 3FF0000–3FFFFFFh for Linear ROM0 Decode** 0 = Linear accesses to this region cause ROMCS0 to be generated even if ROM shadowing is enabled via bit 5 of this register 1 = Linear accesses to this region do not generate ROMCS0 Accesses to this region will go to system DRAM, or to the target device and address pointed to by one of MMS Windows C-F if such window is located in this region of CPU address space. This bit will be cleared whenever an SRESET of the CPU core is generated (i.e., slow reset, via port EEh, or via port 92h).
2	CSEG_EN	**Enable 00C0000–00CFFFFh for Linear ROM0 Decode** 0 = Linear accesses to this region will be directed to the ISA bus or to the target device and address pointed to by one of MMS Windows C–F if such window is located in this region of CPU address space 1 = Linear accesses to this region cause either ROMCS0 to be generated or a DRAM cycle to be generated based on bit 5 of this register

Bit	Name	Function
1	DSEG_EN	**Enable 00D0000–00DFFFFh for Linear $\overline{\text{ROMCS0}}$ Decode** 0 = Linear accesses to this region will be directed to the ISA bus or to the target device and address pointed to by one of MMS Windows C–F if such window is located in this region of CPU address space 1 = Linear accesses to this region cause either $\overline{\text{ROMCS0}}$ to be generated or a DRAM cycle to be generated based on bit 5 of this register
0	ESEG_EN	**Enable 00E0000-00EFFFFh for Linear $\overline{\text{ROMCS0}}$ Decode** 0 = Linear accesses to this region will be directed to the ISA bus or to the target device and address pointed to by one of MMS Windows C–F if such window is located in this region of CPU address space 1 = Linear accesses to this region cause either $\overline{\text{ROMCS0}}$ to be generated or a DRAM cycle to be generated based on bit 5 of this register

Programming Notes

MMS Windows may not be located in any segment that is enabled for linear $\overline{\text{ROMCS0}}$ decode.

	7	6	5	4	3	2	1	0
Bit	CSEG_ CACHE_EN	DSEG_ CACHE_EN	ESEG_ CACHE_EN	FSEG_ CACHE_EN	CSEG_WP	DSEG_WP	ESEG_WP	FSEG_WP
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	CSEG_CACHE_EN	**Enable Caching for 00C0000–00CFFFFh Linear Accesses** 0 = Caching is disabled for this region 1 = This region is cacheable
6	DSEG_CACHE_EN	**Enable Caching for 00D0000–00DFFFFh Linear Accesses** 0 = Caching is disabled for this region 1 = This region is cacheable
5	ESEG_CACHE_EN	**Enable Caching for 00E0000–00EFFFFh Linear Accesses** 0 = Caching is disabled for this region 1 = This region is cacheable
4	FSEG_CACHE_EN	**Enable Caching for 00F0000–00FFFFFh Linear Accesses** 0 = Caching is disabled for this region 1 = This region is cacheable
3	CSEG_WP	**Write Protect 00C0000–00CFFFFh Linear Accesses** 0 = Linear writes to this shadow-able region are enabled 1 = Linear writes to this shadow-able region are inhibited
2	DSEG_WP	**Write Protect 00D0000–00DFFFFh Linear Accesses** 0 = Linear writes to this shadow-able region are enabled 1 = Linear writes to this shadow-able region are inhibited
1	ESEG_WP	**Write Protect 00E0000–00EFFFFh Linear Accesses** 0 = Linear writes to this shadow-able region are enabled 1 = Linear writes to this shadow-able region are inhibited
0	FSEG_WP	**Write Protect 00F0000–00FFFFFh Linear Accesses** 0 = Linear writes to this shadow-able region are enabled 1 = Linear writes to this shadow-able region are inhibited

Programming Notes

Individual bits in the above register become don't cares if the segment(s) that they refer to are not enabled for linear decode via CSC index 21h.

Use caution when write-protecting a cached region of memory. Doing so can result in cache incoherency since the write protection applies to the external memory device only. Therefore, a write to a cached, write-protected region will result in only the cached copy of the data being updated, but not the copy stored in an external memory device. Caching should be disabled for regions where write protection is enabled.

	7	6	5	4	3	2	1	0
Bit	Reserved		CS_EARLY0	FAST_ ROMCS0	ROMCS0_ WS_SLCT	DSIZE0[1–0]		Reserved
Default	x	x	0	0	0	?	?	x
R/W			R/W	R/W	R/W	R		

Bit	Name	Function
7–6	Reserved	**Reserved** During read/modify/write operations, software must preserve these bits.
5	CS_EARLY0	**Early Chip Select** This bit enables an early chip select for the ROMCS0 interface when set. 0 = In Fast ROMCS0 mode, ROMCS0 is asserted when the ROMCS0 address decode is true and is synchronized with the ROM controller clock edge. When Fast_ROMCS0 mode is not enabled, ROMCS0 is additionally qualified with the ROM command signal. 1 = The ROMCS0 signal is generated as a simple, unqualified address decode thus causing the chip select to be generated earlier in the cycle For a write cycle where FAST_ROMCS0 = 0, this bit must always be ‘1b’. For all other cases, this bit can be either ‘0b’ or ‘1b’.
4	FAST_ROMCS0	**ROMCS0 Access Speed** This bit selects the speed of the ROMCS0 interface. Fast ROMCS0 mode must be selected in order to enable burst mode ROM operation. In addition, if the x32 interface is configured, the ROM controller is forced into Fast ROM mode always, and this bit becomes a don’t care for either read or write access. 0 = Normal speed (ISA timings) Wait states can be added to the ROM cycles by deasserting the ISA IOCHRDY signal. 1 = Fast speed Cycles run at the CPU 1x clock rate up to a maximum of 33 MHz. Wait states are controlled by bit 3 of this register and the ROMCS0 Configuration Register B.

Bit	Name	Function
3	$\overline{\text{ROMCS0}}$_WS_SLCT	**Select $\overline{\text{ROMCS0}}$ Wait State Control** New "burst mode" ROM devices support faster access times when performing transfers on paragraph boundaries. The first 32-bit access of the transfer occurs using the same timings as a non-burst access, but the subsequent accesses (7 or 3 respectively) that make up a paragraph occur using faster timings. Support for Burst mode ROM devices requires special $\overline{\text{ROMCS0}}$ wait state handling which is enabled by this bit. This bit selects how CPU wait states will be inserted for $\overline{\text{ROMCS0}}$ accesses. By default, the ROM controller does not use Burst mode ROM timings. All of the following must be true for the ÉlanSC400 ROM controller to use Burst mode ROM timings: – This bit must be set. – The CPU is requesting a burst transfer to support a cache line fill, etc. Thus, the burst mode ROM access will occur on a paragraph boundary only. – Caching is enabled for the ROM access. – The Fast_$\overline{\text{ROMCS0}}$ bit is set. This bit has no effect if FAST_$\overline{\text{ROMCS0}}$ = '0b'. Assuming FAST_$\overline{\text{ROMCS0}}$ = '1b': 0 = Use wait states as defined by CSC index 24h[2–0] for all $\overline{\text{ROMCS0}}$ accesses. 1 = Use wait states as defined by CSC index 24h[2–0] for all non-burst $\overline{\text{ROMCS0}}$ accesses. For burst $\overline{\text{ROMCS0}}$ accesses, use the wait states as defined by CSC index 24h[2-0] for the first access of the burst, and then use the wait states defined by CSC index 24h[4-3] for the other accesses which make up the burst.
2–1	DSIZE0[1–0]	**$\overline{\text{ROMCS0}}$ Data Bus Width Status** These two bits can be read back to determine the $\overline{\text{ROMCS0}}$ data bus width which is set via pin strapping options, and latched at power-on reset. 0 x = 8-bit device 1 0 = 16-bit device 1 1 = 32-bit device If the x32 interface is configured, the ROM controller is forced into Fast ROM mode always. See bits 1–0 of the Pin Strap Status Register (CSC index register 20h) for more detail on $\overline{\text{ROMCS0}}$ width control.
0	Reserved	**Reserved** During read/modify/write operations, software must preserve this bit.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved			WAIT_BRST0[1–0]		WAIT_NBRST0[2–0]		
Default	x	x	x	0	0	0	0	1
R/W				R/W		R/W		

Bit	Name	Function
7–5	Reserved	**Reserved** During read/modify/write operations, software must preserve these bits.
4–3	WAIT_BRST0[1–0]	**Wait States for Burst Access** When Fast ROM mode for ROMCS0 is enabled, these two bits program the ROMCS0 interface wait states for all cycles of a burst access that occur after the initial cycle. The wait state starts at the transition of SA3–SA0 during burst cycles. When Fast ROM mode for ROMCS0 is disabled, these bits have no effect. 0 0 = 0 wait states 0 1 = 1 wait states 1 0 = 2 wait states 1 1 = 3 wait states
2–0	WAIT_NBRST0[2–0]	**Wait States for Non-burst Mode** When Fast ROM mode for ROMCS0 is enabled, these three bits program the ROMCS0 interface wait states for non-burst mode. In addition, if Fast ROM mode and burst mode for ROMCS0 is enabled, these three bits select the number of wait states generated for the initial access of burst sequence. When FAST_ROMCS0 is disabled, these bits have no effect. 0 0 0 = 0 wait states 0 0 1 = 1 wait states ..... 1 1 1 = 7 wait states

Programming Notes

The value programmed for the WAIT_NBRST0 parameter must be greater than or equal to the value programmed for the WAIT_BRST0 parameter. Failure to do this will result in incorrect ROM controller operation.

	7	6	5	4	3	2	1	0
Bit	Reserved		CS_EARLY1	FAST_ ROMCS1	ROMCS1_ WS_SLCT	DSIZE1[1–0]		Reserved
Default	x	x	0	0	0	0	0	x
R/W			R/W	R/W	R/W	R/W		

Bit	Name	Function
7–6	Reserved	**Reserved** During read/modify/write operations, software must preserve these bits.
5	CS_EARLY1	**Early Chip Select** This bit enables an early chip select for the ROMCS1 interface when set. 0 = In Fast ROMCS1 mode, ROMCS1 is asserted when the ROMCS1 address decode is true and is synchronized with the ROM controller clock edge. When Fast_ROMCS1 mode is not enabled, ROMCS1 is additionally qualified with the ROM command signal. 1 = The ROMCS1 signal is generated as a simple, unqualified address decode thus causing the chip select to be generated earlier in the cycle For a write cycle where FAST_ROMCS1 = 0, this bit must always be '1b'. For all other cases, this bit can be either '0b' or '1b'.
4	FAST_ROMCS1	**ROMCS1 Access Speed** This bit selects the speed of the ROMCS1 interface. Fast ROMCS1 mode must be selected in order to enable burst mode ROM operation. In addition, if the x32 interface is configured, the ROM controller is forced into Fast ROMCS1 mode always, and this bit becomes a don't care for either read or write access. 0 = Normal speed (ISA timings) Wait states can be added to the ROM cycles by deasserting the ISA IOCHRDY signal. 1 = Fast speed Cycles run at the CPU 1X clock rate up to a maximum of 33 MHz. Wait states are controlled by bit 3 of this register and the ROMCS1 Configuration Register B.

Bit	Name	Function
3	ROMCS1_WS_SLCT	**Select ROMCS1 Wait State Control** New "burst mode" ROM devices support faster access times when performing transfers on paragraph boundaries. The first 32-bit access of the transfer occurs using the same timings as a non-burst access, but the subsequent accesses (7 or 3 respectively) that make up a paragraph occur using faster timings. Support for Burst mode ROM devices requires special ROMCS1 wait state handling which is enabled by this bit. This bit selects how CPU wait states will be inserted for ROMCS1 accesses. By default, the ROM controller does not use Burst mode ROM timings. All of the following must be true for the ÉlanSC400 ROM controller to use Burst mode ROM timings: – This bit must be set. – The CPU is requesting a burst transfer to support a cache line fill, etc. Thus, the burst mode ROM access will occur on a paragraph boundary only. – Caching is enabled for the ROM access. – The Fast_ROMCS1 bit is set. This bit has no effect if FAST_ROMCS1 = '0b'. Assuming FAST_ROMCS1 = '1b': 0 = Use wait states as defined by CSC index 24h[2–0] for all ROMCS1 accesses. 1 = Use wait states as defined by CSC index 24h[2–0] for all non-burst ROMCS1 accesses. For burst ROMCS1 accesses, use the wait states as defined by CSC index 24h[2-0] for the first access of the burst, and then use the wait states defined by CSC index 24h[4-3] for the other accesses which make up the burst.
2–1	DSIZE1[1–0]	**ROMCS1 Data Bus Width Status and Control** These two bits can be read back to determine the ROMCS1 data bus width which is set via pin strapping options, and latched at power-on reset. 0 x = 8-bit device 1 0 = 16-bit device 1 1 = 32-bit device If the 32-bit interface is configured for ROMCS1, the ROMCS1 interface is always forced into Fast ROM mode. Also note that configuring the ROMCS1 or ROMCS2 interfaces for 32-bit operation does not enable the R32BFOE signal. This signal will be driven from the ÉlanSC400 microcontroller only if the ROMCS0 interface is enabled for 32-bit operation, and only then for accesses to the ROMCS0 interface.
0	Reserved	**Reserved** During read/modify/write operations, software must preserve this bit.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved			WAIT_BRST1[1–0]		WAIT_NBRST1[2–0]		
Default	x	x	x	0	0	0	0	0
R/W				R/W		R/W		

Bit	Name	Function
7–5	Reserved	**Reserved** During read/modify/write operations, software must preserve these bits.
4–3	WAIT_BRST1[1–0]	**Wait States for Burst Access** When Fast ROM mode for ROMCS1 is enabled, these two bits program the ROMCS1 interface wait states for all cycles of a burst access that occur after the initial cycle. The wait state starts at the transition of SA3–SA0 during burst cycles. When Fast ROM mode for ROMCS1 is disabled, these bits have no effect. 0 0 = 0 wait states 0 1 = 1 wait states 1 0 = 2 wait states 1 1 = 3 wait states
2–0	WAIT_NBRST1[2–0]	**Wait States for Non-burst Access** When Fast ROM mode for ROMCS1 is enabled, these three bits program the ROMCS1 interface wait states for non-burst mode. In addition, if Fast ROM mode and burst mode for ROMCS1 is enabled, these three bits select the number of wait states generated for the initial access of burst sequence. When FAST_ROMCS1 is disabled, these bits have no effect. 0 0 0 = 0 wait states 0 0 1 = 1 wait states ..... 1 1 1 = 7 wait states

Programming Notes

The value programmed for the WAIT_NBRST1 parameter must be greater than or equal to the value programmed for the WAIT_BRST1 parameter. Failure to do this will result in incorrect ROM controller operation.

	7	6	5	4	3	2	1	0
Bit	Reserved		CS_EARLY2	FAST_ ROMCS2	ROMCS2_ WS_SLCT	DSIZE2[1–0]		Reserved
Default	x	x	0	0	0	0	0	x
R/W			R/W	R/W	R/W	R/W		

Bit	Name	Function
7–6	Reserved	**Reserved** During read/modify/write operations, software must preserve these bits.
5	CS_EARLY2	**Early Chip Select** This bit enables an early chip select for the ROMCS2 interface when set. 0 = In Fast ROMCS2 mode, ROMCS2 is asserted when the ROMCS2 address decode is true and is synchronized with the ROM controller clock edge. When Fast_ROMCS2 mode is not enabled, ROMCS2 is additionally qualified with the ROM command signal. 1 = The ROMCS2 signal is generated as a simple, unqualified address decode thus causing the chip select to be generated earlier in the cycle. For a write cycle where FAST_ROMCS2 = 0, this bit must always be '1b'. For all other cases, this bit can be either '0b' or '1b'.
4	FAST_ROMCS2	**ROMCS2 Access Speed** This bit selects the speed of the ROMCS2 interface. Fast ROMCS2 mode must be selected in order to enable burst mode ROM operation. In addition, if the x32 interface is configured, the ROM controller is forced into Fast ROMCS2 mode always, and this bit becomes a don't care for either read or write access. 0 = Normal speed (ISA timings) Wait states can be added to the ROM cycles by deasserting the ISA IOCHRDY signal. 1 = Fast speed Cycles run at the CPU 1X clock rate up to a maximum of 33 MHz. Wait states are controlled by bit 3 of this register and the ROMCS2 Configuration Register B.

Bit	Name	Function
3	ROMCS2_WS_SLCT	**Select ROMCS2 Wait State Control** New "burst mode" ROM devices support faster access times when performing transfers on paragraph boundaries. The first 32-bit access of the transfer occurs using the same timings as a non-burst access, but the subsequent accesses (7 or 3 respectively) that make up a paragraph occur using faster timings. Support for Burst mode ROM devices requires special ROMCS2 wait state handling which is enabled by this bit. This bit selects how CPU wait states will be inserted for ROMCS2 accesses. By default, the ROM controller does not use Burst mode ROM timings. All of the following must be true for the ÉlanSC400 ROM controller to use Burst mode ROM timings: – This bit must be set. – The CPU is requesting a burst transfer to support a cache line fill, etc. Thus, the burst mode ROM access will occur on a paragraph boundary only. – Caching is enabled for the ROM access. – The Fast_ROMCS2 bit is set. This bit has no effect if FAST_ROMCS2 = '0b'. Assuming FAST_ROMCS2 = '1b': 0 = Use wait states as defined by CSC index 24h[2–0] for all ROMCS2 accesses. 1 = Use wait states as defined by CSC index 24h[2–0] for all non-burst ROMCS2 accesses. For burst ROMCS2 accesses, use the wait states as defined by CSC index 24h[2-0] for the first access of the burst, and then use the wait states defined by CSC index 24h[4-3] for the other accesses which make up the burst.
2–1	DSIZE2[1–0]	**ROMCS2 Data Bus Width Status and Control** These two bits can be read back to determine the ROMCS2 data bus width which is set via pin strapping options, and latched at power-on reset. 0 x = 8-bit device 1 0 = 16-bit device 1 1 = 32-bit device If the 32-bit interface is configured for ROMCS2, the ROMCS2 interface is always forced into Fast ROMCS2 mode. Also note that configuring the ROMCS1 or ROMCS2 interfaces for 32-bit operation does not enable the R32BFOE signal. This signal will be driven from the ÉlanSC400 microcontroller only if the ROMCS0 interface is enabled for 32-bit operation, and only then for accesses to the ROMCS0 interface.
0	Reserved	**Reserved** During read/modify/write operations, software must preserve this bit.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved			WAIT_BRST2[1–0]		WAIT_NBRST2[2–0]		
Default	x	x	x	0	0	0	0	0
R/W				R/W		R/W		

Bit	Name	Function
7–5	Reserved	**Reserved** During read/modify/write operations, software must preserve these bits.
4–3	WAIT_BRST2[1–0]	**Wait States for Burst Mode** When Fast ROM mode for ROMCS2 is enabled, these two bits program the ROMCS2 interface wait states for all cycles of a burst access that occur after the initial cycle. The wait state starts at the transition of SA3–SA0 during burst cycles. When Fast ROM mode for ROMCS2 is disabled, these bits have no effect. 0 0 = 0 wait states 0 1 = 1 wait states 1 0 = 2 wait states 1 1 = 3 wait states
2–0	WAIT_NBRST2[2–0]	**Wait States for Non-burst Mode** When Fast ROM mode for ROMCS2 is enabled, these three bits program the ROMCS2 interface wait states for non-burst mode. In addition, if Fast ROM mode and burst mode for ROMCS2 is enabled, these three bits select the number of wait states generated for the initial access of burst sequence. When FAST_ROMCS2 is disabled, these bits have no effect. 0 0 0 = 0 wait states 0 0 1 = 1 wait states ..... 1 1 1 = 7 wait states

Programming Notes

The value programmed for the WAIT_NBRST2 parameter must be greater than or equal to the value programmed for the WAIT_BRST2 parameter. Failure to do this will result in incorrect ROM controller operation.

DSIZEx[1–0]	ROMCSx Data Bus Width	ROMCS0_ WS_SLCT	FAST_ ROMCSx	ROMCSx Configuration Summary
0 x	8-bit	don't care	0	8-bit, Normal Speed, Non-Burst
0 x	8-bit	don't care	1	8-bit, Fast Speed, Non-Burst
1 0	16-bit	don't care	0	16-bit, Normal Speed, Non-Burst
1 0	16-bit	0	1	16-bit, Fast Speed, Non-Burst
1 0	16-bit	1	1	16-bit, Fast Speed, Burst Capable
1 1	32-bit	0	don't care	32-bit, Fast Speed, Non-Burst
1 1	32-bit	1	don't care	32-bit, Fast Speed, Burst Capable

Non-burst = Not capable of Burst mode ROM interface timings
Burst capable = Capable of Burst mode ROM interface timings under conditions specified in bit 3 of registers 23h, 25h, and 27h.

	7	6	5	4	3	2	1	0
Bit	MMSF_ CACHE_EN	MMSE_ CACHE_EN	MMSD_ CACHE_EN	MMSC_ CACHE_EN	MMSF_WP	MMSE_WP	MMSD_WP	MMSC_WP
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	MMSF_CACHE_EN	**Cache Enable Control for MMS Window F** 0 = MMS Window F is non-cacheable 1 = MMS Window F is cacheable
6	MMSE_CACHE_EN	**Cache Enable Control for MMS Window E** 0 = MMS Window E is non-cacheable 1 = MMS Window E is cacheable
5	MMSD_CACHE_EN	**Cache Enable Control for MMS Window D** 0 = MMS Window D is non-cacheable 1 = MMS Window D is cacheable
4	MMSC_CACHE_EN	**Cache Enable Control for MMS Window C** 0 = MMS Window C is non-cacheable 1 = MMS Window C is cacheable
3	MMSF_WP	**Write Protect Control for MMS Window F** 0 = MMS Window F is not write-protected 1 = MMS Window F is write-protected
2	MMSE_WP	**Write Protect Control for MMS Window E** 0 = MMS Window E is not write-protected 1 = MMS Window E is write-protected
1	MMSD_WP	**Write Protect Control for MMS Window D** 0 = MMS Window D is not write-protected 1 = MMS Window D is write-protected
0	MMSC_WP	**Write Protect Control for MMS Window C** 0 = MMS Window C is not write-protected 1 = MMS Window C is write-protected

Programming Notes

To enable the set up of MMS Windows C–F, the internal PC Card controller must be enabled (D0h[1] = 1), and operating in standard mode (F1h[0] = 0). Once any of MMS Windows C–F are opened (via 3E0/3E1h index space), disabling the internal PC Card controller does not disable the MMS window(s), but disallows re-configuration of them until the internal PC Card controller is re-enabled. MMS Windows C–F are not restricted from being opened in system address space below 64 Kbytes as are the PC Card Socket B Windows 1–4 (which share resources with MMS Windows C–F).

	7	6	5	4	3	2	1	0
Bit	MMSF_DEVICE[1–0]		MMSE_DEVICE[1–0]		MMSD_DEVICE[1–0]		MMSC_DEVICE[1–0]	
Default	0	0	0	0	0	0	0	0
R/W	R/W		R/W		R/W		R/W	

Bit	Name	Function
7–6	MMSF_DEVICE[1–0]	**Physical Device Selection Bits for MMS Window F** Selects one of the four devices that MMS Window F can be pointed to. 0 0 = $\overline{\text{ROMCS0}}$ 0 1 = $\overline{\text{ROMCS1}}$ 1 0 = $\overline{\text{ROMCS2}}$ 1 1 = System memory (DRAM) MMS Windows C–F are only available if the Mode bit of the PC Card Mode and DMA Control Register (CSC index F1h[0]) is cleared.
5–4	MMSE_DEVICE[1–0]	**Physical Device Selection Bits for MMS Window E** Selects one of the four devices that MMS Window E can be pointed to. 0 0 = $\overline{\text{ROMCS0}}$ 0 1 = $\overline{\text{ROMCS1}}$ 1 0 = $\overline{\text{ROMCS2}}$ 1 1 = System memory (DRAM) MMS Windows C–F are only available if the Mode bit of the PC Card Mode and DMA Control Register (CSC index F1h[0]) is cleared.
3–2	MMSD_DEVICE[1–0]	**Physical Device Selection Bits for MMS Window D** Selects one of the four devices that MMS Window D can be pointed to. 0 0 = $\overline{\text{ROMCS0}}$ 0 1 = $\overline{\text{ROMCS1}}$ 1 0 = $\overline{\text{ROMCS2}}$ 1 1 = System memory (DRAM) MMS Windows C–F are only available if the Mode bit of the PC Card Mode and DMA Control Register (CSC index F1h[0]) is cleared.
1–0	MMSC_DEVICE[1–0]	**Physical Device Selection Bits for MMS Window C** Selects one of the four devices that MMS Window C can be pointed to. 0 0 = $\overline{\text{ROMCS0}}$ 0 1 = $\overline{\text{ROMCS1}}$ 1 0 = $\overline{\text{ROMCS2}}$ 1 1 = System memory (DRAM) MMS Windows C–F are only available if the Mode bit of the PC Card Mode and DMA Control Register (CSC index F1h[0]) is cleared.

Programming Notes

To enable the set up of MMS Windows C–F, the internal PC Card controller must be enabled (D0h[1] = 1), and operating in standard mode (F1h[0] = 0). Once any of MMS Windows C–F are opened (via 3E0h/3E1h index space), disabling the internal PC Card controller does not disable the MMS window(s), but disallows re-configuration of them until the internal PC Card controller is re-enabled.

	7	6	5	4	3	2	1	0
Bit	MMSA_DEST_START[22–15]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	MMSA_DEST_START[22–15]	**MMSA Destination Start Address Bits SA22–SA15** Contains bits 22–15 of the destination start address for the 32 Kbytes MMS Window A. There are 2048 possible start addresses which reside on 32 Kbytes boundaries in the destination address space. If all eleven MMS Window A destination bits are cleared, the translated destination start address SA25–SA0 would be 0000000h.

Programming Notes

MMS Window A is 32 Kbytes wide with a fixed address CPU address range of B0000–B7FFFh.

	7	6	5	4	3	2	1	0
Bit	MMSA_DEVICE[1–0]		MMSA_WP	MMSA_ CACHE_EN	MMSA_EN	MMSA_DEST_START[25–23]		
Default	0	0	0	0	0	0	0	0
R/W	R/W		R/W	R/W	R/W	R/W		

Bit	Name	Function
7–6	MMSA_DEVICE[1–0]	**Physical Device Selection Bits for MMS Window A** Selects one of the four devices that MMS Window A can be pointed to: 0 0 = $\overline{\text{ROMCS0}}$ 0 1 = $\overline{\text{ROMCS1}}$ 1 0 = $\overline{\text{ROMCS2}}$ 1 1 = System memory (DRAM)
5	MMSA_WP	**Write Protect bit for MMS Window A** 0 = MMS Window A is not write-protected 1 = MMS Window A is write-protected
4	MMSA_CACHE_EN	**Cache Enable Bit for MMS Window A** 0 = MMS Window A is non-cacheable 1 = MMS Window A is cacheable
3	MMSA_EN	**Enable Bit for MMS Window A** 0 = MMS Window A is disabled 1 = MMS Window A is enabled
2–0	MMSA_DEST_START[25–23]	**MMSA Destination Start Address Bits SA25–SA23** Contains bits 25–23 of the destination start address for the 32 Kbytes MMS Window A. There are 2048 possible start addresses which reside on 32 Kbytes boundaries in the destination address space. If all eleven MMS Window A destination bits are cleared, the translated destination start address SA25–SA0 would be 0000000h.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	MMSB_DEST_START[22–15]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	MMSB_DEST_START[22–15]	**MMSB Destination Start Address Bits SA22–SA15** Contains bits 22–15 of the destination start address for the 64 Kbytes MMS Window B. There are 2048 possible start addresses which reside on 32 Kbytes boundaries in the destination address space. If all eleven MMS Window B destination bits are cleared, the translated destination start address SA25–SA0 would be 0000000h.

Programming Notes

MMS Window B is 64 Kbytes wide with a fixed address CPU address range of 100000–10FFFFh. This memory region is known as the High Memory Area (HMA). It is mappable in target address space on 32 Kbytes boundaries. The 32 Kbytes boundary support allows access of all target address space even when operating in x86 real mode where the top 16 bytes of the HMA are not available due to CPU addressing limitations.

	7	6	5	4	3	2	1	0
Bit	MMSB_DEVICE[1–0]		MMSB_WP	MMSB_ CACHE_EN	MMSB_EN	MMSB_DEST_START[25–23]		
Default	0	0	0	0	0	0	0	0
R/W	R/W		R/W	R/W	R/W	R/W		

Bit	Name	Function
7–6	MMSB_DEVICE[1–0]	**Physical Device Selection Bits for MMS Window B** Selects one of the four devices that MMS Window B can be pointed to. 0 0 = $\overline{\text{ROMCS0}}$ 0 1 = $\overline{\text{ROMCS1}}$ 1 0 = $\overline{\text{ROMCS2}}$ 1 1 = System memory (DRAM)
5	MMSB_WP	**Write Protect Bit for MMS Window B** 0 = MMS Window B is not write-protected 1 = MMS Window B is write-protected
4	MMSB_CACHE_EN	**Cache Enable Bit for MMS Window B** 0 = MMS Window B is non-cacheable 1 = MMS Window B is cacheable
3	MMSB_EN	**Enable Bit for MMS Window B** 0 = MMS Window B is disabled 1 = MMS Window B is enabled
2–0	MMSB_DEST_START[25–23]	**MMSB Destination Start Address Bits SA25–SA23** Contains bits 25–23 of the destination start address for the 64 Kbytes MMS Window B. There are 2048 possible start addresses which reside on 32 Kbytes boundaries in the destination address space. If all eleven MMS Window B destination bits are cleared, the translated destination start address SA25–SA0 would be 0000000h.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	BL0_CLKIO_SLCT[1–0]		Reserved	GP_EQU_ IOCS16	GP_EQU_ IOCHRDY	GP_EQU_ PIRQ1	GP_EQU_ PIRQ0	GP_EQU_ DMA
Default	0	0	0	0	0	0	0	0
R/W	R/W		R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7–6	BL0_CLKIO_SLCT[1–0]	**Select $\overline{BL0}$ or CLK_IO Signal** If CLK_IO is selected as an input, be sure the pin has a stable 1.19318 MHz frequency on it as it will immediately be switched in as the 8254 timer clock. If CLK_IO is selected as an output, program the frequency to output in the clock control registers of the PMU. 0 0 = Pin disabled 0 1 = $\overline{BL0}$ input is available 1 0 = CLK_IO is available as an output 1 1 = CLK_IO is available as the timer clock input
5	Reserved	**Reserved**
4	GP_EQU_IOCS16	**Select GPIO_CS5 Signal or ISA $\overline{IOCS16}$** 0 = GPIO_CS signal is available on this pin 1 = ISA signal is available on this pin
3	GP_EQU_IOCHRDY	**Select GPIO_CS6 Signal or ISA IOCHRDY** 0 = GPIO_CS signal is available on this pin 1 = ISA signal is available on this pin
2	GP_EQU_PIRQ1	**Select GPIO_CS7 Signal or ISA PIRQ1** 0 = GPIO_CS signal is available on this pin 1 = ISA signal is available on this pin
1	GP_EQU_PIRQ0	**Select GPIO_CS8 Signal or ISA PIRQ0** 0 = GPIO_CS signal is available on this pin 1 = ISA signal is available on this pin
0	GP_EQU_DMA	**Select GPIO_CS12–GPIO_CS9 Signals or ISA DMA Signals: PDRQ0, $\overline{PDACK0}$, AEN, TC** 0 = GPIO_CS signals are available on these pins 1 = ISA signals are available on these pins

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved	GPIO_PCPWR_SLCTB	GPIO_PCPWR_SLCTA	GPIO_LBL2_SLCT	GPIO_KBDCOL_SLCT	ISA_KBDROW_SLCT	PP_PCMB_SLCT[1–0]	
Default	x	0	0	0	0	0	0	0
R/W		R/W	R/W	R/W	R/W	R/W	R/W	

Bit	Name	Function
7	Reserved	**Reserved** During read/modify/write operations, software must preserve this bit.
6	GPIO_PCPWR_SLCTB	**Select PC Card Socket B VCC and VPP Control Signals or GPIO Signals** 0 = GPIO18–GPIO16 signals available on these pins 1 = PC Card PCMB_VPP2, PCMB_VPP1, PCMB_VCC (PC Card Socket B VCC/VPP control signals) available on these pins
5	GPIO_PCPWR_SLCTA	**Select PC Card Socket A VCC and VPP Control Signals or GPIO/ GPIO_CS Signals** 0 = GPIO15, GPIO_CS14–GPIO_CS13 signals available on these pins 1 = PC Card PCMA_VPP2, PCMA_VPP1, PCMA_VCC (PC Card Socket A VCC/VPP control signals) available on these pins
4	GPIO_LBL2_SLCT	**Select GPIO19 Signal or $\overline{\text{LBL2}}$ Signal** 0 = GPIO19 signal available on this pin 1 = $\overline{\text{LBL2}}$ signal available on this pin $\overline{\text{LBL2}}$ is simply an indicator that the PMU is currently in Critical Suspend mode.
3	GPIO_KBDCOL_SLCT	**Select Keyboard Column Signals or XT Keyboard Signals** 0 = Keyboard Column signals KBD_COL0–KBD_COL1 are available on the pins 1 = XT keyboard signals XT_CLK, XT_DATA are available on the pins
2	ISA_KBDROW_SLCT	**Select Keyboard Row Signals: KBD_ROW12–KBD_ROW7 or Additional ISA Controls: $\overline{\text{MCS16}}$, $\overline{\text{SBHE}}$, BALE, PIRQ2, PDRQ1, $\overline{\text{PDACK1}}$** 0 = Keyboard Row signals are available on these pins 1 = Additional ISA controls available
1–0	PP_PCMB_SLCT[1–0]	**Select Parallel Port Signals, PC Card Socket B Signals, or GPIOs** 0 0 = GPIO 21–GPIO31 signals available on the pins 0 1 = PC Card Socket B signals $\overline{\text{WP\_B}}$,$\overline{\text{BVD2\_B}}$, $\overline{\text{BVD1\_B}}$, RDY_B, $\overline{\text{CD\_B}}$, $\overline{\text{REG\_B}}$, $\overline{\text{RST\_B}}$, $\overline{\text{MCEH\_B}}$, $\overline{\text{MCEL\_B}}$ are available on the pins 1 0 = Parallel Port signals $\overline{\text{PPDWE}}$, $\overline{\text{PPOEN}}$, SLCT, BUSY, $\overline{\text{ACK}}$, PE, ERROR, INIT, $\overline{\text{SLCTIN}}$, $\overline{\text{AFDT}}$, and $\overline{\text{STRB}}$ are available on the pins 1 1 = Reserved The pins used for GPIO21 and GPIO22 are three stated if this option is selected.

Programming Notes

	7	6	5	4	3	2	1	0
Bit			Reserved				WIRED_PIRQ	GPIO_ PCACD_SLCT
Default	x	x	x	x	x	x	0	0
R/W							R/W	R/W

Bit	Name	Function
7–6	Reserved	**Reserved** During read/modify/write operations, software must preserve these bits.
5–2	Reserved	**Reserved** During read/modify/write operations, software must preserve these bits.
1	WIRED_PIRQ	**Wired for PIRQ Inputs** Indicator to the ÉlanSC400 microcontroller that the system is wired to use the PIRQ pins as programmable IRQ inputs versus the matrix keyboard column signals. This allows the hardware to switch in the proper pin termination during suspend when the ISA interface (Power Down Group C) signals are powered won via CSC index E3h[2]. System firmware should initialize this bit at boot time to reflect the pin usage as follows: 0 = KBD_COL6–KBD_COL2 selected (pull-up used when ISA interface is powered down during suspend) 1 = PIRQ7–PIRQ3 selected (pull-down used when ISA interface is powered down during suspend) This allows the hardware to switch in the proper pin termination during suspend when the ISA interface (Power Down Group C) signals are powered down via CSC index E3h[2].
0	GPIO_PCACD_SLCT	**Enable PC Card Socket A Second Card Detect Input or GPIO Signal** 0 = GPIO 20 signal available on this pin 1 = PC Card Socket A second card detect input $\overline{\text{CD\_A2}}$ available

Programming Notes

	7	6	5	4	3	2	1	0
Bit	CS7_PUEN	CS6_PUEN	CS5_PUEN	CS4_PUEN	CS3_PUEN	CS2_PUEN	CS1_PUEN	CS0_PUEN
Default	1	1	1	1	1	1	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	CS7_PUEN	**GPIO_CS7 Pull-up Enable/Disable** 0 = Disabled 1 = Enabled
6	CS6_PUEN	**GPIO_CS6 Pull-up Enable/Disable** 0 = Disabled 1 = Enabled
5	CS5_PUEN	**GPIO_CS5 Pull-up Enable/Disable** 0 = Disabled 1 = Enabled
4	CS4_PUEN	**GPIO_CS4 Pull-up Enable/Disable** 0 = Disabled 1 = Enabled
3	CS3_PUEN	**GPIO_CS3 Pull-up Enable/Disable** 0 = Disabled 1 = Enabled
2	CS2_PUEN	**GPIO_CS2 Pull-up Enable/Disable** 0 = Disabled 1 = Enabled
1	CS1_PUEN	**GPIO_CS1 Pull-up Enable/Disable** 0 = Disabled 1 = Enabled
0	CS0_PUEN	**GPIO_CS0 Pull-up Enable/Disable** 0 = Disabled 1 = Enabled

Programming Notes

Bits 7–0: The termination state that is specified using bits 7–0 is not actually felt at the pins until the Pin Termination Latch Command bit at CSC index E5[0] is set.

	7	6	5	4	3	2	1	0
Bit	GPIO15_ PDEN	CS14_PDEN	CS13_PDEN	CS12_PDEN	CS11_PUEN	CS10_PUEN	CS9_PUEN	CS8_PUEN
Default	1	1	1	1	1	1	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	GPIO15_PDEN	**GPIO15 Pull-down Enable/Disable** 0 = Disabled 1 = Enabled
6	CS14_PDEN	**GPIO_CS14 Pull-down Enable/Disable** 0 = Disabled 1 = Enabled
5	CS13_PDEN	**GPIO_CS13 Pull-down Enable/Disable** 0 = Disabled 1 = Enabled
4	CS12_PDEN	**GPIO_CS12 Pull-down Enable/Disable** 0 = Disabled 1 = Enabled
3	CS11_PUEN	**GPIO_CS11 Pull-up Enable/Disable** 0 = Disabled 1 = Enabled
2	CS10_PUEN	**GPIO_CS10 Pull-up Enable/Disable** 0 = Disabled 1 = Enabled
1	CS9_PUEN	**GPIO_CS9 Pull-up Enable/Disable** 0 = Disabled 1 = Enabled
0	CS8_PUEN	**GPIO_CS8 Pull-up Enable/Disable** 0 = Disabled 1 = Enabled

Programming Notes

Bits 7–0: The termination state that is specified using bits 7–0 is not actually felt at the pins until the Pin Termination Latch Command bit at CSC index E5[0] is set.

	7	6	5	4	3	2	1	0
Bit	GPIO23_ PUEN	GPIO22_ PUEN	GPIO21_ PUEN	GPIO20_ PUEN	GPIO19_ PUEN	GPIO18_ PDEN	GPIO17_ PDEN	GPIO16_ PDEN
Default	1	1	1	1	1	1	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	GPIO23_PUEN	**GPIO23 Pull-up Enable/Disable** 0 = Disabled 1 = Enabled
6	GPIO22_PUEN	**GPIO22 Pull-up Enable/Disable** 0 = Disabled 1 = Enabled
5	GPIO21_PUEN	**GPIO21 Pull-up Enable/Disable** 0 = Disabled 1 = Enabled
4	GPIO20_PUEN	**GPIO20 Pull-up Enable/Disable** 0 = Disabled 1 = Enabled
3	GPIO19_PUEN	**GPIO19 Pull-up Enable/Disable** 0 = Disabled 1 = Enabled
2	GPIO18_PDEN	**GPIO18 Pull-down Enable/Disable** 0 = Disabled 1 = Enabled
1	GPIO17_PDEN	**GPIO17 Pull-down Enable/Disable** 0 = Disabled 1 = Enabled
0	GPIO16_PDEN	**GPIO16 Pull-down Enable/Disable** 0 = Disabled 1 = Enabled

Programming Notes

Bits 7–0: The termination state that is specified using bits 7–0 is not actually felt at the pins until the Pin Termination Latch Command bit at CSC index E5h[0] is set.

	7	6	5	4	3	2	1	0
Bit	GPIO31_ PUEN	GPIO30_ PUEN	GPIO29_ PUEN	GPIO28_ PUEN	GPIO27_ PUEN	GPIO26_ PUEN	GPIO25_ PUEN	GPIO24_ PUEN
Default	1	1	1	1	1	1	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	GPIO31_PUEN	**GPIO31 Pull-up Enable/Disable** 0 = Disabled 1 = Enabled
6	GPIO30_PUEN	**GPIO30 Pull-up Enable/Disable** 0 = Disabled 1 = Enabled
5	GPIO29_PUEN	**GPIO29 Pull-up Enable/Disable** 0 = Disabled 1 = Enabled
4	GPIO28_PUEN	**GPIO28 Pull-up Enable/Disable** 0 = Disabled 1 = Enabled
3	GPIO27_PUEN	**GPIO27 Pull-up Enable/Disable** 0 = Disabled 1 = Enabled
2	GPIO26_PUEN	**GPIO26 Pull-up Enable/Disable** 0 = Disabled 1 = Enabled
1	GPIO25_PUEN	**GPIO25 Pull-up Enable/Disable** 0 = Disabled 1 = Enabled
0	GPIO24_PUEN	**GPIO24 Pull-up Enable/Disable** 0 = Disabled 1 = Enabled

Programming Notes

Additional pin termination controls can be found in the keyboard section. See CSC index CAh for more information.

Bits 7–0: The termination state that is specified using bits 7–0 is not actually felt at the pins until the Pin Termination Latch Command bit at CSC index E5h[0] is set.

	7	6	5	4	3	2	1	0
Bit	LS_TIMER_CNT	EN_HYPER	EN_SB_LCD	FAST_TIMEO	HS_COUNTING	PMU_FORCE[2–0]		
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R	R/W		

Bit	Name	Function
7	LS_TIMER_CNT	**Low-Speed Timer Count Reset** Enable a secondary activity received in Low-Speed mode to reset the Low-Speed Timer. 0 = Timer not reset 1 = Timer is reset and count starts over
6	EN_HYPER	**Enable Hyper-Speed Mode** Enable Hyper-Speed mode to be entered as a result of wake-ups or activity: 0 = The highest PMU mode that can be entered as a result of wake-ups or activities is High-Speed mode. When this bit is cleared, Hyper-Speed mode can be entered only by writing to the PMU mode force bits in 2–0 of this register. 1 = Hyper-Speed mode entry as a result of wake-ups or activity is enabled. If this bit is set, and the PMU mode is forced to High Speed mode via bits 2–0 of this register, the PMU mode will jump to High Speed mode momentarily, and will then immediately transition to Hyper-Speed mode.
5	EN_SB_LCD	**Enable Graphics Operation in Standby Mode** 0 = Graphics controller is disabled when system enters Standby mode 1 = Graphics controller is still enabled when system enters Standby mode (only if the Graphics mode was enabled before the PMU enters Standby mode)
4	FAST_TIMEO	**Speed Up Suspend and Standby Mode Timers for PMU Code Debug** 0 = Use normal PMU mode timer time-outs 1 = Speed up PMU mode timer time-outs Setting this bit speeds up (only) the PMU mode timer time-outs to aid software debug for routines that use the mode timers. When this bit is set, the new delay can be calculated from the following formula: log2(OldValue/30.5) * 30.5 Where OldValue = the time-out value (in micro seconds) that would be in effect if FAST_TIMEO = 0.

Bit	Name	Function
3	HS_COUNTING	**High-Speed Clock Delay Timer Status** 0 = Timer has expired, and high speed clock is being used 1 = Timer is counting, and intermediate clock is being used Refer to CSC index 45h[5–3] for more detail.
2–0	PMU_FORCE[2–0]	**Force Mode of PMU** When written, these bits will force the system into selected PMU mode. When read, these bits return the last value written. 000 = High-Speed mode 001 = Hyper-Speed mode 010 = Low-Speed mode 011 = Reserved 100 = Standby mode 101 = Suspend mode 110–111 = Reserved After writing to CSC index 40h for any reason, at least two falling edges of the 32 KHz clock (as monitored at CSC index 82h[3]) must transpire before subsequent forced PMU mode changes will be recognized.

Programming Notes

Use caution when performing read/modify/write operations to this register. Since the PMU_FORCE bits retain the value of what was last written (as opposed to the current PMU mode), a read/modify/write operation targeted at changing some other bit in this register may force the ÉlanSC400 microcontroller into an unexpected PMU mode. After reading and modifying the desired bits, either explicitly set the PMU mode bits to a new desired value or read the current mode from CSC index 41h[1–0] and use this value to program the mode force bits.

