Élan™SC310 Microcontroller

Programmer's Reference Manual

Rev. A, April 1996

AMD

Trademarks

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

INTRODUCTION

The Élan™SC310 microcontroller is a highly integrated, low-power, single-chip system composed of the Am386®SXLV microprocessor plus the additional logic needed for an AT-compatible personal computer. It is designed for embedded PC solutions. The ÉlanSC310 microcontroller features precise power management and high integration, while providing the user with industry-standard interfaces.

The ÉlanSC310 microcontroller's architecture consists of the following functional modules:

- The Am386SXLV CPU itself, which is optimized for 3.3-V operation and includes System Management mode (SMM) power-management hardware
- A power-management controller that interfaces to the CPU's SMM and is tightly integrated with the internal clock-generator hardware
- A PC/AT-compatible memory controller and associated mapping hardware
- An optional CPU local bus or full ISA-bus controller
- Standard PC/AT system logic and peripheral controllers (DMA, interrupt controller, timer, ISA-bus controller, and EPP parallel port)
- Additional peripheral controllers (UART and real-time clock)

PURPOSE OF THIS MANUAL

This manual provides supplemental information unique to the ÉlanSC310 microcontroller, power-management and memory-management programming considerations, and a detailed description of the additional registers.

For logic and software descriptions applicable to the Am386SXLV CPU, see the Am386DXLV and Am386SXLV Microprocessors Technical Reference Manual, PID 16944.

For information on the PC/AT-compatible peripherals in the ÉlanSC310 microcontroller (e.g., 8259A, 8254), refer to the documentation for the individual part.

IN THIS MANUAL

This manual is organized in the following manner:

Chapter 1, “Power Management,” explains how to use the power-management features of the ÉlanSC310 microcontroller to conserve battery power.

Chapter 2, “Memory Management,” describes the PC/AT-compatible memory controller and the MMS mapping hardware and address spaces.

Chapter 3, “PC/AT Peripheral Registers,” provides information on the standard PC/AT peripheral registers.

Chapter 4, “Configuration Registers,” provides information on the ÉlanSC310 microcontroller’s configuration registers.

Appendix A, “Configuration Index Register Reference,” lists the configuration index registers in alphabetical order.

Appendix B, “XT-Keyboard Interface,” explains how to implement the XT keyboard function in the ÉlanSC310 microcontroller.

RELATED AMD PUBLICATIONS

PID No.	Publication and Description
20668	***ÉlanSC310 Microcontroller Data Sheet*** Describes the ÉlanSC310 microcontroller’s technical features, signal pins, internal controllers, and electrical specifications.
16944	***Am386DXLV and Am386SXLV Microprocessors Technical Reference Manual*** Describes System Management mode (SMM) and explains how to use the System Management Interrupt (SMI).
19255	***FusionE86℠ Catalog*** Provides information on tools that speed an E86™ family embedded product to market. Includes products from expert suppliers of embedded development software.

To order publications, see “Literature Ordering” on the back of this manual.

CHAPTER

1 POWER MANAGEMENT

In general, the purpose of today's power management is to reduce the wattage consumption of battery-powered computers in order to extend the useful amount of computing that can be done with a single battery charge. Power management is based on the following principal techniques:

- Employing process technologies (e.g., CMOS) that have intrinsically low power requirements
- Employing lower voltages whenever possible
- Continuously monitoring the activity of the computer and either turning off components that are not in use or reducing their clock speed

All these techniques are extensively employed in the ÉlanSC310 microcontroller, but only the third technique is under programmer control. Power management monitors all system activities (e.g., keyboard, screen, and disk events) and—based on the state of the system—determines in which operating mode the system should be running for best power conservation.

In addition, the ÉlanSC310 microcontroller can manage the power consumption of peripheral devices. This control can be integrated into the operating-mode mechanism, or it can be handled separately via the System Management Interrupt (SMI). The ÉlanSC310 microcontroller can be programmed so that various conditions, such as peripheral accesses, can cause an SMI to occur. An SMI causes the CPU to save the operating state of the processor and switch to a special interrupt service routine. This routine can then be used to turn on peripherals.

The ÉlanSC310 microcontroller's Power Management Unit (PMU) controls five Power-Management Control (PMC) pins and four Programmable General-Purpose I/O (PGP) pins that can be used by system designers to control different external peripherals.

External pins, such as the four low-battery interrupts ($\overline{BL4}$–$\overline{BL1}$), the AC power detect (ACIN), the external SMI (EXTSMI), and the $\overline{SUS}$/$\overline{RES}$ pin, can also cause SMIs or mode changes to occur. Certain power-management functions are disabled when the ACIN input is detected because it is assumed that the system is no longer using a battery.