	7	6	5	4	3	2	1	0
Bit	Reserved	TIME0_NOW	LAST_MODE[2–0]			Reserved	PRES_MODE[1–0]	
Default	x	0	1	1	0	x	0	0
R/W		W	R				R	

Bit	Name	Function
7	Reserved	**Reserved** During read/modify/write operations, software must preserve this bit.
6	TIME0_NOW	**Time Out Now** Set this bit to immediately time out the current mode timer. This bit does not need to be cleared and is not read back. Software should not invoke this feature for at least 15 microseconds after a wake-up from Suspend mode. Software can read the last mode from bits 5–3 of this register and delay 15 microseconds if the last mode was Suspend mode before setting this bit.
5–3	LAST_MODE[2–0]	**Read Last Mode of PMU** When Read will indicate which mode the PMU state machine was in before the present mode. 0 0 0 = High-Speed mode 0 0 1 = Hyper-Speed mode 0 1 0 = Low-Speed mode 0 1 1 = Temporary Low-Speed mode 1 0 0 = Standby mode 1 0 1 = Suspend or Critical Suspend mode 1 1 0 -1 1 1 = Reserved Power-on reset will set these bits to ‘110b’. This will change as a result of the first PMU mode change to occur.
2	Reserved	**Reserved** During read/modify/write operations, software must preserve this bit.
1–0	PRES_MODE[1–0]	**Read Present Mode of PMU** When Read will indicate which mode the PMU state machine is in. 0 0 = High-Speed mode 0 1 = Hyper-Speed mode 1 0 = Low-Speed mode 1 1 = Temporary Low-Speed mode

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved		HS_TIM[2–0]			HYPER_TIM[2–0]		
Default	x	x	0	0	0	0	0	0
R/W			R/W			R/W		

Bit	Name	Function
7–6	Reserved	**Reserved** During read/modify/write operations, software must preserve these bits.
5–3	HS_TIM[2–0]	**High-Speed Timer Value** Timer value to count down in High-Speed mode before dropping to Low-Speed mode. Read returns the last value written: 0 0 0 = Disabled 0 0 1 = 0.125 seconds 0 1 0 = 0.25 seconds 0 1 1 = 0.5 seconds 1 0 0 = 1 second 1 0 1 = 4 seconds 1 1 0 = 8 seconds 1 1 1 = 16 seconds
2–0	HYPER_TIM[2–0]	**Hyper-Speed Timer Value** Timer value to count down in Hyper-Speed mode before dropping to High-Speed mode. Read returns the last value written: 0 0 0 = Disabled 0 0 1 = 0.125 seconds 0 1 0 = 0.25 seconds 0 1 1 = 0.5 seconds 1 0 0 = 1 second 1 0 1 = 4 seconds 1 1 0 = 8 seconds 1 1 1 = 16 seconds

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved		SB_TIM[2–0]			LS_TIM[2–0]		
Default	x	x	0	0	0	0	0	0
R/W			R/W			R/W		

Bit	Name	Function
7–6	Reserved	**Reserved** During read/modify/write operations, software must preserve these bits.
5–3	SB_TIM[2–0]	**Standby Timer Value** Timer value to count down in Standby mode before dropping to Suspend mode. Read returns the last value written. 0 0 0 = Disabled 0 0 1 = 1 minute 0 1 0 = 2 minutes 0 1 1 = 4 minutes 1 0 0 = 8 minutes 1 0 1 = 16 minutes 1 1 0 = 32 minutes 1 1 1 = 60 minutes
2–0	LS_TIM[2–0]	**Low-Speed Timer Value** Timer value to count down in Low-Speed mode before dropping to Standby mode. Read returns the last value written. 0 0 0 = Disabled 0 0 1 = 8 seconds 0 1 0 = 16 seconds 0 1 1 = 32 seconds 1 0 0 = 1 minute 1 0 1 = 4 minutes 1 1 0 = 8 minutes 1 1 1 = 16 minutes

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved		SUS_TIM[2–0]			TLS_TIM[2–0]		
Default	x	x	0	0	0	0	0	0
R/W			R/W			R/W		

Bit	Name	Function
7–6	Reserved	**Reserved** During read/modify/write operations, software must preserve these bits.
5–3	SUS_TIM[2–0]	**Suspend Mode Timer Value** Timer value to count down in Suspend mode. On a time-out the timer can cause a wake- up, an SMI/NMI, or nothing. Read returns the last value written. 0 0 0 = Disabled 0 0 1 = 1 minute 0 1 0 = 8 minutes 0 1 1 = 16 minutes 1 0 0 = 32 minutes 1 0 1 = 1 hour 1 1 0 = 2 hours 1 1 1 = 4 hours
2–0	TLS_TIM[2–0]	**Return from Temporary Low-Speed Mode Timer** Timer value to count down in Temporary Low-Speed mode before returning to the mode called by. Read returns the last value written. 000 = 30 µseconds 001 = 61 µseconds 010 = 122 µseconds 011 = 244 µseconds 100 = 488 µseconds 101 = 976 µseconds 110 = 1.95 milliseconds 111 = 3.9 milliseconds When temporary low-speed PMU mode is entered for the purpose of allowing an unmasked NMI or SMI to be serviced, the temporary low-speed mode timer begins counting down. The PMU will not return to its previous mode until the temporary low-speed timer times out. You must clear all pending NMIs before the temporary low-speed time-out will be felt by the PMU. Thus, if you gate off NMIs via port 70h while further NMIs are still pending, the PMU will remain in temporary low-speed mode past the point where the temporary low speed mode timer timed-out.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved		HSCLKSTEP[2–0]			WAKE_DLY[2–0]		
Default	x	x	0	0	0	0	0	0
R/W			R/W			R/W		

Bit	Name	Function
7–6	Reserved	**Reserved** During read/modify/write operations, software must preserve these bits.
5–3	HSCLKSTEP[2–0]	**High-Speed Clock Stepping** When entering High-Speed mode from Suspend mode or from Standby mode, the clock will default to 8 MHz. When this Timer times out the CPU clock will switch to the programmed speed if it is higher than 8 MHz. If this Timer is disabled the CPU clock will restart at the programmed speed. Read returns the last value written. 0 0 0 = Disabled 0 0 1 = 0.125 seconds 0 1 0 = 0.25 seconds 0 1 1 = 0.5 seconds 1 0 0 = 1 seconds 1 0 1 = 2 seconds 1 1 0 = 4 seconds 1 1 1 = Reserved CSC index 40h[3] allows software to determine if this timer is still running (intermediate 8 MHz clock in effect) or whether this timer has expired.
2–0	WAKE_DLY[2–0]	**Wake-Up Timer Delay** Timer value to count down after a wake up is sensed and the PLLs are started up (if necessary) and the GPIO_CSx signals are switched to High-Speed (or Low-Speed) mode levels (for those GPIO_CSx signals that are programmed to change based on PMU mode) to allow the power supplies to stabilize before the ÉlanSC400 microcontroller starts driving its outputs or using its inputs. Read returns the last value written. 0 0 0 = Disabled 0 0 1 = 0.125 seconds 0 1 0 = 0.25 seconds 0 1 1 = 0.5 seconds 1 0 0 = 1 seconds 1 0 1 = 2 seconds 1 1 0 = 4 seconds 1 1 1 = Reserved

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved				SUS_RES_CFG[1–0]		EN_SUS_RES[1–0]	
Default	x	x	x	x	0	0	0	0
R/W					R/W		R/W	

Bit	Name	Function
7–4	Reserved	**Reserved** During read/modify/write operations, software must preserve these bits.
3–2	SUS_RES_CFG[1–0]	**SUS_RES Pin Trigger Configuration** 0 0 = Rising edge Suspend/Resume 0 1 = Falling edge Suspend/Resume 1 0 = Rising edge Suspend, falling edge Resume 1 1 = Rising edge Resume, falling edge Suspend
1–0	EN_SUS_RES[1–0]	**Enable SUS_RES Pin to Cause Suspend/Resume Function** 0 0 = Disabled 0 1 = Enabled to cause Suspend 1 0 = Enabled to cause Resume 1 1 = Enabled to cause both Suspend and Resume

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved	MKYPRS_ WAKE	SUS_TIMR_ WAKE	SIN_WAKE	Reserved	RIN_WAKE	RTC_WAKE	Reserved
Default	x	0	0	0	0	0	0	x
R/W		R/W	R/W	R/W	R/W	R/W	R/W	

Bit	Name	Function
7	Reserved	**Reserved** During read/modify/write operations, software must preserve this bit.
6	MKYPRS_WAKE	**Matrix Keyboard Key Pressed Wake-Up Control** 0 = Do not wake up system 1 = Wake up system If this option is enabled, and the system is awakened due to a keypress, no further keypressed wake up will be possible until all matrix keys are released, and a key is again pressed, even after the keypressed wake up status bit has been cleared.
5	SUS_TIMR_WAKE	**Suspend Mode Timer Time-Out Wake-Up Control** 0 = Do not wake up system 1 = Wake up system
4	SIN_WAKE	**Falling Edge on the Internal UART's Serial Input Pin (SIN) Wake-Up Control** 0 = Do not wake up system 1 = Wake up system
3	Reserved	**Reserved**
2	RIN_WAKE	**Falling Edge on the Internal UART's Ring Indicate Pin ($\overline{\text{RIN}}$) Wake-Up Control** 0 = Do not wake up system 1 = Wake up system
1	RTC_WAKE	**RTC Alarm (IRQ8) Wake-Up Control** 0 = Do not wake up system 1 = Wake up system
0	Reserved	**Reserved** During read/modify/write operations, software must preserve this bit.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	ACIN_WAKE[1–0]		BL2_WAKE[1–0]		BL1_WAKE[1–0]		BL0_WAKE[1–0]	
Default	0	0	0	0	0	0	0	0
R/W	R/W		R/W		R/W		R/w	

Bit	Name	Function
7–6	ACIN_WAKE[1–0]	**ACIN Rising or Falling Edge Wake-Up Control** 0 0 = Do not wake up system 0 1 = Falling edge wake up system 1 0 = Rising edge wake up system 1 1 = Either edge wake up system
5–4	BL2_WAKE[1–0]	**$\overline{\text{BL2}}$ Rising or Falling Edge Wake-Up Control** 0 0 = Do not wake up system 0 1 = Falling edge wake up system 1 0 = Rising edge wake up system 1 1 = Either edge wake up system
3–2	BL1_WAKE[1–0]	**$\overline{\text{BL1}}$ Rising or Falling Edge Wake-Up Control** 0 0 = Do not wake up system 0 1 = Falling edge wake up system 1 0 = Rising edge wake up system 1 1 = Either edge wake up system
1–0	BL0_WAKE[1–0]	**$\overline{\text{BL0}}$ Rising or Falling Edge Wake-Up Control** 0 0 = Do not wake up system 0 1 = Falling edge wake up system 1 0 = Rising edge wake up system 1 1 = Either edge wake up system

Programming Notes

	7	6	5	4	3	2	1	0
Bit	PDRQ1_WAKE	PDRQ0_WAKE	PIRQ5_WAKE	PIRQ4_WAKE	PIRQ3_WAKE	PIRQ2_WAKE	PIRQ1_WAKE	PIRQ0_WAKE
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	PDRQ1_WAKE	**Programmable DMA Request 1 (PDRQ1) Wake-Up Control** 0 = Do not wake up system 1 = Wake up system
6	PDRQ0_WAKE	**Programmable DMA Request 0 (PDRQ0) Wake-Up Control** 0 = Do not wake up system 1 = Wake up system
5	PIRQ5_WAKE	**Programmable Interrupt Request 5 (PIRQ5) Wake-Up Control** 0 = Do not wake up system 1 = Wake up system
4	PIRQ4_WAKE	**Programmable Interrupt Request 4 (PIRQ4) Wake-Up Control** 0 = Do not wake up system 1 = Wake up system
3	PIRQ3_WAKE	**Programmable Interrupt Request 3 (PIRQ3) Wake-Up Control** 0 = Do not wake up system 1 = Wake up system
2	PIRQ2_WAKE	**Programmable Interrupt Request 2 (PIRQ2) Wake-Up Control** 0 = Do not wake up system 1 = Wake up system
1	PIRQ1_WAKE	**Programmable Interrupt Request 1 (PIRQ1) Wake-Up Control** 0 = Do not wake up system 1 = Wake up system
0	PIRQ0_WAKE	**Programmable Interrupt Request 0 (PIRQ0) Wake-Up Control** 0 = Do not wake up system 1 = Wake up system

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved	PCMB_SC_ WAKE	PCMA_SC_ WAKE	PCMB_CD_ WAKE	PCMA_CD_ WAKE	PCMB_INT_ WAKE	PCMA_INT_ WAKE	RI_WAKE
Default	x	0	0	0	0	0	0	0
R/W		R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	Reserved	**Reserved** During read/modify/write operations, software must preserve this bit.
6	PCMB_SC_WAKE	**PC Card Socket B Status Change Wake-Up Control** 0 = Do not wake up system 1 = Wake up system
5	PCMA_SC_WAKE	**PC Card Socket A Status Change Wake-Up-up Control** 0 = Do not wake up system 1 = Wake up system
4	PCMB_CD_WAKE	**PC Card Socket B Card Detect Wake-Up Control** 0 = Do not wake up system 1 = Wake up system
3	PCMA_CD_WAKE	**PC Card Socket A Card Detect Wake-Up Control** 0 = Do not wake up system 1 = Wake up system
2	PCMB_INT_WAKE	**PC Card Socket B Interrupt Request Wake-Up Control** 0 = Do not wake up system 1 = Wake up system
1	PCMA_INT_WAKE	**PC Card Socket A Interrupt Request Wake-Up Control** 0 = Do not wake up system 1 = Wake up system
0	RI_WAKE	**PC Card Ring Indicate Wake-Up Control** 0 = Do not wake up system 1 = Wake up system

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved	MKYPRS_ WOKE	SUS_TIMR_ WOKE	SIN_WOKE	Reserved	RIN_WOKE	RTC_WOKE	SUS_RES_ WOKE
Default	x	0	x	0	0	0	0	0
R/W		R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	Reserved	**Reserved** During read/modify/write operations, software must preserve this bit.
6	MKYPRS_WOKE	**Matrix Keyboard Key Pressed Wake-Up Status** Write to ’0b’ to clear wake up status. 0 = Did not wake up system 1 = Awakened system
5	SUS_TIMR_WOKE	**Suspend Mode Timer Time-Out Wake-Up Status** 0 = Did not wake up system 1 = Awakened system
4	SIN_WOKE	**Internal UART Receive Signal Wake-Up Status** Write to ’0b’ to clear. 0 = Did not wake up system 1 = Awakened system
3	Reserved	**Reserved**
2	RIN_WOKE	**Internal UART Ring Indicate Wake-Up Status** Write to ’0b’ to clear. 0 = Did not wake up system 1 = Awakened system
1	RTC_WOKE	**RTC Alarm (IRQ8) Wake-Up Status** Write to ’0b’ to clear. 0 = Did not wake up system 1 = Awakened system
0	SUS_RES_WOKE	**SUS_RES Pin Wake-Up Status** (Write to ’0b’ to clear. 0 = Did not wake up system 1 = Awakened system

Programming Notes

	7	6	5	4	3	2	1	0
Bit	ACIN_WOKE		BL2_WOKE		BL1_WOKE		BL0_WOKE	
Default	0	0	0	0	0	0	0	0
R/W	R/W		R/W		R/W		R/W	

Bit	Name	Function
7–6	ACIN_WOKE	**ACIN Rising or Falling Edge Wake-Up Status** Write to ’00b’ to clear. 0 0 = Did not wake up system 0 1 = Falling edge awakened system 1 0 = Rising edge awakened system 1 1 = Reserved
5–4	BL2_WOKE	**$\overline{\text{BL2}}$ Rising or Falling Edge Wake-Up Status** Write to ’00b’ to clear. 0 0 = Did not wake up system 0 1 = Falling edge awakened system 1 0 = Rising edge awakened system 1 1 = Reserved
3–2	BL1_WOKE	**$\overline{\text{BL1}}$ Rising or Falling Edge Wake-Up Status** Write to ’00b’ to clear. 0 0 = Did not wake up system 0 1 = Falling edge awakened system 1 0 = rising Edge awakened system 1 1 = Reserved
1–0	BL0_WOKE	**$\overline{\text{BL0}}$ Rising or Falling Edge Wake-Up Status** Write to ’00b’ to clear. 0 0 = Did not wake up system 0 1 = Falling edge awakened system 1 0 = Rising edge awakened system 1 1 = Reserved

Programming Notes

	7	6	5	4	3	2	1	0
Bit	PDRQ1_ WOKE	PDRQ0_ WOKE	PIRQ5_ WOKE	PIRQ4_ WOKE	PIRQ3_ WOKE	PIRQ2_ WOKE	PIRQ1_ WOKE	PIRQ0_ WOKE
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	PDRQ1_WOKE	**Programmable DMA Request 1 (PDRQ1) Wake-Up Status** Write to ‘0b to clear. 0 = Did not wake up system 1 = Awakened system
6	PDRQ0_WOKE	**Programmable DMA Request 0 (PDRQ0) Wake-Up Status** Write to ’0b’ to clear. 0 = Did not wake up system 1 = Awakened system
5	PIRQ5_WOKE	**Programmable Interrupt Request 5 (PIRQ5) Wake-Up Status** Write to ’0b’ to clear. 0 = Did not wake up system 1 = Awakened system
4	PIRQ4_WOKE	**Programmable Interrupt Request 4 (PIRQ4) Wake-Up Status** Write to ’0b’ to clear. 0 = Did not wake up system 1 = Awakened system
3	PIRQ3_WOKE	**Programmable Interrupt Request 3 (PIRQ3) Wake-Up Status** Write to ’0b’ to clear. 0 = Did not wake up system 1 = Awakened system
2	PIRQ2_WOKE	**Programmable Interrupt Request 2 (PIRQ2) Wake-Up Status** Write to ’0b’ to clear. 0 = Did not wake up system 1 = Awakened system
1	PIRQ1_WOKE	**Programmable Interrupt Request 1 (PIRQ1) Wake-Up Status** Write to ’0b’ to clear. 0 = Did not wake up system 1 = Awakened system
0	PIRQ0_WOKE	**Programmable Interrupt Request 0 (PIRQ0) Wake-Up Status** Write to ’0b’ to clear. 0 = Did not wake up system 1 = Awakened system

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved	PCMB_SC_WOKE	PCMA_SC_WOKE	PCMB_CD_WOKE	PCMA_CD_WOKE	PCMB_INT_WOKE	PCMA_INT_WOKE	RI_WOKE
Default	x	0	0	0	0	0	0	0
R/W		R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	Reserved	**Reserved** During read/modify/write operations, software must preserve this bit.
6	PCMB_SC_WOKE	**PC Card Socket B Status Change Wake-Up Status** Write to ’0b’ to clear. 0 = Did not wake up system 1 = Awakened system
5	PCMA_SC_WOKE	**PC Card Socket A Status Change Wake-Up Status** Write to ’0b’ to clear. 0 = Did not wake up system 1 = Awakened system
4	PCMB_CD_WOKE	**PC Card Socket B Card Detect Wake-Up Status** Write to ’0b’ to clear. 0 = Did not wake up system 1 = Awakened system
3	PCMA_CD_WOKE	**PC Card Socket A Card Detect Wake-Up Status** Write to ’0b’ to clear. 0 = Did not wake up system 1 = Awakened system
2	PCMB_INT_WOKE	**PC Card Socket B Interrupt Request Wake-Up Status** Write to ’0b’ to clear. 0 = Did not wake up system 1 = Awakened system
1	PCMA_INT_WOKE	**PC Card Socket A Interrupt Request Wake-Up Status** Write to ’0b’ to clear. 0 = Did not wake up system 1 = Awakened system
0	RI_WOKE	**PC Card Ring Indicate Wake-Up Status** Write to ’0b’ to clear. 0 = Did not wake up system 1 = Awakened system

Programming Notes

	7	6	5	4	3	2	1	0
Bit	GP7_WOKE	GP6_WOKE	GP5_WOKE	GP4_WOKE	GP3_WOKE	GP2_WOKE	GP1_WOKE	GP0_WOKE
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	GP7_WOKE	**GPIO_CS7 Awakened System or Caused Activity** Write to '0b' to clear. 0 = Did not wake up system or cause activity 1 = Awakened system or caused activity
6	GP6_WOKE	**GPIO_CS6 Awakened System or Caused Activity** Write to '0b' to clear. 0 = Did not wake up system or cause activity 1 = Awakened system or caused activity
5	GP5_WOKE	**GPIO_CS5 Awakened System or Caused Activity** Write to '0b' to clear. 0 = Did not wake up system or cause activity 1 = Awakened system or caused activity
4	GP4_WOKE	**GPIO_CS4 Awakened System or Caused Activity** Write to '0b' to clear. 0 = Did not wake up system or cause activity 1 = Awakened system or caused activity
3	GP3_WOKE	**GPIO_CS3 Awakened System or Caused Activity** Write to '0b' to clear. 0 = Did not wake up system or cause activity 1 = Awakened system or caused activity
2	GP2_WOKE	**GPIO_CS2 Awakened System or Caused Activity** Write to '0b' to clear. 0 = Did not wake up system or cause activity 1 = Awakened system or caused activity
1	GP1_WOKE	**GPIO_CS1 Awakened System or Caused Activity** Write to '0b' to clear. 0 = Did not wake up system or cause activity 1 = Awakened system or caused activity
0	GP0_WOKE	**GPIO_CS0 Awakened System or Caused Activity** Write to '0b' to clear. 0 = Did not wake up system or cause activity 1 = Awakened system or caused activity

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved	GP14_WOKE	GP13_WOKE	GP12_WOKE	GP11_WOKE	GP10_WOKE	GP9_WOKE	GP8_WOKE
Default	x	0	0	0	0	0	0	0
R/W		R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	Reserved	**Reserved** During read/modify/write operations, software must preserve this bit.
6	GP14_WOKE	**GPIO_CS14 Awakened System or Caused Activity** Write to '0b' to clear. 0 = Did not wake up system or cause activity 1 = Awakened system or caused activity
5	GP13_WOKE	**GPIO_CS13 Awakened System or Caused Activity** Write to '0b' to clear. 0 = Did not wake up system or cause activity 1 = Awakened system or caused activity
4	GP12_WOKE	**GPIO_CS12 Awakened System or Caused Activity** Write to '0b' to clear. 0 = Did not wake up system or cause activity 1 = Awakened system or caused activity
3	GP11_WOKE	**GPIO_CS11 Awakened System or Caused Activity** Write to '0b' to clear. 0 = Did not wake up system or cause activity 1 = Awakened system or caused activity
2	GP10_WOKE	**GPIO_CS10 Awakened System or Caused Activity** Write to '0b' to clear. 0 = Did not wake up system or cause activity 1 = Awakened system or caused activity
1	GP9_WOKE	**GPIO_CS9 Awakened System or Caused Activity** Write to '0b' to clear. 0 = Did not wake up system or cause activity 1 = Awakened system or caused activity
0	GP8_WOKE	**GPIO_CS8 Awakened System or Caused Activity** Write to '0b' to clear. 0 = Did not wake up system or cause activity 1 = Awakened system or caused activity

Programming Notes

	7	6	5	4	3	2	1	0
Bit	CSD_PRI_ ACT	CSC_PRI_ ACT	CSB_PRI_ ACT	CSA_PRI_ ACT	CSD_IS_ACT	CSC_IS_ACT	CSB_IS_ACT	CSA_IS_ACT
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	CSD_PRI_ACT	**GP_CSD Activity Level** 0 = Secondary activity 1 = Primary activity
6	CSC_PRI_ACT	**GP_CSC Activity Level** 0 = Secondary activity 1 = Primary activity
5	CSB_PRI_ACT	**GP_CSB Activity Level** 0 = Secondary activity 1 = Primary activity
4	CSA_PRI_ACT	**GP_CSA Activity Level** 0 = Secondary activity 1 = Primary activity
3	CSD_IS_ACT	**GP_CSD Activity Enable** 0 = Not activity 1 = Cause activity
2	CSC_IS_ACT	**GP_CSC Activity Enable** 0 = Not activity 1 = Cause activity
1	CSB_IS_ACT	**GP_CSB Activity Enable** 0 = Not activity 1 = Cause activity
0	CSA_IS_ACT	**GP_CSA Activity Enable** 0 = Not activity 1 = Cause activity

Programming Notes

GP_CSA–GP_CSD are logical concepts with two main functional capabilities that are separately configurable, and in no way dependent upon each other. The GP_CSA–GP_CSD features allow CPU accesses to user defined address ranges to:

1. Serve as stimulus to the PMU. GP_CS hits can result in the generation of a PMU primary or secondary activity, or they can cause the PMU to generate an SMI.
2. Cause a chip select to be driven off chip. The actual pin that the chip select signal will appear on is user mappable by programming index registers B2h and B3h. GP_CSA and GP_CSB refer primarily to I/O accesses while GP_CSC and GP_CSD refer primarily to memory accesses.

	7	6	5	4	3	2	1	0
Bit	Reserved				CSD_WAS_ ACT	CSC_WAS_ ACT	CSB_WAS_ ACT	CSA_WAS_ ACT
Default	x	x	x	x	0	0	0	0
R/W					R	R	R/W	R/W

Bit	Name	Function
7–4	Reserved	**Reserved** During read/modify/write operations, software must preserve these bits.
3	CSD_WAS_ACT	**GP_CSD Activity State** Write to ’0b’ to clear. 0 = Not detected as activity 1 = Detected as activity
2	CSC_WAS_ACT	**GP_CSC Activity Status** Write to ’0b’ to clear. 0 = Not detected as activity 1 = Detected as activity
1	CSB_WAS_ACT	**GP_CSB Activity Status** Write to ’0b’ to clear. 0 = Not detected as activity 1 = Detected as activity
0	CSA_WAS_ACT	**GP_CSA Activity Status** Write to ’0b’ to clear. 0 = Not detected as activity 1 = Detected as activity

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved	VL_IS_ACT	CS1-2_IS_ACT	CS0_IS_ACT	VID_RAM_IS_ACT	VID_IO_IS_ACT	UART2_IO_IS_ACT	UART1_IO_IS_ACT
Default	x	0	0	0	0	0	0	0
R/W		R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	Reserved	**Reserved** During read/modify/write operations, software must preserve this bit.
6	VL_IS_ACT	**Any VL Bus Cycle (memory and I/O) Will Be Activity** 0 = Not an activity 1 = Will be activity
5	CS1-2_IS_ACT	**CPU Access to $\overline{\text{ROMCS2}}$–$\overline{\text{ROMCS1}}$ Will Be Activity** 0 = Not an activity 1 = Will be activity
4	CS0_IS_ACT	**CPU Access to $\overline{\text{ROMCS0}}$ Will Be Activity** 0 = Not an activity 1 = Will be activity
3	VID_RAM_IS_ACT	**CPU Access to Internal Graphics Memory Will Be Activity** 0 = Not an activity 1 = Will be activity
2	VID_IO_IS_ACT	**CPU Access to Internal Graphics I/O Will Be Activity** 0 = Not an activity 1 = Will be activity
1	UART2_IO_IS_ACT	**CPU Access to UART at COM2 Will Be Activity** 0 = Not an activity 1 = Will be activity
0	UART1_IO_IS_ACT	**CPU Access to UART at COM1 Will Be Activity** 0 = Not an activity 1 = Will be activity

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved		KYIO_IS_ACT	KYTIM_IS_ACT	TICK_IS_ACT	IRQ_IS_ACT	DRAM_IS_ACT	MKYPRS_IS_ACT
Default	x	x	0	0	0	0	0	0
R/W			R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7–6	Reserved	**Reserved** During read/modify/write operations, software must preserve these bits.
5	KYIO_IS_ACT	**CPU Access to Keyboard Registers (60h and 64h) Will Be Activity** 0 = Not an activity 1 = Will be activity
4	KYTIM_IS_ACT	**Keyboard Timer Time-Out Will Be Activity** 0 = Not an activity 1 = Will be activity
3	TICK_IS_ACT	**Timer Tick Interrupt Active (IRQ0) Will Be Activity** 0 = Not an activity 1 = Will be activity
2	IRQ_IS_ACT	**Interrupt Active (Not Timer Tick) Will Be Activity** 0 = Not an activity 1 = Will be activity
1	DRAM_IS_ACT	**CPU Access to DRAM (Non-Graphics Section Access) Will Be Activity** 0 = Not an activity 1 = Will be activity
0	MKYPRS_IS_ACT	**Matrix Keyboard Key Pressed Will Be Activity** 0 = Not an activity 1 = Will be activity If this option is enabled, and PMU activity is generated due to a key press, no further keypressed PMU activity will be possible until all matrix keys are released, and a key is again pressed, even after the keypressed activity status bit has been cleared.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	ACIN_IS_ACT	DRQ_IS_ACT	IDEIO_IS_ACT	FDDIO_IS_ACT	XVGARAM_IS_ACT	XVGAIO_IS_ACT	SIN_IS_ACT	RIN_IS_ACT
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

\

Bit	Name	Function
7	ACIN_IS_ACT	**ACIN Signal Will Be Activity** 0 = Not an activity 1 = Will be activity
6	DRQ_IS_ACT	**DMA Request Will Be Activity** 0 = Not an activity 1 = Will be activity
5	IDEIO_IS_ACT	**CPU Access to IDE Hard Drive Registers Will Be Activity** 0 = Not an activity 1 = Will be activity
4	FDDIO_IS_ACT	**CPU Access to Floppy Controller Registers Will Be Activity** 0 = Not an activity 1 = Will be activity
3	XVGARAM_IS_ACT	**CPU Access to External VGA Controller Memory Will Be Activity** 0 = Not an activity 1 = Will be activity
2	XVGAIO_IS_ACT	**CPU Access to External VGA Controller I/O Will Be Activity** 0 = Not an activity 1 = Will be activity
1	SIN_IS_ACT	**A Falling Edge on the Internal UART's Serial Input Pin (SIN) Will Be Activity** 0 = Not an activity 1 = Will be activity
0	RIN_IS_ACT	**A Falling Edge on the Internal UART's Ring Indicate Pin ($\overline{\text{RIN}}$) Will Be Activity** 0 = Not an activity 1 = Will be activity

Programming Notes

	7	6	5	4	3	2	1	0
Bit	INTREG_ WAS_ACT	PCINTR_IS_ ACT	PCMRI_IS_ ACT	PCMB_IO_ IS_ACT	PCMB_RAM_ IS_ACT	PCMA_IO_ IS_ACT	PCMA_RAM_ IS_ACT	PP_IO_IS_ ACT
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	INTREG_WAS_ACT	**CPU Access to Internal System Registers Will Be Activity** 0 = Not an activity 1 = Will be activity
6	PCINTR_IS_ACT	**PMU's PC Card Status Change Interrupt Will Be Activity** 0 = Not an activity 1 = Will be activity
5	PCMRI_IS_ACT	**PC Card Ring Indicate Will Be Activity** 0 = Not an activity 1 = Will be activity
4	PCMB_IO_IS_ACT	**CPU I/O Access to PC Card Socket B Will Be Activity** 0 = Not an activity 1 = Will be activity
3	PCMB_RAM_IS_ACT	**CPU Memory Access to PC Card Socket B Will Be Activity** 0 = Not an activity 1 = Will be activity
2	PCMA_IO_IS_ACT	**CPU I/O Access to PC Card Socket A Will Be Activity** 0 = Not an activity 1 = Will be activity
1	PCMA_RAM_IS_ACT	**CPU Memory Access to PC Card Socket A Will Be Activity** 0 = Not an activity 1 = Will be activity
0	PP_IO_IS_ACT	**CPU Access to Parallel Port Will Be Activity** 0 = Not an activity 1 = Will be activity

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved	VL_WAS_ACT	2_WAS_ACT	CS0_WAS_ ACT	VID_RAM_ WAS_ACT	VID_IO_ WAS_ACT	UART2_IO_ WAS_ACT	UART1_IO_ WAS_ACT
Default	x	0	0	0	0	0	0	0
R/W		R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	Reserved	**Reserved** During read/modify/write operations, software must preserve this bit.
6	VL_WAS_ACT	**Any VL Bus Cycle (Memory and I/O) Activity Status** Write to '0b' to clear. 0 = Not detected as activity 1 = Detected as activity
5	CS1-2_WAS_ACT	**CPU Access to $\overline{\text{ROMCS2}}$–$\overline{\text{ROMCS1}}$ Activity Status** Write to '0b' to clear. 0 = Not detected as activity 1 = Detected as activity
4	CS0_WAS_ACT	**CPU Access to $\overline{\text{ROMCS0}}$ Activity Status** Write to '0b' to clear. 0 = Not detected as activity 1 = Detected as activity
3	VID_RAM_WAS_ACT	**CPU Access to Internal Graphics Memory Activity Status** Write to '0b' to clear. 0 = Not detected as activity 1 = Detected as activity
2	VID_IO_WAS_ACT	**CPU Access to Internal Graphics I/O Activity Status** Write to '0b' to clear. 0 = Not detected as activity 1 = Detected as activity
1	UART2_IO_WAS_ACT	**CPU Access to UART at COM2 Activity Status** Write to '0b' to clear. 0 = Not detected as activity 1 = Detected as activity
0	UART1_IO_WAS_ACT	**CPU Access to UART at COM1 Activity Status** Write to '0b' to clear. 0 = Not detected as activity 1 = Detected as activity

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved		KYIO_WAS_ACT	KYTIM_WAS_ACT	TICK_WAS_ACT	IRQ_WAS_ACT	DRAM_WAS_ACT	MKYPRS_WAS_ACT
Default	x	x	0	0	0	0	0	0
			R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7–6	Reserved	**Reserved** During read/modify/write operations, software must preserve these bits.
5	KYIO_WAS_ACT	**CPU Access to Keyboard Registers Active (60h and 64h) Activity Status** Write to ’0b’ to clear. 0 = Not detected as activity 1 = Detected as activity
4	KYTIM_WAS_ACT	**Keyboard Timer Time-Out Active Activity Status** Write to ’0b’ to clear. 0 = Not detected as activity 1 = Detected as activity
3	TICK_WAS_ACT	**Timer Tick Interrupt Active (IRQ0) Activity Status** Write to ’0b’ to clear. 0 = Not detected as activity 1 = Detected as activity
2	IRQ_WAS_ACT	**Interrupt Active (Not Timer Tick) Activity Status** Write to ’0b’ to clear. 0 = Not detected as activity 1 = Detected as activity
1	DRAM_WAS_ACT	**CPU Access to DRAM (Non-Graphics Section Access) Activity Status** Write to ’0b’ to clear. 0 = Not detected as activity 1 = Detected as activity
0	MKYPRS_WAS_ACT	**Matrix Keyboard Key Pressed Activity Status** Write to ’0b’ to clear. 0 = Not detected as activity 1 = Detected as activity

Programming Notes

	7	6	5	4	3	2	1	0
Bit	ACIN_WAS_ ACT	DRQ_WAS_ ACT	IDEIO_WAS_ ACT	FDDIO_ WAS_ACT	XVGARAM_ WAS_ACT	XVGAIO_ WAS_ACT	SIN_WAS_ ACT	RIN_WAS_ ACT
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	ACIN_WAS_ACT	**ACIN Signal Activity Status** Write to '0b' to clear. 0 = Not detected as activity 1 = Detected as activity
6	DRQ_WAS_ACT	**DMA Request Activity Status** Write to '0b' to clear. 0 = Not detected as activity 1 = Detected as activity
5	IDEIO_WAS_ACT	**CPU Access to IDE Hard Drive Registers Activity Status** Write to '0b' to clear. 0 = Not detected as activity 1 = Detected as activity
4	FDDIO_WAS_ACT	**CPU Access to Floppy Controller Registers Activity Status** Write to '0b' to clear. 0 = Not detected as activity 1 = Detected as activity
3	XVGARAM_WAS_ACT	**CPU Access to External VGA Controller Memory Activity Status** Write to '0b' to clear. 0 = Not detected as activity 1 = Detected as activity
2	XVGAIO_WAS_ACT	**CPU Access to External VGA Controller I/O Activity Status** Write to '0b' to clear. 0 = Not detected as activity 1 = Detected as activity
1	SIN_WAS_ACT	**Internal UART Receive Toggle Activity Status** Write to '0b' to clear. 0 = Not detected as activity 1 = Detected as activity
0	RIN_WAS_ACT	**Internal UART Ring Indicate Activity Status** Write to '0b' to clear. 0 = Not detected as activity 1 = Detected as activity

Programming Notes

	7	6	5	4	3	2	1	0
Bit	INTREG_ WAS_ACT	PCMINTR_ WAS_ACT	PCMRI_ WAS _ACT	PCMB_IO_ WAS_ACT	PCMB_RAM_ WAS_ACT	PCMA_IO_ WAS_ACT	PCMA_RAM_ WAS_ACT	PP_IO_ WAS_ACT
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	INTREG_WAS_ACT	**CPU Access to Internal System Registers Activity Status** Write to ’0b’ to clear. 0 = Not detected as activity 1 = Detected as activity
6	PCMINTR_WAS_ACT	**PMU’s PC Card Status Change Interrupt Activity Status** Write to ’0b’ to clear. 0 = Not detected as activity 1 = Detected as activity
5	PCMRI_WAS _ACT	**PC Card Ring Indicate Activity Status** Write to ’0b’ to clear. 0 = Not detected as activity 1 = Detected as activity
4	PCMB_IO_WAS_ACT	**CPU I/O Access to PC Card Socket B Activity Status** Write to ’0b’ to clear. 0 = Not detected as activity 1 = Detected as activity
3	PCMB_RAM_WAS_ACT	**CPU Memory Access to PC Card Socket B Activity Status** Write to ’0b’ to clear. 0 = Not detected as activity 1 = Detected as activity
2	PCMA_IO_WAS_ACT	**CPU I/O Access to PC Card Socket A Activity Status** Write to ’0b’ to clear. 0 = Not detected as activity 1 = Detected as activity
1	PCMA_RAM_WAS_ACT	**CPU Memory Access to PC Card Socket A Activity Status** Write to ’0b’ to clear. 0 = Not detected as activity 1 = Detected as activity
0	PP_IO_WAS_ACT	**CPU Access to Parallel Port Activity Status** Write to ’0b’ to clear. 0 = Not detected as activity 1 = Detected as activity

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved	VL_SEC_ACT	CS1-2_ SEC_ACT	CS0_SEC_ ACT	VID_RAM_ SEC_ACT	VID_IO_ SEC_ACT	UART2_IO_ SEC_ACT	UART1_IO_ SEC_ACT
Default	x	0	0	0	0	0	0	0
R/W		R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	Reserved	**Reserved** During read/modify/write operations, software must preserve these bits.
6	VL_SEC_ACT	**Any VL Bus Cycle (Memory and I/O) Activity Class** 0 = Primary activity 1 = Secondary activity
5	CS1-2_SEC_ACT	**CPU Access to $\overline{\text{ROMCS2}}$–$\overline{\text{ROMCS1}}$ Activity Class** 0 = Primary activity 1 = Secondary activity
4	CS0_SEC_ACT	**CPU Access to $\overline{\text{ROMCS0}}$ Activity Class** 0 = Primary activity 1 = Secondary activity
3	VID_RAM_SEC_ACT	**CPU Access to Internal Graphics Memory Activity Class** 0 = Primary activity 1 = Secondary activity
2	VID_IO_SEC_ACT	**CPU Access to Internal Graphics I/O Activity Class** 0 = Primary activity 1 = Secondary activity
1	UART2_IO_SEC_ACT	**CPU Access to UART at COM 2 Activity Class** 0 = Primary activity 1 = Secondary activity
0	UART1_IO_SEC_ACT	**CPU Access to UART at COM1 Activity Class** 0 = Primary activity 1 = Secondary activity

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved		KYIO_SEC_ ACT	KYTIM_SEC_ ACT	IRQ0_SEC_ ACT	IRQ_SEC_ ACT	DRAM_SEC_ ACT	MKYPRS_ SEC_ACT
Default	x	x	0	0	0	0	0	0
			R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7–6	Reserved	**Reserved** During read/modify/write operations, software must preserve these bits.
5	KYIO_SEC_ACT	**CPU Access to Keyboard Registers (60h and 64h) Activity Class** 0 = Primary activity 1 = Secondary activity
4	KYTIM_SEC_ACT	**Keyboard Timer Time-Out Activity Class** 0 = Primary activity 1 = Secondary activity
3	IRQ0_SEC_ACT	**Timer Tick Interrupt Active (IRQ0) Activity Class** 0 = Primary activity 1 = Secondary activity
2	IRQ_SEC_ACT	**Interrupt Active (Not Timer Tick) Activity Class** 0 = Primary activity 1 = Secondary activity
1	DRAM_SEC_ACT	**CPU Access to DRAM (Non-Graphics Section Access) Activity Class** 0 = Primary activity 1 = Secondary activity
0	MKYPRS_SEC_ACT	**Matrix Keyboard Key Pressed Activity Class** 0 = Primary activity 1 = Secondary activity

Programming Notes

	7	6	5	4	3	2	1	0
Bit	ACIN_SEC_ ACT	DRQ_SEC_ ACT	IDEIO_SEC_ ACT	FDDIO_ SEC_ACT	XVGARAM_ SEC_ACT	XVGAIO_ SEC_ACT	SIN_SEC_ ACT	RIN_SEC_ ACT
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	ACIN_SEC_ACT	**ACIN Signal Activity Class** 0 = Primary activity 1 = Secondary activity
6	DRQ_SEC_ACT	**DMA Request Activity Class** 0 = Primary activity 1 = Secondary activity
5	IDEIO_SEC_ACT	**CPU Access to IDE Hard Drive Registers Activity Class** 0 = Primary activity 1 = Secondary activity
4	FDDIO_SEC_ACT	**CPU Access to Floppy Controller Registers Activity Class** 0 = Primary activity 1 = Secondary activity
3	XVGARAM_SEC_ACT	**CPU Access to External VGA Controller Memory Activity Class** 0 = Primary activity 1 = Secondary activity
2	XVGAIO_SEC_ACT	**CPU Access to External VGA Controller I/O Activity Class** 0 = Primary activity 1 = Secondary activity
1	SIN_SEC_ACT	**Falling Edge on Internal UART Serial Input Pin Activity Class** 0 = Primary activity 1 = Secondary activity
0	RIN_SEC_ACT	**Falling Edge on Internal UART Ring Indicate Pin Activity Class** 0 = Primary activity 1 = Secondary activity

Programming Notes

	7	6	5	4	3	2	1	0
Bit	INTREG_ SEC_ACT	PCINTR_ SEC_ACT	PCRI_ SEC _ACT	PCMB_IO_ SEC_ACT	PCMB_RAM_ SEC_ACT	PCMA_IO_ SEC_ACT	PCMA_RAM_ SEC_ACT	PP_IO_SEC_ ACT
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	INTREG_SEC_ACT	**CPU Access to Internal System Registers Activity Class** 0 = Primary activity 1 = Secondary activity
6	PCINTR_SEC_ACT	**PMU's PC Card Status Change Interrupt Activity Class** 0 = Primary activity 1 = Secondary activity
5	PCRI_SEC _ACT	**PC Card Ring Indicate Activity Class** 0 = Primary activity 1 = Secondary activity
4	PCMB_IO_SEC_ACT	**CPU I/O Access to PC Card Socket B Activity Class** 0 = Primary activity 1 = Secondary activity
3	PCMB_RAM_SEC_ACT	**CPU Memory Access to PC Card Socket A Activity Class** 0 = Primary activity 1 = Secondary activity
2	PCMA_IO_SEC_ACT	**CPU I/O Access to PC Card Socket B Activity Class** 0 = Primary activity 1 = Secondary activity
1	PCMA_RAM_SEC_ACT	**CPU Memory Access to PC Card Socket A Activity Class** 0 = Primary activity 1 = Secondary activity
0	PP_IO_SEC_ACT	**CPU Access to Parallel Port Activity Class** 0 = Primary activity 1 = Secondary activity

Programming Notes

	7	6	5	4	3	2	1	0
Bit	WAS_ SUSPEND	WAS_CRIT_ SUS	SW_ACIN	BL2_CRIT_ SUS	BL1_CFG[1–0]		BL0_CFG[1–0]	
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W		R/W	

Bit	Name	R/W	Function
7	WAS_SUSPEND	R/W	**Suspend Status** System has been in Suspend indication. Cleared by writing a 0. 0 = The system has not been in Suspend mode 1 = The system has been in Suspend mode since the last time this bit was cleared
6	WAS_CRIT_SUS	R/W	**Resumed from Critical Suspend Indication** Cleared by writing a 0. 0 = The system was not forced to Critical Suspend mode by $\overline{BL2}$ 1 = The system has been forced to Critical Suspend mode by $\overline{BL2}$ since the last time this bit was cleared
5	SW_ACIN	R/W	**Software ACIN** 0 = Do not force software ACIN 1 = Force software ACIN
4	BL2_CRIT_SUS	R/W	**$\overline{BL2}$ Forces Critical Suspend Mode** $\overline{BL2}$ pin forces PMU Mode to Critical Suspend when $\overline{BL2}$ is low 0 = Disabled, does not force mode change 1 = Force PMU to Critical Suspend mode when $\overline{BL2}$ is low If $\overline{BL2}$ is configured to generate an XMI by setting CSC index 93h[5–4] to either 01b or 11b, and $\overline{BL2}$ is also configured to go in to Critical Suspend mode when it is low by setting this bit, then Critical Suspend mode will be entered immediately if $\overline{BL2}$ occurs. Upon exit from Critical Suspend mode, the XMI will automatically occur. Since $\overline{LBL2}$ is really a Critical Suspend mode indication, it will never go active if this bit is cleared.

Bit	Name	R/W	Function
3–2	BL1_CFG[1–0]	R/W	**$\overline{\text{BL1}}$ Configuration** $\overline{\text{BL1}}$ pin disables Hyper-Speed mode and forces High-Speed clock to 8.29MHz when $\overline{\text{BL1}}$ is active or $\overline{\text{BL1}}$ pin forces PMU to disable High-Speed mode when active. If these bits = '01b' or '10b' to limit the PMU mode to High speed or High-Speed modes respectively (via CSC index 40h[2–0]) will result in a momentary change to the higher power PMU mode followed by an immediate drop-back to High-Speed or Low-Speed mode respectively. Activities and wake-ups will only go to Low-Speed mode. 0 0 = Disabled, does not force anything 0 1 = Force CPU clock to 8MHz in High-Speed mode 1 0 = Force PMU to Low-Speed as highest mode, disable High-Speed 1 1 = Reserved
1–0	BL0_CFG[1–0]	R/W	**$\overline{\text{BL0}}$ Configuration** $\overline{\text{BL0}}$ pin disables Hyper-Speed mode and forces High-Speed clock to 8.29MHz when $\overline{\text{BL0}}$ is active or $\overline{\text{BL0}}$ pin forces PMU to disable High-Speed mode when active. If these bits = '01b' or '10b' to limit the PMU mode to High speed or High-Speed modes respectively (via CSC index 40h[2–0]) will result in a momentary change to the higher power PMU mode followed by an immediate drop-back to High-Speed or Low-Speed mode respectively. Activities and wake ups will only go to Low-Speed mode. 0 0 = Disabled, does not force anything 0 1 = Force CPU clock to 8 MHz in High-Speed mode 1 0 = Force PMU to Low-Speed as highest mode, disable High-Speed 1 1 =Reserved

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved				NOBL1_CS_ LOCK	NOBL2_CS_ LOCK	ACIN_CS_ LOCK	ACIN_KILL_ PMU
Default	x	x	x	x	0	0	0	0
R/W					R/W	R/W	R/W	R/W

Bit	Name	Function
7–4	Reserved	**Reserved** During read/modify/write operations, software must preserve these bits.
3	NOBL1_CS_LOCK	**$\overline{\text{BLx}}$ Critical Suspend Unlock** Control to force the PMU mode from Critical Suspend to Suspend mode when both $\overline{\text{BL2}}$ and $\overline{\text{BL1}}$ are inactive. 0 = Will change to Suspend mode 1 = Will stay in Critical Suspend mode
2	NOBL2_CS_LOCK	**Inactive $\overline{\text{BL2}}$ Critical Suspend Unlock** Control to allow $\overline{\text{BL2}}$ being inactive to force the PMU mode from Critical Suspend mode to Suspend mode. 0 = Will change to Suspend mode 1 = Will stay in Critical Suspend mode
1	ACIN_CS_LOCK	**ACIN Signal Critical Suspend Unlock** Disallow ACIN transitioning active from unlocking Critical Suspend mode. Assuming that the PMU is in Critical Suspend mode due to the BL0 pin being driven low, and that this bit is cleared, a transition of the ACIN pin from inactive to active will unlock the PMU from Critical Suspend mode, and cause it to transition into Suspend mode. If BL0 is still active when the ACIN pin transitions active, the PMU will return immediately to Critical Susupend mode. If CSC index 71h[0] is set, this bit will have no effect. 0 = Will change to Suspend mode 1 = Will stay in Critical Suspend mode
0	ACIN_KILL_PMU	**ACIN Signal Disable PMU Functions** Control to allow the assertion of ACIN to disable PMU functions 0 = Disabled 1 = Enabled If this bit is set, and either the ACIN pin or the software ACIN bit is set (see CSC index 70h[5]), forcing the PMU mode to any mode other than Suspend mode (forcing is done via CSC index 40h[2–0]) will result in the PMU going to the forced mode, and then immediately jumping back to Hyper-Speed or High-Speed PMU mode, whichever is the highest allowed (via CSC index 40h[6]). Forcing the PMU to Suspend mode when this bit is set will cause the PMU to enter Suspend mode.8

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved			SUS_RES_STATE	ACIN_STATE	BL2_STATE	BL1_STATE	BL0_STATE
Default	x	x	x	0	0	0	0	0
R/W				R	R	R	R	R

Bit	Name	Function
7–5	Reserved	**Reserved** During read/modify/write operations, software must preserve these bits.
4	SUS_RES_STATE	**SUS_RES Pin State**
3	ACIN_STATE	**ACIN Pin State**
2	BL2_STATE	**$\overline{\text{BL2}}$ Pin State**
1	BL1_STATE	**$\overline{\text{BL1}}$ Pin State**
0	BL0_STATE	**$\overline{\text{BL0}}$ Pin State**

Programming Notes

	7	6	5	4	3	2	1	0
Bit	CPUCLK_PRES_SP[2–0]			LS_CPUCLK[1–0]		HS_CPUCLK[1–0]		HYS_CPUCLK
Default	0	1	1	0	0	0	0	
R/W	R			R/W		R/W		R/W

Bit	Name	Function
7–5	CPUCLK_PRES_SP[2–0]	**Present Speed of the CPU Clock** For all bit patterns except ‘110b’, the 1x CPU clock is referenced. For bit pattern ‘110b’, refer to bit 0 of this register to determine whether the 2x (66 MHz) is being used. A write to these bits has no effect. 0 0 0 = 1 MHz 0 0 1 = 2 MHz 0 1 0 = 4 MHz 0 1 1 = 8 MHz 1 0 0 = 16 MHz 1 0 1 = 33 MHz 1 1 0 = Clock-multiplied mode enabled 1 1 1 = Reserved
4–3	LS_CPUCLK[1–0]	**CPU Clock Speed in Low-Speed Mode** 0 0 = 8.29 MHz 0 1 = 4.15 MHz 1 0 = 2.07 MHz 1 1 = 1.04 MHz
2–1	HS_CPUCLK[1–0]	**CPU Clock Speed in High-Speed Mode** 0 0 = 8.29 MHz 0 1 = 16.59 MHz 1 0 = 33.18 MHz 1 1 = Reserved
0	HYS_CPUCLK	**CPU Clock Speed in Hyper-Speed Mode** 0 = Clock doubled (66 MHz) 1 = Clock tripled (99 MHz) The ÉlanSC400 microcontroller may require special packaging to safely support clock-tripled mode. Selection of clock-tripled mode results in much higher heat generation by the device. Selecting clock-tripled mode for an ÉlanSC400 microcontroller device which uses a package that is not specifically approved by AMD for use at clock-tripled speed may result in erratic system operation, loss of data, or damage to the ÉlanSC400 microcontroller device.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved		ACIN_X	AUTO_ SLOWDOWN	SLOW_PERIOD[1–0]		FAST_PERIOD[1–0]	
Default	x	x	0	0	0	0	0	0
R/W			R/W	R/W	R/W		R/W	

Bit	Name	Function
7–6	Reserved	**Reserved** During read/modify/write operations, software must preserve these bits.
5	ACIN_X	**Override ACIN for Auto Slowdown** Enable auto slowdown to occur even if ACIN is enabled and active. This is a thermal protection measure. 0 = Disabled 1 = Enabled, use bits 4–0 to control period
4	AUTO_SLOWDOWN	**Enable Auto Slowdown** The PMU modes or states provide a way to trade off performance for processing power in discrete steps. The auto slowdown feature allows the (average power/average performance) to be fine tuned when the PMU is operating in the highest available PMU mode. This can be either be Hyper-Speed mode or High-Speed mode, depending on whether or not Hyper-Speed mode is disabled. Enabling this feature causes the CPU clock to switch between the fast speed and slow speed operating frequencies as defined by the FAST_PERIOD and SLOW_PERIOD bit fields. 0 = Auto slowdown disabled 1 = Auto slowdown enabled, use bits 3–0 to control period
3–2	SLOW_PERIOD[1–0]	**Slow Clock Duration** When Auto Slowdown is enabled, these bits define how long the CPU clock runs at the slow clock frequency before switching up to run at the fast clock frequency. When the Hyper-Speed PMU mode is enabled, the slow clock frequency will be the High-Speed mode clock frequency. When the Hyper-Speed PMU mode is disabled, the slow clock will equal the Low-Speed mode clock frequency. See the FAST_PERIOD field for further explanation. 0 0 = 0.25 seconds 0 1 = 0.5 seconds 1 0 = 1 seconds 1 1 = 2 seconds

Bit	Name	Function
1–0	FAST_PERIOD[1–0]	**Fast Clock Duration** When Auto Slowdown is enabled, these bits define how long the CPU clock runs at the fast clock frequency before switching down to run at the slow clock frequency. When the Hyper-Speed PMU mode is enabled, the fast clock frequency will be the Hyper-Speed mode clock frequency. When the Hyper-Speed PMU mode is disabled, the fast clock will equal the High-Speed mode clock frequency. See the SLOW_PERIOD field for further explanation. 0 0 = 4 seconds 0 1 = 8 seconds 1 0 = 16 seconds 1 1 = 32 seconds

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved	DMA_CLK[1–0]		STB_HS _PLL_DIS	RD_32KHZ	PLL_DLY[1–0]		SUS_PLL_EN
Default	0	0	1	0	0	0	0	1
R/W	R/W	R/W		R/W	R	R/W		R/W

Bit	Name	Function
7	Reserved	**Reserved**
6–5	DMA_CLK[1–0]	**DMA Controller Clock Frequency Select** 0 0 = DC 0 1 = 4 MHz or CPUCLK, whichever is slower 1 0 = 8 MHz or CPUCLK, whichever is slower 1 1 = 16 MHz or CPUCLK, whichever is slower 8 and 16 MHz operation of the DMA controller is intended for high speed IrDA operation only. This higher speed DMA operation for IrDA transactions may or may not be required depending on overall system latency issues. Timing for ISA and PC Card DMA cycles at DMA controller clock speeds higher than 4 MHz is not specified or guaranteed. Since CPUCLK can run at 2 MHz or 1 MHz, the possible DMA clock speeds are 16, 8, 4, 2, and 1 MHz.
4	STB_HS_PLL_DIS	**Disable the High-Speed PLL in Standby mode** 0 = High-Speed PLL is enabled 1 = High-Speed PLL is disabled in Standby mode
3	RD_32KHZ	**State of the 32 KHz Clock** Read high or low depending on the 32 KHz clock. Writing this bit has no effect.
2–1	PLL_DLY[1–0]	**System PLL Restart Delay Time (not the CPU PLL)** 0 0 = 32 milliseconds 0 1 = 16 milliseconds 1 0 = 8 milliseconds 1 1 = 4 milliseconds
0	SUS_PLL_EN	**All System PLLs enable/disable in Suspend mode (not the CPU PLL)** 0 = PLLs are disabled in Suspend mode 1 = PLLs are enabled in Suspend mode

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved				CLK_IO_ SEL[3–0]			
Default	x	x	x	x	1	0	0	0
R/W					R/W			

Bit	Name	Function
7–4	Reserved	**Reserved** During read/modify/write operations, software must preserve these bits
3–0	CLK_IO_ SEL[3–0]	**CLI_IO Pin Select** When the CLK_IO pin is selected as an output clock source, these bits select the internal clock that will be multiplexed out onto this pin. 0 0 0 0 = UART clock (18.432 MHz) 0 0 0 1 = Reserved 0 0 1 0 = Graphics dot clock (20.736 MHz to 36.864 MHz depending on Graphics controller mode) 0 0 1 1 = CPU clock (33.18 MHz,16.59 MHz, 8.29 MHz, 4.15 MHz, 2.07 MHz, 1.04 MHz) 0 1 0 0 = System clock (8.29 MHz, 4.15 MHz, 2.07 MHz, 1.04 Mhz) 0 1 0 1 = 2x system clock 0 1 1 0 = DMA clock (16.59 MHz, 8.29 MHz, 4.15 MHz, 2.07 MHz, 1.04 MHz) 0 1 1 1 = 2x DMA clock 1 X X X = DC The CLK_IO output signal will be DC whenever the selected clock source is DC. During operation, when the CLK_IO clock source selection is changed, the output clock signal is not guaranteed to switch cleanly.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	FactoryA[7–0]							
Default	–	–	–	–	–	–	–	–
R/W								

Bit	Name	Function
7–0	FactoryA[7–0]	**Reserved**

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved							
Default	–	–	–	–	–	–	–	–
R/W								

Bit	Name	R/W	Function
7–0	Reserved	R/W	**Reserved**

Programming Notes

	7	6	5	4	3	2	1	0
Bit	WAKE_ WIL_XMI	SIN_WILL_XMI	RIN_WILL_XMI	RTC_WILL_XMI	SUS_RES_CFG		SW_NMI	SW_SMI
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W		W	W

Bit	Name	Function
7	WAKE_WILL_XMI	**Wake-Up XMI Enable** 0 = Do not cause XMI 1 = Cause SMI/NMI
6	SIN_WILL_XMI	**Falling Edge on the Internal UART's Serial Input Pin (SIN) XMI Enable** 0 = Do not cause XMI 1 = Cause SMI/NMI
5	RIN_WILL_XMI	**Falling Edge on the Internal UART's Ring Indicate Pin ($\overline{\text{RIN}}$) Enable** 0 = Do not cause XMI 1 = Cause SMI/NMI
4	RTC_WILL_XMI	**RTC Alarm XMI Enable** 0 = Do not cause XMI 1 = Cause SMI/NMI
3	SR_RISE_WILL_XMI	**Rising Edge on SUS_RES Pin XMI Enable** 0 = Do not cause SMI/NMI 1 = Cause SMI/NMI
2	SR_FALL_WILL_XMI	**Rising Edge on SUS_RES Pin XMI Enable** 0 = Do not cause SMI/NMI 1 = Cause SMI/NMI
1	SW_NMI	**Force NMI** 0 = Do not cause NMI 1 = Cause NMI Refer to CSC index 9Dh[1] for a related bit.
0	SW_SMI	**Force SMI** 0 = Do not cause SMI 1 = Cause SMI

Programming Notes

Bits 6–2: If, when an XMI is enabled, the signal that is being used to generate the XMI is already in the state where it would have been after the enabled edge occurred, an XMI will be generated immediately, without need for another edge. For example, if SIN is low, and then SIN is configured to XMI on a falling edge, the SMI will occur immediately as a result of enabling it. To avoid unwanted XMIs, follow these steps:

1. Disable the SMI at its master control.
2. Enable the XMI.
3. Clear the XMI status bit.
4. Re-enable the SMI at its master source (port 70h[7] for NMIs, CSC index 9Dh[0] for SMIs).

Bits 1–0: These bits behave as if they were self-clearing. When generating an SMI via bit 0 of this register, be sure to clear CSC index 94h[0] while inside the SMI handler prior to exiting SMM mode (by executing a resume instruction) or another SMI will be generated as soon as the resume instruction is executed.

	7	6	5	4	3	2	1	0
Bit	KYOBR_ WILL_XMI	KYIBW_ WILL_XMI	KYTIM_ WILL_XMI	MKYPRS_ WILL_XMI	PCM_INT_ WILL_XMI	PCM_RI_ WILL_XMI	PCMB_CD_ WILL_XMI	PCMA_CD_ WILL_XMI
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	KYOBR_WILL_XMI	**Keyboard Output Buffer Read XMI Enable** 0 = Do not cause XMI 1 = Cause SMI/NMI
6	KYIBW_WILL_XMI	**Keyboard Input Buffer Written XMI Enable** 0 = Do not cause XMI 1 = Cause SMI/NMI
5	KYTIM_WILL_XMI	**Keyboard Timer XMI Enable** 0 = Do not cause XMI 1 = Cause SMI/NMI
4	MKYPRS_WILL_XMI	**Matrix Keyboard Key Pressed/XT Keyboard XMI Enable** 0 = Do not cause XMI 1 = Cause SMI/NMI Keyboard row 14 is shared with the SUS_RES pin, and is excluded from causing an XMI by default. See CSC index C0h[3]. If this option is enabled, and an XMI occurs due to a key press, no further key press XMI will be possible until all matrix keys are released and a key is again pressed, even after the keypressed XMI status bit has been cleared. If XT keyboard interface mode is enabled and is configured to generate XMIs (see CSC index C1h[5–4]), the resultant XMI will only be properly cleared once CSC index C1h[1] has been toggled. See CSC index C1h[1–0] for more detail.
3	PCM_INT_WILL_XMI	**PMU's PC Card Status Change Interrupt XMI Enable** 0 = Do not cause XMI 1 = Cause SMI/NMI
2	PCM_RI_WILL_XMI	**PC Card Ring Indicate XMI Enable** 0 = Do not cause XMI 1 = Cause SMI/NMI
1	PCMB_CD_WILL_XMI	**PC Card Socket B Card Detect XMI Enable** 0 = Do not cause XMI 1 = Cause SMI/NMI
0	PCMA_CD_WILL_XMI	**PC Card Socket A Card Detect XMI Enable** 0 = Do not cause XMI 1 = Cause SMI/NMI

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved			SUSTO_ WILL_XMI	SBTO_WILL_ XMI	LOTO_WILL_ XMI	HITO_WILL_ XMI	HYTO_WILL_ XMI
Default	x	x	x	0	0	0	0	0
R/W				R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7–5	Reserved	**Reserved** During read/modify/write operations, software must preserve these bits.
4	SUSTO_WILL_XMI	**Suspend Mode Timer Time-Out XMI Enable** 0 = Do not cause XMI 1 = Cause SMI/NMI
3	SBTO_WILL_XMI	**Standby Mode Timer Time-Out XMI Enable** 0 = Do not cause XMI 1 = Cause SMI/NMI
2	LOTO_WILL_XMI	**Low-Speed Mode Timer Time-Out XMI Enable** 0 = Do not cause XMI 1 = Cause SMI/NMI
1	HITO_WILL_XMI	**High-Speed Mode Timer Time-Out XMI Enable** 0 = Do not cause XMI 1 = Cause SMI/NMI
0	HYTO_WILL_XMI	**Hyper-Speed Mode Timer Time-Out XMI Enable** 0 = Do not cause XMI 1 = Cause SMI/NMI

Programming Notes

	7	6	5	4	3	2	1	0
Bit	ACIN_RISE_WAS_XMI	ACIN_FALL_WAS_XMI	BL2_RISE_WAS_XMI	BL2_FALL_WAS_XMI	BL1_RISE_WAS_XMI	BL1_FALL_WAS_XMI	BL0_RISE_WAS_XMI	BL0_FALL_WAS_XMI
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	ACIN_RISE_WAS_XMI	**ACIN Pin Edge XMI Enable** A Rising Edge on ACIN 0 = Did not cause SMI/NMI 1 = Caused SMI/NMI
6	ACIN_FALL_WAS_XMI	**ACIN Pin Edge XMI Enable** A Falling Edge on ACIN 0 = Did not cause SMI/NMI 1 = Caused SMI/NMI
5	BL2_RISE_WAS_XMI	**$\overline{\text{BL2}}$ Pin Edge XMI Enable** A Rising Edge on $\overline{\text{BL2}}$ 0 = Did not cause SMI/NMI 1 = Caused SMI/NMI
4	BL2_FALL_WAS_XMI	**$\overline{\text{BL2}}$ Pin Edge XMI Enable** A Falling Edge on $\overline{\text{BL2}}$ 0 = Did not cause SMI/NMI 1 = Caused SMI/NMI
3	BL1_RISE_WAS_XMI	**$\overline{\text{BL1}}$ Pin Edge XMI Enable** A Rising Edge on $\overline{\text{BL1}}$ 0 = Did not cause SMI/NMI 1 = Caused SMI/NMI
2	BL1_FALL_WAS_XMI	**$\overline{\text{BL1}}$ Pin Edge XMI Enable** A Falling Edge on $\overline{\text{BL1}}$ 0 = Did not cause SMI/NMI 1 = Caused SMI/NMI
1	BL0_RISE_WAS_XMI	**$\overline{\text{BL0}}$ Pin Edge XMI Enable** A Rising Edge on $\overline{\text{BL0}}$ 0 = Did not cause SMI/NMI 1 = Caused SMI/NMI
0	BL0_FALL_WAS_XMI	**$\overline{\text{BL0}}$ Pin Edge XMI Enable** A Falling Edge on $\overline{\text{BL0}}$ 0 = Did not cause SMI/NMI 1 = Caused SMI/NMI

Programming Notes

For all bit fields in this register, if, when an XMI is enabled, the signal that is being used to generate the XMI (BLx, ACIN) is already in the state where it would have been after the enabled edge occurred, an XMI will be generated immediately, without need for another edge. For example, if BL0 is low,

and then BL0 is configured to XMI on a falling edge, the SMI will occur immediately as a result of enabling it. To avoid unwanted XMIs, follow these steps:

1. Disable the SMI at its master control.
2. Enable the XMI.
3. Clear the XMI status bit.
4. Re-enable the SMI at its master source (port 70h[7] for NMIs, CSC index 9Dh[0] for SMIs).

	7	6	5	4	3	2	1	0
Bit	WAKE_WAS_XMI	SIN_WAS_XMI	RIN_WAS_XMI	RTC_WAS_XMI	SUS_RISE_WAS_XMI	SUS_FALL_WAS_XMI	SW_NMI	SW_SMI
Default	0	0	0	0	0	0	0	0
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	WAKE_WAS_XMI	**Wake-Up XMI Status** Write to '0b' to clear. 0 = Did not cause SMI/NMI 1 = Caused SMI/NMI
6	SIN_WAS_XMI	**UART Receive XMI Status** Write to '0b' to clear. 0 = Did not cause SMI/NMI 1 = Caused SMI/NMI
5	RIN_WAS_XMI	**UART Ring Indicate XMI Status** Write to '0b' to clear. 0 = Did not cause SMI/NMI 1 = Caused SMI/NMI
4	RTC_WAS_XMI	**RTC Alarm (IRQ8) XMI Status** Write to '0b' to clear. 0 = Did not cause SMI/NMI 1 = Caused SMI/NMI
3	SUS_RISE_WAS_XMI	**SUS_RES Pin XMI Status** Rising Edge on SUS_RES caused XMI 0 = Did not cause SMI /NMI 1 = Caused SMI/NMI
2	SUS_FALL_WAS_XMI	**SUS_RES Pin XMI Status** Falling Edge on SUS_RES caused XMI 0 = Did not cause SMI /NMI 1 = Caused SMI/NMI
1	SW_NMI	**Force NMI XMI Status** Write to '0b' to clear. 0 = Did not cause NMI 1 = Caused NMI
0	SW_SMI	**Force SMI XMI Status** Write to '0b' to clear. 0 = Did not cause SMI 1 = Caused SMI

Programming Notes

Although bits 3 and 2 are shown above as separate bits, the XMI that is associated with either event will be cleared only when both of these bits are written to '00b'.

	7	6	5	4	3	2	1	0
Bit	KYOBR_ WAS_XMI	KYIBW_ WAS_XMI	KYTIM_ WAS_XMI	MKYPRS_ WAS_XMI	PCM_INT_ WAS_XMI	PCM_RI_ WAS_XMI	PCMB_CD_ WAS_XMI	PCMA_CD_ WAS_XMI
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	KYOBR_WAS_XMI	**Keyboard Output Buffer Read XMI Status** Write to ’0b’ to clear. 0 = Did not cause SMI/NMI 1 = Caused SMI/NMI
6	KYIBW_WAS_XMI	**Keyboard Input Buffer Written XMI Status** Write to ’0b’ to clear. 0 = Did not cause SMI/NMI 1 = Caused SMI/NMI
5	KYTIM_WAS_XMI	**Keyboard Timer XMI Status** Write to ’0b’ to clear. 0 = Did not cause SMI/NMI 1 = Caused SMI/NMI
4	MKYPRS_WAS_XMI	**Matrix Keyboard Key Pressed/XT Keyboard XMI Status** 0 = Did not cause SMI/NMI 1 = Caused SMI/NMI If XT keyboard interface mode is enabled and is configured to generate XMIs (see CSC index C1h[5–4]), the resultant XMI will only be properly cleared once CSC index C1h[1] has been toggled. See CSC index C1h[1–0] for more details.
3	PCM_INT_WAS_XMI	**PMU’s PC Card Status Change XMI Status** 0 = Did not cause SMI/NMI 1 = Caused SMI/NMI
2	PCM_RI_WAS_XMI	**PC Card Ring Indicate XMI Status** 0 = Did not cause SMI/NMI 1 = Caused SMI/NMI
1	PCMB_CD_WAS_XMI	**PC Card Socket B Card Detect XMI Status** Write to ’0b’ to clear. 0 = Did not cause SMI/NMI 1 = Caused SMI/NMI
0	PCMA_CD_WAS_XMI	**PC Card Socket A Card Detect XMI Status** Write to ’0b’ to clear. 0 = Did not cause SMI/NMI 1 = Caused SMI/NMI

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved		GPIO_WAS_XMI	SUSTO_WAS_XMI	SBTO_WAS_XMI	LOTO_WAS_XMI	HITO_WAS_XMI	HYTO_WAS_XMI
Default	x	x	0	0	0	0	0	0
R/W			R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7–6	Reserved	**Reserved** During read/modify/write operations, software must preserve these bits.
5	GPIO_WAS_XMI	**GPIO_XMI XMI Status** Write to ’0b’ to clear. 0 = Did not cause SMI/NMI 1 = Caused SMI/NMI
4	SUSTO_WAS_XMI	**Suspend Mode Timer Time-Out XMI Status** Write to ’0b’ to clear. 0 = Did not cause SMI/NMI 1 = Caused SMI/NMI
3	SBTO_WAS_XMI	**Stand-by Mode Timer Time-Out XMI Status** Write to ’0b’ to clear. 0 = Did not cause SMI/NMI 1 = Caused SMI/NMI
2	LOTO_WAS_XMI	**Low-Speed Mode Timer Time-Out XMI Status** Write to ’0b’ to clear. 0 = Did not cause SMI/NMI 1 = Caused SMI/NMI
1	HITO_WAS_XMI	**High-Speed Mode Timer Time-Out XMI Status** Write to ’0b’ to clear. 0 = Did not cause SMI/NMI 1 = Caused SMI/NMI
0	HYTO_WAS_XMI	**Hyper-Speed Mode Timer Time-Out XMI Status** Write to ’0b’ to clear. 0 = Did not cause SMI/NMI 1 = Caused SMI/NMI

Programming Notes

	7	6	5	4	3	2	1	0
Bit	ACIN_RISE_ WAS_XMI	ACIN_FALL_ WAS_XMI	BL2_RISE_ WAS_XMI	BL2_FALL_ WAS_XMI	BL1_RISE_ WAS_XMI	BL1_FALL_ WAS_XMI	BL0_RISE_ WAS_XMI	BL0_FALL_ WAS_XMI
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	ACIN_RISE_WAS_XMI	**ACIN Pin Edge XMI Status** Rising Edge on ACIN caused XMI 0 = Did not cause SMI/NMI 1 = Caused SMI/NMI
6	ACIN_FALL_WAS_XMI	**ACIN Pin Edge XMI Status** Falling Edge on ACIN caused XMI 0 = Did not cause SMI/NMI 1 = Caused SMI/NMI
5	BL2_RISE_WAS_XMI	**$\overline{BL2}$ Pin Edge XMI Status** Rising Edge on $\overline{BL2}$ caused XMI 0 = Did not cause SMI/NMI 1 = Caused SMI/NMI
4	BL2_FALL_WAS_XMI	**$\overline{BL2}$ Pin Edge XMI Status** Falling Edge on $\overline{BL2}$ caused XMI 0 = Did not cause SMI/NMI 1 = Caused SMI/NMI
3	BL1_RISE_WAS_XMI	**$\overline{BL1}$ Pin Edge XMI Status** Rising Edge on $\overline{BL1}$ caused XMI 0 = Did not cause SMI/NMI 1 = Caused SMI/NMI
2	BL1_FALL_WAS_XMI	**$\overline{BL1}$ Pin Edge XMI Status** Falling Edge on $\overline{BL1}$ caused XMI 0 = Did not cause SMI/NMI 1 = Caused SMI/NMI
1	BL0_RISE_WAS_XMI	**$\overline{BL0}$ Pin Edge XMI Status** Rising Edge on $\overline{BL0}$ caused XMI 0 = Did not cause SMI/NMI 1 = Caused SMI/NMI
0	BL0_FALL_WAS_XMI	**$\overline{BL0}$ Pin Edge XMI Status** Falling Edge on $\overline{BL0}$ caused XMI 0 = Did not cause SMI/NMI 1 = Caused SMI/NMI

Programming Notes

Although all bits in this register are shown above as separate bits, they must be treated as four pairs of bits when clearing an XMI. The ACIN bits are grouped as a pair and the BLx bits are grouped as three additional pairs. The XMI that is associated with either the rising or falling edge event associated with a pair will be cleared only when both of the bits in the pair are written to ’00b’.

For all bit fields in this register, if, when an XMI is enabled, the signal that's being used to generate the XMI (BLx, ACIN) is already in the state where it would have been after the enabled edge occurred, an XMI will be generated immediately, without requiring another edge. For example, if BL0 is low, and then BL0 is configured to XMI on a falling edge, the XMI will occur immediately as a result of enabling it. To avoid unwanted XMIs, perform the following steps:

1. Disable the XMI at its master control.
2. Enable the XMI.
3. Clear the XMI status bit.
4. Re-enable the XMI at its master control (CSC index 9Dh[2] for NMIs, CSC index 9Dh[0] for SMIs).

	7	6	5	4	3	2	1	0
Bit	RTC_XMI_SEL	UART_XMI_SEL	PCM_INT_XMI_SEL	WAKE_XMI_SEL	KEYINT_XMI_SEL	SUS_RES_XMI_SEL	BLX_ACIN_XMI_SEL	MODTIM_XMI_SEL
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	RTC_XMI_SEL	**RTC Alarm XMI Select** 0 = SMI/NMI interrupts are SMI 1 = SMI/NMI interrupts are NMI
6	UART_XMI_SEL	**Falling Edge on Either UART Ring Indicate/UART Receive XMI Select** 0 = SMI/NMI interrupts are SMI 1 = SMI/NMI interrupts are NMI
5	PCM_INT_XMI_SEL	**PC Card XMI Select** 0 = SMI/NMI interrupts are SMI 1 = SMI/NMI interrupts are NMI
4	WAKE_XMI_SEL	**Wake-Ups XMI Select** 0 = SMI/NMI interrupts are SMI 1 = SMI/NMI interrupts are NMI
3	KEYINT_XMI_SEL	**Internal Keyboard XMI Select** Internally generated keyboard interrupts due to keyboard timer time-out, input buffer written, output buffer read, matrix key pressed, or XT keyboard byte received. Each of these can have individual enables. See the keyboard index registers for more detail. 0 = SMI/NMI interrupts are SMI 1 = SMI/NMI interrupts are NMI
2	SUS_RES_XMI_SEL	**SUS_RES Signal XMI Select** 0 = SMI/NMI interrupts are SMI 1 = SMI/NMI interrupts are NMI
1	BLX_ACIN_XMI_SEL	**Battery Management (BL0, BL1, BL2, and ACIN) XMI Select** 0 = SMI/NMI interrupts are SMI 1 = SMI/NMI interrupts are NMI
0	MODTIM_XMI_SEL	**High-Speed, Low-Speed, Standby, and Suspend Mode Timer XMI Select** 0 = SMI/NMI interrupts are SMI 1 = SMI/NMI interrupts are NMI

Programming Notes

NMI handlers will need to set the NMI_DONE bit at CSC index 9Dh[1] prior to returning control to the interrupted routine.

	7	6	5	4	3	2	1	0
Bit	Reserved			KEYIOTRAP	INTVIDTRAP	PPTRAP	UART2TRAP	UART1TRAP
Default	x	x	x	0	0	0	0	0
R/W				R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7–5	Reserved	**Reserved** During read/modify/write operations, software must preserve these bits.
4	KEYIOTRAP	**Keyboard Access SMI Enable** 0 = Do not cause an SMI due to an access to this I/O device 1 = Cause an SMI due to an access to this I/O device When the XT keyboard interface is enabled, accesses to port 60h can be configured to generate an SMI. When the PC/AT keyboard interface is enabled, accesses to either port 60h or 64h can be configured to generate an SMI.
3	INTVIDTRAP	**Internal Graphics I/O SMI Enable** 0 = Do not cause an SMI due to an access to this I/O device 1 = Cause an SMI due to an access to this I/O device The address range that will generate the SMI is based on the internal graphics mode that is currently enabled.
2	PPTRAP	**LPT1/LPT2 Parallel Port I/O SMI Enable** 0 = Do not cause an SMI due to an access to either LPT1 or LPT2 1 = Cause an SMI due to an access to either LPT1 or LPT2
1	UART2TRAP	**UART COM2 SMI Enable** 0 = Do not cause an SMI due to an I/O access to UART COM2 1 = Cause an SMI due to an I/O access to UART COM2
0	UART1TRAP	**UART COM1 SMI Enable** 0 = Do not cause an SMI due to an I/O access to UART COM1 1 = Cause an SMI due to an I/O access to UART COM1

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved	GPCSB_ TRAP	GPCSA_ TRAP	PCMB_ IOTRAP	PCMA_ IOTRAP	XVGA_ IOTRAP	HDD_IOTRAP	FDD_IOTRAP
Default	x	0	0	0	0	0	0	0
R/W		R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	Reserved	**Reserved** During read/modify/write operations, software must preserve this bit.
6	GPCSB_TRAP	**GP_CSB SMI Enable** 0 = Do not cause an SMI due to an access to this I/O range 1 = Cause an SMI due to an access to this I/O range
5	GPCSA_TRAP	**GP_CSA SMI Enable** 0 = Do not cause an SMI due to an access to this I/O range 1 = Cause an SMI due to an access to this I/O range
4	PCMB_IOTRAP	**PC Card Socket B I/O Access SMI Enable** 0 = Do not cause an SMI due to an access to any enabled I/O windows supported by PC Card Socket B 1 = Cause an SMI due to an access to any enabled I/O windows supported by PC Card Socket B
3	PCMA_IOTRAP	**PC Card Socket A I/O Access SMI Enable** 0 = Do not cause an SMI due to an access to any enabled I/O windows supported by PC Card Socket A 1 = Cause an SMI due to an access to any enabled I/O windows supported by PC Card Socket A
2	XVGA_IOTRAP	**External VGA Graphics I/O Access SMI Enable** 0 = Do not cause an SMI due to an I/O access to an External VGA Graphics controller 1 = Cause an SMI due to an I/O access to an External VGA Graphics controller
1	HDD_IOTRAP	**IDE Hard Drive Access SMI Enable** 0 = Do not cause an SMI due to an I/O access to the primary fixed disk controller 1 = Cause an SMI due to an I/O access to the primary fixed disk controller
0	FDD_IOTRAP	**Floppy Disk Controller Access SMI Enable** 0 = Do not cause an SMI due to an I/O access to the primary floppy disk controller 1 = Cause an SMI due to an I/O access to the primary floppy controller

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved			KEYIOTRAP-PED	INTVIDTRA-PPED	PPTRAPPED	UART2TRAP-PED	UART1TRAP-PED
Default	x	x	x	0	0	0	0	0
R/W				R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7–5	Reserved	**Reserved** During read/modify/write operations, software must preserve these bits.
4	KEYIOTRAPPED	**Keyboard Access SMI Status** Write to ’0b’ to clear. 0 = Did not cause an SMI due to an access to this I/O device 1 = Caused an SMI due to an access to this I/O device
3	INTVIDTRAPPED	**Internal Graphics I/O SMI Status** Write to ‘0b’ to clear. 0 = Did not cause an SMI due to an access to this I/O device 1 = Caused an SMI due to an access to this I/O device The address range that will generate the SMI is based on the internal graphics mode that is currently enabled.
2	PPTRAPPED	**LPT1/LPT2 Parallel Port I/O SMI Status** Write to ‘0b’ to clear. 0 = Did not cause an SMI due to an access to LPT1 or LPT2 1 = Caused an SMI due to an access to LPT1 or LPT2
1	UART2TRAPPED	**UART COM2 SMI Status** Write to ’0b’ to clear. 0 = Did not cause an SMI due to an I/O access to UART COM2 1 = Caused an SMI due to an I/O access to UART COM2
0	UART1TRAPPED	**UART COM1 SMI Status** Write to ’0b’ to clear. 0 = Did not cause an SMI due to an I/O access to UART COM1 1 = Caused an SMI due to an I/O access to UART COM1

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved	GPCSB_ TRAPPED	GPCSA_ TRAPPED	PCMB_ IOTRAPPED	PCMA_ IOTRAPPED	XVGA_ IOTRAPPED	HDD_ IOTRAPPED	FDD_ IOTRAPPED
Default	x	0	0	0	0	0	0	0
R/W		R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	R/W	Function
7	Reserved		**Reserved** During read/modify/write operations, software must preserve this bit.
6	GPCSB_TRAPPED	R/W	**GP_CSB SMI Status** Write to ‘0b’ to clear. 0 =Did not cause an SMI due to an access to this I/O range 1 =Caused an SMI due to an access to this I/O range
5	GPCSA_TRAPPED	R/W	**GP_CSA SMI Status** Write to ‘0b’ to clear. 0 =Did not cause an SMI due to an access to this I/O range 1 =Caused an SMI due to an access to this I/O range
4	PCMB_IOTRAPPED	R/W	**PC Card Socket B I/O Access SMI Status** Write to ‘0b’ to clear. 0 = Did not cause an SMI due to an access to any enabled I/O windows supported by PC Card Socket B 1 = Caused an SMI due to an access to any enabled I/O windows supported by PC Card Socket B
3	PCMA_IOTRAPPED	R/W	**PC Card Socket A I/O Access SMI Status** Write to ‘0b’ to clear. 0 = Did not cause an SMI due to an access to any enabled I/O windows supported by PC Card Socket A 1 = Caused an SMI due to an access to any enabled I/O windows supported by PC Card Socket A
2	XVGA_IOTRAPPED	R/W	**External VGA Graphics I/O Access SMI Status** Write to ‘0b’ to clear. 0 = Did not cause an SMI due to an I/O access to an external VGA graphics controller 1 = Caused an SMI due to an I/O access to an external VGA graphics controller
1	HDD_IOTRAPPED	R/W	**IDE Hard Drive Access SMI Status** Write to ‘0b’ to clear. 0 = Did not cause an SMI due to an I/O access to the primary fixed disk controller 1 = Caused an SMI due to an I/O access to the primary fixed disk controller
0	FDD_IOTRAPPED	R/W	**Floppy Disk Controller Access SMI Status** Write to ‘0b’ to clear. 0 = Did not cause an SMI due to an I/O access to the primary floppy disk controller 1 = Caused an SMI due to an I/O access to the primary floppy disk controller

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved					NMI_GATE	NMI_DONE	SMI_GATE
Default	x	x	x	x	x	x	0	0
R/W							W	R/W

Bit	Name	Function
7–3	Reserved	**Reserved** During read/modify/write operations, software must preserve these bits.
2	NMI_GATE	**Master NMI Enable** 0 = NMI events are gated off from reaching the core 1 = NMI events will propagate to the CPU core This bit is a read/write version of the NMI Gate bit which typically resides at direct-mapped port 70h[7] on a PC/AT Compatible system. It has been moved here to facilitate internal design integration with the ÉlanSC400 microcontroller PMU.
1	NMI_DONE	**NMI Routine Done** Write a ‘1b’ to this bit to indicate to the PMU that the NMI routine is done. This bit is used by, and cleared automatically by, the PMU state machine. After setting NMI_DONE, at least three cycles of the 32 KHz clock (as monitored at CSC index 82h[3]) must transpire before a subsequent pending NMI will be felt at the CPU.
0	SMI_GATE	**Master SMI Enable** 0 = SMI events are gated off from reaching the CPU core 1 = SMI events will propagate to the CPU core This bit will be automatically cleared whenever the internal master reset is asserted. This occurs during a power-on reset and as a result of the CPU executing a shutdown cycle during SMM mode. Clearing this bit does not disable operation of the SMI status bits in CSC indexes 94–97h.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	CS3_PRI	CS3_DIR	CS2_PRI	CS2_DIR	CS1_PRI	CS1_DIR	CSO_PRI	CS0_DIR
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	CS3_PRI	**GPIO_CS3 Signal is a Primary Activity and Wake-up (Falling Edge) Enable** 0 = Not activity or wake up 1 = Cause activity or wake up
6	CS3_DIR	**GPIO_CS3 Signal is an Input/Output** 0 = Input 1 = Output
5	CS2_PRI	**GPIO_CS2 Signal is a Primary Activity and Wake-up (Falling Edge) Enable** 0 = Not activity or wake up 1 = Cause activity or wake up
4	CS2_DIR	**GPIO_CS2 Signal is an Input/Output** 0 = Input 1 = Output
3	CS1_PRI	**GPIO_CS1 Signal is a Primary Activity and Wake-up (Falling Edge) Enable** 0 = Not activity or wake up 1 = Cause activity or wake up
2	CS1_DIR	**GPIO_CS1 Signal is an Input/Output** 0 = Input 1 = Output
1	CSO_PRI	**GPIO_CS0 Signal is a Primary Activity and Wake-up (Falling Edge) Enable** 0 = Not activity or wake up 1 = Cause activity or wake up
0	CS0_DIR	**GPIO_CS0 Signal is an Input/Output** 0 = Input 1 = Output

Programming Notes

	7	6	5	4	3	2	1	0
Bit	CS7_PRI	CS7_DIR	CS6_PRI	CS6_DIR	CS5_PRI	CS5_DIR	CS4_PRI	CS4_DIR
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	CS7_PRI	**GPIO_CS7 Signal is a Primary Activity and Wake-up (Falling Edge) Enable** 0 = Not activity or wake up 1 = Cause activity or wake up
6	CS7_DIR	**GPIO_CS7 Signal is an Input/Output** 0 = Input 1 = Output
5	CS6_PRI	**GPIO_CS6 Signal is a Primary Activity and Wake-up (Falling Edge) Enable** 0 = Not activity or wake up 1 = Cause activity or wake up
4	CS6_DIR	**GPIO_CS6 Signal is an Input/Output** 0 = Input 1 = Output
3	CS5_PRI	**GPIO_CS5 Signal is a Primary Activity and Wake-up (Falling Edge) Enable** 0 = Not activity or wake up 1 = Cause activity or wake up
2	CS5_DIR	**GPIO_CS5 Signal is an Input/Output** 0 = Input 1 = Output
1	CS4_PRI	**GPIO_CS4 Signal is a Primary Activity and Wake-up (Falling Edge) Enable** 0 = Not activity or wake up 1 = Cause activity or wake up
0	CS4_DIR	**GPIO_CS4 Signal is an Input/Output** 0 = Input 1 = Output

Programming Notes

	7	6	5	4	3	2	1	0
Bit	CS11_PRI	CS11_DIR	CS10_PRI	CS10_DIR	CS9_PRI	CS9_DIR	CS8_PRI	CS8_DIR
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	CS11_PRI	**GPIO_CS11 Signal is a Primary Activity and Wake-up (Falling Edge) Enable** 0 = Not activity or wake up 1 = Cause activity or wake up
6	CS11_DIR	**GPIO_CS11 Signal is an Input/Output** 0 = Input 1 = Output
5	CS10_PRI	**GPIO_CS10 Signal is a Primary Activity and Wake-up (Falling Edge) Enable** 0 = Not activity or wake up 1 = Cause activity or wake up
4	CS10_DIR	**GPIO_CS10 Signal is an Input/Output** 0 = Input 1 = Output
3	CS9_PRI	**GPIO_CS9 Signal is a Primary Activity and Wake-up (Falling Edge) Enable** 0 = Not activity or wake up 1 = Cause activity or wake up
2	CS9_DIR	**GPIO_CS9 Signal is an Input/Output** 0 = Input 1 = Output
1	CS8_PRI	**GPIO_CS8 Signal is a Primary Activity and Wake-up (Falling Edge) Enable** 0 = Not activity or wake up 1 = Cause activity or wake up
0	CS8_DIR	**GPIO_CS8 Signal is an Input/Output** 0 = Input 1 = Output

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved	GPIO15_DIR	CS14_PRI	CS14_DIR	CS13_PRI	CS13_DIR	CS12_PRI	CS12_DIR
Default	x	0	0	0	0	0	0	0
R/W		R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	Reserved	**Reserved** During read/modify/write operations, software must preserve this bit.
6	GPIO15_DIR	**GPIO15 Signal is an Input/Output** 0 = Input 1 = Output
5	CS14_PRI	**GPIO_CS14 Signal is a Primary Activity and Wake-up (Falling Edge) Enable** 0 = Not activity or wake up 1 = Cause activity or wake up
4	CS14_DIR	**GPIO_CS14 Signal is an Input/Output** 0 = Input 1 = Output
3	CS13_PRI	**GPIO_CS13 Signal is a Primary Activity and Wake-up (Falling Edge) Enable** 0 = Not activity or wake up 1 = Cause activity or wake up
2	CS13_DIR	**GPIO_CS13 Signal is an Input/Output** 0 = Input 1 = Output
1	CS12_PRI	**GPIO_CS12 Signal is a Primary Activity and Wake-up (Falling Edge) Enable** 0 = Not activity or wake up 1 = Cause activity or wake up
0	CS12_DIR	**GPIO_CS12 Signal is an Input/Output** 0 = Input 1 = Output

Programming Notes

	7	6	5	4	3	2	1	0
Bit	GPIO23_DIR	GPIO22_DIR	GPIO21_DIR	GPIO20_DIR	GPIO19_DIR	GPIO18_DIR	GPIO17_DIR	GPIO16_DIR
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	GPIO23_DIR	**GPIO23 Signal is an Input/Output** 0 = Input 1 = Output
6	GPIO22_DIR	**GPIO22 Signal is an Input/Output** 0 = Input 1 = Output
5	GPIO21_DIR	**GPIO21 Signal is an Input/Output** 0 = Input 1 = Output
4	GPIO20_DIR	**GPIO20 Signal is an Input/Output** 0 = Input 1 = Output
3	GPIO19_DIR	**GPIO19 Signal is an Input/Output** 0 = Input 1 = Output
2	GPIO18_DIR	**GPIO18 Signal is an Input/Output** 0 = Input 1 = Output
1	GPIO17_DIR	**GPIO17 Signal is an Input/Output** 0 = Input 1 = Output
0	GPIO16_DIR	**GPIO16 Signal is an Input/Output** 0 = Input 1 = Output

Programming Notes

	7	6	5	4	3	2	1	0
Bit	GPIO31_DIR	GPIO30_DIR	GPIO29_DIR	GPIO28_DIR	GPIO27_DIR	GPIO26_DIR	GPIO25_DIR	GPIO24_DIR
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	GPIO31_DIR	**GPIO31 Signal is an Input/Output** 0 = Input 1 = Output
6	GPIO30_DIR	**GPIO30 Signal is an Input/Output** 0 = Input 1 = Output
5	GPIO29_DIR	**GPIO29 Signal is an Input/Output** 0 = Input 1 = Output
4	GPIO28_DIR	**GPIO28 Signal is an Input/Output** 0 = Input 1 = Output
3	GPIO27_DIR	**GPIO27 Signal is an Input/Output** 0 = Input 1 = Output
2	GPIO26_DIR	**GPIO26 Signal is an Input/Output** 0 = Input 1 = Output
1	GPIO25_DIR	**GPIO25 Signal is an Input/Output** 0 = Input 1 = Output
0	GPIO24_DIR	**GPIO24 Signal is an Input/Output** 0 = Input 1 = Output

Programming Notes

	7	6	5	4	3	2	1	0
Bit	GP7STAT_CTL	GP6STAT_CTL	GP5STAT_CTL	GP4STAT_CTL	GP3STAT_CTL	GP2STAT_CTL	GP1STAT_CTL	GP0STAT_CTL
Default	x	x	x	x	x	x	x	x
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	GP7STAT_CTL	**GPIO_CS7 Status and Control** Read GPIO_CS7 value and write GPIO_CS7 value when pin = output
6	GP6STAT_CTL	**GPIO_CS6 Status and Control** Read GPIO_CS6 value and write GPIO_CS6 value when pin = output
5	GP5STAT_CTL	**GPIO_CS5 Status and Control** Read GPIO_CS5 value and write GPIO_CS5 value when pin = output
4	GP4STAT_CTL	**GPIO_CS4 Status and Control** Read GPIO_CS4 value and write GPIO_CS4 value when pin = output
3	GP3STAT_CTL	**GPIO_CS3 Status and Control** Read GPIO_CS3 value and write GPIO_CS3 value when pin = output
2	GP2STAT_CTL	**GPIO_CS2 Status and Control** Read GPIO_CS2 value and write GPIO_CS2 value when pin = output
1	GP1STAT_CTL	**GPIO_CS1 Status and Control** Read GPIO_CS1 value and write GPIO_CS1 value when pin = output
0	GP0STAT_CTL	**GPIO_CS0 Status and Control** Read GPIO_CS0 value and write GPIO_CS0 value when pin = output

Programming Notes

At power-on reset, the GPIO pins referenced by this register default to inputs, hence the values read back from this register depend on what is driving them external to the ÉlanSC400 microcontroller. The value read back can also be influenced by the use of the programmable termination. See GPIO Termination Control Register A at CSC index 3Bh for more detail. Also note that if, while its associated pin is configured as an input, any these bits is written to, the value written will be latched. If the pin is then configured as an output, the latched value will be driven to the GPIO_CS pin.

The value read back from this register may not necessarily correspond to what was last written to this register.

	7	6	5	4	3	2	1	0
Bit	GP15STAT_CTL	GP14STAT_CTL	GP13STAT_CTL	GP12STAT_CTL	GP11STAT_CTL	GP10STAT_CTL	GP9STAT_CTL	GP8STAT_CTL
Default	x	x	x	x	x	x	x	x
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	GP15STAT_CTL	**GPIO_CS15 Status and Control** Read GPIO_CS15 value and write GPIO15 value when pin = output
6	GP14STAT_CTL	**GPIO_CS14 Status and Control** Read GPIO_CS14 value and write GPIO_CS14 value when pin = output
5	GP13STAT_CTL	**GPIO_CS13 Status and Control** Read GPIO_CS13 value and write GPIO_CS13 value when pin = output
4	GP12STAT_CTL	**GPIO_CS12 Status and Control** Read GPIO_CS12 value and write GPIO_CS12 value when pin = output
3	GP11STAT_CTL	**GPIO_CS11 Status and Control** Read GPIO_CS11 value and write GPIO_CS11 value when pin = output
2	GP10STAT_CTL	**GPIO_CS10 Status and Control** Read GPIO_CS10 value and write GPIO_CS10 value when pin = output
1	GP9STAT_CTL	**GPIO_CS9 Status and Control** Read GPIO_CS9 value and write GPIO_CS9 value when pin = output
0	GP8STAT_CTL	**GPIO_CS8 Status and Control** Read GPIO_CS8 value and write GPIO_CS8 value when pin = output

Programming Notes

At power-on reset, the GPIO pins referenced by this register default to inputs, hence the values read back from this register depend on what is driving them external to the ÉlanSC400 microcontroller. The value read back can also be influenced by the use of the programmable termination. See GPIO Termination Control Register B at CSC index 3Ch for more detail. Also note that if, while its associated pin is configured as an input, any these bits is written to, the value written will be latched. If the pin is then configured as an output, the latched value will be driven to the GPIO_CS pin.

The value read back from this register may not necessarily correspond to what was last written to this register.

	7	6	5	4	3	2	1	0
Bit	GP23STAT_CTL	GP22STAT_CTL	GP21STAT_CTL	GP20STAT_CTL	GP19STAT_CTL	GP18STAT_CTL	GP17STAT_CTL	GP16STAT_CTL
Default	x	x	x	x	x	x	x	x
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	GP23STAT_CTL	**GPIO23 Status and Control** Read GPIO23 value and write GPIO23 value when pin = output
6	GP22STAT_CTL	**GPIO22 Status and Control** Read GPIO22 value and write GPIO22 value when pin = output
5	GP21STAT_CTL	**GPIO21 Status and Control** Read GPIO21 value and write GPIO21 value when pin = output
4	GP20STAT_CTL	**GPIO20 Status and Control** Read GPIO20 value and write GPIO20 value when pin = output
3	GP19STAT_CTL	**GPIO19 Status and Control** Read GPIO19 value and write GPIO19 value when pin = output
2	GP18STAT_CTL	**GPIO18 Status and Control** Read GPIO18 value and write GPIO18 value when pin = output
1	GP17STAT_CTL	**GPIO17 Status and Control** Read GPIO17 value and write GPIO17 value when pin = output
0	GP16STAT_CTL	**GPIO16 Status and Control** Read GPIO16 value and write GPIO16 value when pin = output

Programming Notes

At power-on reset, the GPIO pins referenced by this register default to inputs, hence the values read back from this register depend on what's driving them external to the ÉlanSC400 microcontroller. The value read back can also be influenced by the use of the programmable termination. See GPIO Termination Control Register C at CSC index 3Dh for more detail. Also note that if, while its associated pin is configured as an input, any these bits is written to, the value written will be latched. If the pin is then configured as an output, the latched value will be driven to the GPIO_CS pin.

	7	6	5	4	3	2	1	0
Bit	GP31STAT_CTL	GP30STAT_CTL	GP29STAT_CTL	GP28STAT_CTL	GP27STAT_CTL	GP26STAT_CTL	GP25STAT_CTL	GP24STAT_CTL
Default	x	x	x	x	x	x	x	x
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	GP31STAT_CTL	**GPIO31 Status and Control** Read GPIO31 value and write GPIO31 value when pin = output
6	GP30STAT_CTL	**GPIO30 Status and Control** Read GPIO30 value and write GPIO30 value when pin = output
5	GP29STAT_CTL	**GPIO29 Status and Control** Read GPIO29 value and write GPIO29 value when pin = output
4	GP28STAT_CTL	**GPIO28 Status and Control** Read GPIO28 value and write GPIO28 value when pin = output
3	GP27STAT_CTL	**GPIO27 Status and Control** Read GPIO27 value and write GPIO27 value when pin = output
2	GP26STAT_CTL	**GPIO26 Status and Control** Read GPIO26 value and write GPIO26 value when pin = output
1	GP25STAT_CTL	**GPIO25 Status and Control** Read GPIO25 value and write GPIO25 value when pin = output
0	GP24STAT_CTL	**GPIO24 Status and Control** Read GPIO24 value and write GPIO24 value when pin = output

Programming Notes

At power-on reset, the GPIO pins referenced by this register default to inputs, hence the values read back from this register depend on what's driving them external to the ÉlanSC400 microcontroller. The value read back can also be influenced by the use of the programmable termination. See GPIO Termination Control Register D at CSC index 3Eh for more detail.

Also note that if, while its associated pin is configured as an input, any these bits is written to, the value written will be latched. If the pin is then configured as an output, the latched value will be driven to the GPIO_CS pin.

	7	6	5	4	3	2	1	0
Bit	Reserved		GPMUA_HY_LEV	GPMUA_HS_LEV	GPMUA_LS_LEV	GPMUA_TLS_LEV	GPMUA_SB_LEV	GPMUA_SU_LEV
Default	x	x	0	0	0	0	0	0
R/W			R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7–6	Reserved	**Reserved** During read/modify/write operations, software must preserve these bits.
5	GPMUA_HY_LEV	**Drive GPIO_PMUA Signal with Programmed Value in Hyper-Speed Mode** 0 = Drive GPIO pin associated with GPIO_PMUA Low when Hyper-Speed mode entered 1 = Drive GPIO pin associated with GPIO_PMUA High when Hyper-Speed mode entered
4	GPMUA_HS_LEV	**Drive GPIO_PMUA Signal with Programmed Value in High-Speed Mode** 0 = Drive GPIO pin associated with GPIO_PMUA Low when High-Speed mode entered 1 = Drive GPIO pin associated with GPIO_PMUA High when High-Speed mode entered
3	GPMUA_LS_LEV	**Drive GPIO_PMUA Signal with Programmed Value in Low-Speed Mode** 0 = Drive GPIO pin associated with GPIO_PMUA Low when Low-Speed mode entered 1 = Drive GPIO pin associated with GPIO_PMUA High when Low-Speed mode entered
2	GPMUA_TLS_LEV	**Drive GPIO_PMUA Signal with Programmed Value in Temporary Low-Speed Mode** 0 = Drive GPIO pin associated with GPIO_PMUA Low when Temporary Low-Speed mode entered 1 = Drive GPIO pin associated with GPIO_PMUA High when Temporary Low-Speed mode entered
1	GPMUA_SB_LEV	**Drive GPIO_PMUA Signal with Programmed Value in Standby Mode** 0 = Drive GPIO pin associated with GPIO_PMUA Low when Standby mode entered 1 = Drive GPIO pin associated with GPIO_PMUA High when Standby mode entered
0	GPMUA_SU_LEV	**Drive GPIO_PMUA Signal with Programmed Value in Suspend or Critical Suspend Mode** 0 = Drive GPIO pin associated with GPIO_PMUA Low when Suspend or Critical Suspend mode entered 1 = Drive GPIO pin associated with GPIO_PMUA High when Suspend or Critical Suspend mode entered

Programming Notes

Setting bit 0 of this register will cause the GPIO pin, which is selected via CSC index AEh[3–0], to go active when the PMU is either in Suspend or Critical Suspend mode. To distinguish between

Suspend and Critical Suspend modes, look at the state of the $\overline{\text{LBL2}}$ pin which, when active, indicates that the PMU is in Critical Suspend PMU mode.

See CSC index AEh[3–0] for GPIO_PMUA to GPIO_CS pin mapping.

	7	6	5	4	3	2	1	0
Bit	Reserved		GPMUB_HY_LEV	GPMUB_HS_LEV	GPMUB_LS_LEV	GPMUB_TLS_LEV	GPMUB_SB_LEV	GPMUB_SU_LEV
Default	x	x	0	0	0	0	0	0
R/W			R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7–6	Reserved	**Reserved** During read/modify/write operations, software must preserve these bits.
5	GPMUB_HY_LEV	**Drive GPIO_PMUB Signal with Programmed Value in Hyper-Speed Mode** 0 = Drive GPIO pin associated with GPIO_PMUB Low when Hyper-Speed mode entered 1 = Drive GPIO pin associated with GPIO_PMUB High when Hyper-Speed mode entered
4	GPMUB_HS_LEV	**Drive GPIO_PMUB Signal with Programmed Value in Hyper-Speed Mode** 0 = Drive GPIO pin associated with GPIO_PMUB Low when High-Speed mode entered 1 = Drive GPIO pin associated with GPIO_PMUB High when High-Speed mode entered
3	GPMUB_LS_LEV	**Drive GPIO_PMUB Signal with Programmed Value in Low-Speed Mode** 0 = Drive GPIO pin associated with GPIO_PMUB Low when Low-Speed mode entered 1 = Drive GPIO pin associated with GPIO_PMUB High when Low-Speed mode entered
2	GPMUB_TLS_LEV	**Drive GPIO_PMUB Signal with Programmed Value in Temporary Low-Speed Mode** 0 = Drive GPIO pin associated with GPIO_PMUB Low when Temporary Low-Speed mode entered 1 = Drive GPIO pin associated with GPIO_PMUB High when Temporary Low-Speed mode entered
1	GPMUB_SB_LEV	**Drive GPIO_PMUB Signal with Programmed Value in Standby Mode** 0 = Drive GPIO pin associated with GPIO_PMUB Low when Standby mode entered 1 = Drive GPIO pin associated with GPIO_PMUB High when Standby mode entered
0	GPMUB_SU_LEV	**Drive GPIO_PMUB Signal with Programmed Value in Suspend or Critical Suspend Mode** 0 = Drive GPIO pin associated with GPIO_PMUB Low when Suspend or Critical Suspend mode entered 1 = Drive GPIO pin associated with GPIO_PMUB High when Suspend or Critical Suspend mode entered

Programming Notes

Setting bit 0 of this register will cause the GPIO pin, which is selected via CSC index AEh[7–4], to go active when the PMU is either in Suspend or Critical Suspend mode. To distinguish between

Suspend and Critical Suspend modes, look at the state of the $\overline{\text{LBL2}}$ pin which, when active, indicates that the PMU is in Critical Suspend PMU mode.

See CSC index AEh[7–4] for GPIO_PMUB to GPIO_CS pin mapping.

	7	6	5	4	3	2	1	0
Bit	Reserved		GPMUC_HY_LEV	GPMUC_HS_LEV	GPMUC_LS_LEV	GPMUC_TLS_LEV	GPMUC_SB_LEV	GPMUC_SU_LEV
Default	x	x	0	0	0	0	0	0
R/W			R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7–6	Reserved	**Reserved** During read/modify/write operations, software must preserve these bits.
5	GPMUC_HY_LEV	**Drive GPIO_PMUC Signal with Programmed Value in Hyper-Speed Mode** 0 = Drive GPIO pin associated with GPIO_PMUC Low when Hyper-Speed mode entered 1 = Drive GPIO pin associated with GPIO_PMUC High when Hyper-Speed mode entered
4	GPMUC_HS_LEV	**Drive GPIO_PMUC Signal with Programmed Value in Hyper-Speed Mode** 0 = Drive GPIO pin associated with GPIO_PMUC Low when High-Speed mode entered 1 = Drive GPIO pin associated with GPIO_PMUC High when High-Speed mode entered
3	GPMUC_LS_LEV	**Drive GPIO_PMUC Signal with Programmed Value in Low-Speed Mode** 0 = Drive GPIO pin associated with GPIO_PMUC Low when Low-Speed mode entered 1 = Drive GPIO pin associated with GPIO_PMUC High when Low-Speed mode entered
2	GPMUC_TLS_LEV	**Drive GPIO_PMUC Signal with Programmed Value in Temporary Low-Speed Mode** 0 = Drive GPIO pin associated with GPIO_PMUC Low when Temporary Low-Speed mode entered 1 = Drive GPIO pin associated with GPIO_PMUC High when Temporary Low-Speed mode entered
1	GPMUC_SB_LEV	**Drive GPIO_PMUC Signal with Programmed Value in Standby Mode** 0 = Drive GPIO pin associated with GPIO_PMUC Low when Standby mode entered 1 = Drive GPIO pin associated with GPIO_PMUC High when Standby mode entered
0	GPMUC_SU_LEV	**Drive GPIO_PMUC Signal with Programmed Value in Suspend or Critical Suspend Mode** 0 = Drive GPIO pin associated with GPIO_PMUC Low when Suspend or Critical Suspend mode entered 1 = Drive GPIO pin associated with GPIO_PMUC High when Suspend or Critical Suspend mode entered

Programming Notes

Setting bit 0 of this register will cause the GPIO pin, which is selected via CSC index AFh[3–0], to go active when the PMU is either in Suspend or Critical Suspend mode. To distinguish between

Suspend and Critical Suspend modes, look at the state of the $\overline{\text{LBL2}}$ pin which, when active, indicates that the PMU is in Critical Suspend PMU mode.

See CSC index AFh[3–0] for GPIO_PMUC to GPIO_CS pin mapping.

	7	6	5	4	3	2	1	0
Bit	Reserved		GPMUD_HY_LEV	GPMUD_HS_LEV	GPMUD_LS_LEV	GPMUD_TLS_LEV	GPMUD_SB_LEV	GPMUD_SU_LEV
Default	x	x	0	0	0	0	0	0
R/W			R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7–6	Reserved	**Reserved** During read/modify/write operations, software must preserve these bits.
5	GPMUD_HY_LEV	**Drive GPIO_PMUD Signal with Programmed Value in Hyper-Speed Mode** 0 = Drive GPIO pin associated with GPIO_PMUD Low when Hyper-Speed mode entered 1 = Drive GPIO pin associated with GPIO_PMUD High when Hyper-Speed mode entered
4	GPMUD_HS_LEV	**Drive GPIO_PMUD Signal with Programmed Value in Hyper-Speed Mode** 0 = Drive GPIO pin associated with GPIO_PMUD Low when High-Speed mode entered 1 = Drive GPIO pin associated with GPIO_PMUD High when High-Speed mode entered
3	GPMUD_LS_LEV	**Drive GPIO_PMUD Signal with Programmed Value in Low-Speed Mode** 0 = Drive GPIO pin associated with GPIO_PMUD Low when Low-Speed mode entered 1 = Drive GPIO pin associated with GPIO_PMUD High when Low-Speed mode entered
2	GPMUD_TLS_LEV	**Drive GPIO_PMUD Signal with Programmed Value in Temporary Low-Speed Mode** 0 = Drive GPIO pin associated with GPIO_PMUD Low when Temporary Low-Speed mode entered 1 = Drive GPIO pin associated with GPIO_PMUD High when Temporary Low-Speed mode entered
1	GPMUD_SB_LEV	**Drive GPIO_PMUD Signal with Programmed Value in Standby Mode** 0 = Drive GPIO pin associated with GPIO_PMUD Low when Standby mode entered 1 = Drive GPIO pin associated with GPIO_PMUD High when Standby mode entered
0	GPMUD_SU_LEV	**Drive GPIO_PMUD Signal with Programmed Value in Suspend or Critical Suspend Mode** 0 = Drive GPIO pin associated with GPIO_PMUD Low when Suspend or Critical Suspend mode entered 1 = Drive GPIO pin associated with GPIO_PMUD High when Suspend or Critical Suspend mode entered

Programming Notes

Setting bit 0 of this register will cause the GPIO pin, which is selected via CSC index AFh[7–4], to go active when the PMU is either in Suspend or Critical Suspend mode. To distinguish between

Suspend and Critical Suspend modes, look at the state of the $\overline{\text{LBL2}}$ pin which, when active, indicates that the PMU is in Critical Suspend PMU mode.

See CSC index AFh[7–4] for GPIO_PMUD to GPIO_CS pin mapping.

	7	6	5	4	3	2	1	0
Bit	GPMUB_MUX[3–0]				GPMUA_MUX[3–0]			
Default	1	1	1	1	1	1	1	1
R/W	R/W				R/W			

Bit	Name	Function
7–4	GPMUB_MUX[3–0]	**Map GPIO_PMUB to One of the GPIO_CS Pins** The number programmed corresponds to the GPIO_CS that is mapped: 0 0 0 0 = GPIO_CS0 0 0 0 1 = GPIO_CS1 . . 1 1 1 0 = GPIO_CS14 1 1 1 1 = Disabled
3–0	GPMUA_MUX[3–0]	**Map GPIO_PMUA to One of the GPIO_CS Pins** The number programmed corresponds to the GPIO_CS that is mapped: 0 0 0 0 = GPIO_CS0 0 0 0 1 = GPIO_CS1 . . 1 1 1 0 = GPIO_CS14 1 1 1 1 = Disabled

Programming Notes

When mapping GPMUA/B to external GPIO_CS pins for the purpose of controlling the GPIO_CS pins based on the PMU mode, you must also configure the desired GPIO_CS pins to be outputs (see CSC indexes A0–A3h). You must also ensure that the bits in index register A6h or A7h that correspond to the GPIO_CS pins you are controlling based on PMU mode have been cleared.

	7	6	5	4	3	2	1	0
Bit	GPMUD_MUX[3–0]				GPMUC_MUX[3–0]			
Default	1	1	1	1	1	1	1	1
R/W	R/W				R/W			

Bit	Name	Function
7–4	GPMUD_MUX[3–0]	**Map GPIO_PMUD to One of the GPIO_CS Pins** The number programmed corresponds to the GPIO_CS that is mapped: 0 0 0 0 = GPIO_CS0 0 0 0 1 = GPIO_CS1 . . 1 1 1 0 = GPIO_CS14 1 1 1 1 = Disabled
3–0	GPMUC_MUX[3–0]	**Map GPIO_PMUC to One of the GPIO_CS Pins** The number programmed corresponds to the GPIO_CS that is mapped: 0 0 0 0 = GPIO_CS0 0 0 0 1 = GPIO_CS1 . . 1 1 1 0 = GPIO_CS14 1 1 1 1 = Disabled

Programming Notes

When mapping GPMUC/D to external GPIO_CS pins for the purpose of controlling the GPIO_CS pins based on the PMU mode, you must also configure the desired GPIO_CS pins to be outputs (see CSC indexes A0–A3h). You must also ensure that the bits in index register A6h or A7h that correspond to the GPIO_CS pins you are controlling based on PMU mode have been cleared.

	7	6	5	4	3	2	1	0
Bit	Reserved			GPIO_XMI_SEL	GPXMI_MUX[3–0]			
Default	x	x	x	0	1	1	1	1
R/W				R/W	R/W			

Bit	Name	Function
7–5	Reserved	**Reserved** During read/modify/write operations, software must preserve these bits.
4	GPIO_XMI_SEL	**GPIO_XMI SMI/NMI Selection** 0 = GPIO_XMI interrupts are SMI 1 = GPIO_XMI interrupts are NMI
3–0	GPXMI_MUX[3–0]	**Map one of the GPIO_CS Pins to the Internal GPIO_XMI Input Signal** This allows logic external to the ÉlanSC400 microcontroller to generate an SMI/NMI by driving a falling edge onto the GPIO_CS pin selected via this register. The number programmed into this register corresponds to the GPIO_CS pin that is to be internally routed to the GPIO_XMI input: 0 0 0 0 = GPIO_CS0 0 0 0 1 = GPIO_CS1 . . 1 1 1 0 = GPIO_CS14 1 1 1 1 = Disabled To avoid generating unwanted XMIs when mapping GPIO_XMI to a supported GPIO_CS pin, follow these steps: 1. Disable XMIs to the CPU core. For SMIs, this is done via CSC index 9Dh[0]. For NMIs, this is done via CSC index 9Dh[2]. 2. Select whether GPIO_XMI produces an SMI or NMI via CSC index B0h[4]. 3. Map a GPIO_CS pin to the GPIO_XMI signal using CSC index B0h[3–0]. 4. Clear the GPIO_WAS_XMI bit at CSC index 96h[5]. 5. Enable XMIs to the CPU core by reversing the action from step 1 above. XMIs are fully enabled if steps 1–5 have been performed. Reading status and clearing the XMI for GPIO_XMI is performed via index 96h[5].

Programming Notes

	7	6	5	4	3	2	1	0
Bit	GPROM_CS2-MUX[3–0]				GPSCP_MUX[3–0]			
Default	1	1	1	1	1	1	1	1
R/W	R/W				R/W			

Bit	Name	Function
7–4	GPROM_CS2-MUX[3–0]	**Map ROM Chip Select 2 (ROMCS2) to One of the GPIO_CS Pins** The number programmed corresponds to the GPIO_CS that is mapped: 0 0 0 0 = GPIO_CS0 0 0 0 1 = GPIO_CS1 . . 1 1 1 0 = GPIO_CS14 1 1 1 1 = Disabled
3–0	GPSCP_MUX[3–0]	**Map Keyboard Controller (external SCP) Chip Select (60h and 64h) to One of the GPIO_CS Pins** The number programmed corresponds to the GPIO_CS that is mapped: 0 0 0 0 = GPIO_CS0 0 0 0 1 = GPIO_CS1 . . 1 1 1 0 = GPIO_CS14 1 1 1 1 = Disabled If this feature is enabled (i.e., external SCP chip select mapped to one of the GPIO_CS pins), then the chip select will be generated regardless of the setting of CSC index C1h[3–2].

Programming Notes

When mapping either the internal SCP address decoder or ROMCS2 signal to external GPIO_CS pins for the purpose of generating external chip selects, you must also configure the desired GPIO_CS pins to be outputs (see CSC indexes A0–A3h). In addition, you must ensure that the bits in index register A6h or A7h that correspond to the GPIO_CS pins you are using as chip selects have been cleared.

	7	6	5	4	3	2	1	0
Bit	GPCSB_MUX[3–0]				GPCSA_MUX[3–0]			
Default	1	1	1	1	1	1	1	1
R/W	R/W				R/W			

Bit	Name	Function
7–4	GPCSB_MUX[3–0]	**Map GP_CSB to One of the GPIO_CS Pins** The number programmed corresponds to the GPIO_CS that is mapped: 0 0 0 0 = GPIO_CS0 0 0 0 1 = GPIO_CS1 . . 1 1 1 0 = GPIO_CS14 1 1 1 1 = Disabled
3–0	GPCSA_MUX[3–0]	**Map GP_CSA to One of the GPIO_CS Pins** The number programmed corresponds to the GPIO_CS that is mapped: 0 0 0 0 = GPIO_CS0 0 0 0 1 = GPIO_CS1 . . 1 1 1 0 = GPIO_CS14 1 1 1 1 = Disabled

Programming Notes

When mapping GP_CSA/B to external GPIO_CS pins for the purpose of generating external chip selects, you must also configure the desired GPIO_CS pins to be outputs (see CSC indexes A0–A3h). In addition, you must ensure that the bits in index register A6h or A7h that correspond to the GPIO_CS pins you are using as chip selects have been cleared.

	7	6	5	4	3	2	1	0
Bit	GPCSD_MUX[3–0]				GPCSC_MUX[3–0]			
Default	1	1	1	1	1	1	1	1
R/W	R/W				R/W			

Bit	Name	Function
7–4	GPCSD_MUX[3–0]	**Map GP_CSD to One of the GPIO_CS Pins** The number programmed corresponds to the GPIO_CS that is mapped: 0 0 0 0 = GPIO_CS0 0 0 0 1 = GPIO_CS1 . . 1 1 1 0 = GPIO_CS14 1 1 1 1 = Disabled
3–0	GPCSC_MUX[3–0]	**Map GP_CSC to One of the GPIO_CS Pins** The number programmed corresponds to the GPIO_CS that is mapped: 0 0 0 0 = GPIO_CS0 0 0 0 1 = GPIO_CS1 . . 1 1 1 0 = GPIO_CS14 1 1 1 1 = Disabled

Programming Notes

When mapping GP_CSC/D to external GPIO_CS pins for the purpose of generating external chip selects, you must also configure the desired GPIO_CS pins to be outputs (see CSC indexes A0–A3h). You must also ensure that the bits in index register A6h or A7h that correspond to the GPIO_CS pins you are using as chip selects have been cleared.

	7	6	5	4	3	2	1	0
Bit	CSA_ADDR[7–0]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	CSA_ADDR[7–0]	**Chip Select A Address Bits 7–0** SA 7–0 for I/O chip select A generation.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved		CSA_SA3_ MASK	CSA_SA2_ MASK	CSA_SA1_ MASK	CSA_SA0_ MASK	CSA_ADDR[9–8]	
Default	x	x	0	0	0	0	0	0
R/W			R/W	R/W	R/W	R/W		

Bit	Name	Function
7–6	Reserved	**Reserved** During read/modify/write operations, software must preserve these bits.
5	CSA_SA3_MASK	**Mask SA3** 0 = Masked 1 = Not masked
4	CSA_SA2_MASK	**Mask SA2** 0 = Masked 1 = Not masked
3	CSA_SA1_MASK	**Mask SA1** 0 = Masked 1 = Not masked
2	CSA_SA0_MASK	**Mask SA0** 0 = Masked 1 = Not masked
1–0	CSA_ADDR[9–8]	**Chip Select A Address Bits 9–8** SA 9–8 for I/O Chip Select A generation.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	CSB_ADDR[7–0]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	CSB_ADDR[7–0]	**Chip Select B Address Bits 7–0** SA 7–0 for I/O Chip Select B generation.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved		CSB_SA3_ MASK	CSB_SA2_ MASK	CSB_SA1_ MASK	CSB_SA0_ MASK	CSB_SA0_MASK	
Default	x	x	0	0	0	0	0	0
R/W			R/W	R/W	R/W	R/W	R/W	

Bit	Name	Function
7–6	Reserved	**Reserved** During read/modify/write operations, software must preserve these bits.
5	CSB_SA3_MASK	**Mask SA3** 0 = Masked 1 = Not masked
4	CSB_SA2_MASK	**Mask SA2** 0 = Masked 1 = Not masked
3	CSB_SA1_MASK	**Mask SA1** 0 = Masked 1 = Not masked
2	CSB_SA0_MASK	**Mask SA0** 0 = Masked 1 = Not masked
1–0	CSB_ADDR[9–8]	**Chip Select B Address Bits 9–8** SA 9–8 for I/O Chip Select B generation.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved	GP_CSB_X8_X16	CSB_GATED_IOX[1–0]		Reserved	GP_CSA_X8_X16	CSA_GATED_IOX[1–0]	
Default	x	0	0	0	x	0	0	x
R/W		R/W	R/W			R/W	R/W	

Bit	Name	Function
7	Reserved	**Reserved** During read/modify/write operations, software must preserve this bit.
6	GP_CSB_X8_X16	**GP_CSB ISA I/O Cycle Data Bus Width and Timing Selector** 0 = 8-bit data bus size and timings will be used for ISA bus I/O transactions using GP_CSB 1 = 16-bit data size and timings will be used for GP_CSB ISA bus I/O transactions If IOCS16 is sampled active at the proper time per the ISA bus I/O timing requirements, 16-bit data size and timings will be used.
5–4	CSB_GATED_IOX[1–0]	**Qualify GP_CSB with $\overline{\text{IOR}}$/$\overline{\text{IOW}}$** GP_CSB generation, whose address decode is specified by indexes B6h and B7h, is always qualified with AEN being deasserted. It can be additionally qualified by $\overline{\text{IOR}}$ and/or $\overline{\text{IOW}}$ as follows: 0 0 = Not additionally qualified by $\overline{\text{IOR}}$ or by $\overline{\text{IOW}}$ 0 1 = Additionally qualified by $\overline{\text{IOR}}$ only 1 0 = Additionally qualified by $\overline{\text{IOW}}$ only 1 1 = Additionally qualified by either $\overline{\text{IOR}}$ or $\overline{\text{IOW}}$ These bits qualify an I/O address range hit to help define what will constitute a I/O chip select. A chip select has two main (and not necessarily related) uses which are to drive an external chip select pin for the purpose of selecting an external device and for use as an internally routed input to the PMU for generating PMU activity. In order to generate a CSB external chip select, an external pin must be selected via CSC index B2h[7–4]. In order to generate CSB activity, this must be enabled via CSC index 60h[1].
3	Reserved	**Reserved** During read/modify/write operations, software must preserve this bit.

Bit	Name	Function
2	GP_CSA_X8_X16	**GP_CSA ISA I/O Cycle Data Bus Width and Timing Selector** 0 = 8-bit data bus size and timings will be used for ISA bus I/O transactions using GP_CSA 1 = 16-bit data size and timings will be used for GP_CSA ISA bus I/O transactions. If IOCS16 is sampled active at the proper time per the ISA bus I/O timing requirements, 16-bit data size and timings will be used.
1–0	CSA_GATED_IOX[1–0]	**Qualify GP_CSA with $\overline{\text{IOR}}$/$\overline{\text{IOW}}$** GP_CSA generation whose address decode is specified by indexes B6h and B7h is always qualified with AEN being deasserted. It may be additionally qualified by $\overline{\text{IOR}}$ and/or $\overline{\text{IOW}}$ as follows: 0 0 = Not additionally qualified by $\overline{\text{IOR}}$ or by $\overline{\text{IOW}}$ 0 1 = Additionally qualified by $\overline{\text{IOR}}$ only 1 0 = Additionally qualified by $\overline{\text{IOW}}$ only 1 1 = Additionally qualified by either $\overline{\text{IOR}}$ or $\overline{\text{IOW}}$ These bits qualify an I/O address range hit to help define what will constitute a I/O chip select. A chip select has two main (and not necessarily related) uses: to drive an external chip select pin for the purpose of selecting an external device and for use as an internally routed input to the PMU for generating PMU activity. In order to generate a CSA external chip select, an external pin must be selected via CSC index B2h[3–0]. In order to generate CSA activity, this must be enabled via CSC index 60h[0].

Programming Notes

	7	6	5	4	3	2	1	0
Bit	CSC_ADDR[25–18]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	CSC_ADDR[25–18]	**Chip Select C Address Bits 25–18** SA25–18 for memory Chip Select C generation.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	CSC_SA17_ MASK	CSC_SA16_ MASK	CSC_SA15_ MASK	CSC_SA14_ MASK	CSC_ADDR[17–14]			
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W			

Bit	Name	Function
7	CSC_SA17_MASK	**Mask SA17** 0 = Masked 1 = Not masked
6	CSC_SA16_MASK	**Mask SA16** 0 = Masked 1 = Not masked
5	CSC_SA15_MASK	**Mask SA15** 0 = Masked 1 = Not masked
4	CSC_SA14_MASK	**Mask SA14** 0 = Masked 1 = Not masked
3–0	CSC_ADDR[17–14]	**Chip Select C Address Bits 17–14** SA17–14 for Memory Chip Select C generation.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	CSD_ADDR[25–18]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	CSD_ADDR[25–18]	**Chip Select D Address Bits 25–18** SA25–18 for Memory Chip Select D generation.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	CSD_SA17_ MASK	CSD_SA16_ MASK	CSD_SA15_ MASK	CSD_SA14_ MASK	CSD_ADDR[17–14]			
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W			

Bit	Name	Function
7	CSD_SA17_MASK	**Mask SA17** 0 = Masked 1 = Not masked
6	CSD_SA16_MASK	**Mask SA16** 0 = Masked 1 = Not masked
5	CSD_SA15_MASK	**Mask SA15** 0 = Masked 1 = Not masked
4	CSD_SA14_MASK	**Mask SA14** 0 = Masked 1 = Not masked
3–0	CSD_ADDR[17–14]	**Chip Select D Address Bits 17–14** SA17–14 for Memory Chip Select D generation.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	GP_CSD_X8_X16	CSD_GATED_MEMX[2–0]			GP_CSC_X8_X16	CSC_GATED_MEMX[2–0]		
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W			R/w	R/w		

Bit	Name	Function
7	GP_CSD_X8_X16	**GP_CSD ISA Memory Cycle Data Bus Width and Timing Selector** 0 = 8-bit data bus size and timings will be used for ISA bus memory transactions using GP_CSD 1 = 16-bit data size and timings will be used for GP_CSD ISA bus memory transactions If MCS16 is sampled active at the proper time per the ISA bus memory timing requirements, 16-bit data size and timings will be used.
6–4	CSD_GATED_MEMX[2–0]	**Qualify GP_CSD with $\overline{\text{MEMR}}$/$\overline{\text{MEMW}}$ or CPU Address Valid** 0 0 0 = Chip select D is a function of SA25–SA14 decode only 0 0 1 = Qualify SA25–SA14 decode with $\overline{\text{MEMR}}$ 0 1 0 = Qualify SA25–SA14 decode with $\overline{\text{MEMW}}$ 0 1 1 = Qualify SA25–SA14 decode with either $\overline{\text{MEMR}}$ or $\overline{\text{MEMW}}$ 1 0 0 = Qualify SA25–SA14 with CPU address valid 1 0 1 – 1 1 1 = Reserved These bits qualify a memory address range hit to help define what will constitute a memory chip select. A chip select has two main (and not necessarily related) uses which are to drive an external chip select pin for the purpose of selecting an external device and for use as an internally routed input to the PMU for generating PMU activity. In order to generate a CSD external chip select, an external pin must be selected via CSC index B3h[7–4]. In order to generate CSD activity, this must be enabled via CSC index 60h[3]. Bit pattern ‘000b’ should not be selected If GP_CSD is to be used as a PMU activity source since this setting does not qualify GP_CSD with the CPU address being valid. Use of ‘000b’ can result in the generation of false activities.

Bit	Name	Function
3	GP_CSC_X8_X16	**GP_CSC ISA Memory Cycle Data Bus Width and Timing Selector** 0 = 8-bit data bus size and timings will be used for ISA bus memory transactions using GP_CSC 1 = 16-bit data size and timings will be used for GP_CSC ISA bus memory transactions If MCS16 is sampled active at the proper time per the ISA bus memory timing requirements, 16-bit data size and timings will be used.
2–0	CSC_GATED_MEMX[2–0]	**Qualify GP_CSC with $\overline{\text{MEMR}}$/$\overline{\text{MEMW}}$ or CPU Address Valid** 0 0 0 = Chip Select C is a function of SA25–SA14 decode only 0 0 1 = Qualify SA25–SA14 decode with $\overline{\text{MEMR}}$ 0 1 0 = Qualify SA25–SA14 decode with $\overline{\text{MEMW}}$ 0 1 1 = Qualify SA25–SA14 decode with either $\overline{\text{MEMR}}$ or $\overline{\text{MEMW}}$ 1 0 0 = Qualify SA25–SA14 with CPU address valid 1 0 1 – 1 1 1 = Reserved These bits qualify a memory address range hit to help define what will constitute a memory chip select. A chip select has two main, (and not necessarily related) uses which are to drive an external chip select pin for the purpose of selecting an external device and for use as an internally routed input to the PMU for generating PMU activity. In order to generate an ÉlanSC400 microcontroller external chip select, an external pin must be selected via CSC index B3h[3–0]. In order to generate CSC activity, this must be enabled via CSC index 60h[2]. Bit pattern ‘000b’ should not be selected if GP_CSC is to be used as a PMU activity source since this setting does not qualify GP_CSC with the CPU address being valid. Use of ‘000b’ can result in the generation of false activities.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	INT_ON_ RST_CMD	INT_ON_ A20_CMD	MOU_ IRQ12_EN	KEY_IRQ1_ EN	KBROW14_ CAN_XMI	RST_ SNOOP_DIS	A20G_ SNOOP_DIS	M_BUF_ FULL_SLCT
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	INT_ON_RST_CMD	**SCP Reset Command XMI Enable** Control SMI/NMI generation if SCP CPU core reset command sequence ('FE') is seen: 0 = If CSC index 91h[6] = '1b', clearing this bit overrides the XMI generation for an input buffer write via port 64h when the command written = FEh. In addition, the port 64h write does not cause the input buffer full flag to be set in the keyboard status register which can be read back from port 64h. 1 = The above-mentioned I/O writes to ports 64/60h are not treated as special cases, and an XMI will be generated in both cases if CSC index 91h[6] = 1. This bit has no effect if CSC index 91h[6] = '0b'.
6	INT_ON_A20_CMD	**SCP GateA20 Command XMI Enable** Control SMI/NMI generation if SCP GateA20 command sequence ('D1', data) is seen: 0 = If CSC index 91h[6] = '1b', clearing this bit overrides the XMI generation for an input buffer write via port 64h when the command written = D1h. The XMI is also inhibited for the next byte written to the input buffer providing that it occurs via port 60h. In addition, neither the port 64h write nor the port 60h write will cause the input buffer full flag to be set in the Keyboard Status Register, which can be read back from port 64h. 1 = The above-mentioned I/O writes to ports 64h/60h are not treated as special cases and an XMI will be generated in both cases if CSC index 91h[6] = 1. This bit has no effect if CSC index 91h[6] = '0b'.
5	MOU_IRQ12_EN	**SCP Mouse Emulation IRQ12 Control** 0 = IRQ12 won't be generated as a result of a write to CSC index C4h 1 = IRQ12 will be generated as a result of a write to CSC index C4h This bit has no function when the external SCP is enabled C1h[3–2] = '10b'. Mouse IRQ12 won't be generated if CSC index C0h[0] is not set.
4	KEY_IRQ1_EN	**SCP Emulation/XT Keyboard IRQ1 Control** 0 = IRQ1 will not be generated by C3h writes or XT keyboard data in 1 = IRQ1 will be generated by subsequent C3h writes or XT data in See the table below for more information.

Bit	Name	Function
3	KBROW14_CAN_XMI	**Keyboard Row 14 XMI Enable** The SUS_RES and matrix keyboard row 14 inputs share the same ÉlanSC400 microcontroller pin, making row 14 a special case. If CSC index 91h[4] is set, matrix scan key presses can generate an SMI/NMI. Key presses that involve row 14 are excluded from this by default. This bit allows key presses that involve row 14 to be included in the SMI/NMI generation. Systems that use the pin as a SUS_RES input only can thus be configured have matrix key presses generate an SMI/NMI without forcing this attribute on SUS_RES. Selection of SMI versus NMI is via CSC index 98[3]. 0 = SUS_RES/KBD_ROW14 pin won't generate an SMI/NMI 1 = SUS_RES/KBD_ROW14 pin will generate a keypressed SMI/NMI if so enabled for the other rows (CSC index 91h[4] = 1)
2	RST_SNOOP_DIS	**SCP Reset Command Snooping Disable** The ÉlanSC400 microcontroller has no external input pin for the Hot Reset signal that would normally be driven by a PC/AT compatible external System Control Processor (SCP). In order to maintain software compatibility, hardware is provided to watch the busses for SCP Hot Reset pin control command sequences. When this bit is cleared, these command sequences will be automatically intercepted and processed by a hardware state machine within the ÉlanSC400 microcontroller. When set, the ÉlanSC400 microcontroller will not snoop for SCP Hot Reset control commands. This bit has no effect on whether or not the SCP Hot Reset control command sequence is driven off the ÉlanSC400 microcontroller. If a Hot Reset command is detected, the processor SRESET signal will be pulsed. 0 = Enabled 1 = Disabled
1	A20G_SNOOP_DIS	**SCP A20 Gate Command Snooping Disable** The ÉlanSC400 microcontroller has no external input pin for the GateA20 signal that would normally be driven by a PC/AT-compatible external System Control Processor (SCP). In order to maintain software compatibility, hardware is provided to watch the buses for SCP GateA20 pin control command sequences. When this bit is cleared, these command sequences will be automatically intercepted and processed by a hardware state machine within the ÉlanSC400 microcontroller. When set, the ÉlanSC400 microcontroller will not snoop for SCP GateA20 control commands This bit has no effect on whether or not the SCP GateA20 control command sequence is driven off the ÉlanSC400 microcontroller. 0 = Enabled 1 = Disabled The A20 line must propagate immediately following master reset to enable proper system operation. The state machine underlying this register bit ensures that the internal GateA20 control will force CPU A20 to propagate at this time. A20 will continue to propagate as a result of this control until software turns it off via the appropriate SCP command. When this bit is disabled, the A20Gate command snoop state machine is disabled and will no longer force CPU A20 to propagate.
0	M_BUF_FULL_SLCT	**Mouse Output Buffer Full Select** This bit selects whether bit 5 of the emulated Keyboard Status Register (direct-mapped port 64h) is the PC/AT compatible keyboard interface transmit time-out indicator, or the PS/2 compatible mouse output buffer full indicator. When used as the time-out bit, SCP emulation software must control port 64h[5] via the Keyboard Status Register Write Register (CSC index C5h). When configured to emulate the mouse output buffer full status bit, port 64h[5] is automatically set when the Mouse Output Buffer Write Register (CSC index C4h) is written to, and cleared when the Keyboard Output Buffer Write Register (CSC index C3h) is written to. No mouse IRQ12 will be generated unless this bit is set regardless of CSC index C0h[5]. 0 = Emulated SCP port 64h[5] = PC/AT keyboard interface time-out 1 = Emulated SCP port 64h[5] = PS/2 mouse output buffer full bit

SCP Support	XT KB Enabled	XT Intr Type	Effect of Setting Index Bit C0h[4]
External	N/A	N/A	No effect
None	No	N/A	No effect
None	Yes	XMI	No effect
None	Yes	IRQ1	Allows IRQ1 to be generated as a result of data being received from the XT keyboard interface.
Internal	No	N/A	Allows IRQ1 to be generated as a result of CSC index C3h being written to.
Internal	Yes	XMI	Allows IRQ1 to be generated as a result of CSC index C3h being written to.
Internal	Yes	IRQ1	Allows IRQ1 to be generated as a result of data being received from the XT keyboard interface OR as a result of CSC index C3h being written to. When configured like this, if either IRQ1 source is being asserted, the other source will not be able to generate an IRQ1. Care must be taken in the IRQ1 handler for this specialized case since if both sources of IRQ1 are asserted simultaneously (i.e., a write to CSC index C3h is quickly followed by arrival of data from the XT keyboard interface). The handler must not exit until one of the following has occurred: - Both IRQ1 sources are cleared - CSC index C0[4] has been toggled after one of the IRQ1 sources has been cleared to generate an edge for the remaining IRQ source. Failure to perform one of the above can result in the loss of further IRQ1 requests being detected by the PIC, as an edge is required for this.

This table shows the operation of CSC index C0h, bit 4 as it relates to other keyboard bit settings. SCP support is controlled by CSC index C1h[3–2], XT keyboard interface enable is controlled via CSC index C1h[4], and the XT interrupt type is controlled by CSC index C1h[5].

	7	6	5	4	3	2	1	0
Bit	Reserved	KEY_TIM_ STAT_CLR	XTKB_INTR_ SEL	XT_IF_ ENABLE	SCP_IF_SLCT		XTKB_ACK	XTKB_IF_EN
Default	x	0	0	0	0	0	0	0
R/W		R/W	R/W	R/W	R/W		R/W	R/W

Bit	Name	Function
7	Reserved	**Reserved** During read/modify/write operations, software must preserve this bit.
6	KEY_TIM_STAT_ CLR	**Keyboard Timer SMI/NMI Status/Clear** This bit is set when an SMI or NMI occurs as a result of a time-out on the keyboard-specific timer. Clearing this bit resets the SMI/NMI. Setting this bit has no effect. Keyboard timer SMIs/NMIs are enabled via CSC index 91h[5].
5	XTKB_INTR_SEL	**XT Keyboard SMI/NMI, or IRQ1 Generation Select** This bit has no meaning if the XT keyboard is disabled via bit 4 of this register. 0 = A byte received from the XT keyboard interface will generate IRQ1. 1 = A byte received from the XT keyboard interface will generate either an SMI or an NMI depending on the setting of CSC index 98h[3]. In order to generate an XT keyboard SMI/NMI, CSC index 91h[4] must also be set. SMI/NMI status for these interrupts is then read back from (and the SMI/NMI is cleared at CSC index 95h[4]. When this bit is set, key presses from the matrix keyboard will not generate SMI/NMIs, regardless of the state of CSC index 91h[4]. C0h[4] must also be set in order to generate an IRQ1 to the PIC.
4	XT_IF_ENABLE	**XT Keyboard Interface Enable** When this bit is set, data that is shifted in from the XT keyboard data line will be latched into the on-chip output buffer so that it can be read by the CPU at direct-mapped port 60h. 0 = XT keyboard interface is disabled 1 = XT keyboard interface is enabled Bits 3–2 of this register must = ‘01b’, for port 60h reads to return data shifted in from the XT keyboard interface. Besides enabling the basic XT keyboard interface, this configuration also allows SCP emulation to be performed while using a PC/XT keyboard versus a scanned matrix keyboard. If the external SCP interface is enabled bits 3–2 of this register = ‘10b’ at the same time as the XT keyboard interface, reads from direct-mapped port 60h will be from the external SCP.

Bit	Name	Function
3–2	SCP_IF_SLCT	**Enable Internal SCP Emulation Registers or External SCP Chip Select at 60h and 64h** 0 0 = I/O accesses to ports 60h and 64h go neither off the ÉlanSC400 microcontroller to the ISA bus nor to the internally emulated SCP input buffer, output buffer, or status register 0 1 = Enable Internal SCP emulation registers (Input Buffer, Output Buffer, and Status Register). I/O cycles to ports 60h and 64h will not go to the external ISA bus, but will instead access the internal emulation of the SCP input buffer, output buffer, and status register. 1 0 = Enable external SCP support. I/O cycles to ports 60h and 64h will go to the external ISA bus 1 1 = Same as 0 0 Bits 3–2 of this register in no way affect the A20Gate/hot reset snooping functionality. These features are independently controlled via the Keyboard Configuration Register A at CSC index C0h[2–1]. In addition, I/O writes to the output buffer at port 60h or 64h will put data in the internal emulation of the SCP output buffer. The external SCP chip select is set up using CSC index B1h. The internal IRQ1 and IRQ12 for supporting keyboard/mouse emulation are disabled when the external SCP support is enabled.
1	XTKB_ACK	**XT Keyboard Acknowledge** When the XT keyboard function is enabled, writing this bit has three functions. This bit can be set by software to disable the keyboard interface by holding the data line low. In order for the XT keyboard to be able to transmit date, this bit should be cleared. In addition, for each byte that the system receives from the keyboard, this bit must be toggled–first written high and then low. This performs the other two functions: clears the XT keyboard data shift register so that false keyboard interrupts aren't generated (this also results in the XT keyboard data port at I/O location 60h being cleared), and acknowledges to the keyboard that the system has read the last byte. Reading this bit returns the last value written. When the XT keyboard function is disabled, this bit has no effect on system operation. In a PC/XT Compatible system, this bit normally resides at direct-mapped port 61h[7].
0	XTKB_IF_EN	**XT Keyboard Interface Enable** When the XT keyboard function is enabled, set this bit to enable the keyboard interface, clear this bit to disable the XT keyboard interface (by holding the clock line low). Reading this bit returns the last value written. When the XT keyboard function is disabled, this bit has not effect on system operation. In a PC/XT Compatible system, this bit normally resides at direct-mapped port 61h[6].

Programming Notes

	7	6	5	4	3	2	1	0
Bit	IB_BACKDOOR[7–0]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	IB_BACKDOOR[7–0]	**Input Buffer Back Door** Writes to direct-mapped ports 60h or 64h will be stored in this register regardless of CSC index C1h[3–2]. SCP emulation software can read from this register to obtain the byte written to ports 60h/64h. This read will automatically clear both the IBF flag in the internally emulated Keyboard Status Register and the Input Buffer Write SMI/NMI (enabled via CSC index 91h[6]). Writing to this index register places data in the input buffer without setting the IBF flag in the Keyboard Status Register. An SMI or NMI will be generated based on CSC index 98h[3].

Programming Notes

	7	6	5	4	3	2	1	0
Bit	OB_BACKDOOR[7–0]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	OB_BACKDOOR[7–0]	**Output Buffer Back Door** Writes to this index will place a byte in the keyboard/mouse output buffer, and set the OBF flag in the internally emulated Keyboard Status Register (direct-mapped 64h) regardless of CSC index C1h[3–2]. In addition, port 64h[5] will be cleared if it has been configured to emulate the PS/2 compatible Mouse OBF flag. Writing this register will also clear the Output Buffer Read SMI/NMI (enabled via CSC index 91h[7]). The choice of SMI versus NMI is based on CSC index 98h[3]. A read of CSC index C3h or C4h returns the byte in the emulated SCP Output Buffer without clearing the OBF flag in the Keyboard Status Register. The same physical SCP emulation register stores both keyboard and mouse output buffer writes.

Programming Notes

Bits 7–0: If the XT keyboard interface is enabled (CSC index C1h[4] = ‘1b’), then data shifted in from the XT keyboard data pin will be placed in the emulated keyboard output buffer. If internal SCP emulation is enabled as well, then the output buffer can contain data from either the XT keyboard data input, writes to CSC index C3h, or writes to CSC index C4h, whichever occurred last.

	7	6	5	4	3	2	1	0
Bit	MOU_BACKDOOR[7–0]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	MOU_BACKDOOR[7–0]	**Mouse Output Buffer Back Door** Writes to this index will place a byte in the keyboard/mouse output buffer, and set the OBF flag in the Keyboard Status Register (direct-mapped 64h) regardless of CSC index C1h[3–2]. In addition, port 64[5] will be set if it has been configured to emulate the PS/2 compatible Mouse OBF flag. Writing this register will also clear the Output Buffer Read SMI/NMI (enabled via CSC index 91h[7]). The choice of SMI versus NMI is based on CSC index 98h[3]. A read of CSC index C3h or C4h returns the byte in the emulated SCP Output Buffer without clearing the OBF flag in the Keyboard Status Register. The same physical SCP emulation register stores both keyboard and mouse output buffer writes.

Programming Notes

Bits 7–0: If the XT keyboard interface is enabled, (CSC index C1h[4] = ‘1b’) then data shifted in from the XT keyboard data pin will be placed in the emulated keyboard output buffer. If internal SCP emulation is enabled as well, then the output buffer can contain data from either the XT keyboard data input, writes to CSC index C3h, or writes to CSC index C4h, whichever occurred last.

	7	6	5	4	3	2	1	0
Bit	KB_STAT_BACKDOOR[7–6,4,2]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	KB_STAT_BACKDOOR [7–6,4,2]	**Keyboard Status Back Door** Writes to this index will allow software to control bits 7, 6, 4, and 2 of the internal SCP emulation Keyboard Status Register (direct-mapped 64h) regardless of CSC index C1h[3–2]. Bits 3, 1, and 0 are controlled automatically, and are not directly controllable via software. Bit 5 can either be controlled automatically or by software depending on whether it is defined to be PC/AT keyboard Transmit Time-out, or PS/2 Mouse OBF (see CSC index C0h[0]). Reads of this register return all 8 bits of the emulated SCP status register.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	KEY_TIMER[7–0]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	KEY_TIMER[7–0]	**Keyboard Timer** Time delay that will elapse before the keyboard timer SMI/NMI will occur. A read of this register returns the last value written. The timer starts counting down as soon as a non-zero byte is written to it. For each time-out, it must be reprogrammed first to zero and then to a new non-zero value in order to generate another keyboard timer SMI/NMI. CSC index 91h[5] enables the SMI/NMI. The choice of SMI versus NMI is based on CSC index 98h[3]. 00h = Disable timer 01–FFh =Delay for (n+2.5)*3.91 milliseconds (± .5 milliseconds) after writing to this register and then generate a keyboard timer SMI/NMI The n term in the above equation is the value programmed into this register.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	MATKB_ COL7	MATKB_ COL6	MATKB_ COL5	MATKB_ COL4	MATKB_ COL3	MATKB_ COL2	MATKB_ COL1	MATKB_ COL0
Default	1	1	1	1	1	1	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	MATKB_COL7	**COL7 Signal** Write = Set Column pin High or Low Read = State of Column pin 0 = Signal drives Low 1 = I/O pin driver is three-stated with termination as selected by CSC index CAh
6	MATKB_COL6	**COL6 Signal** Write = Set Column pin High or Low Read = State of Column pin 0 = Signal drives Low 1 = I/O pin driver is three-stated with termination as selected by CSC index CAh
5	MATKB_COL5	**COL5 Signal** Write = Set Column pin High or Low Read = State of Column pin 0 = Signal drives Low 1 = I/O pin driver is three-stated with termination as selected by CSC index CAh
4	MATKB_COL4	**COL4 Signal** Write = Set Column pin High or Low Read = State of Column pin 0 = Signal drives Low 1 = I/O pin driver is three-stated with termination as selected by CSC index CAh
3	MATKB_COL3	**COL3 Signal** Write = Set Column pin High or Low Read = State of Column pin 0 = Signal drives Low 1 = I/O pin driver is three-stated with termination as selected by CSC index CAh
2	MATKB_COL2	**COL2 Signal** Write = Set Column pin High or Low Read = State of Column pin 0 = Signal drives Low 1 = I/O pin driver is three-stated with termination as selected by CSC index CAh

Bit	Name	Function
1	MATKB_COL1	**COL1 Signal** Write = Set Column pin High or Low Read = State of Column pin 0 = Signal drives Low 1 = I/O pin driver is three-stated with termination as selected by CSC index CAh
0	MATKB_COL0	**COL0 Signal** Write = Set Column pin High or Low Read = State of Column pin 0 = Signal drives Low 1 = I/O pin driver is three-stated with termination as selected by CSC index CAh

Programming Notes

When the XT keyboard interface is enabled (CSC index 39h[3] = ‘1b’), writes to bits 1–0 of this register are undefined. Reads from these bits return the states of the respective pins.

	7	6	5	4	3	2	1	0
Bit	MATKB_ ROW7	MATKB_ ROW6	MATKB_ ROW5	MATKB_ ROW4	MATKB_ ROW3	MATKB_ ROW2	MATKB_ ROW1	MATKB_ ROW0
Default	1	1	1	1	1	1	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	MATKB_ROW7	**ROW7 Signal** Write = Set row pin High or Low Read = State of row pin 0 = Signal drives Low 1 = I/O pin driver is three-stated
6	MATKB_ROW6	**ROW6 Signal** Write = Set row pin High or Low Read = State of row pin 0 = Signal drives Low 1 = I/O pin driver is three-stated
5	MATKB_ROW5	**ROW5 Signal** Write = Set row pin High or Low Read = State of row pin 0 = Signal drives Low 1 = I/O pin driver is three-stated
4	MATKB_ROW4	**ROW4 Signal** Write = Set row pin High or Low Read = State of row pin 0 = Signal drives Low 1 = I/O pin driver is three-stated
3	MATKB_ROW3	**ROW3 Signal** Write = Set row pin High or Low Read = State of row pin 0 = Signal drives Low 1 = I/O pin driver is three-stated
2	MATKB_ROW2	**ROW2 Signal** Write = Set row pin High or Low Read = State of row pin 0 = Signal drives Low 1 = I/O pin driver is three-stated

Bit	Name	Function
1	MATKB_ROW1	**ROW1 Signal** Write = Set row pin High or Low Read = State of row pin 0 = Signal drives Low 1 = I/O pin driver is three-stated
0	MATKB_ROW0	**ROW0 Signal** Write = Set row pin High or Low Read = State of row pin 0 = Signal drives Low 1 = I/O pin driver is three-stated

Programming Notes

No programmable termination is provided for matrix keyboard row I/O pins. These pins are pulled up internally with the option to disable the pull-ups in Suspend mode.

	7	6	5	4	3	2	1	0
Bit	Reserved	MATKB_ SUS_RES	MATKB_ ROW13	MATKB_ ROW12	MATKB_ ROW11	MATKB_ ROW10	MATKB_ ROW9	MATKB_ ROW8
Default	x	1	1	1	1	1	1	1
R/W		R	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	Reserved	**Reserved** During read/modify/write operations, software must preserve this bit.
6	MATKB_SUS_RES	**Row 14/Suspend-Resume Signal** Write = No effect Read = State of shared keyboard ROW14/SUS_RES pin
5	MATKB_ROW13	**ROW13 Signal** Write = Set row pin High or Low Read = State of row pin 0 = Signal drives Low 1 = I/O pin driver is three-stated
4	MATKB_ROW12	**ROW12 Signal** Write = Set row pin High or Low Read = State of row pin 0 = Signal drives Low 1 = I/O pin driver is three-stated
3	MATKB_ROW11	**ROW11 Signal** Write = Set row pin High or Low Read = State of row pin 0 = Signal drives Low 1 = I/O pin driver is three-stated
2	MATKB_ROW10	**ROW10 Signal** Write = Set row pin High or Low Read = State of row pin 0 = Signal drives Low 1 = I/O pin driver is three-stated
1	MATKB_ROW9	**ROW9 Signal** Write = Set row pin High or Low Read = State of row pin 0 = Signal drives Low 1 = I/O pin driver is three-stated
0	MATKB_ROW8	**ROW8 Signal** Write = Set row pin High or Low Read = State of row pin 0 = Signal drives Low 1 = I/O pin driver is three-stated

Programming Notes

No programmable termination is provided for matrix keyboard row I/O pins. These pins are pulled up internally with the option to disable the pull-ups in Suspend mode.

	7	6	5	4	3	2	1	0
Bit	COL7PULLUP	COL6PULLUP	COL5PULLUP	COL4PULLUP	COL3PULLUP	COL2PULLUP	COL1PULLUP	COL0PULLUP
Default	1	1	1	1	1	1	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	COL7PULLUP	**COL7 Termination** Write = Set column resistor pull-up or pull-down Read = Last value written 0 = Pull-down resistor 1 = Pull-up resistor
6	COL6PULLUP	**COL6 Termination** Write = Set column resistor pull-up or pull-down Read = Last value written 0 = Pull-down resistor 1 = Pull-up resistor
5	COL5PULLUP	**COL5 Termination** Write = Set column resistor pull-up or pull-down Read = Last value written 0 = Pull-down resistor 1 = Pull-up resistor
4	COL4PULLUP	**COL4 Termination** Write = Set column resistor pull-up or pull-down Read = Last value written 0 = Pull-down resistor 1 = Pull-up resistor
3	COL3PULLUP	**COL3 Termination** Write = Set column resistor pull-up or pull-down Read = Last value written 0 = Pull-down resistor 1 = Pull-up resistor
2	COL2PULLUP	**COL2 Termination** Write = Set column resistor pull-up or pull-down Read = Last value written 0 = Pull-down resistor 1 = Pull-up resistor

Bit	Name	Function
1	COL1PULLUP	**COL1 Termination** Write = Set column resistor pull-up or pull-down Read = Last value written 0 = Pull-down resistor 1 = Pull-up resistor
0	COL0PULLUP	**COL0 Termination** Write = Set column resistor pull-up or pull-down Read = Last value written 0 = Pull-down resistor 1 = Pull-up resistor

Programming Notes

The termination state that is specified using bits 7–0 is not actually felt at the pins until the Pin Termination Latch Command bit at CSC index E5[0] is set.

	7	6	5	4	3	2	1	0
Bit	Reserved		EXT_ECHO_ ZBUS	IO_ECHO _ZBUS	DMA1_DIS	DMA0_DIS	PCC_ENB	RTC_DIS
Default	x	x	0	0	0	0	0	0
R/W			R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7–6	Reserved	**Reserved** During read/modify/write operations, software must preserve these bits.
5	EXT_ECHO_ZBUS	**Chip Setup and Control Index Register Echo** 0 = CSC indexed register accesses will not be echoed to the ISA bus 1 = CSC indexed register accesses will echo address 22h/23h and control signals to the ISA bus Data bus signals will exit the ÉlanSC400 microcontroller, but will not make it to the ISA bus since the $\overline{\text{DBUFOE}}$ signal is not driven off the microcontroller for echoed internal register accesses. For an I/O read from an internal register, the data read will be driven off the microcontroller for debug purposes. All echoed internal register accesses will occur at ISA speed. During ECHO_ZBUS cycles, AEN is also asserted so that ISA I/O devices will not decode the ECHO_ZBUS cycles.
4	IO_ECHO_ZBUS	**Internal I/O Register Echo** 0 = Internal I/O device accesses will not be echoed to the ISA bus 1 = Internal I/O device accesses will echo address and control signals to the ISA Bus Regardless of the state of this bit, I/O writes to port 80h are a special case and always propagate to the ISA bus to support port 80h BIOS checkpoint display cards. Writes to ports 84h, 86h, 88h, 8Ch, and 8Eh operate the same as port 80h.
3	DMA1_DIS	**DMA 1 Slave Controller Disable** This bit controls the I/O address decode for the internal slave DMA controller core. 0 = DMA 1(Slave) controller enabled 1 = DMA 1 (Slave) controller disabled When this bit is cleared, I/O accesses to the slave DMA controller I/O control and status ports go exclusively to the slave DMA controller. Accesses to direct mapped ports C0h, C2h, C4h, C6h, C8h, CAh, CCh, CEh, D0h, D2h, D4h, D6h, D8h, DAh, DCh, and DEh are considered accesses to internal registers and no VL or ISA bus cycle will be generated for accesses to them. When this bit is set, the internal I/O address decode for these ports is disabled, so these same accesses generate external bus cycles. Disabling the I/O decode does not disable or reset the master/slave DMA controller core in any way. Accesses to these ports are not considered accesses to internal registers and VL or ISA bus cycles will be generated when they are accessed.

Bit	Name	Function
2	DMA0_DIS	**DMA 0 Slave Controller Disable** This bit controls the I/O address decode for the internal master DMA controller core. 0 = DMA 0 (master) controller enabled 1 = DMA 0 (master) controller disabled When this bit is cleared, I/O accesses to the master DMA controller I/O control and status ports go exclusively to the master/slave DMA controller. Accesses to direct mapped ports 00–0Fh are considered internal registers and no VL or ISA bus cycle will be generated for accesses to them. When this bit is set, the internal I/O address decode for these ports is disabled, so these same accesses generate external bus cycles. Disabling the I/O decode does not disable or reset the master DMA controller core in any way. Accesses to direct mapped ports 00–0Fh are not considered internal registers and VL or ISA bus cycles will be generated when they are accessed.
1	PCC_ENB	**PC Card Controller Enable** This bit controls the I/O address decode for the internal PC Card controller. When cleared, all I/O accesses to ports 3E0h and 3E1h generate external bus cycles. When set, I/O writes to port 3E0h go to the internal PC Card controller and to the external buses to support an external secondary PC Card controller. 0 = Internal PC Card controller disabled Accesses to addresses 3E0h and 3E1h go off the ÉlanSC400 microcontroller to the VL or ISA bus. 1 = Internal PC Card controller enabled In order to support an external second PC Card controller, writes to 3E0h go to both the internal PC Card controller and to the ISA bus. I/O reads from port 3E0h come from the internal PC Card controller only. Destination selection for writes to port 3E1h are based on the last value written to port 3E0h. If the last value written to port 3E0h was less than 80h, the destination of data written to port 3E1h will be in the internal PC Card controller. If the last value written to port 3E0h was greater than 80h, the write to 3E1h will generate an external bus cycle. Disabling the I/O decode does not disable or reset the PC Card controller core in any way, nor does it disable redirection of $\overline{\text{ROMCSx}}$ to PC Card Socket A.
0	RTC_DIS	**RTC Controller Enable** This bit controls the I/O address decode for the internal RTC core. When cleared, I/O accesses to ports 70h and 71h go exclusively to the internal RTC core. When set, the internal I/O decode the 70h and 71h are disabled, so these same accesses generate external bus cycles. Disabling the I/O decode does not disable or reset the core in any way. 0 = Internal RTC enabled 1 = Internal RTC disabled When this bit is cleared, accesses to direct-mapped ports 70h and 71h are considered internal registers and VL or ISA bus cycles will be generated for accesses to them. When this bit is set, accesses to direct mapped ports 70h and 71h are not considered internal registers and VL or ISA bus cycles will be generated when they are accessed. This allows use of an ISA-based external RTC module if required by the system design. Disabling the internal RTC does not automatically disable the internal IRQ8 connection between the RTC and the PIC. If an external RTC is used, all internal RTC IRQs must be masked off at the RTC (i.e., the PIE, AIE, and UIE bits must all be cleared).

Programming Notes

To enable the set up of MMS Windows C–F, the internal PC Card controller must be enabled (D0h[1] = 1) and operating in standard mode (F1h[0] = 0). Once any of MMS Windows C–F are opened (via 3E0h/3E1h index space), disabling the internal PC Card controller does not disable the

MMS window(s), but disallows their re-configuration until the internal PC Card controller is re-enabled. When the internal PC Card controller is disabled, I/O accesses to 3E0h/3E1h go off the ÉlanSC400 microcontroller to the sub-ISA bus.

	7	6	5	4	3	2	1	0
Bit	Reserved				PP_CONFIG	PP_ENB	SP_CONFIG	UART_ENB
Default	x	x	x	x	0	0	0	1
R/W					R/W	R/W	R/W	R/W

Bit	Name	Function
7–4	Reserved	**Reserved** During read/modify/write operations, software must preserve these bits.
3	PP_CONFIG	**Parallel Port Base Address Configuration** 0 = Internal parallel port base address = 378h (LPT1) 1 = Internal parallel port base address = 278h (LPT2) Parallel port IRQ levels are separately mapped via CSC index D8[4–3].
2	PP_ENB	**Internal Parallel Port Enable** 0 = Internal Parallel Port disabled 1 = Internal Parallel Port enabled When the internal parallel port is disabled, accesses to I/O locations in the 378–37Fh and 278–27Fh ranges go off the ÉlanSC400 microcontroller to the ISA bus.
1	SP_CONFIG	**Serial Port Base Address Configuration** 0 = Internal serial port base address = 3F8h (COM1) 1 = Internal serial port base address = 2F8h (COM2) Serial port IRQ levels are separately mapped via CSC index D8h[6–5].
0	UART_ENB	**Internal UART Enable** 0 = Internal UART disabled 1 = Internal UART enabled When this bit is cleared, the serial port and serial infrared interfaces are powered down (Power Down Group D), and the associated pins are three-stated with internal pull-downs. When the internal UART is disabled, accesses to I/O locations in the 3F8–3FFh and 2F8–2FFh ranges go off the ÉlanSC400 microcontroller to the ISA bus.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved					PP_TO_EN	PP_MODE[1–0]	
Default	x	x	x	x	x	0	0	0
R/W						R/W	R/W	

Bit	Name	Function
7–3	Reserved	**Reserved** During read/modify/write operations, software must preserve these bits.
2	PP_TO_EN	**Parallel port EPP Mode Time-Out Enable Bit** This bit is undefined unless the EPP mode of operation is selected via bits 1–0 of this register. 0 = EPP time-out is disabled 1 = EPP time-out is enabled
1–0	PP_MODE[1–0]	**Enhanced Parallel Port (EPP) Enable** 0 0 = Set/reset DMA Channel 4 DMA request per the REQDMA bit 0 1 = Set/reset DMA Channel 5 DMA request per the REQDMA bit 1 0 = Set/reset DMA Channel 6 DMA request per the REQDMA bit 1 1 = Set/reset DMA Channel 7 DMA request per the REQDMA bit If EPP mode is selected, automatic address and data strobes are generated when I/O writes are performed to the EPP address (port x7Bh) and data ports (x7C–x7Fh where x = 2 or 3) respectively. In addition, the parallel status port meaning is changed slightly. If Bidirectional mode is selected, data can be transmitted over the parallel port in either direction. See direct-mapped port x7Ah (x = 2 or 3), bit 5 for more detail on bidirectional operation.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	RFRT[1–0]		Reserved		Reserved	TFCLR	RFCLR	FIFOEN
Default	0	0	x	x	x	0	0	0
R/W					R	R	R	R

Bit	Name	Function
7–6	RFRT[1–0]	**Receiver FIFO Register Trigger Bits** When in 16550-compatible mode, this bit field specifies the trigger level at which the Interrupt Identification Register will report that a received data available interrupt is pending. If received data available interrupts are enabled in the IER, the system will be interrupted when the receive FIFO fills to the trigger as follows: 0 0 = 1 byte 0 1 = 4 bytes 1 0 = 8 bytes 1 1 = 14 bytes When the data in the receive FIFO falls below this trigger level, the interrupt will be cleared.
5–4	Reserved	**Reserved** These bits will read back ‘00b’.
3	Reserved	**Reserved** This bit will read back ‘1b’.
2	TFCLR	**Transmit FIFO Clear** Since the direct-mapped version of this bit (see direct-mapped 03FAh/02FAh) is self-clearing, it always reads back ‘0b’.
1	RFCLR	**Receive FIFO Clear** Since the direct-mapped version of this bit (see direct-mapped 03FAh/02FAh) is self-clearing, it always reads back ‘0b’.
0	FIFOEN	**FIFOs Enabled (16550-Compatible Mode Enabled)** 0 = UART is in 16450 compatible mode 1 = UART is in 16550 compatible mode This bit must be ‘1b’ when other FIFO control register bits are written to or they will not be programmed. Any mode switch will clear both FIFOs. The FIFOs must be enabled for the IrDA interface to operate in High-Speed mode. Accesses to receive and transmit FIFOs, and to all FIFO control bits in the write-only FIFO Control Register at direct-mapped I/O address 03FAh/02FAh (except bit 0) are disabled. Accesses to receive and transmit FIFOs, and to all FIFO control bits in the write-only FIFO Control Register at direct-mapped I/O address 03FAh/02FAh are enabled.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	PIRQ1S[3–0]				PIRQ0S[3–0]			
Default	0	0	0	0	0	0	0	0
R/W	R/W				R/W			

Bit	Name	Function
7–4	PIRQ1S[3–0]	**Programmable Interrupt Request 1 Routing** The value in this 4-bit field maps Programmable Interrupt Request 1 (PIRQ1) to an IRQ on the cascaded 8259. 0 0 0 0 = Disables PIRQ1 as an input 0 0 0 1 = IRQ1 0 0 1 0 = Reserved 0 0 1 1 = IRQ3 0 1 0 0 = IRQ4 0 1 0 1 = IRQ5 0 1 1 0 = IRQ6 0 1 1 1 = IRQ7 1 0 0 0 = IRQ8 1 0 0 1 = IRQ9 1 0 1 0 = IRQ10 1 0 1 1 = IRQ11 1 1 0 0 = IRQ12 1 1 0 1 = IRQ13 1 1 1 0 = IRQ14 1 1 1 1 = IRQ15
3–0	PIRQ0S[3–0]	**Programmable Interrupt Request 0 Routing** The value in this 4-bit field maps Programmable Interrupt Request 0 (PIRQ0) to an IRQ on the cascaded 8259. 0 0 0 0 = Disables PIRQ0 as an input 0 0 0 1 = IRQ1 0 0 1 0 = Reserved 0 0 1 1 = IRQ3 0 1 0 0 = IRQ4 0 1 0 1 = IRQ5 0 1 1 0 = IRQ6 0 1 1 1 = IRQ7 1 0 0 0 = IRQ8 1 0 0 1 = IRQ9 1 0 1 0 = IRQ10 1 0 1 1 = IRQ11 1 1 0 0 = IRQ12 1 1 0 1 = IRQ13 1 1 1 0 = IRQ14 1 1 1 1 = IRQ15

Programming Notes

	7	6	5	4	3	2	1	0
Bit	PIRQ3S[3–0]				PIRQ2S[3–0]			
Default	0	0	0	0	0	0	0	0
R/W	R/W				R/W			

Bit	Name	Function
7–4	PIRQ3S[3–0]	**Programmable Interrupt Request 3 Routing** The value in this 4-bit field maps Programmable Interrupt Request 3 (PIRQ3) to an IRQ on the cascaded 8259. 0 0 0 0 = Disables PIRQ3 as an input 0 0 0 1 = IRQ1 0 0 1 0 = Reserved 0 0 1 1 = IRQ3 0 1 0 0 = IRQ4 0 1 0 1 = IRQ5 0 1 1 0 = IRQ6 0 1 1 1 = IRQ7 1 0 0 0 = IRQ8 1 0 0 1 = IRQ9 1 0 1 0 = IRQ10 1 0 1 1 = IRQ11 1 1 0 0 = IRQ12 1 1 0 1 = IRQ13 1 1 1 0 = IRQ14 1 1 1 1 = IRQ15
3–0	PIRQ2S[3–0]	**Programmable Interrupt Request 2 Routing** The value in this 4-bit field maps Programmable Interrupt Request 2 (PIRQ2) to an IRQ on the cascaded 8259. 0 0 0 0 = Disables PIRQ2 as an input 0 0 0 1 = IRQ1 0 0 1 0 = Reserved 0 0 1 1 = IRQ3 0 1 0 0 = IRQ4 0 1 0 1 = IRQ5 0 1 1 0 = IRQ6 0 1 1 1 = IRQ7 1 0 0 0 = IRQ8 1 0 0 1 = IRQ9 1 0 1 0 = IRQ10 1 0 1 1 = IRQ11 1 1 0 0 = IRQ12 1 1 0 1 = IRQ13 1 1 1 0 = IRQ14 1 1 1 1 = IRQ15

Programming Notes

	7	6	5	4	3	2	1	0
Bit	PIRQ5S[3–0]				PIRQ4S[3–0]			
Default	0	0	0	0	0	0	0	0
R/W	R/W				R/W			

Bit	Name	Function
7–4	PIRQ5S[3–0]	**Programmable Interrupt Request 5 Routing** The value in this 4-bit field maps Programmable Interrupt Request 5 (PIRQ5) to an IRQ on the cascaded 8259. 0 0 0 0 = Disables PIRQ5 as an input 0 0 0 1 = IRQ1 0 0 1 0 = Reserved 0 0 1 1 = IRQ3 0 1 0 0 = IRQ4 0 1 0 1 = IRQ5 0 1 1 0 = IRQ6 0 1 1 1 = IRQ7 1 0 0 0 = IRQ8 1 0 0 1 = IRQ9 1 0 1 0 = IRQ10 1 0 1 1 = IRQ11 1 1 0 0 = IRQ12 1 1 0 1 = IRQ13 1 1 1 0 = IRQ14 1 1 1 1 = IRQ15
3–0	PIRQ4S[3–0]	**Programmable Interrupt Request 4 Routing** The value in this 4-bit field maps Programmable Interrupt Request 4 (PIRQ4) to an IRQ on the cascaded 8259. 0 0 0 0 = Disables PIRQ4 as an input 0 0 0 1 = IRQ1 0 0 1 0 = Reserved 0 0 1 1 = IRQ3 0 1 0 0 = IRQ4 0 1 0 1 = IRQ5 0 1 1 0 = IRQ6 0 1 1 1 = IRQ7 1 0 0 0 = IRQ8 1 0 0 1 = IRQ9 1 0 1 0 = IRQ10 1 0 1 1 = IRQ11 1 1 0 0 = IRQ12 1 1 0 1 = IRQ13 1 1 1 0 = IRQ14 1 1 1 1 = IRQ15

Programming Notes

	7	6	5	4	3	2	1	0
Bit	PIRQ7S[3–0]				PIRQ6S[3–0]			
Default	0	0	0	0	0	0	0	0
R/W	R/W				R/W			

Bit	Name	Function
7–4	PIRQ7S[3–0]	**Programmable Interrupt Request 7 Routing** The value in this 4-bit field maps Programmable Interrupt Request 7 (PIRQ7) to an IRQ on the cascaded 8259. 0 0 0 0 = Disables PIRQ7 as an input 0 0 0 1 = IRQ1 0 0 1 0 = Reserved 0 0 1 1 = IRQ3 0 1 0 0 = IRQ4 0 1 0 1 = IRQ5 0 1 1 0 = IRQ6 0 1 1 1 = IRQ7 1 0 0 0 = IRQ8 1 0 0 1 = IRQ9 1 0 1 0 = IRQ10 1 0 1 1 = IRQ11 1 1 0 0 = IRQ12 1 1 0 1 = IRQ13 1 1 1 0 = IRQ14 1 1 1 1 = IRQ15
3–0	PIRQ6S[3–0]	**Programmable Interrupt Request 6 Routing** The value in this 4-bit field maps Programmable Interrupt Request 6 (PIRQ6) to an IRQ on the cascaded 8259. 0 0 0 0 = Disables PIRQ6 as an input 0 0 0 1 = IRQ1 0 0 1 0 = Reserved 0 0 1 1 = IRQ3 0 1 0 0 = IRQ4 0 1 0 1 = IRQ5 0 1 1 0 = IRQ6 0 1 1 1 = IRQ7 1 0 0 0 = IRQ8 1 0 0 1 = IRQ9 1 0 1 0 = IRQ10 1 0 1 1 = IRQ11 1 1 0 0 = IRQ12 1 1 0 1 = IRQ13 1 1 1 0 = IRQ14 1 1 1 1 = IRQ15

Programming Notes

	7	6	5	4	3	2	1	0
Bit	CURCON_IRQ_EN	UART_IRDA_IRQS[1–0]		PPORT_IRQS[1–0]		Reserved		
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W		R/W		R/W		

Bit	Name	Function
7	CURCON_IRQ_EN	**Cursor Address Register Access IRQ Enable** When this bit is set, writes to the 6845-compatible cursor address registers at CSC indexes 0Eh and 0Fh of 3x4/3x5h address space will generate IRQ9 to the system. This interrupt can be used to keep track of the cursor location independent of application software when the physical display screen is smaller than the frame buffer. 0 = IRQ9 is not generated to the PIC by writes to cursor address high/low registers 1 = IRQ9 is generated to the PIC by writes to cursor address high/low registers
6–5	UART_IRDA_IRQS[1–0]	**UART and IrDA IRQ Routing** Controls mapping of either the UART or the IrDA core's internal IRQ to the programmable interrupt controller. 0 0 = Disables UART/IRDA IRQs as an input to the PIC 0 1 = UART/IRDA IRQ mapped to the IRQ3 input of the PIC 1 0 = UART/IRDA IRQ mapped to the IRQ4 input of the PIC 1 1 = Disables UART/IRDA IRQ as an input to the PIC In regard to UART/IrDA IRQs, when CSC index EAh[0] = 0 (UART mode), only normal UART (16450/550) IRQ generation sources are available. When CSC index EAh[0] = 1, and CSC index EAh[1] = 0 (Slow-Speed IrDA mode), only normal UART (16450/550) IRQ generation sources are available. When CSC index EAh[0] = 1, and CSC index EAh[1] = 1 (High-Speed IrDA mode), only IrDA IRQ sources (see CSC index EBh[6–4]) are available.
4–3	PPORT_IRQS[1–0]	**Parallel Port IRQ Routing** Controls mapping of the on board parallel port's internal IRQ to the programmable interrupt controller. 0 0 = Disables PPORT_IRQ as an input to the PIC 0 1 = PPORT IRQ mapped to the IRQ5 input of the PIC 1 0 = PPORT IRQ mapped to the IRQ7 input of the PIC 1 1 = Disables PPORT IRQ as an input
2–0	Reserved	**Reserved**

Programming Notes

	7	6	5	4	3	2	1	0
Bit	DMA3[25–24]		DMA2[25–24]		DMA1[25–24]		DMA0[25–24]	
Default	0	0	0	0	0	0	0	0
R/W	R/W		R/W		R/W		R/W	

Bit	Name	Function
7–6	DMA3[25–24]	**DMA Channel 3 Page Register Extension** Highest two bits of memory address [A25–A24] for Channel 3.
5–4	DMA2[25–24]	**DMA Channel 2 Page Register Extension** Highest two bits of memory address [A25–A24] for Channel 2.
3–2	DMA1[25–24]	**DMA Channel 1 Page Register Extension** Highest two bits of memory address [A25–A24] for Channel 1.
1–0	DMA0[25–24]	**DMA Channel 0 Page Register Extension** Highest two bits of memory address [A25–A24] for Channel 0.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved		DMA7[25–24]		DMA6[25–24]		DMA5[25–24]	
Default	x	x	0	0	0	0	0	0
R/W			R/W		R/W		R/W	

Bit	Name	Function
7–6	Reserved	**Reserved** During read/modify/write operations, software must preserve these bits.
5–4	DMA7[25–24]	**DMA Channel 7 Page Register Extension** Highest two bits of memory address [A25–A24] for Channel 7.
3–2	DMA6[25–24]	**DMA Channel 6 Page Register Extension** Highest two bits of memory address [A25–A24] for Channel 6.
1–0	DMA5[25–24]	**DMA Channel 5 Page Register Extension** Highest two bits of memory address [A25–A24] for Channel 5.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	IRDA_CH_SEL[1–0]		PDMA1_CH_SEL[2–0]			PDMA0_CH_SEL[2–0]		
Default	0	0	0	0	0	0	0	0
R/W	R/W		R/W			R/W		

Bit	Name	Function
7–6	IRDA_CH_SEL[1–0]	**IRDA Controller DMA Channel Routing** 0 0 = Unconnected 0 1 = Channel 0 (8-bit channel) 1 0 = Channel 1 (8-bit channel) 1 1 = Unconnected
5–3	PDMA1_CH_SEL[2–0]	**Programmable DMA Controller Channel 1 Routing** Map PDRQ1/$\overline{\text{PDACK1}}$ programmable DMA pins to an internal DMA controller channel: 0 0 0 = Unconnected 0 0 1 = Channel 0 (8-bit channel) 0 1 0 = Channel 1 (8-bit channel) 0 1 1 = Channel 2 (8-bit channel) 1 0 0 = Channel 3 (8-bit channel) 1 0 1 = Channel 5 (16-bit channel) 1 1 0 = Channel 6 (16-bit channel) 1 1 1 = Channel 7 (16-bit channel)
2–0	PDMA0_CH_SEL[2–0]	**Programmable DMA Controller Channel 0 Routing** Map PDRQ0/$\overline{\text{PDACK0}}$ programmable DMA pins to an internal DMA controller channel: 0 0 0 = Unconnected 0 0 1 = Channel 0 (8-bit channel) 0 1 0 = Channel 1 (8-bit channel) 0 1 1 = Channel 2 (8-bit channel) 1 0 0 = Channel 3 (8-bit channel) 1 0 1 = Channel 5 (16-bit channel) 1 1 0 = Channel 6 (16-bit channel) 1 1 1 = Channel 7 (16-bit channel)

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved				PCMB_CH_SEL[1–0]		PCMA_CH_SEL[1–0]	
Default	x	x	x	x	0	0	0	0
R/W					R/W		R/W	

Bit	Name	Function
7–4	Reserved	**Reserved** During read/modify/write operations, software must preserve these bits.
3–2	PCMB_CH_SEL[1–0]	**PC Card Socket B DMA Channel Routing** 0 0 = Unconnected 0 1 = Channel 3 (8-bit channel) 1 0 = Channel 2 (8-bit channel) 1 1 = Channel 5 (16-bit channel)
1–0	PCMA_CH_SEL[1–0]	**PC Card Socket A DMA Channel Routing** 0 0 = Unconnected 0 1 = Channel 3 (8-bit channel) 1 0 = Channel 2 (8-bit channel) 1 1 = Channel 5 (16-bit channel)

Programming Notes

	7	6	5	4	3	2	1	0
Bit	VERTDOUB	DOT320	DOTDOUB	DRWIDTH	DUAL_SCRN	VID_ENB	FLATENB	COMP_MOD
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	VERTDOUB	**Vertical Line Doubling Enable** 0 = Normal (no doubling) 1 = Double each vertical line (e.g., display 120 lines on 240 screen)
6	DOT320	**Horizontal Dot Doubling Qualifier** 0 = Unconditionally double dots as controlled by bit 5 1 = Double horizontal dots only in CGA modes which display 320 horizontal pixels (e.g., double in 40-column mode, but not in 80-column mode) To use this mode, bit 0 of this register must be cleared.
5	DOTDOUB	**Horizontal Dot Doubling Enable** 0 = Normal (no doubling). 1 = Double horizontal dots (e.g., display 320 dots on 640 screen)
4	DRWIDTH	**Drive Width Select (Applies to Single Scan Panels Only)** 0 = Select 4 bits per shift 1 = Select 8 bits per shift (requires 8-bit data interface to panel)
3	DUAL_SCRN	**Panel Type Select** 0 = Single scan, single drive 1 = Dual scan, dual drive
2	VID_ENB	**Enable Graphics Controller** 0 = Internal graphics disabled 1 = Internal graphics enabled The internal graphics controller will not work when any other 32-bit interface is enabled such as the DRAM and ROM interfaces. In addition, the graphics controller should not be enabled at the same time as the VESA bus interface (see CSC index 14h[3]) since they share pins on the ÉlanSC400 microcontroller.
1	FLATENB	**Select Linear Flat-mapped Graphics Mode** 0 = CGA/MDA compatible mode 1 = Enable linear flat-mapped graphics
0	COMP_MOD	**Compatibillty Mode** 0 = 3D4/3D5h visible, compatibility mode will be CGA 1 = 3B4/3B5h visible, compatibility mode will be MDA

Programming Notes

	7	6	5	4	3	2	1	0
Bit	UNDERENB	REGLOCK	PIXDEPTH[1–0]		GRAPOL	TXTPOL	BLANKDAT	NONDISP
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W		R/W	R/W	R/W	R/W

Bit	Name	Function
7	UNDERENB	**Underline Attribute Enable** 0 = Disable underline attribute. Bit 3 of the character attribute byte will be ignored in MDA mode, but will be used in character color processing in CGA mode. 1 = Enable underline attribute. Bit 3 of the character attribute byte will add an underline to a character. Bit 3 will not be used in character color processing.
6	REGLOCK	**Lockout for Graphics Controller Indexed Registers at 3x4/3x5h** 0 = Enable graphics controller indexed registers at 3x4/3x5h (Graphics controller must be enabled also) 1 = Lockout graphics controller indexed registers at 3x4/3x5h
5–4	PIXDEPTH[1–0]	**Pixel Depth for Linear Flat-mapped Modes** 0 0 = 1 bit per pixel 0 1 = 2 bits per pixel 1 0 = 4 bits per pixel 1 1 = Reserved
3	GRAPOL	**Graphic Modes Data Polarity** 0 = Do not invert output data in Graphics modes 1 = Invert all output data in Graphics modes
2	TXTPOL	**Text Mode Data Polarity** 0 = Do not invert output data in text modes 1 = Invert all output data in text modes
1	BLANKDAT	**Blanked Data Value** This bit controls the shading of data that is within the displayed screen region but contains no memory mapped pixel information (i.e., pixels between Display End and Border End). The shade set by this bit can be overridden in some modes by setting bit 7 of the Gray Shade Mode Register (ports 3x4h/3x5h index 43h). 0 = Set blanked data to 0 1 = Set blanked data to 1
0	NONDISP	**Non-display Data Value** This bit controls the shading of data which can be output on excess lines beyond the limit of the displayed area. Applicable only if the non-display lines register is programmed to a non-zero value. Bits 2–3 do not affect this setting. 0 = Set non-display data to 0 1 = Set non-display data to 1

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved	ISAWIN_EN	WPWIN_STOP[25–20]					
Default	x	0	0	0	0	0	0	0
R/W		R/W	R/W					

Bit	Name	Function
7	Reserved	**Reserved** During read/modify/write operations, software must preserve this bit.
6	ISAWIN_EN	**Enable Overlapping ISA Window** 0 = Programmable ISA window disabled 1 = Programmable ISA window enabled This ISA Window is defined as an address range that, when enabled, will overlap physical system memory (DRAM). When this window is enabled, CPU accesses to this address range will generate an VL or ISA bus cycle instead of a DRAM cycle. This feature allows an ISA memory hole to be created within DRAM space anywhere below 16 Mbyte.
5–0	WPWIN_STOP[25–20]	**Stop Address Bits SA25–SA20** Contains bits 5–0 of the stop address bits for this write-protected window. There are 63 valid stop addresses which reside on 1 Mbyte boundaries between 64 Mbytes and 1 Mbyte of the physical DRAM address space. The start address is always 3FFFFFFh, so the write protect window grows downward. [3Fh] = 1 Mbyte window size (from 3FFFFFF–3F00000h) [3Eh] = 2 Mbyte window size (from 3FFFFFF–3E00000h) [01h] = 63 Mbyte window size (from 3FFFFFF–0100000h) [00h] = System DRAM write protect window is disabled Note that this write protect window applies only to system DRAM. If system DRAM resides from 0–4MB, and other memory mapped resources reside above that, the DRAM write protect window will have no effect on those memory-mapped resources.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	ISAWIN_START[23–16]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	ISAWIN_START[23–16]	**ISA Window Size Control** Contains bits 23–16 of the start address for the overlapping ISA window. There are 256 possible start addresses which reside on 64 Kbytes boundaries in the destination address space below 16 Mbytes. If all eight bits are cleared, the window start address SA25–SA0 would be 0000000h.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	ISAWIN_SIZE[7–0]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	ISAWIN_SIZE[7–0]	**ISA Window Size Control** Contains bits 7–0 of the encoded window size for this overlapping ISA window. Each encoding represents one of 256 possible window sizes in terms of 64 Kbytes multiples. 00h = 64 Kbytes window size (1 x 64 Kbytes) FFh = 16 Mbytes window size (256 x 64 Kbytes)

Programming Notes

	7	6	5	4	3	2	1	0
Bit	NOPWR_ PCMB_SUS	NOPWR_ PCMA_SUS	NOPWR_VL_ SUS	Reserved	NOPWR_ UART_SUS	NOPWR_ ISA_SUS	NOPWR_ ROM_SUS	NOPWR_ DRAM_SUS
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	NOPWR_PCMB_SUS	**PC Card Socket B Interface Powered or Not Powered in Suspend Mode** $\overline{IOR}$ and $\overline{IOW}$ need special handling if the ISA interface is not powered down (Power Down Group H). 0 = Powered, signals stopped inactive 1 = Not powered, signals three-state with pull-downs
6	NOPWR_PCMA_SUS	**PC Card Socket A Interface Powered or Not Powered in Suspend Mode** $\overline{IOR}$ and $\overline{IOW}$ need special handling if the ISA interface is not powered down (Power Down Group G). 0 = Powered, signals stopped inactive 1 = Not powered, signals three-state with pull-downs
5	NOPWR_VL_SUS	**VL Local Bus Interface Powered or Not Powered in Suspend Mode (Power Down Group F)** 0 = Powered, signals stopped inactive 1 = Not powered, signals three-state with pull-downs
4	Reserved	**Reserved**
3	NOPWR_UART_SUS	**Serial Port and Serial Infrared Interfaces Powered or Not Powered in Suspend Mode (Power Down Group D)** 0 = Powered, signals stopped inactive 1 = Not powered, signals three-state with pull-downs If CSC index D1h[0] is cleared, the serial port and serial infrared interfaces are powered down with pins three-stated and internally pulled down in all modes, so this bit has no effect.
2	NOPWR_ISA_SUS	**ISA Bus Interface Powered or Not Powered in Suspend Mode** The ISA signals that share pins with other functions will only be affected by this register if they are enabled rather than the alternate function. $\overline{IOR}$ and $\overline{IOW}$ need special handling if both PC Card interfaces are not powered down. (Power Down Group C) 0 = Powered, signals stopped inactive 1 = Not powered, signals three-state with pull-downs
1	NOPWR_ROM_SUS	**ROM Interface Powered or Not Powered in Suspend Mode (Power Down Group B)** 0 = Powered, signals stopped inactive 1 = Not powered, signals three-state with pull-downs
0	NOPWR_DRAM_SUS	**DRAM Interface Powered or Not Powered in Suspend Mode (Power Down Group A)** 0 = Powered, signals active for DRAM refresh 1 = Not powered, signals three-state with pull-downs

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved							NOPWR_ SDBUF_SUS
Default	x	x	x	x	x	x	x	0
R/W								R/W

Bit	Name	Function
7–1	Reserved	**Reserved** During read/modify/write operations, software must preserve these bits.
0	NOPWR_SDBUF_SUS	**$\overline{\text{DBUFOE}}$, $\overline{\text{R32BFOE}}$ Powered or Not Powered in Suspend Mode (Power Down Group A)** 0 = Powered 1 = Not powered, signals three-state with pull-downs This bit has no meaning unless either pin straps are set up to enable the $\overline{\text{DBUFOE}}$ or $\overline{\text{R32BFOE}}$ signals ($\overline{\text{R32BFOE}}$ is automatically enabled if the $\overline{\text{ROMCS0}}$ 32-bit interface is selected via pin strap), or software programs the $\overline{\text{ROMCS0}}$ interface to be 32-bit (this also enables $\overline{\text{R32BFOE}}$). See CSC index 20h[1–0] for $\overline{\text{ROMCS0}}$ width programming details.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	SKTB_OVRD	SKTA_OVRD	SDBUF_OVRD	Reserved	Reserved	ISA_OVRD	ROM_OVRD	TERM_LATCH
Default	0	0	0	x	x	0	0	0
R/W	R/W	R/W	R/W			R/W	R/W	R/W

Bit	Name	Function
7	SKTB_OVRD	**PC Card Socket B Termination Override** When the PC Card Socket B interface remains powered in Suspend mode, setting this bit will cause the Power Down Group H output signals, which are normally driven high, to be three-stated instead. This can be used to override the suspend pin state setting that results when bit 7 of the Suspend Pin State Register A = ‘0b’. 0 = The Power Down Group H signals are driven high if the interface is to remain powered in Suspend mode as configured via bit 7 of the Suspend Pin State Register A 1 = The Power Down Group H signals are three-stated if the interface is to remain powered in Suspend mode
6	SKTA_OVRD	**PC Card Socket A Termination Override** When the PC Card Socket A interface remains powered in Suspend mode, setting this bit will cause the Power Down Group G output signals, which are normally driven high, to be three-stated instead. This can be used to override the suspend pin state setting that results when bit 6 of the Suspend Pin State Register A = ‘0b’. 0 = The Power Down Group G signals are driven high if the interface is to remain powered in Suspend mode as configured via bit 6 of the Suspend Pin State Register A 1 = The Power Down Group G signals are three-stated if the interface is to remain powered in Suspend mode
5	SDBUF_OVRD	**Suspend Mode Termination Override** When the $\overline{\text{DBUFOE}}$, $\overline{\text{R32BFOE}}$ signals remain powered in Suspend mode, setting this bit will cause these output signals, which are normally driven high, to be three-stated instead. This can be used to override the suspend pin state setting that results when bit 0 of the Suspend Pin State Register B = ‘0b’. 0 = The $\overline{\text{DBUFOE}}$, $\overline{\text{R32BFOE}}$ signals are driven high if the interface is to remain powered in Suspend mode as configured via bit 0 of the Suspend Pin State Register B 1 = The $\overline{\text{DBUFOE}}$, $\overline{\text{R32BFOE}}$ signals are three-stated if the interface is to remain powered in Suspend mode This bit has no meaning unless either pin straps are set up to enable the $\overline{\text{DBUFOE}}$ or $\overline{\text{R32BFOE}}$ signals ($\overline{\text{R32BFOE}}$ is automatically enabled if the $\overline{\text{ROMCS0}}$ 32-bit interface is selected via pin strap), or software programs the $\overline{\text{ROMCS0}}$ interface to be 32-bit (this also enables $\overline{\text{R32BFOE}}$). See CSC index 20h[1–0] for $\overline{\text{ROMCS0}}$ width programming details.
4	Reserved	**Reserved** During read/modify/write operations, software must preserve this bit.

Bit	Name	Function
3	Reserved	**Reserved** During read/modify/write operations, software must preserve this bit.
2	ISA_OVRD	**ISA Interface Termination Override** When the ISA interface remains powered in Suspend mode, setting this bit will cause the Power Down Group C output signals, which are normally driven high, to be three-stated instead. This can be used to override the suspend pin state setting that results when bit 2 of the Suspend Pin State Register A = ‘0b’. 0 =The Power Down Group C signals are driven high if the interface is to remain powered in Suspend mode as configured via bit 2 of the suspend Pin State Register A 1 =The Power Down Group C signals are three-stated if the interface is to remain powered in Suspend mode
1	ROM_OVRD	**ROM Interface Termination Override** When the ROM interface remains powered in Suspend mode, setting this bit will cause the Power Down Group B output signals, which are normally driven high, to be three-stated instead. This can be used to override the suspend pin state setting that results when bit 1 of the Suspend Pin State Register A = ‘0b’. 0 = The Power Down Group B signals are driven high if the interface is to remain powered in Suspend mode as configured via bit 1 of the Suspend Pin State Register A 1 = The Power Down Group B signals are three-stated if the interface is to remain powered in Suspend mode
0	TERM_LATCH	**Pin Termination Latch Command** This bit affects CSC index registers 3B–3Eh, CAh, EA[6], and F2h, all of which control termination for certain pins on the device. Writing to these CSC index registers does not result in a change in the termination at the pins until this bit is set. While the actual pin terminations are in the process of being configured internally, this bit will read back ‘1b’ to indicate that further pin termination changes should not be attempted by software. When this bit reads back ‘0b’, it is safe to reconfigure terminations. Writing this bit to ‘0b’ has no effect. For termination control bits that only affect termination during suspend, it is not necessary to set this bit for the changes to take effect since this will automatically occur when suspend is entered.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	EOT_SLCT	SIRIN_PD_DIS	RECV_ BLOCKING	IRDA_RECEIVE	IRQ_ENABLE	START_DMA	SELMODE	SELDEVICE
Default	x	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	EOT_SLCT	**End of Transmission Status Bit Select** 0 = TC/EOT status bit in CSC index EBh = terminal count status 1 = TC/EOT status bit in CSC index EBh = end of transmission status See the TC/EOT bit at CSC index EBh[6] for more detail.
6	SIRIN_PD_DIS	**Serial Infra Red SIRIN Pin Internal Pull-down Resistor Disable (regardless of PMU mode).** 0 = SIRIN pin internal pull-down resistor is enabled 1 = SIRIN pin internal pull-down resistor is disabled The pull-down resistor state that is specified using this bit is not actually felt at the pin until the Pin Termination Latch Command bit at CSC index E5[0] is set.
5	RECV_BLOCKING	**Receive Blocking** 0 = Enables receive blocking during transmit 1 = Disables receive blocking during transmit
4	IRDA_RECEIVE	**IrDA Receive** This bit determines the IrDA data direction. 0 = IrDA is in Transmit mode 1 = IrDA is in Receive mode
3	IRQ_ENABLE	**High-Speed IrDA IRQ Enable** This bit controls the generation of high speed mode (DMA-based) IrDA interrupt requests. This bit is only valid when the IrDA interface is operated in High-Speed (DMA) IrDA mode. 0 = Disables the overflow, underflow, and terminal count IrDA IRQs 1 = Enables the overflow, underflow, and terminal count (TC) IRQs generated from IrDA
2	START_DMA	**IrDA DMA Start-Up Control** 0 = No effect 1 = Indicates to the High-Speed IrDA block to initiate DRQs in Transmit mode The bit is automatically cleared when the TC (terminal count) is reached on a DMA channel that is routed to the IrDA block during the High-Speed DMA transmit modes.

Bit	Name	Function
1	SELMODE	**IrDA Data Rate Select** 0 = IrDA Slow-Speed mode (115 Kbits/second) 1 = IrDA High-Speed mode (1.15 Mbits/second) High-Speed IrDA mode uses DMA only. Normal UART interrupts generated due to receive buffer full, Transmit Holding Register empty, and status changes do not apply, and will not be generated. In order to operate in High-Speed IrDA mode, the UART must be set up for 16550-compatible mode, that is, FIFOs enabled, via the FIFO Control Register at direct-mapped I/O location 03FAh/02FAh. Slow-Speed IrDA mode operation requires that the UART be set up for 8 data bits, no parity, and 1 stop bit via the UART Line Control Register at direct-mapped I/O location 03FBh/02FBh. Slow-Speed IrDA mode does not use DMA, and uses all of the traditional UART interrupts and controls for data transfer.
0	SELDEVICE	**UART or IrDA Mode Select** 0 = UART mode 1 = IrDA mode When UART mode is selected, the IrDA interface and all associated control and status bits have no meaning except this bit. When IrDA mode is selected and is operated in Slow-Speed IrDA mode by clearing this bit, use all of the normal 16550 UART control and status bits to transmit and receive data over the IrDA interface.

Programming Notes

The use of either Low- or High-Speed IrDA requires that the on-board UART be enabled by setting CSC index D1h[0].

	7	6	5	4	3	2	1	0
Bit	HS_IRDA_ IRQ_STATUS	TC/EOT	RECV_OVER FLOW	XMIT_ UNDERFLOW	RECV_FIFO_ FULL	RECV_FIFO_ EMPTY	XMIT_FIFO_ FULL	XMIT_FIFO_ EMPTY
Default	0	0	0	0	0	1	0	1
R/W	R	R/W	R/W	R/W	R	R	R	R

Bit **Name** **Function**

7 HS_IRDA_IRQ_STATUS

High-Speed Mode IrDA Interrupt Request Status

This bit indicates the status of a High-Speed mode IrDA interrupt request. It is a logical OR of bits 4, 5, and 6 of this register, only if CSC index EAh[3] (IRQ_ENABLE) is set; otherwise, this bit reads back '0b'. This bit is cleared automatically when all of the latched IRQ status represented by bits 4, 5, and 6 of this register have been cleared.

0 = High-Speed mode IRQ from IrDA has not occurred
When slow speed IrDA is selected, the normal 16550-compatible registers are used to control the FIFO, generate IRQs, and receive status.

1 = High-Speed mode IRQ from IrDA has occurred

6 TC/EOT

IrDA DMA Terminal Count/End of Transmission Status

This bit is only valid when operating the IrDA interface in High-Speed (DMA) mode.

0 = DMA TC/EOT did not generate an IrDA IRQ
If CSC index EAh[7] = 0, this is the IrDA TC (Terminal Count) status and interrupt clear bit. This bit will be set, and an IRQ will be generated, if IrDA DMA mode related IRQs are enabled via CSC index EAh[3], and the DMA controller signals that it has transferred all data possible without further software intervention (the DMA transfer count has expired). Software should write a '0b' to this bit to clear the TC interrupt (a write of '1b' has no effect).

1 = DMA TC/EOT did generate an IrDA IRQ
If CSC index EAh[7] = 1, this is the IrDA EOT (End Of Transmission) status and interrupt clear bit. This bit will be set, and an IRQ will be generated, if IrDA DMA mode-related IRQs are enabled via CSC index EAh[3], and all data that has been transferred to the transmit FIFO for a given IrDA DMA block transfer has actually been transmitted out the IrDA port. This condition will be true when 8 bit-times have transpired after having received the internal IrDA DMA TC signal.

5 RECV_OVERFLOW

Receive FIFO Overflow Status

This bit is only valid when operating the IrDA interface in High-Speed (DMA) mode. This bit will be set and an IRQ will be generated, if IrDA DMA mode related IRQs are enabled via CSC index EAh[3], and the IrDA receiver attempts to place received a byte into the receive FIFO in preparation for storage to memory by the DMA controller, but the receive FIFO is completely full. This is an indication to software that the DMA controller could not transfer data from the receive FIFO to system memory as fast as the IrDA receiver is filling the receive FIFO. Software should write a '0b' to this bit to clear the receive FIFO overflow interrupt (a write of '1b' has no effect).

0 = Overflow did not generate an IrDA IRQ

1 = Overflow generated an IrDA IRQ

Bit	Name	Function
4	XMIT_UNDERFLOW	**Transmit FIFO Underflow Status** This bit is only valid when operating the IrDA interface in High-Speed (DMA) mode. This bit will be set and an IRQ will be generated, if IrDA DMA mode related IRQs are enabled via CSC index EAh[3], and the IrDA transmitter attempts to remove a byte from the transmit buffer in preparation for transmission, but the transmit FIFO buffer is completely empty. This is an indication to software that the DMA controller could not transfer data from system memory to the transmit FIFO as fast as the IrDA transmitter is emptying the transmit FIFO. Software should write a '0b' to this bit to clear the transmit FIFO underflow interrupt (a write of '1b' has no effect). 0 = Underflow did not generate an IrDA IRQ 1 = Underflow generated an IrDA IRQ
3	RECV_FIFO_FULL	**Receive FIFO Full Status** 0 = Receive FIFO is not full 1 = 16 byte receive FIFO is full
2	RECV_FIFO_EMPTY	**Receive FIFO Empty Status** 0 = Receive FIFO is not empty 1 = 16 byte receive FIFO is empty
1	XMIT_FIFO_FULL	**Transmit FIFO Full Status** 0 = Transmit FIFO is not full 1 = 16 byte transmit FIFO is full
0	XMIT_FIFO_EMPTY	**Transmit FIFO Empty Status** 0 = Transmit FIFO is not empty 1 = 16 byte transmit FIFO is empty

Programming Notes

If CSC index EAh[3] is ever disabled and bit 7 = 1, bits 6–4 of this register should be cleared by software. Also note that if more than one of bits 6–4 is set and software does not clear all of the pending interrupts during the write to CSC index EBh, special hardware will automatically generate another edge to the PIC so that another IRQ will occur when the current Interrupt Handling Routine (IHR) finishes and exits. This allows the IHR to handle one event per IRQ without risk of losing an IRQ. If software clears all pending IRQ sources simultaneously, hardware will not generate this automatic edge to the PIC.

	7	6	5	4	3	2	1	0
Bit	RECV_ABORT	FRM7_CRC_ERR	FRM6_CRC_ERR	FRM5_CRC_ERR	FRM4_CRC_ERR	FRM3_CRC_ERR	FRM2_CRC_ERR	FRM1_CRC_ERR
Default	x	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R	R

Bit	Name	Function
7	RECV_ABORT	**Receive Abort** 0 = Receive abort has not occurred 1 = Receive abort has occurred
6	FRM7_CRC_ERR	**Frame #7 CRC Error Detect** 0 = No CRC error detected in frame #7 1 = CRC error detected in frame #7
5	FRM6_CRC_ERR	**Frame #6 CRC Error Detect** 0 = No CRC error detected in frame #6 1 = CRC error detected in frame #6
4	FRM5_CRC_ERR	**Frame #5 CRC Error Detect** 0 = No CRC error detected in frame #5 1 = CRC error detected in frame #5
3	FRM4_CRC_ERR	**Frame #4 CRC Error Detect** 0 = No CRC error detected in frame #4 1 = CRC error detected in frame #4
2	FRM3_CRC_ERR	**Frame #3 CRC Error Detect** 0 = No CRC error detected in frame #3 1 = CRC error detected in frame #3
1	FRM2_CRC_ERR	**Frame #2 CRC Error Detect** 0 = No CRC error detected in frame #2 1 = CRC error detected in frame #2
0	FRM1_CRC_ERR	**Frame #1 CRC Error Detect** 0 = No CRC error detected in frame #1 1 = CRC error detected in frame #1

Programming Notes

Bits 7–0: These bits are cleared by writing data of arbitrary value to this index register.

	7	6	5	4	3	2	1	0
Bit	OWN_ADDR[7–0]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	OWN_ADDR[7–0]	**Own Address** This register holds the 8-bit address that is currently assigned to the on-board IrDA interface.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	FRAME_LEN[7–0]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	FRAME_LEN[7–0]	**IrDA Data Frame Length Register Bits 7–0** Bits 7–0 of the 11-bit IrDA Data Frame Length Register. See CSC index EFh for bits 10–8. The 11-bit value written across these two registers is the length of a transmitted IrDA data frame in bytes. It includes a count of the actual data bytes in a frame, plus one byte for the address and two bytes for the CRC. Therefore, the number programmed in this 11-bit register will be the data size plus 3. If multiple consecutive frames are sent (up to the seven allowed by the IrDA standard), the length of each of these frames must conform to FRAME_LEN10–FRAME_LEN0.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved					FRAME_LEN[10–8]		
Default	x	x	x	x	x	0	0	0
R/W						R/W		

Bit	Name	Function
7–3	Reserved	**Reserved** During read/modify/write operations, software must preserve these bits.
2–0	FRAME_LEN[10–8]	**IrDA Data Frame Length Register Bits 10–8** Bits 10–8 of the 11-bit IrDA Data Frame Length Register. See CSC index EFh for bits 7–0. The 11-bit value written across these two registers is the length of a transmitted IrDA data frame in bytes. It includes a count of the actual data bytes in a frame, plus one byte for the address and two bytes for the CRC. Therefore, the number programmed in this 11-bit register will be the data size plus 3. If multiple consecutive frames are sent (up to the seven allowed by the IrDA standard), the length of each of these frames must conform to FRAME_LEN10–FRAME_LEN0.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	MEM_WIN_SEL[3–0]				Reserved		FORCE_CD_B	FORCE_CD_A
Default	0	0	0	0	x	x	0	0
R/W	R/W						R/W	R/W

Bit	Name	Function
7–4	MEM_WIN_SEL[3–0]	**PC Card Memory Window Select** These bits have no effect in PC Card Enhanced mode (i.e., when 10 PC Card memory windows are enabled). In Standard mode, six total PC Card memory windows are available: Windows 0–4 from Socket A, and Window 0 from Socket B. To support resource sharing between the sockets, Windows 1–4 from Socket A can be redirected to Socket B using this bit field.The socket mappings are shown in the table below.
3–2	Reserved	**Reserved** During read/modify/write operations, software must preserve these bits.
1	FORCE_CD_B	**Socket B Force Card Detect Event** This bit is reset to 0 when this register is read. Writing to this bit has the following effect on the CD_CHNG bit in the Socket B Card Status Change Register: 0 = No effect (i.e., current state is unaffected) 1 = Set the FORCE_CD_B bit in the PC Card Extended Features Register and the CD_CHNG bit in the Socket B Card Status Change Register to 1 The CD_CHNG bit in the Socket B Card Status Change Register is set asynchronously by writing to the FORCE_CD_B bit. This means that when CD_CHNG is cleared after reading the Socket B Card Status Change Register, another card status change interrupt will not be generated due to the fact that the FORCE_CD_B bit is still set.
0	FORCE_CD_A	**Socket A Force Card Detect Event** This bit is reset to 0 when this register is read. Writing to this bit has the following effect on the CD_CHNG bit in the Socket A Card Status Change Register: 0 = No effect (i.e., current state is unaffected) 1 = Set the FORCE_CD_A bit in the PC Card Extended Features Register and the CD_CHNG bit in the Socket A Card Status Change Register to 1 The CD_CHNG bit in the Socket A Card Status Change Register is set asynchronously by writing to the FORCE_CD_A bit. This means that when CD_CHNG is cleared after reading the Socket A Card Status Change Register, another card status change interrupt will not be generated due to the fact that the FORCE_CD_A bit is still set.

Programming Notes

Index F0h MEM_WIN_SEL [3–0]	**Socket A Memory Windows 1–4 Socket Mappings**		**Index F0h MEM_WIN_SEL [3–0]**	**Socket A Memory Windows 1–4 Socket Mappings**	
	Socket A	**Socket B**		**Socket A**	**Socket B**
0000 (0h)	1,2,3,4	None	1000 (8h)	1,2,3	4
0001 (1h)	2,3,4	1	1001 (9h)	2,3	1,4
0010 (2h)	1,3,4	2	1010 (Ah)	1,3	2,4

Index F0h MEM_WIN_SEL [3–0]	Socket A Memory Windows 1–4 Socket Mappings		Index F0h MEM_WIN_SEL [3–0]	Socket A Memory Windows 1–4 Socket Mappings	
	Socket A	Socket B		Socket A	Socket B
0011 (3h)	3,4	1,2	1011 (Bh)	3	1,2,4
0100 (4h)	1,2,4	3	1100 (Ch)	1,2	3,4
0101 (5h)	2,4	1,3	1101 (Dh)	2	1,3,4
0110 (6h)	1,4	2,3	1110 (Eh)	1	2,3,4
0111 (7h)	4	1,2,3	1111 (Fh)	None	1,2,3,4

	7	6	5	4	3	2	1	0
Bit	DMA_EN_B[1–0]		DMA_EN_A[1–0]		Reserved		CLK_SEL	MODE
Default	0	0	0	0	x	x	0	0
R/W	R/W		R/W				R/W	R/W

Bit	Name	Function
7–6	DMA_EN_B[1–0]	**Socket B DMA Enable** These bits enable and disable DMA mode for Socket B. When enabled, these bits also determine which pin is used at the DMA Request signal to forward to the DRQs via the PC CARD Controller Data Request Signal (Socket B). The actual DRQ that the PC CARD Controller Data Request Signal (Socket B) is mapped to is controlled by the DMA controller block. 0 0 = DMA disabled 0 1 = DMA disabled 1 0 = WP_B/$\overline{\text{IOS16\_B}}$ is DMA request 1 1 = BVD2_B/$\overline{\text{SPKR\_B}}$ is DMA request The MODE bit must be set (Enhanced mode) for bits 7–6 to have any meaning.
5–4	DMA_EN_A[1–0]	**Socket A DMA Enable** These bits enable and disable DMA mode for Socket A. When enabled, these bits also determine which pin is used at the DMA Request signal to forward to the DRQs via the PC CARD Controller Data Request Signal (Socket A). The actual DRQ that the PC CARD Controller Data Request Signal (Socket A) is mapped to is controlled by the DMA controller block. 0 0 = DMA Disabled 0 1 = DMA Disabled 1 0 = WP_A/$\overline{\text{IOS16\_A}}$ is DMA Request 1 1 = BVD2_A/$\overline{\text{SPKR\_A}}$ is DMA Request The MODE bit must be set (Enhanced mode) for bits 5–4 to have any meaning.
3–2	Reserved	**Reserved** During read/modify/write operations, software must preserve these bits.

Bit	Name	Function
1	CLK_SEL	**Clock Select** Selects the clock speed at which the PC Card controller runs. See PC Card Timing Control Registers for details on the effects this bit has on PC Card cycles. 0 = Nominal 8 MHz ISA bus clock When the CPU clock is ≥ 8 Mhz, the PC Card controller will run at 8 MHz. When the CPU clock is < 8 Mhz, the PC Card controller will run at the CPU clock rate. 1 = Nominal 33 MHz local bus clock (can be reduced under PMU control)
0	MODE	**PC Card Controller Mode** Selects the PC Card controller operating mode. 0 = Standard mode In this mode, only six of the memory windows are available for PC Cards (two are dedicated and four are floating), the CLK_SEL bit has no meaning (only nominal 8 MHz ISA bus clock is available), and all DMA features are disabled. 1 = Enhanced mode In this mode, all ten memory windows are available for PC Cards, the CLK_SEL bit can be used to select a high speed operating mode, and the DMA features are enabled.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved						SKB_PU_EN	SKA_PU_EN
Default	x	x	x	x	x	x	1	1
R/W							R/W	R/W

Bit	Name	Function
7–2	Reserved	**Reserved** During read/modify/write operations, software must preserve these bits.
1	SKB_PU_EN	**PC Card Socket B Input Pull-up Resistor Enable/Disable** 0 = Disabled 1 = Enabled
0	SKA_PU_EN	**PC Card Socket A Input Pull-up Resistor Enable/Disable** 0 = Disabled 1 = Enabled

Programming Notes

The termination state that is specified using bits 1–0 is not actually felt at the pins until the Pin Termination Latch Command bit at CSC index E5h[0] is set.

	7	6	5	4	3	2	1	0
Bit	MAJORSTEP[3–0]				MINORSTEP[3–0]			
Default	0	0	0	0	0	0	0	0
R/W	R				R			

Bit	Name	Function
7–4	MAJORSTEP[3–0]	**Major Stepping Level as in Revision MAJOR.MINOR**
3–0	MINORSTEP[3–0]	**Minor Stepping Level**

Programming Notes

The value read back depends on the current major.minor versions of a specific ÉlanSC400 microcontroller.

CHAPTER

4 RTC AND CMOS RAM INDEXED REGISTERS

4.1 OVERVIEW

The registers described in this chapter function as the configuration, setup, and status for the Real-Time Clock (RTC), as well as user-configurable RAM locations. They are listed in hexadecimal order by function in Table 4-1.

- RTC indexed registers 00–09h contain RTC seconds, minutes, hours, day of week, day of month, months of year, and year status, as well as second, minute, and hour alarm configuration controls. Binary and BCD formats for these registers are listed in a single table on page 4-5.
- Detailed register descriptions are included for the RTC indexed registers 0A–0Dh, which are used to configure the RTC.
- All index values from 0E–7Fh can be used as read/write RAM locations.

The 114 general-purpose RAM bytes are not dedicated within the RTC. They can be used by system or application level software and are fully available during the RTC update cycle. The 114 user RAM bytes provide low-power CMOS battery-backed storage and additional RAM.

RAM locations of the RTC are accessed using a two-step process.

- Identify a RAM location to be accessed by writing the RAM index to I/O port 70h, the RTC/CMOS RAM index register. The address must be in the range of 0E–7Fh.
- After writing the RTC/CMOS RAM Address Port, read the contents of the indexed CMOS RAM location from the RTC/CMOS RAM Data Port, port 71h, or by writing the desired data byte to this port.

The RTC registers are always accessed with ISA bus timings.

Table 4-1 RTC and CMOS RAM Register Map

Register Name	I/O (Port) Address	Index	Page Number
RTC/CMOS RAM Index Register	0070h		page 4-3
RTC/CMOS RAM Data Port	0071h		page 4-4
Time, Calendar, and Alarm Group		00–09h	page 4-5
RTC Current Second Register		00h	page 4-5
RTC Alarm Second Register		01h	page 4-6
RTC Current Minute Register		02h	page 4-7
RTC Alarm Minute Register		03h	page 4-8
RTC Current Hour Register		04h	page 4-9
RTC Alarm Hour Register		05h	page 4-10

Register Name	I/O (Port) Address	Index	Page Number
RTC Current Day of Week Register		06h	page 4-10
RTC Current Day of Month Register		07h	page 4-11
RTC Current Month Register		08h	page 4-12
RTC Current Year Register		09h	page 4-13
RTC Configuration Group		0A–0Dh	
Register A		0Ah	page 4-16
Register B		0Bh	page 4-18
Register C		0Ch	page 4-19
Register D		0Dh	page 4-20
CMOS RAM		0E–7Fh	

4.2 REGISTER DESCRIPTIONS

Each RTC and CMOS RAM indexed register is described on the following pages. Additional information about using these registers to program the ÉlanSC400 microcontroller can be found in the *ÉlanSC400 User's Manual* (order #21030).

	7	6	5	4	3	2	1	0
Bit	Reserved	CMOSIDX						
Default	–	x	x	x	0	0	0	0
R/W		W						

Bit	Name	Function
7	Reserved	**Reserved**
6–0	CMOSIDX	**CMOS RAM Index** CMOS RAM index to read/write.

Programming Notes

Bit 7 of this register is the master NMI gate control in a typical PC/AT Compatible system. For various reason, this bit has been made to reside at CSC index 9Dh[2] on the ÉlanSC400 microcontroller. Compatibility issues are minimized since the ÉlanSC400 microcontroller does not support generation of either of the legacy NMI sources (channel check or parity error).

	7	6	5	4	3	2	1	0
Bit	CMOSDATA							
Default	x	x	x	x	x	x	x	x
R/W	W							

Bit	Name	Function
7–0	CMOSDATA	**RTC/CMOS Data Port** Data to be written or read.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	RTC_SECOND							
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7–0	RTC_SECOND	**RTC Current Second Initialization and Readback Register** Software may initialize the seconds value for the RTC by writing data to this register in either binary or BCD (Binary Coded Decimal) formats. The seconds component of the RTC time may be read from this register. The RTC logic updates this register once per second. Valid values for this register range from 0 to 59.

Programming Notes

See the SET bit (RTC index 0BH[7]) and the DM bit (RTC index 0BH[2]) for further information that is pertinent to programming this register.

	7	6	5	4	3	2	1	0
Bit	RTC_ALARM_SECOND							
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7–0	RTC_ALARM_SECOND	**RTC Alarm Second Initialization and Readback Register** Software may initialize the alarm seconds value for the RTC by writing data to this register in either binary or BCD (Binary Coded Decimal) formats. The alarm seconds component of the RTC time may be read from this register. Writing any value from C0H to FFH to this register makes the seconds component of the alarm a wild card. For example, setting the hours, minutes, and seconds alarm registers to C0H will cause an RTC alarm event to be generated once per second. The wild card based once-per-second alarm will not occur unless the hours and minutes alarm settings are also wild cards. The RTC logic checks once per second to see if an alarm has occurred. Valid values for this register range from 0 to 59 and all wild card values.

Programming Notes

See the SET bit (RTC index 0BH[7]), the DM bit (RTC index 0BH[2]), and the RTC alarm IRQ status and control bits in RTC index 0CH for further information that is pertinent to programming this register.

	7	6	5	4	3	2	1	0
Bit	RTC_MINUTE							
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7–0	RTC_MINUTE	**RTC Current Minute Initialization and Readback Register** Software may initialize the minutes value for the RTC by writing data to this register in either binary or BCD (Binary Coded Decimal) formats. The minutes component of the RTC time may be read from this register. The RTC logic updates this register once per second. Valid values for this register range from 0 to 59.

Programming Notes

See the SET bit (RTC index 0BH[7]) and the DM bit (RTC index 0BH[2]) for further information that is pertinent to programming this register.

	7	6	5	4	3	2	1	0
Bit	RTC_ALARM_MINUTE							
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7–0	RTC_ALARM_MINUTE	**RTC Alarm Minute Initialization and Readback Register** Software may initialize the alarm minutes value for the RTC by writing data to this register in either binary or BCD (Binary Coded Decimal) formats. The alarm minutes component of the RTC time may be read from this register. Writing a value of C0H-FFH to this register makes the minutes component of the alarm a wild card. For example, setting the hours and minutes alarm registers to C0H will cause an RTC alarm event to be generated once per minute. The wild card based once-per-minute alarm will not occur unless the hours alarm setting is also a wild card. The RTC logic checks once per second to see if an alarm has occurred. Valid values for this register range from 0 to 59 and all wild card values.

Programming Notes

See the SET bit (RTC index 0BH[7]), the DM bit (RTC index 0BH[2]), and the RTC alarm IRQ status and control bits in RTC index 0CH for further information that is pertinent to programming this register.

	7	6	5	4	3	2	1	0
Bit	RTC_HOUR							
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7–0	RTC_HOUR	**RTC Current Hour Initialization and Readback Register** Software may initialize the hours value for the RTC by writing data to this register in either binary or BCD (Binary Coded Decimal) formats. The hours component of the RTC time may be read from this register. The RTC logic updates this register once per second. If the 24/12 bit (see RTC index 0BH[1])is cleared, bit 7 of this register indicates whether the current hour is AM or PM. If the high order bit is set, the current hour is PM. If the high order bit is cleared, the current hour is AM. In 24 hour mode, valid values for this register range from 0 to 23. In 12 hour mode, valid values for this register range from 1 to 12.

Programming Notes

See the SET bit (RTC index 0BH[7]), the DM bit (RTC index 0BH[2]) and the 24/12 bit (RTC index 0BH[1]) for further information that is pertinent to programming this register.

	7	6	5	4	3	2	1	0
Bit	RTC_ALARM_HOUR							
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7–0	RTC_ALARM_HOUR	**RTC Alarm Hour Initialization and Readback Register** Software may initialize the alarm hour value for the RTC by writing data to this register in either binary or BCD (Binary Coded Decimal) formats. The alarm hours component of the RTC time may be read from this register. Writing a value of C0H-FFH to this register makes the hours component of the alarm a wild card. For example setting the hours alarm register to C0H will cause an RTC alarm event to be generated once per hour. The RTC logic checks once per second to see if an alarm has occurred. If the 24/12 bit (see RTC index 0BH[1]) is cleared, bit 7 of this register indicates whether the alarm hour is AM or PM. If the high order bit is set, the alarm hour is PM. If the high order bit is cleared, the alarm hour is AM. In 24 hour mode, valid values for this register range from 0 to 23 and all wild card values. In 12 hour mode, valid values for this register range from 1 to 12 and all wild card values. Note that if this register is written with a wild card when 12 hour mode is selected, the AM/PM bit is a don't care since an alarm will occur every hour regardless.

Programming Notes

See the SET bit (RTC index 0BH[7]), the DM bit (RTC index 0BH[2]), the 24/12 bit (RTC index 0BH[1]), and the RTC alarm IRQ status and control bits in RTC index 0CH for further information that is pertinent to programming this register.

	7	6	5	4	3	2	1	0
Bit	RTC_DAY							
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7–0	RTC_DAY	**RTC Current Day of the Week** Valid values for this register range from 1 to 7 where: 1 = Sunday 2 = Monday 3 = Tuesday 4 = Wednesday 5 = Thursday 6 = friday 7 = Saturday

Programming Notes

See the SET bit (RTC index 0BH[7]) and the DM bit (RTC index 0BH[2]) for further information that is pertinent to programming this register.

	7	6	5	4	3	2	1	0
Bit	RTC_MONTH							
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7–0	RTC_MONTH	**RTC Current Day of the Month** Valid values for this register range from 1 to 31.

Programming Notes

See the SET bit (RTC index 0BH[7]) and the DM bit (RTC index 0BH[2]) for further information that is pertinent to programming this register.

	7	6	5	4	3	2	1	0
Bit	RTC_MONTH							
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7–0	RTC_MONTH	**RTC Current Month** Valid values for this register range from 1 to 12 where: 1 = January 2 = February 3 = March 4 = April 5 = May 6 = June 7 = July 8 = August 9 = September 10 = October 11 = November 12 = December

Programming Notes

See the SET bit (RTC index 0BH[7]) and the DM bit (RTC index 0BH[2]) for further information that is pertinent to programming this register.

	7	6	5	4	3	2	1	0
Bit	RTC_YEAR							
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7–0	RTC_YEAR	**RTC Current Year** Valid values for this register range from 0 to 99.

Programming Notes

See the SET bit (RTC index 0BH[7]) and the DM bit (RTC index 0BH[2]) for further information that is pertinent to programming this register.

	7	6	5	4	3	2	1	0
Bit	RTC_CMOS_REG_X							
Default	x	x	x	x	x	x	x	x
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7–0	RTC_CMOS_REG_X	**General Purpose, Battery-Backed (Nonvolatile) Storage Location for Use by System Firmware, Applications, etc.** There are 114 bytes of CMOS RAM available to the system. In a PC/AT-compatible system many of these bytes can be used by the system BIOS. The number of bytes used, and the meaning of data stored in a given CMOS RAM byte location may vary between different BIOS vendors or even between different versions of a single BIOS. Accesses to CMOS RAM locations can be performed without any regard for RTC operations. For example, DM bit (RTC index 0BH[2]) has no effect on CMOS RAM data. If CSC index D0H[0] (RTC_DIS) is set, the CMOS RAM will be unavailable, but not lost (unless both main and backup power to the RTC core is removed). Re-enabling the RTC IO decode via the RTC_DIS bit will allow access to the CMOS RAM with its contents intact.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	UIP	DV2–DV0			RS3–RS0			
Default	0	0	0	0	0	0	0	0
R/W	R	R/W			R/W			

Bit	Name	Function
7	UIP	**Update in Progress** This bit is provided for use by software that needs to modify the time, calendar or alarm registers in the real time clock. When this bit reads back '1b', these internal registers are unavailable for access by software since internal RTC logic is using them. When this bit reads back '0b', software will have a guaranteed minimum window of 244 µs in which modifications to these registers are allowed. Setting RTC index 0BH[7] inhibits RTC register update cycles and clears the UIP status bit. The time, calendar, and alarm information in RAM is fully available to the program when the UIP bit is 0, it is not in transition. The UIP bit is a read-only bit and is not affected by reset. Writing the Set bit in Register B to a 1 inhibits any update cycle and then clears the UIP status bit.
6–4	DV2-DV0	**Internal Oscillator Control Bits** 0 1 0 = Turn the oscillator on, use an internal time base of 32768 Hz, and enable the countdown chain to run at the internal time base frequency. This is the normal operational setting for DV2-DV0. 1 1 X = This value turns the oscillator on, but holds the count down chain in reset. In this mode, the time and date update cycles do not occur. This mode is useful for precision setting of the clock. If entering this mode from the "oscillator off" mode, a 200 millisecond delay must be observed to allow for oscillator stabilization prior to attempting to set the time. Time and date update cycles begin 500 milliseconds after the count down chain reset is removed. All other values = Programming DV2-DV0 to any value except '010b' or '11Xb' turns the oscillator off. In this mode, time and date update cycles do not occur, but the real time clock draws slightly less power. Upon exiting this mode, a 200 millisecond delay should be observed before reconfiguring or using the time and date information to allow for oscillator stabilization. These three bits are not affected by a reset of the RTC subsystem. The divider selection bits are also used to reset the divider chain. When the time/calendar is first initialized, the program can start the divider at the precise time stored in the RAM. When the divider reset is removed, the first update cycle begins 500 ms later. These three read/write bits are not affected by $\overline{\text{RTCRST}}$.

Bit	Name	Function
3–0	RS3-RS0	**Rate Selection** On a discrete RTC implementation, these bits control the rate selection for the square wave output and periodic interrupt features. On the ÉlanSC400 microcontroller, the square wave output feature is not available. However, the periodic interrupt output of the RTC is internally tied to IRQ8 and is available for use. RS3–RS0 control the rate at which periodic interrupts may be driven to IRQ8 as follows: 0 0 0 0 = Periodic interrupt disabled 1 0 0 0 = 3.906 milliseconds 0 0 0 1 = 3.906 milliseconds 1 0 0 1 = 7.812 milliseconds 0 0 1 0 = 7.812 milliseconds 1 0 1 0 = 15.625 milliseconds 0 0 1 1 = 122.070 microseconds 1 0 1 1 = 31.250 milliseconds 0 1 0 0 = 244.141 microseconds 1 1 0 0 = 62.500 milliseconds 0 1 0 1 = 488.281 microseconds 1 1 0 1 = 125.000 milliseconds 0 1 1 0 = 976.563 microseconds 1 1 1 0 = 250.000 milliseconds 0 1 1 1 = 1.953 milliseconds 1 1 1 1 = 500.000 milliseconds

Programming Notes

	7	6	5	4	3	2	1	0
Bit	SET	PIE	AIE	UIE	SQWE	DM	24/12	DSE
Default	x	0	0	0	0	x	x	x
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	SET	**Set Bit** When the SET bit is written to ’0b’, time and date update cycles are enabled, and will occur one-per-second. When the SET bit is written to a ’1b’, time and date update cycles are disabled, and any update in progress is aborted. This feature is useful for allowing time and date registers to be updated by software without being disturbed by an update cycle occurring while initializing. Neither internal functions nor RTC subsystem resets affect this bit. SET should be written to ’1b’ while updating index 0BH[2–0], and cleared afterward.
6	PIE	**Periodic Interrupt Enable** When this bit is set, the internal RTC periodic interrupt signal will latch the IRQF flag to ’1b’ (see RTC index 0CH[7]) upon a PF status bit transition from ’0b’ to ’1b’ (see RTC index 0CH[6]). If the PF status bit = ’1b’ when the PIE bit is set by software, IRQF will be asserted immediately. When this bit is cleared, no RTC periodic interrupt event is possible. PIE is not modified by any internal RTC functions, but is reset by a RTC subsystem reset. See RTC index 0AH[3–0] for interrupt rate configuration. See RTC index 0CH[6] for latched status of RTC periodic interrupt occurrence.
5	AIE	**Alarm Interrupt Enable** When this bit is set, the internal RTC alarm interrupt signal will latch the IRQF flag to ’1b’ (see RTC index 0CH[7]) upon an AF status bit transition from ’0b’ to ’1b’ (see RTC index 0CH[5]). If the AF status bit = ’1b’ when the AIE bit is set by software, IRQF will be asserted immediately. When this bit is cleared, no RTC alarm interrupt event is possible. AIE is not modified by any internal RTC functions, but is reset by a RTC subsystem reset. Use RTC indexes 01H, 03H, and 05H to configure when an RTC alarm interrupt will occur, and RTC index 0CH[5] for latched status of RTC alarm interrupt.
4	UIE	**Update-Ended Interrupt Enable** When this bit is set, the internal RTC update ended interrupt signal will latch the IRQF flag to ’1b’ (see RTC index 0CH[7]) upon a UF status bit transition from ’0b’ to ’1b’ (see RTC index 0CH[4]). If the UF status bit = ’1b’ when the UIE bit is set by software, IRQF will be asserted immediately. When this bit is cleared, no RTC update ended interrupt event is possible. UIE is not modified by any internal RTC functions, but is reset by either an RTC subsystem reset, or by writing the SET bit to ’1b’.
3	SQWE	**Square-Wave Enable** This bit has not been implemented on the ÉlanSC400 microcontroller.
2	DM	**Data Mode** The data mode (DM) bit indicates whether time and calendar updates are to use binary or BCD formats. Neither internal functions nor RTC subsystem resets affect this bit. When DM = ’0’, time and calendar data use BCD (binary coded Decimal) format. When DM = ’1b’, binary format is used. RTC indexes 0–9 must be re-initialized after changing this bit.
1	24/12	**24 Hour Mode Select Bit** When this bit = ’0b’, hours data uses 12-hour format, and the MSB of the hours bytes represents PM when ’1b’, and AM when ’0b’. When this bit = ’1b’, hours data uses 24 hour format. Neither internal functions nor RTC subsystem resets affect this bit. RTC indexes 4 and 5 must be re-initialized after changing this bit.
0	DSE	**Daylight Savings Enable** Setting this bit enables two special time updates to automatically occur. When set, the next time reading after 1:59:59 AM on the last Sunday in April will be 3:00:00 AM. In addition, the next time reading after 1:59:59 on the last Sunday in October will be 1:00:00 AM. These special updates do not occur when the DSE bit = ’0b’. Neither internal functions nor RTC subsystem resets affect this bit.

	7	6	5	4	3	2	1	0
Bit	IRQF	PF	AF	UF	Reserved			
Default	x	x	x	x	0	0	0	0
R/W	R	R	R	R	R			

Bit	Name	Function
7	IRQF	**Interrupt Request Flag** When this bit transitions from ’0b’ to ’1b’, the IRQ8 input of the ÉlanSC400 microcontroller Prioritized Interrupt Controller (PIC) is driven active which will generate a CPU interrupt if IRQ8 is unmasked at the PIC. When this bit = ’0b’, the IRQ8 input of the ÉlanSC400 microcontroller’s PIC will be driven inactive. The IRQF bit will be latched to a ’1b’ when any one (or more) of PF, AF, or UF flags transition from ’0b’ to ’1b’ while its/their respective enable(s) are also asserted. The IRQF bit will also be latched to a ’1b’ if an RTC interrupt source enable is written to ’1b’ when its associated flag bit was already asserted. RTC index 0CH is read/reset, and will also be reset by an RTC subsystem reset. If the ÉlanSC400 microcontroller’s internal RTC I/O address decode is disabled via CSC index D0H[0], the internal signal associated with the IRQF status bit will not automatically be disconnected from the PIC. If the intent is to use an external RTC to drive IRQ8 from an external PIRQ signal, then all internal RTC interrupt sources (PIE, AIE, UIE) must be masked off at the internal RTC prior to disabling the internal RTC port 70h/71h I/O decode via CSC index D0H[0].
6	PF	**Periodic Interrupt Flag** This bit is set when an RTC periodic interrupt event occurs regardless of the state of its individual enable (the PIE bit). Periodic interrupt events will occur at a rate that is configured via RTC index 0AH[3:0]. This bit is read/reset, and will also be reset by an RTC subsystem reset.
5	AF	**Alarm Interrupt Flag** This bit is set when an RTC alarm interrupt event occurs regardless of the state of its individual enable (the AIE bit). Alarm events can only occur with a time resolution of 1 second. An alarm event will occur when the current time (as defined by RTC indexes 0, 2, and 4) is equal to the alarm setting (as configured via RTC indexes 1, 3, and 5). The alarm time can contain wildcards for hour, minute, or second settings. A wild card is any value from C0H to FFH. This bit is read/reset, and will also be reset by an RTC subsystem reset.
4	UF	**Update-Ended Interrupt Flag** This bit is set upon termination of each time/date update cycle. This bit is read/reset, and will also be reset by an RTC subsystem reset.
3–0	Reserved	**Reserved** These bits are reserved. Each of them always reads back as ’0b’.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	VRT	Reserved						
Default	?	0	0	0	0	0	0	0
R/W	R	R						

Bit	Name	Function
7	VRT	**Valid RAM and Time** The Valid RAM and Time (VRT) bit is used to determine the validity of the RTC time, date and CMOS RAM registers. A read back of ’0b’ from this bit indicates that the RTC backup battery, as sensed by the ÉlanSC400 microcontroller’s BBATSEN pin, was below 2.4 volts prior to the application of main system power. Since this is too low to keep the RTC logic operational, the RTC date, time, and CMOS RAM are invalid, and the ÉlanSC400 microcontroller performed an RTC subsystem reset as a result. An RTC subsystem reset clears this latched status bit. Reading this bit will cause it to be latched to ’1b’, and it will remain set until an RTC subsystem reset occurs. Note that this bit is not a real time indication of the state of the external RTC backup battery, although there may be a relationship. For example, if the RTC backup battery is removed while main system power is still applied, this bit will still read back ’1b’ until main system power is cycled. This bit will always be set to 0 upon RTC reset. This bit will always be set to 1 after an initial read to this register is performed. This bit will remain set to 1 until an RTC reset occurs. RTC reset will occur anytime the BBATSEN input is sampled to be below 2.4 V during a power-on reset.
6–0	Reserved	**Reserved** These bits are reserved. Each of them always reads back as ’0b’.

Programming Notes

The default value for this register (index 0Dh) refers to the RTC-only reset. The RTC-only reset may or may not occur when a master power-on reset occurs.

CHAPTER

5 GRAPHICS CONTROLLER INDEXED REGISTERS

5.1 OVERVIEW

The registers described in this chapter function as configuration, setup, and status for the LCD graphics controller. They are listed in hexadecimal order by function in Table 5-2.

Graphics controller registers are indexed using I/O ports 3D4h (index) and 3D5h (data) for Color Graphics Adapter (CGA) mode and I/O ports 3B4h (index) and 3B5h (data) for Monochrome Display Adapter (MDA) mode. Different ports are used, depending on the graphics mode selected. The mode is selected using bit 0 of the Internal Graphics Control Register A (CSC index DDh).

- When MDA mode is selected, the MDA index and data registers are located at 3B4h and 3B5h, respectively, in the I/O address space.
- When CGA mode is selected, the CGA index and data registers are located at 3D4h and 3D5h, respectively, in the I/O address space.

The graphics controller indexed registers are accessed using a two-step process:

- An I/O write to the I/O address port for the chosen mode is performed. The data written is the index of the requested graphics controller register.
- This I/O write is followed by an I/O read or write to the data port for the chosen mode. This access causes the graphics controller to allow access to the addressed configuration register.

The graphics controller indexed registers are typically accessed at CPU speeds.

Table 5-2 Graphics Controller Register Map

Register Name	I/O (Port) Address	Index	Page Number
CGA/MDA Index	03x4h		page 5-4
CGA/MDA Data	03x5h		page 5-5
CGA/MDA Mode	03x8h		
Palette	03D9h		
Status	03xAh		
Clear LPS	03DBh		
Set LPS	03DCh		
HGA Configuration	03BFh		
Legacy CGA/MDA Group		**0A–11h**	
Cursor Start Register		0Ah	page 5-6
Cursor Stop Register		0Bh	page 5-7
Start Address High Register		0Ch	page 5-8

Register Name	I/O (Port) Address	Index	Page Number
Start Address Low Register		0Dh	page 5-9
Cursor Address High Register		0Eh	page 5-10
Cursor Address Low Register		0Fh	page 5-11
Light Pen High Register		10h	page 5-12
Light Pen Low Register		11h	page 5-13
Extended Group		**30–52h**	
Horizontal Total Register		30h	page 5-14
Horizontal Display End Register		31h	page 5-15
Horizontal Line Pulse Start Register		32h	page 5-16
Horizontal Border End Register		33h	page 5-17
Non-Display Lines Register		34h	page 5-18
Vertical Adjust Register		35h	page 5-19
Overflow Register		36h	page 5-20
Vertical Display End Register		37h	page 5-21
Vertical Border End Register		38h	page 5-22
Frame Sync Delay Register		39h	page 5-23
Reserved		3Ah	
Dual Scan Row Adjust Register		3Bh	page 5-24
Dual Scan Offset Address High Register		3Ch	page 5-25
Dual Scan Offset Address Low Register		3Dh	page 5-26
Offset Register		3Eh	page 5-27
Underline Location Register		3Fh	page 5-28
Maximum Scan Line Register		40h	page 5-29
LCD Panel AC Modulation Clock Control Register		41h	page 5-30
Font Table Register		42h	page 5-31
Grayscale Mode Register		43h	page 5-32
Grayscale Remap Register 1		44h	page 5-34
Grayscale Remap Register 2		45h	page 5-34
Grayscale Remap Register 3		46h	page 5-34
Grayscale Remap Register 4		47h	page 5-34
Grayscale Remap Register 5		48h	page 5-34
Grayscale Remap Register 6		49h	page 5-34
Grayscale Remap Register 7		4Ah	page 5-34
Grayscale Remap Register 8		4Bh	page 5-34
Pixel Clock Control Register		4Ch	page 5-36
Frame Buffer Base Address High Byte		4Dh	page 5-37

Register Name	I/O (Port) Address	Index	Page Number
Font Buffer Base Address High Byte		4Eh	page 5-38
Frame/Font Buffer Base Address Low		4Fh	page 5-39
PMU Control Register 1		50h	page 5-40
PMU Control Register 2		51h	page 5-41
Extended Feature Control Register		52h	page 5-42

5.2 REGISTER DESCRIPTIONS

Each graphics controller index register is described on the following pages. Additional information about using these registers to program the ÉlanSC400 microcontroller can be found in the *ÉlanSC400 User's Manual* (order #21030).

	7	6	5	4	3	2	1	0
Bit	GR_IDX[7–0]							
Default	0	0	0	0	0	0	0	0
R/W	W							

Bit	Name	Function
7–0	GR_IDX[7–0]	**Graphics Controller Index Register** Graphics controller indexed register to read/write. The direct mapped I/O address will be 3B4h if operating in Monochrome Display Adapter (MDA) mode, and 3D4h if operating in Color Graphics Adapter (CGA) mode.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	GR_DTA[7–0]							
Default	x	x	x	x	x	x	x	x
R/W	W							

Bit	Name	Function
7–0	GR_DTA[7–0]	**Graphics Controller Data Port** Data to be written or read to register selected via 3x4h. The direct mapped I/O address will be 3B5h if operating in Monochrome Display Adapter (MDA) mode, and 3D5h if operating in Color Graphics Adapter (CGA) mode.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved	CUR_CON[1–0]		CUR_START[4–0]				
Default	x	x	x	x	x	x	x	x
R/W		R/W		R/W				

Bit	Name	Function
7	Reserved	**Reserved**
6–5	CUR_CON[1–0]	**Cursor Control Register** Controls the alphanumeric cursor according to the following table:

Bits 6–5	Cursor Display Mode
0 0	Normal blinking cursor
0 1	No cursor display
1 0	No blinking, half intensity cursor
1 1	No blinking, full intensity cursor

Bit	Name	Function
4–0	CUR_START[4–0]	**Cursor Start Register** Defines the starting line of the alphanumeric cursor for bits 4–0. Compatible with the 6845 definition.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	CUR_END[4–0]			Reserved				
Default	x	x	x	x	x	x	x	x
R/W				R/W				

Bit	Name	Function
7–5	Reserved	**Reserved**
4–0	CUR_END[4–0]	**Cursor End Register Bits 4–0** Defines the last line of the alphanumeric cursor. Compatible with the 6845 definition.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	STRTADDR[15–8]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	STRTADDR[15–8]	**Data Start Address Register Bits 15–8** High-order bits of the starting address that determines the first data to be displayed at the top of the screen. In CGA mode, this register is compatible with the 6845 definition (most significant bit is bit 5). In Linear Packed-Pixel 1bpp mode, the register is extended by one bit, so that bit 6 becomes the most significant bit. In Linear Packed-Pixel 4 and 2bpp modes, the register is extended by two bits, so that bit 7 becomes the most significant bit.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	STRTADDR[7–0]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	STRTADDR[7–0]	**Data Start Address Register Bits 7–0** Low-order bits of the starting address that determines the first data to be displayed at the top of the screen. Compatible with the 6845 definition.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved		CUR_ADDR[13–8]					
Default	x	x	x	x	x	x	x	x
R/W			R/W					

Bit	Name	Function
7–6	Reserved	**Reserved**
5–0	CUR_ADDR[13–8]	**Cursor Address Location Bits 13–8** High-order bits of the address that determines the location of the alphanumeric cursor. Compatible with the 6845 definition.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	CUR_ADDR[7–0]							
Default	x	x	x	x	x	x	x	x
R/W	R/W							

Bit	Name	Function
7–0	CUR_ADDR[7–0]	**Cursor Address Location Bits 7–0** Low-order bits of the address that determines the location of the alphanumeric cursor. Compatible with the 6845 definition.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved		LIGHTPEN[13–8]					
Default	x	x	x	x	x	x	x	x
R/W			R					

Bit	Name	Function
7–6	Reserved	**Reserved**
5–0	LIGHTPEN[13–8]	**Light Pen Location Bits 13–8** This read-only register reads back as a mirror of the value in the Cursor address high register and is provided for CGA software compatibility.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	LIGHTPEN[7–0]							
Default	x	x	x	x	x	x	x	x
R/W	R							

Bit	Name	Function
7–0	LIGHTPEN[7–0]	**Light Pen Location Bits 7–0** This read-only register reads back as a mirror of the value in the Cursor address low register and is provided for CGA software compatibility.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved	HTOTAL[6–0]						
Default	x	x	x	x	x	x	x	x
R/W		R/W						

Bit	Name	Function
7	Reserved	**Reserved** Software should write this bit to 0.
6–0	HTOTAL[6–0]	**Horizontal Total Bits 6–0** The horizontal total minus 2. The horizontal total is the total number of character counts in a horizontal line, where a character is 8–16 pixels wide. Programming this register to a larger value than the Horizontal Display End register allows the frame rate to be slowed.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved	HDEND[6–0]						
Default	x	x	x	x	x	x	x	x
R/W		R/W						

Bit	Name	Function
7	Reserved	**Reserved** Software should write this bit to 0.
6–0	HDEND[6–0]	**Horizontal Display End Bits 6–0** The number of the last character position in a line which will be output from the frame buffer. Value = number of horizontal displayed characters from memory minus 1.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	SELHPW	HLPSTRT[6–0]						
Default	x	x	x	x	x	x	x	x
R/W	R/W	R/W						

Bit	Name	Function
7	SELHPW	**Select Horizontal Line Pulse Width** 0 = Horizontal line pulse width = 1 character clock 1 = Horizontal line pulse width = 2 character clocks
6–0	HLPSTRT[6–0]	**Horizontal Line Pulse Start Bits 6–0** This bit field determines the point at which the horizontal clocking signal (called LC on the chip's graphics controller electrical interface, also known as CP1 on some graphics panels) is sent out to the graphics relative to the pixel data for each scan line. This control is supplied so that the LC pulse location can be adjusted to meet the timing requirements of a wide variety of graphics panels. A typical value is the horizontal border end value plus 3. A maximum value is the horizontal total value plus 2.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved	HBRDEND[6–0]						
Default	x	x	x	x	x	x	x	x
R/W		R/W						

Bit	Name	Function
7	Reserved	**Reserved** Software should write this bit to 0.
6–0	HBRDEND[6–0]	**Horizontal Border End Bits 6–0** Determines the last character position to be displayed at the end of a line. Value = number of horizontal displayed character positions minus 1. All character positions from value programmed to end of horizontal total will not be sent as data to the LCD panel. If the Horizontal Display End register is programmed to a lower value than this register, the character positions between Horizontal Display End and this register will be sent to the panel as overscan values. This is especially useful when the display mode selected does not use the entire screen area.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved						NDISPADJ	
Default	x	x	x	x	x	x	x	x
R/W							R/W	

Bit	Name	Function
7–2	Reserved	**Reserved** Software should write this bit to 0.
1–0	NDISPADJ	**Non-display Adjust Bits 1–0** These bits allow 2 or 4 additional non-display lines to be added to the bottom of the vertical display sequence after vertical adjust rolls over. Data displayed during these lines comes from Internal Graphics Control Register B (ÉlanSC400 microcontroller configuration indexed register DEh), bit 0.

NDISPADJ1–NDISPADJ0:	Number of non-display lines added:
0 0	0
0 1	2
1 0	4
1 1	1

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved		VADJUST[5–0]					
Default	x	x	x	x	x	x	x	x
R/W			R/W					

Bit	Name	Function
7–6	Reserved	**Reserved** Software should write this bit to 0.
5–0	VADJUST[5–0]	**Vertical Adjust Bits 5–0** If the number of scan lines in the display panel is not an integer multiple of the number of scan lines in a character row, program this register to equal the number of excess lines (i.e., VADJUST = (Total_scan_lines) modulo (lines_in_a_character_row)). If the number of scan lines in the display panel is an integer multiple of the number of scan lines in a character row, set this register to 0.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved			FLMMODE	Reserved	VBRDEND	VDEND	Reserved
Default	x	x	x	x	x	x	x	x
R/W				R/W		R/W	R/W	

Bit	Name	Function
7–5	Reserved	**Reserved** Software should write this bit to 0.
4	FLMMODE	**FLM (Vertical Sync) Mode** 0 = FLM will be asserted at the end of the first scan line 1 = FLM will be asserted before the start of the first scan line
3	Reserved	**Reserved** Software should write this bit to 0.
2	VBRDEND[8]	**Vertical Border End Bit 8** This is the eighth bit of the Vertical Border End Register. See graphics index 38h for more detail.
1	VDEND[8]	**Vertical Display Enable End Bit 8** This is the eighth bit of the Vertical Display Enable End Register. See graphics index 37h for more detail.
0	Reserved	**Reserved** Software should write this bit to 0.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	VDEND							
Default	x	x	x	x	x	x	x	x
R/W	R/W							

Bit	Name	Function
7–0	VDEND[7–0]	**Vertical Display End Bits 7–0** Determines the number of the last character line to be output from the frame buffer at the bottom of the screen. VDEND8 is in the Overflow Register at graphics index 36h. Value = (number of horizontal displayed character lines from memory) -1.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	VBRDEND[7–0]							
Default	x	x	x	x	x	x	x	x
R/W	R/W							

Bit	Name	Function
7–0	VBRDEND[7–0]	**Vertical Border End Bits 7–0** Determines the last character line to be output to the panel at the bottom of the display. Value = (number of horizontal displayed character lines) -1. If the Vertical Display End register is programmed to a lower value than this register, the scan lines numbered between Vertical Display End and this register will be sent to the panel with the programmed overscan values. VBRDEND8 is in the Overflow Register at graphics index 36h. Useful when the display mode selected does not use the entire screen area.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved						FRMDLY[1–0]	
Default	x	x	x	x	x	x	x	x
R/W							R/W	

Bit	Name	Function
7–2	Reserved	**Reserved** Software should write this bit to 0.
1–0	FRMDLY[1–0]	**Frame Sync Delay Register Bits 1–0** Number of character clocks to delay Frame sync from beginning of horizontal line pulse. Delay = (value programmed * 4) + 1.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved		DSUPRWAJ[5–0]					
Default	x	x	x	x	x	x	x	x
R/W			R/W					

Bit	Name	Function
7–6	Reserved	**Reserved** Software should write this bit to 0.
5–0	DSUPRWAJ[5–0]	**Dual-Scan Row Adjust Register Bits 5–0** When a character row overlaps the upper and lower screens in a dual-scan display, this register must be programmed with the number of scan lines in the portion of the character row which is within the lower screen. If the number of scan lines in a half-screen is an integer multiple of the number of scan lines in a character row, program this register to 0. Has no effect in single-scan mode.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	DSOFFADR[15–8]							
Default	x	x	x	x	x	x	x	x
R/W	R/W							

Bit	Name	Function
7–0	DSOFFADR[15–8]	**Dual-Scan Offset Address Bits 15–8** Address offset between lower and upper screens of a dual-scan display panel, upper byte. No effect in single-scan mode.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	DSOFFADR[7–0]							
Default	x	x	x	x	x	x	x	x
R/W	R/W							

Bit	Name	Function
7–0	DSOFFADR[7–0]	**Dual-Scan Offset Address Bits 7–0** Address offset between lower and upper screens of a dual-scan display panel, lower byte. No effect in single-scan mode.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	OFFSET[7–0]							
Default	x	x	x	x	x	x	x	x
R/W	R/W							

Bit	Name	Function
7–0	OFFSET[7–0]	**Offset Register Bits 7–0** Specifies the width of the frame buffer. Address offset to the next lower vertically adjacent character is given by the value in this register * N, where N = 1 in 1bpp linear-packed pixel mode and 2 in all other modes. Width of frame buffer can be programmed to be larger than physical screen size.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved			ULINE[4–0]				
Default	x	x	x	x	x	x	x	x
R/W				R/W				

Bit	Name	Function
7–5	Reserved	**Reserved**
4–0	ULINE[4–0]	**Underline Location Bits 4–0** Specifies the scan line number of the underline attribute used in MDA modes. Has no effect in CGA or graphics modes.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved			MAXSCAN[4–0]				
Default	x	x	x	x	x	x	x	x
R/W				R/W				

Bit	Name	Function
7–5	Reserved	**Reserved**
4–0	MAXSCAN[4–0]	**Maximum Scan Line Register Bits 4–0** The number of lines in a character row of identically addressed horizontal lines minus 1. Similar to the 6845 definition.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	MOD	MLINES[6–0]						
Default	x	x	x	x	x	x	x	x
R/W	R/W	R/W						

Bit	Name	Function
7	MOD	**AC Modulation Mode** 0 = M changes phase as determined by bits 6–0 1 = M changes phase once per frame
6–0	MLINES[6–0]	**Horizontal Line Divide Ratio Bits 6–0** When bit 7 = 0, the M pin will change phase every N + 1 horizontal lines, where N = the value of these bits.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	FONTWRP	CHARWID[1–0]		FONTOFF[4–0]				
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W		R/W				

Bit	Name	Function
7	FONTWRP	**Font Table Write-protect** 0 = Do not write-protect font table 1 = Write-protect font table
6–5	CHARWID[1–0]	**Character Width Select** Character width in pixels is as follows: 0 0 = 8 pixel width 0 1 = 10 pixel width 1 0 = 16 pixel width
4–0	FONTOFF[4–0]	**Font Plane Offset Bits 4–0** Sets the offset into the font table during display. Useful in displaying alternate font sets.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	BORDER	COLORSTN	GRAYMAP MODE	GRAYCONT	SHAD4SEL[1–0]		GRAYMOD	GRYREMAP
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W		R/W	R/W

Bit	Name	Function
7	BORDER	**Enable Color Border** 0 = Bit 1 of Internal Graphics Control Register B (CSC index DEh) controls the border color in all modes. 1 = Bits 3–0 of 3D9 will be used to set the border color in CGA text modes and flat-mapped packed-pixel modes (bit 1 of Graphics Controller Mode Register 2 always controls the border shade in MDA and CGA graphics modes.).
6	COLORSTN	**Color STN Enable** 0 = Monochrome panel mode 1 = Color STN panel mode (must be single-scan panel)
5	GRAYMAPMODE	**Grayscale Mapping Mode** Should be set only when bit 1 = 0. See also bit 4 of this register. 0 = Color mapping mode 1 = Monochrome mapping mode
4	GRAYCONT	**Grayscale Contrast Enhance** Used when bit 1 = 0; see also bit 5 of this register. 0 = Normal contrast In color mapping mode (normal contrast), the R and G bits are used to select the output gray shade. In monochrome mapping mode, non-intensified characters are mapped to shades 0 and 3, intensified characters are mapped to shades 1 and 2. 1 = Enhanced contrast In color mapping mode (enhanced contrast), enables 16x2 gray shade palette (bit 0 must also be set). The least significant two bits of each of the gray shades remapping registers can be used to map any of the four gray shades to each of the 16 possible IRGB values (see the table below). In monochrome mapping mode, intensified characters are mapped to shades 0 and 3, non-intensified characters are mapped to shades 0 and 2. This bit is ignored when bit 1 = 1.
3–2	SHAD4SEL[1–0]	**Option 1 Shade Mode Select Bits 1–0** Selects between one of four 4-color shading modes used in option 1. See the Gray Shading section for a description. Ignored when option 2 is selected.
1	GRAYMOD	**Grayscale Mode** 0 = Option 1 (4 shades) 1 = Option 2 (16 shades)
0	GRYREMAP	**Grayscale Remapping Enable** 0 = Disable mapping registers 1 = Enable mapping registers

Programming Notes

Relationship of Horizontal/Vertical display and border signals to display output data				
Horizontal Display Enable	**Horizontal Border**	**Vertical Display Enable**	**Vertical Border**	**Output Data**
1	0	1	0	Normal display data
0	0	1	0	Overscan
x	0	0	0	Overscan
x	0	0	1	Non-display (Internal Graphics Control Register B, bit 0)
0	1	x	x	No data output

	7	6	5	4	3	2	1	0
Bit	GRAYMAPODD				GRAYMAPEVEN			
Default	x	x	x	x	x	x	x	x
R/W	R/W				R/W			

Bit	Name	Function
7–4	GRAYMAPODD	**Grayscale Remapping Registers (Odd-Numbered RGBI Codes)** The value programmed in each of these registers becomes the remapped grayscale code for the corresponding input grayscale code. The first register corresponds to input code 1, the second corresponds to input code 3, etc.
3–0	GRAYMAPEVEN	**Grayscale Remapping Registers (Even-Numbered RGBI Codes)** The value programmed in each of these registers becomes the remapped grayscale code for the corresponding input grayscale code. The first register corresponds to input code 0, the second corresponds to input code 2, etc.

Programming Notes

The graphics controller supports 16 or four gray levels of CGA text mode shades. To allow contrast adjustment between various panels, a Color Mapping function is provided to allow software adjustment of the mapping of color to the gray level displayed by the LCD controller. This color mapping is controlled by the Color Mapping Register(s). This function is enabled by graphics index 43h[0].There are eight color mapping registers, located at graphics indexes 44–4Bh. Each register contains two 4-bit nibbles, for a total of 16 nibbles. Each nibble controls the mapping of one of the 16 or four CGA gray levels; each gray level is defined by the 4-bit quantity: R, G, B, I. The mapping is shown in the chart below:

R	G	B	I	Register Index	Bit Field (16 Gray Shades)	Bit Field (4 Gray Shades)
0	0	0	0	44h	3–0	1–0
0	0	0	1	44h	7–4	5–4
0	0	1	0	45h	3–0	1–0
0	0	1	1	45h	7–4	5–4
0	1	0	0	46h	3–0	1–0
0	1	0	1	46h	7–4	5–4
0	1	1	0	47h	3–0	1–0
0	1	1	1	47h	7–4	5–4
1	0	0	0	48h	3–0	1–0
1	0	0	1	48h	7–4	5–4
1	0	1	0	49h	3–0	1–0
1	0	1	1	49h	7–4	5–4
1	1	0	0	4Ah	3–0	1–0
1	1	0	1	4Ah	7–4	5–4
1	1	1	0	4Bh	3–0	1–0
1	1	1	1	4Bh	7–4	5–4

CGA Example (16 gray-shades mode):

Write to Index Register 3D4:44h

Write to Data Register 3D5:EFh

CGA gray level RGBI (0000) will be displayed as gray level (1111)

CGA gray level RGBI (0001) will be displayed as gray level (1110)

CGA Example (4 gray-shades mode):

Write to Index Register 3D4:44h

Write to Data Register 3D5:23h

CGA gray level RGBI (0000) will be displayed as gray level (11)

CGA gray level RGBI (0001) will be displayed as gray level (10)

	7	6	5	4	3	2	1	0
Bit	Reserved			CLOCKDIV[1–0]		PLLRATIO[2–0]		
Default	x	x	x	0	0	0	0	0
R/W				R/W		R/W		

Bit	Name	Function
7–5	Reserved	**Reserved**
4–3	CLOCKDIV[1–0]	**Dot Clock Divide Select** Divides the base dot clock (as controlled by bits 2–0 above) by 1, 2, or 4 to generate the dot clock rate that is used by the graphics controller as shown in the table below.
2–0	PLLRATIO[2–0]	**Dot Clock Base Frequency Select** 0 0 0 = 20.736 MHz 0 0 1 = 23.040 MHz 0 1 0 = 25.344 MHz 0 1 1 = 27.648 MHz 1 0 0 = 29.952 MHz 1 0 1 = 32.256 MHz 1 1 0 = 34.560 MHz 1 1 1 = 36.864 MHz These bits are used in conjunction with bits 4–3 to determine the dot clock (also known as the pixel clock) rate as shown in the table below.

Programming Notes

Bits 2–0	Dot Clock Frequency (MHz) Bits 4–3		
	0 0:Divide by 4	**0 1:Divide by 2**	**1 0:Divide by 1**
0 0 0	5.19	10.38	20.76
0 0 1	5.77	11.53	23.06
0 1 0	6.34	12.68	25.36
0 1 1	6.92	13.84	27.68
1 0 0	7.50	14.99	29.98
1 0 1	8.07	16.14	32.28
1 1 0	8.65	17.30	34.60
1 1 1	9.23	18.45	36.90

	7	6	5	4	3	2	1	0
Bit	FRMBUFWIN[23–16]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	FRMBUFWIN[23–16]	**Frame Buffer Window Base Address Bits 23–16** Sets the base address at which the frame buffer is visible within a 16 Mbytes shared memory address space (high order bits, see also graphics index 4Fh[1–0]).

Programming Notes

	7	6	5	4	3	2	1	0
Bit	FONTWIN[23–16]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	FONTWIN[23–16]	**Font Buffer Window Base Address Bits 23–16** Sets the base address at which the font buffer is visible within a 16 Mbytes shared memory address space (high order bits, see also graphics index 4Fh[3–2]). Note that the font buffer and associated font buffer configuration controls have meaning only in text mode.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	VID_DRAM	LCD_FRM_BUF_WIN_ENA	PAGESEL[1–0]		FONTWIN[15–14]		FRMBUFWIN[15–14]	
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W		R/W		R/W	

Bit	Name	Function
7	VID_DRAM	**Video DRAM** 0 = DRAM is not allocated to the graphics controller 1 = Enable address spaces specified by FRMBUFWIN, FONTWIN, and memory size as DRAM allocated to the graphics controller
6	LCD_FRM_BUF_WIN_ENA	**LCD Frame Buffer Window** 0 = Graphics controller MMS window is disabled 1 = Enable the 16 Kbytes graphics frame buffer MMS window which is fixed in system address space from 0B8000–0BBFFFh This MMS window not available when the graphics controller is configured for text mode. This window provides access to one of four contiguous 16 Kbytes windows of system DRAM only. The 64 Kbytes block of system DRAM accessible via this window starts at the base address which is programmed via index 4Dh[7–0]. The window (C–F) visible through the window is selected using bits 5–4 below.
5–4	PAGESEL[1–0]	**Page Select Bits 1–0** Select one of four contiguous 16 Kbytes MMS windows to become visible at address 0B8000–0BBFFFh. See bit 6 of this register.
3–2	FONTWIN[15–14]	**Font Buffer Window Base Address Bits 15–14** Sets the base address at which the font buffer is visible within a 16 Mbytes shared memory address space (low order bits).
1–0	FRMBUFWIN[15–14]	**Frame Buffer Window Base Address Bits 15–14** Sets the base address at which the frame buffer is visible within a 16 Mbytes shared memory address space (low order bits). If bit 6 of this register is set, these two bits must be written to ‘00b’. If the frame buffer is larger than 16 Kbytes (such as when the graphics controller is configured for a linear packed pixel mode), bit 0 of this register must be written to ‘0b’.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	LCDPWREN	PMUMODE	PWRUPDLY2[2–0]			PWRUPDLY1[2–0]		
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W			R/W		

Bit	Name	Function
7	LCDPWREN	**LCD Software Power-up and Down** 0 = Disable LCD power and signals (sequencing time controlled by bits 5–0) 1 = Enable LCD power and signals (sequencing time controlled by bits 5–0) This bit, when set, will be overridden by a PMU initiated disable of the LCD (see bit 6 of this register). This bit, when cleared, will override LCD enable control by the PMU (see bit 6 of this register).
6	PMUMODE	**External PMU Control Enable** 0 = Do not allow PMU to disable power to LCD (bit 7 can still be used) 1 = Allow PMU to disable/enable power to LCD (sequencing time controlled by bits 5–0, except during emergency power-down) If the internal graphics controller is enabled (CSC index register DDh[2]='1b'), and the PMU is enabled to shut down the LCD in standby mode (CSC index register 40h[5]='1b'), this bit (graphics index 50h[6]) must also be set.
5–3	PWRUPDLY2[2–0]	**Power-up Power Sequencing Delay 2** Programs the delay time between the sequential events of LCD panel data/control signal activation and VEE activation during power-up. The actual delay time value equals (N + 1)*7.81 ms, where N is the value programmed into bits 5–3.
2–0	PWRUPDLY1[2–0]	**Power-up Power Sequencing Delay 1** Programs the delay time between the sequential events of VDD activation and LCD panel data/control signal activation during power-up. The actual delay time value equals (N + 1)*7.81 ms, where N is the value programmed into bits 2–0.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	Reserved		PWRDWNDLY2[2–0]			PWRDWNDLY1[2–0]		
Default	x	x	0	0	0	0	0	0
R/W			R/W			R/W		

Bit	Name	Function
7–6	Reserved	**Reserved**
5–3	PWRDWNDLY2[2–0]	**Power-down Power Sequencing Delay 2** Programs the delay time between the sequential events of LCD panel data/control signal deactivation and VDD deactivation during normal power-down. The actual delay time value equals (N + 1)*62.5ms, where N is the value programmed into bits 5–3.
2–0	PWRDWNDLY1[2–0]	**Power-down Power Sequencing Delay 1** Programs the delay time between the sequential events of VEE deactivation and LCD panel data/control signal deactivation during normal power-down. The actual delay time value equals (N + 1)*62.5 ms, where N is the value programmed into bits 2–0.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	HIDDEN_ FLUSH	PGM1_E_RB	HGAGR_E_RB	HGA_EN	FCURSOR	FRAME_ DLY_DIS	LEG_TRAP_ EN	RGBEN
Default	0	0	0	0	0	0	0	0
R/W	R/W	R	R	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	HIDDEN_FLUSH	**Graphics Controller FIFO Hidden Flush Enable** 0 = Graphics controller will perform FIFO flush/reload at the end of the last scan line, whether it is displayed or not 1 = Graphics controller will perform FIFO flush/reload at the beginning of the last non-displayed scan line When setting this bit, bit 2 of the Extended Feature Control Register must also be set. In this case, the Horizontal Total Register (graphics index 30h) and Horizontal Border End Register (graphics index 33h) can be programmed to the same value. In addition, the Non-Display Lines Register (graphics index 34h) must be programmed to a non-zero value (1, 2 or 3). When this option is used, there is no need for any additional delay at the end of a horizontal scan line for FIFO flush/reloads.
6	PGM1_E_RB	**Page Memory Enable Readback** This bit is the readback for the write-only PG_MEM1_EN bit that is located in the HGA Configuration Register at direct mapped register 03BFh[1]. This bit is valid only when the Comp_Mod bit in the Internal Graphics Control Register A (CSC index register DDh[0]) is set.
5	HGAGR_E_RB	**HGA Readback** This bit is the readback for the write-only HGA_GR_EN bit that is located in the HGA Configuration Register at direct mapped register 03BFh[0]. This bit is valid only when the Comp_Mod bit in the Internal Graphics Control Register A (CSC index register DDh[0]) is set.
4	HGA_EN	**Enable HGA Register Extensions in MDA Mode** This bit has effect only when the Comp_Mod bit in the Internal Graphics Control Register A (CSC index register DDh[0]) is set. 0 = Disable HGA register extensions in MDA mode 1 = Enable HGA register extension in MDS mode
3	FCURSOR	**Normal Cursor Blink Rate Control** 0 = 1 Hz 1 = 2 Hz
2	FRAME_DLY_DIS	**Inter-Frame FIFO Flush/refill Delay Disable** 0 =The graphics controller inserts a delay between the last LCD panel horizontal line pulse signal at the end of a frame, and before shifting data out for the first line of the next frame to allow for a graphics controller FIFO flush/refill 1 =Automatic FIFO flush/refill delay disabled

Bit	Name	Function
1	LEG_TRAP_EN	**CGA Legacy I/O Trap SMI/NMI Generation Enable** 0 = Disables I/O Trap SMI/NMI generation for accesses to the CGA/MDA legacy registers [Cursor Start (index 0Ah), Cursor End (index 0Bh), Start Address High (index 0Ch), Start Address Low (index 0Dh), Cursor Address High (index 0Eh), Cursor Address Low (index 0Fh), Light Pen High (index 10h), and Light Pen Low (index 11h)] when the internal graphics I/O Trap SMI/NMI generation feature is enabled via setting bit 3 of the graphics controller index register at index location 99h 1 = Enables I/O Trap SMI/NMI generation for accesses to the CGA/ MDA legacy registers [Cursor Start (index 0Ah), Cursor End (index 0Bh), Start Address High (index 0Ch), Start Address Low (index 0Dh), Cursor Address High (index 0Eh), Cursor Address Low (index 0Fh), Light Pen High (index 10h), and Light Pen Low (index 11h) when the internal graphics I/O Trap SMI/NMI generation feature is enabled via setting bit 3 of the graphics controller index register at index location 99h
0	RGBEN	**Switch to RGBI Output** 0 = Normal operation 1 = Low order grayscale bits switched to RGBI, data shift clock becomes pixel clock, M becomes display enable

Programming Notes

Bit 2: When graphics index 4Ch[2–0]←'011b', the delay = 202*(1/dot clock base frequency). When 4Ch[2–0] > '011b', the delay = 256*(1/dot clock base frequency). The delay range is 6.93–9.73 µs.
The "character time" depends upon the dot clock frequency as programmed via graphics index 4Ch, the character width as programmed via graphics index 42h, the horizontal dot doubling enable as programmed via the Internal Graphics Control register A (CSC index DDh[5]), and whether or not a dual scan panel is used as configured via CSC index DDh[3]. If this bit is set, and hidden flush is not enabled via bit 7 of this register, the time defined by the Horizontal Total register setting, the Horizontal Border End register setting must be made long enough to allow the FIFO flush/refill to complete (happens once per frame).

```
Character clock period (character time) = (dot clock period) * (character
width) * (horizontal doubling) * (dual scan)
```

where:

dot clock period = 1/dot clock frequency,

character width = 8, 10, or 16,

horizontal doubling = 2 if doubling enabled, 1 otherwise, and

dual scan = 2 if dual scan panel, 1 otherwise.

The time required for this depends on the mode and panel type as shown below:

Mode	Panel	Flush Delay (Minimum)
Graphics	Single-scan	2.7 µs
Text	Single-scan	3.9 µs
Graphics	Dual-scan	3.9 µs
Text	Dual-scan	5.8 µs

CHAPTER

6 PC CARD CONTROLLER INDEXED REGISTERS

6.1 OVERVIEW

The registers described in this chapter function as configuration, control, and status for the PC Card controller. They are listed in hexadecimal order by function in Table 6-1.

PC Card controller registers are indexed using I/O ports 3E0h (address) and 3E1h (data).

The PC Card controller indexed registers are accessed using a two-step process:

- An I/O write to I/O address 3E0h is first performed. The data written is the actual index address of the PC Card controller index register.
- This I/O write is followed by an I/O read or write to address 3E1h. This access causes the PC Card controller to allow access to the addressed configuration register. For example:

```
mov DX, 3E0h
mov al, index
out DX, al
inc DX
mov al, new data
out DX, al
```

The PC Card controller registers are typically accessed at CPU speeds.

Table 6-1 PC Card Controller Register Map

Register Name	I/O (Port) Address	Index	Page Number
PC Card Index Register	03E0h		page 6-5
PC Card Data Port	03E1h		page 6-6
General Control Group (Socket A)		**00–05h**	
Identification and Revision Register		00h	page 6-7
Interface Status Register		01h	page 6-8
Power and RESETDRV Control Register		02h	page 6-9
Interrupt and General Control Register		03h	page 6-11
Card Status Change Register		04h	page 6-12
Card Status Change Interrupt Configuration Register		05h	page 6-13
Address Window Enable Register		06h	page 6-15
I/O Window Mapping Group (Socket A)		**07–0Fh**	
I/O Window Control Register		07h	page 6-16
I/O Window 0 Start Address Low Register		08h	page 6-17

Register Name	I/O (Port) Address	Index	Page Number
I/O Window 0 Start Address High Register		09h	page 6-18
I/O Window 0 Stop Address Low Register		0Ah	page 6-19
I/O Window 0 Stop Address High Register		0Bh	page 6-20
I/O Window 1 Start Address Low Register		0Ch	page 6-21
I/O Window 1 Start Address High Register		0Dh	page 6-22
I/O Window 1 Stop Address Low Register		0Eh	page 6-23
I/O Window 1 Stop Address High Register		0Fh	page 6-24
Memory Window Mapping Group (Socket A)		**10–35h**	
Memory Window 0 Start Address Low Register		10h	page 6-25
Memory Window 0 Start Address High Register		11h	page 6-26
Memory Window 0 Stop Address Low Register		12h	page 6-27
Memory Window 0 Stop Address High Register		13h	page 6-28
Memory Window 0 Address Offset Low Register		14h	page 6-29
Memory Window 0 Address Offset High Register		15h	page 6-32
Memory Window 1 Start Address Low Register		18h	page 6-31
Memory Window 1 Start Address High Register		19h	page 6-32
Memory Window 1 Stop Address Low Register		1Ah	page 6-33
Memory Window 1 Stop Address High Register		1Bh	page 6-34
Memory Window 1 Address Offset Low Register		1Ch	page 6-35
Memory Window 1 Address Offset High Register		1Dh	page 6-36
Memory Window 2 Start Address Low Register		20h	page 6-37
Memory Window 2 Start Address High Register		21h	page 6-38
Memory Window 2 Stop Address Low Register		22h	page 6-39
Memory Window 2 Stop Address High Register		23h	page 6-40
Memory Window 2 Address Offset Low Register		24h	page 6-41
Memory Window 2 Address Offset High Register		25h	page 6-42
Memory Window 3 Start Address Low Register		28h	page 6-43
Memory Window 3 Start Address High Register		29h	page 6-44
Memory Window 3 Stop Address Low Register		2Ah	page 6-45
Memory Window 3 Stop Address High Register		2Bh	page 6-46
Memory Window 3 Address Offset Low Register		2Ch	page 6-47
Memory Window 3 Address Offset High Register		2Dh	page 6-54
Memory Window 4 Start Address Low Register		30h	page 6-49
Memory Window 4 Start Address High Register		31h	page 6-50
Memory Window 4 Stop Address Low Register		32h	page 6-51

Register Name	I/O (Port) Address	Index	Page Number
Memory Window 4 Stop Address High Register		33h	page 6-52
Memory Window 4 Address Offset Low Register		34h	page 6-53
Memory Window 4 Address Offset High Register		35h	page 6-54
Timing Control Group		3A–3Fh	
Setup Timing 0 Register		3AH	page 6-55
Command Timing 0 Register		3Bh	page 6-56
Recovery Timing 0 Register		3Ch	page 6-57
Setup Timing 1 Register		3DH	page 6-58
Command Timing 1 Register		3Eh	page 6-59
Recovery Timing 1 Register		3Fh	page 6-60
General Control Group (Socket B)		40–05h	
Identification and Revision Register		40h	page 6-7
Interface Status Register		41h	page 6-8
Power and RESETDRV Control Register		42h	page 6-9
Interrupt and General Control Register		43h	page 6-11
Card Status Change Register		44h	page 6-12
Card Status Change Interrupt Configuration Register		45h	page 6-13
Address Window Enable Register		46h	page 6-15
I/O Window Mapping Group (Socket B)		**47–4Fh**	
I/O Window Control Register		47h	page 6-16
I/O Window 0 Start Address Low Register		48h	page 6-17
I/O Window 0 Start Address High Register		49h	page 6-18
I/O Window 0 Stop Address Low Register		4Ah	page 6-19
I/O Window 0 Stop Address High Register		4Bh	page 6-20
I/O Window 1 Start Address Low Register		4Ch	page 6-21
I/O Window 1 Start Address High Register		4Dh	page 6-22
I/O Window 1 Stop Address Low Register		4Eh	page 6-23
I/O Window 1 Stop Address High Register		4Fh	page 6-24
Memory Window Mapping Group (Socket B)		**50–75h**	
Memory Window 0 Start Address Low Register		50h	page 6-25
Memory Window 0 Start Address High Register		51h	page 6-26
Memory Window 0 Stop Address Low Register		52h	page 6-27
Memory Window 0 Stop Address High Register		53h	page 6-28
Memory Window 0 Address Offset Low Register		54h	page 6-29

Register Name	I/O (Port) Address	Index	Page Number
Memory Window 0 Address Offset High Register		55h	page 6-32
Memory Window 1 Start Address Low Register		58h	page 6-31
Memory Window 1 Start Address High Register		59h	page 6-32
Memory Window 1 Stop Address Low Register		5Ah	page 6-33
Memory Window 1 Stop Address High Register		5Bh	page 6-34
Memory Window 1 Address Offset Low Register		5Ch	page 6-35
Memory Window 1 Address Offset High Register		5Dh	page 6-36
Memory Window 2 Start Address Low Register		60h	page 6-37
Memory Window 2 Start Address High Register		61h	page 6-38
Memory Window 2 Stop Address Low Register		62h	page 6-39
Memory Window 2 Stop Address High Register		63h	page 6-40
Memory Window 2 Address Offset Low Register		64h	page 6-41
Memory Window 2 Address Offset High Register		65h	page 6-42
Memory Window 3 Start Address Low Register		68h	page 6-43
Memory Window 3 Start Address High Register		69h	page 6-44
Memory Window 3 Stop Address Low Register		6Ah	page 6-45
Memory Window 3 Stop Address High Register		6Bh	page 6-46
Memory Window 3 Address Offset Low Register		6Ch	page 6-47
Memory Window 3 Address Offset High Register		6Dh	page 6-54
Memory Window 4 Start Address Low Register		70h	page 6-49
Memory Window 4 Start Address High Register		71h	page 6-50
Memory Window 4 Stop Address Low Register		72h	page 6-51
Memory Window 4 Stop Address High Register		73h	page 6-52
Memory Window 4 Address Offset Low Register		74h	page 6-53
Memory Window 4 Address Offset High Register		75h	page 6-54
Timing Control Group		**7A–7Fh**	
Setup Timing 2 Register		7Ah	page 6-55
Command Timing 2 Register		7Bh	page 6-56
Recovery Timing 2 Register		7Ch	page 6-57
Setup Timing 3 Register		7Dh	page 6-58
Command Timing 3 Register		7Eh	page 6-59
Recovery Timing 3 Register		7Fh	page 6-60

6.2 REGISTER DESCRIPTIONS

Each PC Card controller index register is described on the following pages. Additional information about using these registers to program the ÉlanSC400 microcontroller can be found in the *ÉlanSC400 User's Manual* (order #21030).

Primary 82365-Compatible PC Card Controller Index Register

I/O Address 03E0h

	7	6	5	4	3	2	1	0
Bit	PCM_IDX[7–0]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	PCM_IDX[7–0]	**PC Card Controller Index Register** Write the offset of the 82365-compatible index register to access to this port.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	PCM_DATA[7–0]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	PCM_DATA[7–0]	**PC Card Controller Data Port** Access 82365-compatible configuration register values (settings) via this port.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	INFC_ID[1–0]		Reserved		REVISION[3–0]			
Bit	PCC_REV[7–0]							
Default	1	0	0	0	0	0	1	0
R/W	R		R		R			
R/W	R							

Bit	Name	Function
7–6	INFC_ID[1–0]	**Interface ID** These bits identify PC Card interface types supported by the PC Card controller. They read back 10b.
5–4	RESERVED	**Reserved** These bits read back 0.
3–0	REVISION[3–0]	**Revision Level** The initial revision of the PC Card controller will be 0010.
7–0	PCC_REV[7–0]	**PC Card Controller Revision** This is a remapping of the REVISION and INFC_ID bits for identification of this device-specific revision level. These bits will read back F1h only immediately after a write to this register (the data written is not relevant.) If any other read or write precedes a read of this register, the normal REVISION and INFC_ID bits will be read back as 82h.

Programming Notes

All register bit descriptions that are shaded deviate from strict 83865SL implementation.

	7	6	5	4	3	2	1	0
Bit	Scratch Bit	PWR_ACT	RDY_BUSY	WR_PROT	CARD_DET2	CARD_DET1	BVD2–BVD1	
Default	0	0	x	x	x	x	x	x
R/W	R/W	R	R	R	R	R	R	

Bit	Name	Function
7	Scratch Bit	**Scratch Bit** In the original 82365SL design, this bit was the $\overline{GPI}$ status. This bit was used to sample the opposite state of the $\overline{GPI}$ input pin. The intention was to use this input to indicate valid VPP level or other miscellaneous function. In this implementation the I/O pin is not present and therefore this bit is a general read/write bit with no specific hardware status functionality.
6	PWR_ACT	**Slot Power Status** This bit indicates the power status of the socket. If 0, power to the socket is disabled. If this bit is set, power to the socket is enabled. See the table below to see the interaction this bit has with relation to the power control functions of index register 01h and 42h.
5	RDY_BUSY	**Ready/$\overline{Busy}$ Signal Status** This bit reflects the state of the $\overline{RDY\_x}$ pin of the PC Card.
4	WR_PROT	**Memory Write Protect Signal Status** This bit reflects the state of the $\overline{WP\_x}$ pin of the PC Card.
3	CARD_DET2	**Card Detect 2 Signal Status** This bit will always read back the same value as bit 2.
2	CARD_DET1	**Card Detect 1 Signal Status** This bit at index 01h reflects the inverted state of the $\overline{CD\_x}$ of the PC Card for Socket A. This bit at index 41h reflects the inverted state of the $\overline{CD\_x}$ pin of the PC Card for Socket B when Socket B is enabled. If the Socket A interface is configured to support both CD inputs (i.e., $\overline{CD\_A}$ and $\overline{CD\_A2}$) via the pin configuration option, this bit will reflect the logical anding of these inputs. In this configuration, a value of one indicates that both inputs are asserted. A value of zero indicates that these inputs are not both asserted.
1–0	BVD2–BVD1	**Battery Voltage Detect 2 and 1 (Respectively) Signal Status** When the socket is configured for the memory-only interface, this bit is used to detect the state of the $\overline{BVD2\_x}$ and $\overline{BVD1\_x}$ pins. These bits are used to determine the state of the card battery. When the socket is configured for the I/O and memory interface these bits are used to detect the state of the $\overline{BVD2\_x}$ and $\overline{BVD1\_x}$ pins, which function as the card's $\overline{SPKR}$ and $\overline{STSCHG/RI}$ pins. 0 0 = Battery Dead 0 1 = Battery Dead 1 0 = Warning 1 1 = Battery Good

Programming Notes

All register bit descriptions that are shaded deviate from strict 83865SL implementation.

	7	6	5	4	3	2	1	0
Bit	Unused Bit	Unused Bit	AUTO_PWR_ENX	VCC_ENX	Unused Bit	Unused Bit	VPP1X_CNT_1	VPP1X_CNT_0
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	Unused Bit	**Unused Bit** In the original 82365SL implementation, this bit is output enable. This bit was used, in conjunction with the Card Detect inputs, to three-state or enable the socket interface signals and to activate the enable signal for the external buffers (address and data). This implementation does not support an 82365SL compatible enable signal for socket address and data buffers, and this bit is therefore not supported. A buffered PC Card solution can be implemented using one of the GPIO pins as an output to enable or disable the buffers that could be used to isolate a PC Card from the system address and data bus.
6	Unused Bit	**Unused Bit** In the original 82365SL implementation, this bit is Disable RESETDRV. This bit was used to disable the reset of some of the controller's internal configuration registers when an ISA bus RESETDRV pulse was generated due to a RESUME operation. This feature was required when working with particular system chip sets. It is not required for this implementation and will be fully read/writable while not affecting any other hardware. This implementation is compatible with the CL-PD6720.
5	AUTO_PWR_ENX	**Auto Power Enable** This bit in conjunction with bits 0, 1, and 4 are used to control the PCMx_VCC signal if this pin functionality is selected. This output is used to control the socket VCC supply. This bit is used to allow the card detect inputs to automatically cause power to be applied to the socket interface. "VCC Control Signal Definition" on page 6-10.
4	VCC_ENX	**Socket VCC Enable** This bit in conjunction with bits 0, 1, and 5 are used to control the PCMx_VCC signal if this pin functionality is selected. This output is used to control the socket VCC supply. See the following table.
3	Unused Bit	**Unused Bit** In the original 82365SL implementation, this bit is Vpp2 control Bit 1. In this implementation, this bit is fully read writable but does not have any other hardware function. In the original 82365SL controller, an independent control for the two VPP signals of a single PC Card socket existed. Since most designs physically tie together the two VPP pins at the socket interface, this mechanism has been deleted from the controller. This VPP control implementation is consistent with the CL-PD6720.
2	Unused Bit	**Unused Bit** In the original 82365SL implementation, this bit is Vpp2 control Bit 0. In this implementation, this bit is fully read writable but does not have any other hardware function. In the original 82365SL controller, an independent control for the two VPP signals of a single PC Card socket existed. Since most designs physically tie together the two VPP pins at the socket interface, this mechanism has been deleted from the controller. This VPP control implementation is consistent with the CL-PD6720.

Bit	Name	Function
1	VPP1X_CNT_1	**VPP1 Control** This bit, in conjunction with bits 0 and 4 of this register controlled the PCMx_VPP1 and PCMx_VPP2 output signals if this pin functionality is selected. These signals are used to switch the socket Vpp supply between 0, 5, and 12 Volts. See "VPP Control Signal Definition" on page 6-10.
0	VPP1X_CNT_0	**VPP1 Control** This bit, in conjunction with bits 1 and 4 of this register control the PCMx_VPP1 and PCMx_VPP2 output signals if this pin functionality is selected. These signals are used to switch the socket Vpp supply between 0, 5, and 12 Volts. See "VCC Control Signal Definition" on page 6-10.

Programming Notes

All register bit descriptions that are shaded deviate from strict 83865SL implementation.

VPP Control Signal Definition

Bit 4 V_{cc} Power	Bit 1 V_{pp} Control Bit 1	Bit 2 V_{pp} Control Bit 0	PCMX_VPP2	PCMX_VPP1	PCMX_VCC	Comments
1	0	0	0	0	0	V_{pp} is N/C V_{CC} enabled
1	0	1	0	1	0	V_{pp} = VCC V_{CC} enabled
1	1	0	1	0	0	V_{pp} = +12 V V_{CC} enabled
1	1	1	0	0	0	V_{pp} is N/C V_{CC} enabled
0	X	X	0	0	1	V_{pp} is N/C V_{CC} disabled

VCC Control Signal Definition

Socket $\overline{CD\_X}$ Low	Bit 5 Auto Power Control	Bit 4 Vcc Power	PCMX_VCC	PC Card Power Active (Bit 6 of Interface Status Register)	Comments
X	X	0	1	0	Socket forced off
X	0	1	0	1	Socket forced on
No	1	1	1	0	Auto power enabled and no card inserted
Yes	1	1	0	1	Auto power enabled and card inserted/powered

	7	6	5	4	3	2	1	0
Bit	RI_EN	CARD_RST	CARD_IS_IO	CSC_INT_DEST	IRQ[3–0]			
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W			

Bit	Name	Function
7	RI_EN	**Ring Indicate Enable** This bit has no function when the CARD_IS_IO bit in the Interrupt and General Control Register is configured for the memory-only interface. When the CARD_IS_IO bit is configured for the I/O and memory interface, RI_EN has the following meaning: 0 = The BVD1_X ($\overline{\text{STSCHG\_x}}$) signal is used as a status change indication 1 = The BVD1_X ($\overline{\text{STSCHG\_x}}$) signal is used as a ring indication signal and is routed to the PC Card Ring Indicate PMU input signal This disables use of BVD1_X (STSCHG_X) as a status change indication (see the BATT_DEAD bit in the Card Status Change Register).
6	CARD_RST	**PC Card Reset** 0 = RST_X is active 1 = RST_X is inactive
5	CARD_IS_IO	**PC Card Is I/O** 0 = Socket configured for the memory-only interface 1 = Socket configured for I/O and memory interface
4	CSC_INT_DEST	**Card Status Change Interrupt Destination** Controls the destination that the card status change is routed to: an IRQ or the PMU's PC Card Status Change interrupt input. The use of the PMU's PC Card Status Change interrupt input for NMI/SMI generation is controlled in the PMU block. 0 = The card status change interrupt signal is routed to the IRQ encoded in CSC_IRQ3–CSC_IRQ0 in the Card Status Change Interrupt Configuration Register (CSC_IRQ3–CSC_IRQ0 can also be encoded to disable generation of a card status change interrupt on any IRQ) 1 = The card status change interrupt is routed to the PMU's PC Card Status Change interrupt input
3–0	IRQ[3–0]	**PC Card IRQ Routing** Routes the PC Card IRQ to the appropriate PIC IRQ input as follows: 0 0 0 0 = Disabled 0 0 0 1 = Reserved 0 0 1 0 = Reserved 0 0 1 1 = IRQ3 enabled 0 1 0 0 = IRQ4 enabled 0 1 0 1 = IRQ5 enabled 0 1 1 0 = IRQ6 enabled 0 1 1 0 = IRQ6 enabled 0 1 1 1 = IRQ7 enabled 1 0 0 0 = Reserved 1 0 0 1 = IRQ9 enabled 1 0 1 0 = IRQ10 enabled 1 0 1 1 = IRQ11 enabled 1 1 0 0 = IRQ12 enabled 1 1 0 1 = Reserved 1 1 1 0 = IRQ14 enabled 1 1 1 1 = IRQ15 enabled

	7	6	5	4	3	2	1	0
Bit	Reserved				CD_CHNG	RDY_CHNG	BATT_WARN	BATT_DEAD
Default	0	0	0	0	0	0	0	0
R/W	R				R	R	R	R

Bit	Name	Function
7–4	RESERVED	**Reserved** Always reads back 0.
3	CD_CHNG	**Card Detect Change Detected** This bit is reset to 0 when this register is read. 0 = A transition (low to high or high to low) has not occurred on the $\overline{\text{CD\_X}}$ pin since the last time this register was read 1 = A transition (low to high or high to low) has occurred
2	RDY_CHNG	**Ready Change Detected** This bit is reset to 0 when this register is read. This bit has meaning only when the socket is configured for the memory-only interface. In the I/O and memory interface it reads back a 0. 0 = A transition (low to high or high to low) has not occurred on the RDY_X ($\overline{\text{IREQ\_X}}$) pin since the last time this register was read 1 = A transition (low to high or high to low) has occurred
1	BATT_WARN	**Battery Warning Detected** This bit is reset to 0 when this register is read. This bit has meaning only when the socket is configured for the memory-only interface. In the I/O and memory interface it reads back a 0. 0 = A transition (high to low) has not occurred on the BVD2_X pin since the last time this register was read 1 = A transition (high to low) has occurred
0	BATT_DEAD	**Battery Dead Change or Status Change Detected** This bit is reset to 0 when this register is read. 0 = A transition (high to low) has not occurred on the BVD1_X ($\overline{\text{STSCHG\_X}}$) pin since the last time this register was read 1 = A transition (high to low) has occurred When the socket is configured for the memory-only interface, this bit indicates a Battery Dead condition has been asserted on the BVD1_X signal if this bit is set. When the socket is configured for the I/O and memory interface, the state of this bit is determined by the state of the RI_EN bit in the Interrupt and General Control Register and the state of the BVD1_X/STSCHG_X pin. If the RI_EN bit is 0, then the BATT_DEAD bit is set to 1 when the STSCHG_X pin is asserted. If the RI_EN bit is set to 1, then the BATT_DEAD bit will always reads back 0.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	CSC_IRQ[3–0]				CD_EN	RDY_EN	BATT_ WARN_EN	BATT_ DEAD_EN
Default	0	0	0	0	0	0	0	0
R/W	R/W				R/W	R/W	R/W	R/W

Bit	Name	Function
7–4	CSC_IRQ[3–0]	**Card Status Change IRQ Routing** Enables and selects the IRQ line to issue the PC Card status change interrupt on. The state of the CSC_INT_DEST bit in Interrupt and General Control Register determines how CSC_IRQ3–CSC_IRQ0 is interpreted. If CSC_INT_DEST is set, no IRQ line is used; if it is cleared, the IRQ line is selected according to the following CSC_IRQ3–CSC_IRQ0 table: 0 0 0 0 = Disabled 0 0 0 1 = Reserved 0 0 1 0 = Reserved 0 0 1 1 = IRQ3 enabled 0 1 0 0 = IRQ4 enabled 0 1 0 1 = IRQ5 enabled 0 1 1 0 = Reserved 0 1 1 1 = IRQ7 enabled 1 0 0 0 = Reserved 1 0 0 1 = IRQ9 enabled 1 0 1 0 = IRQ10 enabled 1 0 1 1 = IRQ11 enabled 1 1 0 0 = IRQ12 enabled 1 1 0 1 = Reserved 1 1 1 0 = IRQ14 enabled 1 1 1 1 = IRQ15 enabled
3	CD_EN	**Card Detect IRQ Enable** Enables CD_CHNG (bit 3) of the Card Status Change Register (PC Card index 04/44h) to generate a card status change interrupt. 0 = CD_CHNG bit of the Card Status Change Register does not generate a card status change interrupt 1 = CD_CHNG bit of the Card Status Change Register does generate a card status change interrupt
2	RDY_EN	**Ready IRQ Enable** Enables RDY_CHNG (bit 2) of the Card Status Change Register (index 04h/44h) to generate a card status change interrupt. This bit is ignored when the socket is configured for I/O interface. 0 = RDY_CHNG bit of the Card Status Change Register does not generate a card status change interrupt 1 =RDY_CHNG bit of the Card Status Change Register does generate a card status change interrupt

Bit	Name	Function
1	BATT_WARN_EN	**Battery Warning IRQ Enable** Enables BATT_WARN (bit 1) of the Card Status Change Register (PC Card index 04h/44h) to generate a card status change interrupt. This bit is ignored when the socket is configured for I/O interface. 0 = BATT_WARN bit of the Card Status Change Register does not generate a card status change interrupt 1 =BATT_WARN bit of the Card Status Change Register does generate a card status change interrupt
0	BATT_DEAD_EN	**Battery Dead or Status Change IRQ Enable** Enables BATT_DEAD (bit 0) of the Card Status Change Register (PC Card index 04h/44h) to generate a card status change interrupt when BATT_DEAD is 1. 0 = BATT_DEAD bit of the Card Status Change Register does not generate a card status change interrupt 1 =BATT_DEAD bit of the Card Status Change Register does generate a card status change interrupt

Programming Notes

	7	6	5	4	3	2	1	0
Bit	IO_WIN_EN[1]	IO_WIN_EN[0]	Scratch Bit	MEM_WIN_EN[4]	MEM_WIN_EN[3]	MEM_WIN_EN[2]	MEM_WIN_EN[1]	MEM_WIN_EN[0]
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	IO_WIN_EN[1]	**I/O Window 1 Enable**
6	IO_WIN_EN[0]	**I/O Window 0 Enable**
5	Scratch Bit	This is the 'MEMCS16 Decode A23–A12' bit in the 82365SL (Rev. B).
4	MEM_WIN_EN[4]	**Memory Window 4 Enable**
3	MEM_WIN_EN[3]	**Memory Window 3 Enable**
2	MEM_WIN_EN[2]	**Memory Window 2 Enable**
1	MEM_WIN_EN[1]	**Memory Window 1 Enable**
0	MEM_WIN_EN[0]	**Memory Window 0 Enable**

Programming Notes

Bits 4–1: This device has two PC Card modes of operation referred to as Standard mode and Enhanced mode. Standard mode is the default mode of operation. Enhanced mode can be enabled via bit 0 of the Chip Setup and Control (CSC) register F1h. In the Standard PC Card mode of operation, a total of six PC Card memory windows are available. These PC Card windows are enabled via memory window C–F enable bits of the Address Window Enable Register for Socket A and the Memory Window C Enable bit of the Address Window Enable Register for Socket B. The memory window C–F enable bits for Socket B are used to enable/disable the MMS Windows C–F in Standard PC Card mode. This is due to the fact that in Standard PC Card mode, the memory window resources that are typically Socket B memory window control are used for Memory Management System (MMS) Window C–F control. See the table below for the PC Card Socket B resources used for MMS Windows C–F when in Standard PC Card mode.

The MMS windows are disabled when the Enhanced PC Card mode is selected. In the Enhanced PC Card mode the Address Window 1–4 enable bits for Socket B are used to enable or disable the PC Card memory windows for Socket B normally.

All register bit descriptions that are shaded deviate from strict 83865SL implementation.

PC Card Socket B Memory Window Resources Used for MMS

PC Card Socket B Memory Window Control	MMS Window	PC Card Controller Index Registers Used (Index address and data port = 3E0h/3E1h)
Window 0	None	None
Window 1	MMS 0	46h (bit 1), 58–5Dh
Window 2	MMS 1	46h[2], 60–65h
Window 3	MMS 2	46h[3], 68–6Dh
Window 4	MMS 3	46h[4], 70–75h

	7	6	5	4	3	2	1	0
Bit	TIM_SEL_1	Scratch Bit	IO_WIN_AUTO_SIZE_1	IO_WIN_SIZE_1	TIM_SEL_0	Scratch Bit	IO_WIN_AUTO_SIZE_0	IO_WIN_SIZE_0
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	TIM_SEL_1	**I/O Window 1 Timing Set Select** This bit selects one of two timing sets to use for PC Card cycles generated on hits to I/O window 1. 0 = Timing set 0 is used for I/O cycle timings 1 = Timing set 1 is used for I/O cycle timings This is the I/O window 1 Wait State bit in the 82365SL (Rev. B). It selected the number of ISA bus wait states to use during PC Card I/O cycles. Since this design implements cycle timing differently, this bit has a different meaning.
6	Scratch Bit	**Scratch Bit** This is the 'I/O Window 1 Zero Wait State' bit in the 82365SL (Rev. B). In this implementation, this bit does not control any specific hardware.
5	IO_WIN_AUTO_SIZE_1	**I/O Window 1 Auto-Size** 0 = IO_WIN_SIZE_1 bit determines I/O window 1 size 1 = WP_X ($\overline{\text{IOIS16\_x}}$) signal determines I/O window 1 size
4	IO_WIN_SIZE_1	**I/O Window 1 Size** 0 = I/O window 1 is 8 bits wide 1 = I/O window 1 is 16 bits wide
3	TIM_SEL_0	**I/O Window 0 Timing Set Select** This bit selects one of two timing sets to use for PC Card cycles generated on hits to I/O window 0. 0 = Timing set 0 is used for I/O cycle timings 1 = Timing set 1 is used for I/O cycle timings This is the I/O window 0 Wait State bit in the 82365SL (Rev. B). It selected the number of ISA bus wait states to use during PC Card I/O cycles. Since this design implements cycle timing differently, this bit has a different meaning.
2	Scratch Bit	**Scratch Bit** This is the 'I/O Window 0 Zero Wait State' bit in the 82365SL (Rev. B). In this implementation, this bit does not control any specific hardware.
1	IO_WIN_AUTO_SIZE_0	**I/O Window 0 Auto-Size** 0 = IO_WIN_SIZE_0 bit determines I/O window 0 size 1 = WP_X ($\overline{\text{IOIS16\_x}}$) signal determines I/O window 0 size
0	IO_WIN_SIZE_0	**I/O Window 0 Size** 0 = I/O window 0 is 8 bits wide 1 = I/O window 0 is 16 bits wide

Programming Notes

All register bit descriptions that are shaded deviate from strict 83865SL implementation.

I/O Window 0 Start Address Low Register

I/O Address 3E0h/3E1h
(Socket A) Index 08h
(Socket B) Index 48h

	7	6	5	4	3	2	1	0
Bit	IO_WIN0_START[7–0]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	IO_WIN0_START[7–0]	**I/O Window 0 Start Address Bits 7–0** Contains the least significant bits of the system address at which to start mapping into the PC Card address space.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	IO_WIN0_START[15–8]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	IO_WIN0_START[15–8]	**I/O Window 0 Start Address Bits 15–8** Contains the most significant bits of the system address at which to start mapping into the PC Card I/O address space.

Programming Notes

I/O Window 0 Stop Address Low Register

I/O Address 3E0h/3E1h
(Socket A) Index 0Ah
(Socket B) Index 4Ah

	7	6	5	4	3	2	1	0
Bit	IO_WIN0_STOP[7–0]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	IO_WIN0_STOP[7–0]	**I/O Window 0 Stop Address Bits 7–0** Contains the least significant bits of the system address at which to stop mapping into the PC Card I/O address space.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	IO_WIN0_STOP[15–8]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	IO_WIN0_STOP[15–8]	**I/O Window 0 Stop Address Bits 15–8** Contains the most significant bits of the system address at which to stop mapping into the PC Card I/O address space.

Programming Notes

I/O Window 1 Start Address Low Register

I/O Address 3E0h/3E1h
(Socket A) Index 0Ch
(Socket B) Index 4Ch

	7	6	5	4	3	2	1	0
Bit	IO_WIN1_START[7–0]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	IO_WIN1_START[7–0]	**I/O Window 1 Start Address Bits 7–0** Contains the least significant bits of the system address at which to start mapping into the PC Card address space.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	IO_WIN1_START[15–8]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	IO_WIN1_START[15–8]	**I/O Window 1 Start Address Bits 15–8** Contains the most significant bits of the system address at which to start mapping into the PC Card I/O address space.

Programming Notes

I/O Window 1 Stop Address Low Register

I/O Address 3E0h/3E1h
(Socket A) Index 0Eh
(Socket B) Index 4Eh

	7	6	5	4	3	2	1	0
Bit	IO_WIN1_STOP[7–0]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	IO_WIN1_STOP[7–0]	**I/O Window 1 Stop Address Bits 7–0** Contains the least significant bits of the system address at which to stop mapping into the PC Card I/O address space.

Programming Notes

I/O Window 1 Stop Address High Register

I/O Address 3E0h/3E1h
(Socket A) Index 0Fh
(Socket B) Index 4Fh

	7	6	5	4	3	2	1	0
Bit	IO_WIN1_STOP[15–8]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	IO_WIN1_STOP[15–8]	**I/O Window 1 Stop Address Bits 15–8** Contains the most significant bits of the system address at which to stop mapping into the PC Card I/O address space.

Programming Notes

Memory Window 0 Start Address Low Register

I/O Address 3E0h/3E1h
(Socket A) Index 10h
(Socket B) Index 50h

	7	6	5	4	3	2	1	0
Bit	MEM_WIN0_START[19–12]							
Default	o	o	o	o	o	o	o	o
R/W	R/W							

Bit	Name	Function
7–0	MEM_WIN0_ START[19–12]	**Memory Window 0 Start Address Bits 19–12** Contains bits 19–12 of the system address at which to start mapping into the PC Card memory address space.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	DATA_SIZE	Scratch Bit	MEM_WIN0_START[25–24]		MEM_WIN0_START[23–20]			
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W		R/W			

Bit	Name	Function
7	DATA_SIZE	**Window Data Size** 0 = Enables an 8-bit data path 1 = Enables a 16-bit data path
6	Scratch Bit	**Scratch Bit** This is the Zero Wait State bit in the 82365SL (Rev. B).
5–4	MEM_WIN0_ START[25–24]	**Memory Window 0 Start Address Bits 26–24** Contains bits 25–24 of the system address at which to start mapping into the PC Card memory address space. These bits are reserved in the 82365SL (Rev. B).
3–0	MEM_WIN0_ START[23–20]	**Memory Window 0 Start Address Bits 23–20** Contains bits 23–20 of the system address at which to start mapping into the PC Card memory address space.

Programming Notes

All register bit descriptions that are shaded deviate from strict 83865SL implementation.

Memory Window 0 Stop Address Low Register

I/O Address 3E0h/3E1h
(Socket A) Index 12h
(Socket B) Index 52h

	7	6	5	4	3	2	1	0
Bit	MEM_WIN0_STOP[19–12]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	MEM_WIN0_STOP[19–12]	**Memory Window 0 Stop Address Bits 19–12** Contains bits 19–12 of the system address at which to stop mapping into the PC Card address space.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	TIMER_SEL[1–0]		MEM_WIN0_STOP[25–24]		MEM_WIN0_STOP[23–20]			
Default	0	0	0	0	0	0	0	0
R/W	R/W		R/W		R/W			

Bit	Name	Function
7–6	TIMER_SEL[1–0]	**Memory Window Timer Set Select** For this window, selects which of the four timer sets to use for PC Card (non-DMA) memory cycles.
5–4	MEM_WIN0_ STOP[25–24]	**Memory Window 0 Stop Address Bits 25–24** Contains bits 25–24 of the system address at which to stop mapping into the PC Card address space. These bits are reserved in the 82365SL (Rev. B).
3–0	MEM_WIN0_ STOP[23–20]	**Memory Window 0 Stop Address 23–20** Contains bits 23–20 of the system address at which to stop mapping into the PC Card address space.

Programming Notes

All register bit descriptions that are shaded deviate from strict 83865SL implementation.

Memory Window 0 Address Offset Low Register

I/O Address 3E0h/3E1h
(Socket A) Index 14h
(Socket B) Index 54h

	7	6	5	4	3	2	1	0
Bit	MEM_WIN0_CARD_OFS[19–12]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	MEM_WIN0_CARD_ OFS[19–12]	**Memory Window 0 PC Card Offset Address Bits 19–12** Contains bits 19–12 of the offset which is to be added to system address bits SA19–SA12 before mapping into PC Card memory space.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	WR_PROT	REG_ACT	MEM_WIN0_CARD_ OFS[25–24]		MEM_WIN0_CARD_OFS[23–20]			
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W		R/W			

Bit	Name	Function
7	WR_PROT	**Window Write Protect** 0 = Permits writes to go through to the PC Card 1 = Prevents writes from going through to the PC Card
6	REG_ACT	**REG Active** Selects the state of the $\overline{\text{REG\_X}}$ pin for accesses to this window. 0 = Window is mapped to PC Card common memory 1 = Window is mapped to PC Card attribute memory
5–4	MEM_WIN0_CARD_ OFS[25–24]	**Memory Window 0 PC Card Offset Address Bits 25–24** Contains bits 25–24 of the offset which is to be added to system address bits SA25–SA24 before mapping into PC Card memory space. These bits are reserved in the 82365SL (Rev. B).
3–0	MEM_WIN0_CARD_ OFS[23–20]	**Memory Window 0 PC Card Offset Address Bits 23–20** Contains bits 23–20 of the offset which is to be added to system address bits SA23–SA20 before mapping into PC Card memory space.

Programming Notes

All register bit descriptions that are shaded deviate from strict 83865SL implementation.

Memory Window 1 Start Address Low Register

I/O Address 3E0h/3E1h
(Socket A) Index 18h
(Socket B) Index 58h

	7	6	5	4	3	2	1	0
Bit	MEM_WIN1_CARD_START[19–12]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	MEM_WIN1_CARD_ START[19–12]	**Memory Window 1 PC Card Start Address Bits 19–12** Contains bits 19–12 of the system address at which to start mapping into the PC Card memory address space.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	DATA_SIZE	Scratch Bit	MEM_WIN1_CARD_ START[25–24]		MEM_WIN1_CARD_START[23–20]			
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W		R/W			

Bit	Name	Function
7	DATA_SIZE	**Window Data Size** 0 = Enables an 8-bit data path 1 = Enables a 16-bit data path
6	Scratch Bit	**Scratch Bit** This is the Zero Wait State bit in the 82365SL (Rev. B).
5–4	MEM_WIN1_CARD_ START[25–24]	**Memory Window 1 PC Card Start Address Bits 25–24** Contains bits 25–24 of the system address at which to start mapping into the PC Card memory address space. These bits are reserved in the 82365SL (Rev. B).
3–0	MEM_WIN1_CARD_ START[23–20]	**Memory Window 1 PC Card Start Address Bits 23–20** Contains bits 23–20 of the system address at which to start mapping into the PC Card memory address space.

Programming Notes

All register bit descriptions that are shaded deviate from strict 83865SL implementation.

Memory Window 1 Stop Address Low Register

I/O Address 3E0h/3E1h
(Socket A) Index 1Ah
(Socket B) Index 5Ah

	7	6	5	4	3	2	1	0
Bit	MEM_WIN1_CARD_STOP[19–12]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	MEM_WIN1_CARD_STOP[19–12]	**Memory Window 1 PC Card Stop Address Bits 19–12** Contains bits 19–12 of the system address at which to stop mapping into the PC Card address space.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	TIMER_SEL[1–0]		MEM_WIN1_CARD_ STOP[25–24]		MEM_WIN1_CARD_STOP[23–20]			
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W		R/W			

Bit	Name	Function
7–6	TIMER_SEL[1–0]	**Memory Window Timer Set Select** For this window, these bits select which of the four timer sets to use for PC Card (non-DMA) memory cycles.
5–4	MEM_WIN1_CARD_ STOP[25–24]	**Memory Window 1 PC Card Stop Address Bits 25–24** Contains bits 25–24 of the system address at which to stop mapping into the PC Card address space. These bits are reserved in the 82365SL (Rev. B).
3–0	MEM_WIN1_CARD_ STOP[23–20]	**Memory Window 1 PC Card Stop Address Bits 23–20** Contains bits 23–20 of the system address at which to stop mapping into the PC Card address space.

Programming Notes

All register bit descriptions that are shaded deviate from strict 83865SL implementation.

Memory Window 1 Address Offset Low Register

I/O Address 3E0h/3E1h
(Socket A) Index 1Ch
(Socket B) Index 5Ch

	7	6	5	4	3	2	1	0
Bit	MEM_WIN1_CARD_OFS[19–12]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	MEM_WIN1_CARD_ OFS[19–12]	**Memory Window 1 PC Card Offset Bits 19–12** This register contains bits 19–12 of the offset which is to be added to system address bits SA19–SA12 before mapping into PC Card memory space.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	WR_PROT	REG_ACT	MEM_WIN1_CARD_ OFS[25–24]		MEM_WIN1_CARD_OFS[23–20]			
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W		R/W			

Bit	Name	Function
7	WR_PROT	**Window Write Protect** 0 = Permits writes to go through to the PC Card 1 = Prevents writes from going through to the PC Card
6	REG_ACT	**REG Active** Selects the state of the $\overline{\text{REG\_X}}$ pin for accesses to this window. 0 = Window is mapped to PC Card common memory 1 = Window is mapped to PC Card attribute memory
5–4	MEM_WIN1_CARD_ OFS[25–24]	**Memory Window 1 PC Card Offset Bits 25–24** Contains bits 25–24 of the offset which is to be added to system address bits SA25–SA24 before mapping into PC Card memory space. These bits are reserved in the 82365SL (Rev. B).
3–0	MEM_WIN1_CARD_ OFS[23–20]	**Memory Window 1 PC Card Offset Bits 23–20** Contains bits 23–20 of the offset which is to be added to system address bits SA23–SA20 before mapping into PC Card memory space.

Programming Notes

All register bit descriptions that are shaded deviate from strict 83865SL implementation.

Memory Window 2 Start Address Low Register

I/O Address 3E0h/3E1h
(Socket A) Index 20h
(Socket B) Index 60h

	7	6	5	4	3	2	1	0
Bit	MEM_WIN2_START[19–12]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	MEM_WIN2_ START[19–12]	**Memory Window 2 Start Address Bits 19–12** Contains bits 19–12 of the system address at which to start mapping into the PC Card memory address space.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	DATA_SIZE	Scratch Bit	MEM_WIN2_START[25–24]		MEM_WIN2_START[23–20]			
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W		R/W			

Bit	Name	Function
7	DATA_SIZE	**Window Data Size** 0 = Enables an 8-bit data path 1 = Enables a 16-bit data path
6	Scratch Bit	**Scratch Bit** This is the Zero Wait State bit in the 82365SL (Rev. B).
5–4	MEM_WIN2_ START[25–24]	**Memory Window 2 Start Address Bits 26–24** Contains bits 25–24 of the system address at which to start mapping into the PC Card memory address space.
3–0	MEM_WIN2_ START[23–20]	**Memory Window 2 Start Address Bits 23–20** Contains bits 23–20 of the system address at which to start mapping into the PC Card memory address space.

Programming Notes

Bits 5–4: These bits are reserved in the 82365SL (Rev. B).

All register bit descriptions that are shaded deviate from strict 83865SL implementation.

Memory Window 2 Stop Address Low Register

I/O Address 3E0h/3E1h
(Socket A) Index 22h
(Socket B) Index 62h

	7	6	5	4	3	2	1	0
Bit	MEM_WIN2_STOP[19–12]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	MEM_WIN2_ STOP[19–12]	**Memory Window 2 Stop Address Bits 19–12** Contains bits 19–12 of the system address at which to stop mapping into the PC Card address space.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	TIMER_SEL[1–0]		MEM_WIN2_STOP[25–24]		MEM_WIN2_STOP[23–20]			
Default	0	0	0	0	0	0	0	0
R/W	R/W		R/W		R/W			

Bit	Name	Function
7–6	TIMER_SEL[1–0]	**Memory Window Timer Set Select** For this window, these bits select which of the four timer sets to use for PC Card (non-DMA) memory cycles.
5–4	MEM_WIN2_STOP[25–24]	**Memory Window 2 Stop Address Bits 25–24** Contains bits 25–24 of the system address at which to stop mapping into the PC Card address space. These bits are reserved in the 82365SL (Rev. B).
3–0	MEM_WIN2_STOP[23–20]	**Memory Window 2 Stop Address Bits 23–20** This field contains bits 23–20 of the system address at which to stop mapping into the PC Card address space.

Programming Notes

All register bit descriptions that are shaded deviate from strict 83865SL implementation.

Memory Window 2 Address Offset Low Register

I/O Address 3E0h/3E1h
(Socket A) Index 24h
(Socket B) Index 64h

	7	6	5	4	3	2	1	0
Bit	MEM_WIN2_CARD_OFS[19–12]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	MEM_WIN2_CARD_OFS[19–12]	**Memory Window 2 PC Card Offset Address Bits 19–12** This register contains bits 19–12 of the offset which is to be added to system address bits SA19–SA12 before mapping into PC Card memory space.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	WR_PROT	REG_ACT	MEM_WIN2_CARD_ OFS[25–24]		MEM_WIN2_CARD_OFS[23–20]			
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W		R/W			

Bit	Name	Function
7	WR_PROT	**Window Write Protect** 0 = Permits writes to go through to the PC Card 1 = Prevents writes from going through to the PC Card
6	REG_ACT	**REG Active** Selects the state of the $\overline{\text{REG\_X}}$ pin for accesses to this window. 0 = Window is mapped to PC Card common memory 1 = Window is mapped to PC Card attribute memory
5–4	MEM_WIN2_CARD_ OFS[25–24]	**Memory Window 2 PC Card Offset Address Bits 25–24** This field contains bits 25–24 of the offset which is to be added to system address bits SA25–SA24 before mapping into PC Card memory space. These bits are reserved in the 82365SL (Rev. B).
3–0	MEM_WIN2_CARD_ OFS[23–20]	**Memory Window 2 PC Card Offset Address Bits 23–20** This field contains bits 23–20 of the offset which is to be added to system address bits SA23–SA20 before mapping into PC Card memory space.

Programming Notes

All register bit descriptions that are shaded deviate from strict 83865SL implementation.

Memory Window 3 Start Address Low Register

I/O Address 3E0h/3E1h
(Socket A) Index 28h
(Socket B) Index 68h

	7	6	5	4	3	2	1	0
Bit	MEM_WIN3_START[19–12]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	MEM_WIN3_ START[19–12]	**Memory Window 3 Start Address Bits 19–12** This register contains bits 19–12 of the system address at which to start mapping into the PC Card memory address space.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	DATA_SIZE	Scratch Bit	MEM_WIN3_START[25–24]		MEM_WIN3_START[23–20]			
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W		R/W			

Bit	Name	Function
7	DATA_SIZE	**Window Data Size** 0 = Enables an 8-bit data path 1 = Enables a 16-bit data path
6	Scratch Bit	**Scratch Bit** This is the Zero Wait State bit in the 82365SL (Rev. B).
5–4	MEM_WIN3_ START[25–24]	**Memory Window 3 Start Address Bits 26–24** This field contains bits 25–24 of the system address at which to start mapping into the PC Card memory address space. These bits are reserved in the 82365SL (Rev. B).
3–0	MEM_WIN3_ START[23–20]	**Memory Window 3 Start Address Bits 23–20** This field contains bits 23–20 of the system address at which to start mapping into the PC Card memory address space.

Programming Notes

All register bit descriptions that are shaded deviate from strict 83865SL implementation.

Memory Window 3 Stop Address Low Register

I/O Address 3E0h/3E1h
(Socket A) Index 2Ah
(Socket B) Index 6Ah

	7	6	5	4	3	2	1	0
Bit	MEM_WIN3_STOP[19–12]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	MEM_WIN3_STOP[19–12]	**Memory Window 3 Stop Address Bits 19–12** This register contains bits 19–12 of the system address at which to stop mapping into the PC Card address space.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	TIMER_SEL[1–0]		MEM_WIN3_STOP[25–24]		MEM_WIN3_STOP[23–20]			
Default	0	0	0	0	0	0	0	0
R/W	R/W		R/W		R/W			

Bit	Name	Function
7–6	TIMER_SEL[1–0]	**Memory Window Timer Set Select** For this window, this field selects which of the four timer sets to use for PC Card (non-DMA) memory cycles.
5–4	MEM_WIN3_STOP][25–24]	**Memory Window 3 Stop Address Bits 25–24** This field contains bits 25–24 of the system address at which to stop mapping into the PC Card address space. These bits are reserved in the 82365SL (Rev. B).
3–0	MEM_WIN3_STOP][23–20]	**Memory Window 3 Stop Address Bits 23–20** This field contains bits 23–20 of the system address at which to stop mapping into the PC Card address space.

Programming Notes

All register bit descriptions that are shaded deviate from strict 83865SL implementation.

Memory Window 3 Address Offset Low Register

I/O Address 3E0h/3E1h
(Socket A) Index 2Ch
(Socket B) Index 6Ch

	7	6	5	4	3	2	1	0
Bit	MEM_WIN3_CARD_OFS[19–12]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	MEM_WIN3_CARD_OFS[19–12]	**Memory Window 3 PC Card Offset Address Bits 19–12** This register contains bits 19–12 of the offset which is to be added to system address bits SA19–SA12 before mapping into PC Card memory space.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	WR_PROT	REG_ACT	MEM_WIN3_CARD_ OFS[25–24]		MEM_WIN3_CARD_OFS[23–20]			
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W		R/W			

Bit	Name	Function
7	WR_PROT	**Window Write Protect** 0 = Permits writes to go through to the PC Card 1 = Prevents writes from going through to the PC Card
6	REG_ACT	**REG Active** This bit selects the state of the $\overline{\text{REG\_X}}$ pin for accesses to this window. 0 = Window is mapped to PC Card common memory 1 = Window is mapped to PC Card attribute memory
5–4	MEM_WIN3_CARD_ OFS[25–24]	**Memory Window 3 PC Card Offset Address Bits 25–24** This field contains bits 25–24 of the offset which is to be added to system address bits SA25–SA24 before mapping into PC Card memory space. These bits are reserved in the 82365SL (Rev. B).
3–0	MEM_WIN3_CARD_ OFS[23–20]	**Memory Window 3 PC Card Offset Address Bits 23–20** This field contains bits 23–20 of the offset which is to be added to system address bits SA23–SA20 before mapping into PC Card memory space.

Programming Notes

All register bit descriptions that are shaded deviate from strict 83865SL implementation.

Memory Window 4 Start Address Low Register

I/O Address 3E0h/3E1h
(Socket A) Index 30h
(Socket B) Index 70h

	7	6	5	4	3	2	1	0
Bit	MEM_WIN4_START[19–12]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	MEM_WIN4_ START[19–12]	**Memory Window 4 Start Address Bits 19–12** This register contains bits 19–12 of the system address at which to start mapping into the PC Card memory address space.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	DATA_SIZE	Scratch Bit	MEM_WIN4_START[25–24]		MEM_WIN4_START[23–20]			
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W		R/W			

Bit	Name	Function
7	DATA_SIZE	**Window Data Size** 0 = Enables an 8-bit data path 1 = Enables a 16-bit data path
6	Scratch Bit	**Scratch Bit** This is the Zero Wait State bit in the 82365SL (Rev. B).
5–4	MEM_WIN4_ START[25–24]	**Memory Window 4 Start Address Bits 26–24** this field contains bits 25–24 of the system address at which to start mapping into the PC Card memory address space. These bits are reserved in the 82365SL (Rev. B).
3–0	MEM_WIN4_ START[23–20]	**Memory Window 4 Start Address Bits 23–20** This field contains bits 23–20 of the system address at which to start mapping into the PC Card memory address space.

Programming Notes

All register bit descriptions that are shaded deviate from strict 83865SL implementation.

Memory Window 4 Stop Address Low Register

I/O Address 3E0h/3E1h
(Socket A) Index 32h
(Socket B) Index 72hl

	7	6	5	4	3	2	1	0
Bit	MEM_WIN4_STOP[19–12]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	MEM_WIN4_STOP[19–12]	**Memory Window 4 Stop Address Bits 19–12** This register contains bits 19–12 of the system address at which to stop mapping into the PC Card address space.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	TIMER_SEL[1–0]		MEM_WIN4_STOP[25–24]		MEM_WIN4_STOP[23–20]			
Default	0	0	0	0	0	0	0	0
R/W	R/W		R/W		R/W			

Bit	Name	Function
7–6	TIMER_SEL[1–0]	**Memory Window Timer Set Select** For this window, this field selects which of the four timer sets to use for PC Card (non-DMA) memory cycles.
5–4	MEM_WIN4_STOP[25–24]	**Memory Window 4 Stop Address Bits 25–24** This field contains bits 25–24 of the system address at which to stop mapping into the PC Card address space. These bits are reserved in the 82365SL (Rev. B).
3–0	MEM_WIN4_STOP[23–20]	**Memory Window 4 Stop Address Bits 23–20** This field contains bits 23–20 of the system address at which to stop mapping into the PC Card address space.

Programming Notes

All register bit descriptions that are shaded deviate from strict 83865SL implementation.

Memory Window 4 Address Offset Low Register

I/O Address 3E0h/3E1h
(Socket A) Index 34h
(Socket B) Index 74h

	7	6	5	4	3	2	1	0
Bit	MEM_WIN4_CARD_OFS[19–12]							
Default	0	0	0	0	0	0	0	0
R/W	R/W							

Bit	Name	Function
7–0	MEM_WIN4_CARD_ OFS[19–12]	**Memory Window 4 PC Card Offset Address Bits 19–12** This register contains bits 19–12 of the offset which is to be added to system address bits SA19–SA12 before mapping into PC Card memory space.

Programming Notes

	7	6	5	4	3	2	1	0
Bit	WR_PROT	REG_ACT	MEM_WIN4_CARD_ OFS[25–24]		MEM_WIN4_CARD_OFS[23–20]			
Default	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W		R/W			

Bit	Name	Function
7	WR_PROT	**Window Write Protect** 0 = Permits writes to go through to the PC Card 1 = Prevents writes from going through to the PC Card
6	REG_ACT	**REG Active** Selects the state of the $\overline{\text{REG\_X}}$ pin for accesses to this window. 0 = Window is mapped to PC Card common memory 1 = Window is mapped to PC Card attribute memory
5–4	MEM_WIN4_CARD_ OFS[25–24]	**Memory Window 4 PC Card Offset Address Bits 25–24** This field contains bits 25–24 of the offset which is to be added to system address bits SA25–SA24 before mapping into PC Card memory space. These bits are reserved in the 82365SL (Rev. B).
3–0	MEM_WIN4_CARD_ OFS[23–20]	**Memory Window 4 PC Card Offset Address Bits 23–20** This field contains bits 23–20 of the offset which is to be added to system address bits SA23–SA20 before mapping into PC Card memory space.

Programming Notes

All register bit descriptions that are shaded deviate from strict 83865SL implementation.

	7	6	5	4	3	2	1	0
Bit	SETUP_PRESC[1–0]		SETUP_MULT[5–0]					
Default	0	0	0	0	0	0	0	0
R/W	R/W		R/W					

Bit	Name	Function
7–6	SETUP_PRESC[1–0]	**Setup Timing Prescaler Select** For this timing set, this field selects N_{pres} from one of four values to prescale N_{val} to select the number of clocks of setup time from valid address before the PC Card command goes active. See the top of this section for a description of the formula.
5–0	SETUP_MULT[5–0]	**Setup Timing Multiplier Value** For this timing set, this field selects N_{val} from 0 to 63 to be multiplied with N_{pres} to select the number of clocks of setup time from valid address before the PC Card command goes active. See the top of this section for a description of the formula.

Programming Notes

All register bit descriptions that are shaded deviate from strict 83865SL implementation.

The Multiplier Value, N_{val}, is combined with the Prescaler Weighting, N_{pres}, to determine the number of clocks to count for the current state (**S**etup, **C**ommand active, or **R**ecovery). The Current State (S,C, or R) = (N_{pres} x N_{val}) + 1.

Field Value	Prescaler Weighting
00	N_{pres}=1
01	N_{pres}=16
10	N_{pres}=256
11	N_{pres}=4096

	7	6	5	4	3	2	1	0
Bit	CMD_PRESC[1–0]		CMD_MULT[5–0]					
Default	0	0	0	0	1	1	1	1
R/W	R/W		R/W					

Bit	Name	Function
7–6	CMD_PRESC[1–0]	**Command Timing Prescaler Select** For this timing set, this field selects N_{pres} from one of four values to prescale N_{val} to select the number of clocks of command active time. See the top of this section for a description of the formula.
5–0	CMD_MULT[5–0]	**Command Timing Multiplier Value** For this timing set, this field selects N_{val} from 0 to 63 to be multiplied with N_{pres} to select the number of clocks of command active time. See the top of this section for a description of the formula.

Programming Notes

All register bit descriptions that are shaded deviate from strict 83865SL implementation.

The Multiplier Value, N_{val}, is combined with the Prescaler Weighting, N_{pres}, to determine the number of clocks to count for the current state (**S**etup, **C**ommand active, or **R**ecovery). The Current State (S,C, or R) = (N_{pres} x N_{val}) + 1.

Field Value	Prescaler Weighting
00	N_{pres}=1
01	N_{pres}=16
10	N_{pres}=256
11	N_{pres}=4096

	7	6	5	4	3	2	1	0
Bit	REC_PRESC[1–0]		REC_MULT[5–0]					
Default	0	0	0	0	0	0	0	0
R/W	R/W		R/W					

Bit	Name	Function
7–6	REC_PRESC[1–0]	**Recovery Timing Prescaler Select** For this timing set, this field selects N_{pres} from one of four values to prescale N_{val} to select the number of clocks of recovery time (address hold time after the PC Card command goes inactive until the address changes state.) See the top of this section for a description of the formula.
5–0	REC_MULT[5–0]	**Recovery Timing Multiplier Value** For this timing set, this field selects N_{val} from 0 to 63 to be multiplied with N_{pres} to select the number of clocks of recovery time (address hold time after the PC Card command goes inactive until the address changes state.) See the top of this section for a description of the formula.

Programming Notes

All register bit descriptions that are shaded deviate from strict 83865SL implementation.

The Multiplier Value, N_{val}, is combined with the Prescaler Weighting, N_{pres}, to determine the number of clocks to count for the current state (**S**etup, **C**ommand active, or **R**ecovery). The Current State (S,C, or R) = (N_{pres} x N_{val}) + 1.

Field Value	Prescaler Weighting
00	N_{pres}=1
01	N_{pres}=16
10	N_{pres}=256
11	N_{pres}=4096

	7	6	5	4	3	2	1	0
Bit	SETUP_PRESC[1–0]		SETUP_MULT[5–0]					
Default	0	0	0	0	0	0	0	0
R/W	R/W		R/W					

Bit	Name	Function
7–6	SETUP_PRESC[1–0]	**Setup Timing Prescaler Select** For this timing set, this field selects N_{pres} from one of four values to prescale N_{val} to select the number of clocks of setup time from valid address before the PC Card command goes active. See the top of this section for a description of the formula.
5–0	SETUP_MULT[5–0]	**Setup Timing Multiplier Value** For this timing set, this field selects N_{val} from 0 to 63 to be multiplied with N_{pres} to select the number of clocks of setup time from valid address before the PC Card command goes active. See the top of this section for a description of the formula.

Programming Notes

All register bit descriptions that are shaded deviate from strict 83865SL implementation.

The Multiplier Value, N_{val}, is combined with the Prescaler Weighting, N_{pres}, to determine the number of clocks to count for the current state (**S**etup, **C**ommand active, or **R**ecovery). The Current State (S,C, or R) = (N_{pres} x N_{val}) + 1.

Field Value	Prescaler Weighting
00	N_{pres}=1
01	N_{pres}=16
10	N_{pres}=256
11	N_{pres}=4096

	7	6	5	4	3	2	1	0
Bit	CMD_PRESC[1–0]		CMD_MULT[5–0]					
Default	0	0	0	0	2	2	2	2
R/W	R/W		R/W					

Bit	Name	Function
7–6	CMD_PRESC[1–0]	**Command Timing Prescaler Select** For this timing set, this field selects N_{pres} from one of four values to prescale N_{val} to select the number of clocks of command active time. See the top of this section for a description of the formula.
5–0	CMD_MULT[5–0]	**Command Timing Multiplier Value** For this timing set, this field selects N_{val} from 0 to 63 to be multiplied with N_{pres} to select the number of clocks of command active time. See the top of this section for a description of the formula.

Programming Notes

All register bit descriptions that are shaded deviate from strict 83865SL implementation.

The Multiplier Value, N_{val}, is combined with the Prescaler Weighting, N_{pres}, to determine the number of clocks to count for the current state (**S**etup, **C**ommand active, or **R**ecovery). The Current State (S,C, or R) = (N_{pres} x N_{val}) + 1.

Field Value	Prescaler Weighting
00	N_{pres}=1
01	N_{pres}=16
10	N_{pres}=256
11	N_{pres}=4096

	7	6	5	4	3	2	1	0
Bit	REC_PRESC[1–0]		REC_MULT[5–0]					
Default	0	0	0	0	0	0	0	0
R/W	R/W		R/W					

Bit	Name	Function
7–6	REC_PRESC[1–0]	**Recovery Timing Prescaler Select** For this timing set, this field selects N_{pres} from one of four values to prescale N_{val} to select the number of clocks of recovery time (address hold time after the PC Card command goes inactive until the address changes state.) See the top of this section for a description of the formula.
5–0	REC_MULT[5–0]	**Recovery Timing Multiplier Value** For this timing set, this field selects N_{val} from 0 to 63 to be multiplied with N_{pres} to select the number of clocks of recovery time (address hold time after the PC Card command goes inactive until the address changes state.) See the top of this section for a description of the formula.

Programming Notes

All register bit descriptions that are shaded deviate from strict 83865SL implementation.

The Multiplier Value, N_{val}, is combined with the Prescaler Weighting, N_{pres}, to determine the number of clocks to count for the current state (**S**etup, **C**ommand active, or **R**ecovery). The Current State (S,C, or R) = (N_{pres} x N_{val}) + 1.

Field Value	Prescaler Weighting
00	N_{pres}=1
01	N_{pres}=16
10	N_{pres}=256
11	N_{pres}=4096

	7	6	5	4	3	2	1	0
Bit	SETUP_PRESC[1–0]		SETUP_MULT[5–0]					
Default	0	0	0	0	0	0	0	0
R/W	R/W		R/W					

Bit	Name	Function
7–6	SETUP_PRESC[1–0]	**Setup Timing Prescaler Select** For this timing set, this field selects N_{pres} from one of four values to prescale N_{val} to select the number of clocks of setup time from valid address before the PC Card command goes active. See the top of this section for a description of the formula.
5–0	SETUP_MULT[5–0]	**Setup Timing Multiplier Value** For this timing set, this field selects N_{val} from 0 to 63 to be multiplied with N_{pres} to select the number of clocks of setup time from valid address before the PC Card command goes active. See the top of this section for a description of the formula.

Programming Notes

All register bit descriptions that are shaded deviate from strict 83865SL implementation.

The Multiplier Value, N_{val}, is combined with the Prescaler Weighting, N_{pres}, to determine the number of clocks to count for the current state (**S**etup, **C**ommand active, or **R**ecovery). The Current State (S,C, or R) = (N_{pres} x N_{val}) + 1.

Field Value	Prescaler Weighting
00	N_{pres}=1
01	N_{pres}=16
10	N_{pres}=256
11	N_{pres}=4096

	7	6	5	4	3	2	1	0
Bit	CMD_PRESC[1–0]		CMD_MULT[5–0]					
Default	0	0	0	0	3	3	3	3
R/W	R/W		R/W					

Bit	Name	Function
7–6	CMD_PRESC[1–0]	**Command Timing Prescaler Select** For this timing set, this field selects N_{pres} from one of four values to prescale N_{val} to select the number of clocks of command active time. See the top of this section for a description of the formula.
5–0	CMD_MULT[5–0]	**Command Timing Multiplier Value** For this timing set, this field selects N_{val} from 0 to 63 to be multiplied with N_{pres} to select the number of clocks of command active time. See the top of this section for a description of the formula.

Programming Notes

All register bit descriptions that are shaded deviate from strict 83865SL implementation.

The Multiplier Value, N_{val}, is combined with the Prescaler Weighting, N_{pres}, to determine the number of clocks to count for the current state (**S**etup, **C**ommand active, or **R**ecovery). The Current State (S,C, or R) = (N_{pres} x N_{val}) + 1.

Field Value	Prescaler Weighting
00	N_{pres}=1
01	N_{pres}=16
10	N_{pres}=256
11	N_{pres}=4096

	7	6	5	4	3	2	1	0
Bit	REC_PRESC[1–0]		REC_MULT[5–0]					
Default	0	0	0	0	0	0	0	0
R/W	R/W		R/W					

Bit	Name	Function
7–6	REC_PRESC[1–0]	**Recovery Timing Prescaler Select** For this timing set, this field selects N_{pres} from one of four values to prescale N_{val} to select the number of clocks of recovery time (address hold time after the PC Card command goes inactive until the address changes state.) See the top of this section for a description of the formula.
5–0	REC_MULT[5–0]	**Recovery Timing Multiplier Value** For this timing set, this field selects N_{val} from 0 to 63 to be multiplied with N_{pres} to select the number of clocks of recovery time (address hold time after the PC Card command goes inactive until the address changes state.) See the top of this section for a description of the formula.

Programming Notes

All register bit descriptions that are shaded deviate from strict 83865SL implementation.

The Multiplier Value, N_{val}, is combined with the Prescaler Weighting, N_{pres}, to determine the number of clocks to count for the current state (**S**etup, **C**ommand active, or **R**ecovery). The Current State (S,C, or R) = (N_{pres} x N_{val}) + 1.

Field Value	Prescaler Weighting
00	N_{pres}=1
01	N_{pres}=16
10	N_{pres}=256
11	N_{pres}=4096

	7	6	5	4	3	2	1	0
Bit	SETUP_PRESC[1–0]		SETUP_MULT[5–0]					
Default	0	0	0	0	0	0	0	0
R/W	R/W		R/W					

Bit	Name	Function
7–6	SETUP_PRESC[1–0]	**Setup Timing Prescaler Select** For this timing set, this field selects N_{pres} from one of four values to prescale N_{val} to select the number of clocks of setup time from valid address before the PC Card command goes active. See the top of this section for a description of the formula.
5–0	SETUP_MULT[5–0]	**Setup Timing Multiplier Value** For this timing set, this field selects N_{val} from 0 to 63 to be multiplied with N_{pres} to select the number of clocks of setup time from valid address before the PC Card command goes active. See the top of this section for a description of the formula.

Programming Notes

All register bit descriptions that are shaded deviate from strict 83865SL implementation.

The Multiplier Value, N_{val}, is combined with the Prescaler Weighting, N_{pres}, to determine the number of clocks to count for the current state (**S**etup, **C**ommand active, or **R**ecovery). The Current State (S,C, or R) = (N_{pres} x N_{val}) + 1.

Field Value	Prescaler Weighting
00	N_{pres}=1
01	N_{pres}=16
10	N_{pres}=256
11	N_{pres}=4096

	7	6	5	4	3	2	1	0
Bit	CMD_PRESC[1–0]		CMD_MULT[5–0]					
Default	0	0	0	0	0	1	0	0
R/W	R/W		R/W					

Bit	Name	Function
7–6	CMD_PRESC[1–0]	**Command Timing Prescaler Select** For this timing set, this field selects N_{pres} from one of four values to prescale N_{val} to select the number of clocks of command active time. See the top of this section for a description of the formula.
5–0	CMD_MULT[5–0]	**Command Timing Multiplier Value** For this timing set, this field selects N_{val} from 0 to 63 to be multiplied with N_{pres} to select the number of clocks of command active time. See the top of this section for a description of the formula.

Programming Notes

All register bit descriptions that are shaded deviate from strict 83865SL implementation.

The Multiplier Value, N_{val}, is combined with the Prescaler Weighting, N_{pres}, to determine the number of clocks to count for the current state (**S**etup, **C**ommand active, or **R**ecovery). The Current State (S,C, or R) = (N_{pres} x N_{val}) + 1.

Field Value	Prescaler Weighting
00	N_{pres}=1
01	N_{pres}=16
10	N_{pres}=256
11	N_{pres}=4096

	7	6	5	4	3	2	1	0
Bit	REC_PRESC[1–0]		REC_MULT[5–0]					
Default	0	0	0	0	0	0	0	0
R/W	R/W		R/W					

Bit	Name	Function
7–6	REC_PRESC[1–0]	**Recovery Timing Prescaler Select** For this timing set, this field selects N_{pres} from one of four values to prescale N_{val} to select the number of clocks of recovery time (address hold time after the PC Card command goes inactive until the address changes state.) See the top of this section for a description of the formula.
5–0	REC_MULT[5–0]	**Recovery Timing Multiplier Value** For this timing set, this field selects N_{val} from 0 to 63 to be multiplied with N_{pres} to select the number of clocks of recovery time (address hold time after the PC Card command goes inactive until the address changes state.) See the top of this section for a description of the formula.

Programming Notes

All register bit descriptions that are shaded deviate from strict 83865SL implementation.

The Multiplier Value, N_{val}, is combined with the Prescaler Weighting, N_{pres}, to determine the number of clocks to count for the current state (**S**etup, **C**ommand active, or **R**ecovery). The Current State (S,C, or R) = (N_{pres} x N_{val}) + 1.

Field Value	Prescaler Weighting
00	N_{pres}=1
01	N_{pres}=16
10	N_{pres}=256
11	N_{pres}=4096

INDEX

D

E

F

G

H

I

K

L

M

N

O

P

R

S

U

V

W

X