This chapter contains sample microprocessor programs for setting up power-management functions for the ÉlanSC310 microcontroller. However, no attempt is made to cover all possible power-management situations, and the examples given are merely suggestions.

1.1 POWER MANAGEMENT UNIT

The primary design goal of the Power Management Unit (PMU) is to control the power of the entire system so as to eliminate or minimize the excess use of current, particularly when it is not needed at a specific time. The PMU uses the following techniques to conserve power:

- Slows down clocks when the system is not in active use
- Shuts off clocks to parts of the system that are idle
- Shuts off power to parts of the system that are idle
- Reduces power use when batteries are low

An additional goal of a good PMU design is to make these functions as transparent to the user as possible and to avoid any possibility of disastrous side effects, such as accidental loss of data. The ÉlanSC310 microcontroller's PMU includes the following principal components:

- **PMU State Machine** Defines certain levels of system activity and the allowable state transitions. Depending on the current level of activity registered by the PMU state, the system may, for example, run clocks at a high speed, a low speed, or turn them off. The current and previous PMU states may be read; in addition, the PMU may be forced into a specific state by a software command.
- **External Device-Control Interface** Allows the PMU to control external power switches to different devices. The on or off status of most of these devices may be specifically programmed for each of the available power-management states. In addition, dedicated logic enables the power to three specific devices to be automatically turned off after a specified time-out period during which no activity has occurred. If one of these devices is accessed after power has been turned off, an SMI is automatically generated so that the I/O instruction can be retried after powering up the device.
- **Clock-Switching Logic** Synchronously switches different clock sources to the CPU clock, or switches off the CPU clock and phase-locked loops (PLLs).
- **Activity Monitors** Check for certain external or CPU events that indicate system activity. On receipt of an event, this logic causes the PMU state machine to switch to High-Speed PLL mode. By definition, activities function only while the CPUCLK signal is running.
- **State-Transition Timer** Defines the allowable periods of inactivity before PMU state changes occur.
- **Wake-Up Logic** Allows certain events to start the clocks and restart the on-board PLLs. Wake-ups are independent of whether the CPUCLK signal is on or off. Wake-ups force the PMU into High-Speed PLL mode.
- **NMI and SMI Control** Allows certain external, internally generated, or CPU events to generate a nonmaskable interrupt (NMI) or SMI to the CPU. If the CPU clock is not running when a triggering event occurs, this logic can cause the PMU to start the CPU clock to process the interrupt.
- **Battery-Management Logic** Includes four levels of battery-power handling. Certain levels can be programmed to generate SMIs or NMIs, slow the CPU clock, or force the system into Sleep or Suspend mode.
- **Suspend and Resume Pin Logic** Provides a user-operable method of forcing the PMU to enter Sleep mode or to wake up from Sleep, Suspend, or Off mode.

- **Auto Low-Speed Logic** Provides an option of slowing the CPU clock according to a programmable duty cycle while the system is in High-Speed PLL mode, providing additional power savings.
- **Other Power-Saving Features** Include additional power-saving functions which may be utilized at the system designer's discretion.

Each of these PMU components is discussed in greater detail later in this chapter.

1.1.1 Power-Management Modes

The ÉlanSC310 microcontroller's Power-Management Unit (PMU) provides the following power-management states or *modes*:

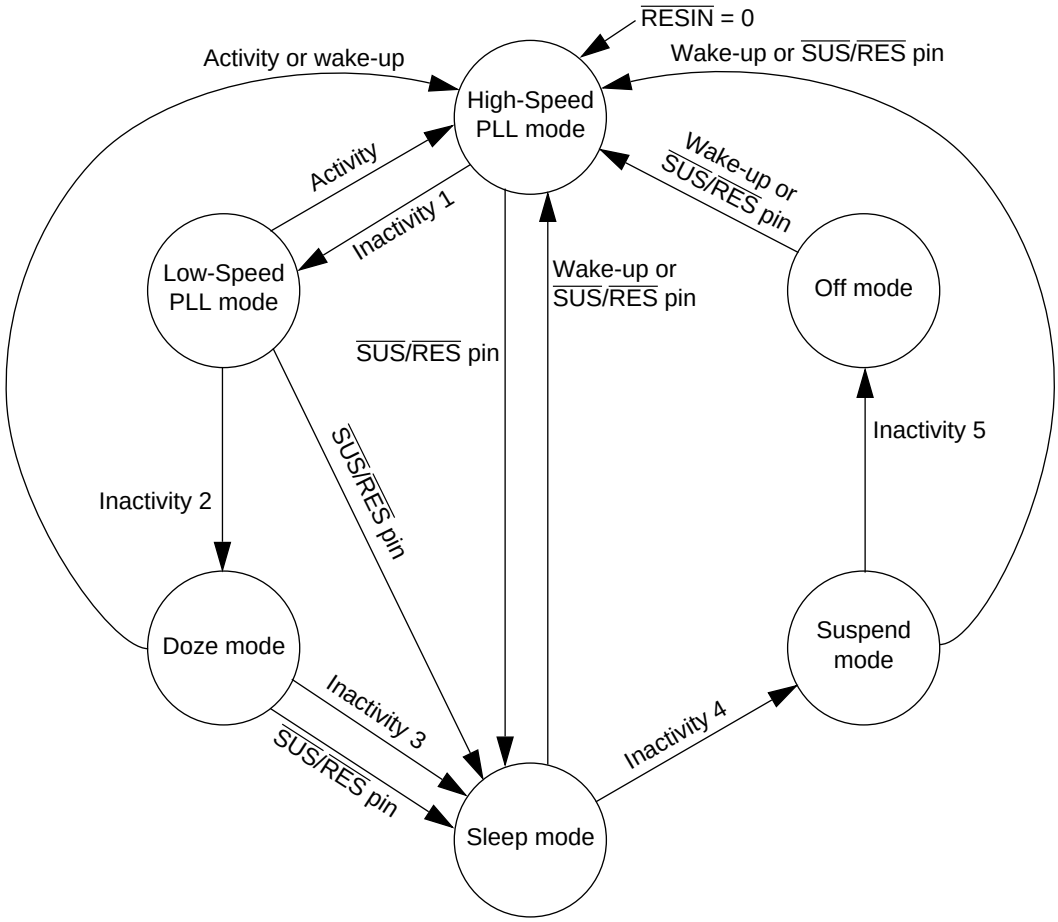
- High-Speed PLL
- Low-Speed PLL
- Doze
- Sleep
- Suspend
- Off

Each mode is defined by a different combination of CPU and peripheral operation. A given system may utilize all six states for the maximum granularity of PMU modes, or it may merge some of the states into as few as three effective PMU modes.

For a given PMU state, the controller can: specify different clock rates to various devices, turn off clocks, control external devices by way of programmable pins, and allow or disallow certain events to cause the unit to enter a subsequent PMU mode.

A state diagram for the PMU state machine is shown in Figure 1-1 on page 1-4. In addition, through the Software Mode Control register at Index 88h, the PMU can be forced into any mode except the Off mode by a software command.

Figure 1-1 PMU Operating-Mode Transitions



Notes:

1. *This picture simplifies the function of the SUS/RES pin. For more details, see "Suspend/Resume Pin Logic" on page 1-34.*
2. *ACIN Low and either BL2 or BL4 Low have the same effect as the SUS/RES pin.*

1.1.1.1 High-Speed PLL Mode

In High-Speed PLL mode, all system clocks run at their highest speeds. For the CPU, the high-speed PLL rate (supplied by the CLK2 signal) is software configurable to 40, 50, or 66 MHz, yielding internal CPU operation speeds (CPUCLK) of 20, 25, or 33 MHz, respectively. The low-speed PLL rate (also supplied by the CLK2 signal) is fixed at 18.432 MHz, yielding an internal CPU operation speed (CPUCLK) of 9.2 MHz.

The high-speed PLL rate only applies to certain types of CPU cycles. Normally, the CPU is clocked at the low-speed PLL rate, even in High-Speed PLL mode. For CPU DRAM, local-bus, fast-ROM, and idle cycles, the clock is dynamically switched to run at the high-speed PLL rate.

Any activity (or wake-up) defined by the software will cause the PMU to select High-Speed PLL mode. In this mode, power conservation takes a back seat to CPU processing power.

Power-Management Control (PMC) pins can be used to control power to peripheral devices on a per-mode basis. Software can restore power to any peripherals whose power (controlled by its associated PMC pin state) was removed by a previous transition to a lower-power mode.

1.1.1.2 Low-Speed PLL Mode

Low-Speed PLL mode is the first level of power conservation. It is entered after a specified elapsed time with no activity, programmed using the High-Speed to Low-Speed Mode Timer register at Index 83h. In this PMU mode, the CPU, DMA, and other internal system clocks run at reduced rates. The low-speed PLL clock, whose rate is always fixed at 18.432 MHz, is sent through a programmable divider. The minimum divisor is 2. This yields a maximum programmable CLK2 rate of 9.216 MHz, which results in a maximum internal operation speed (CPUCLK) of 4.608 MHz. PLL divisors of 2, 4, 8, and 16 can be selected for dividing the low-speed PLL clock in Low-Speed PLL mode. No dynamic switching of CLK2 to the high-speed PLL rate is done in this mode. All other clocks and peripherals run at full speed.

Power-management software may optionally shut off the high-speed PLL. Depending on the frequency of this PLL, up to 750 µA may be saved by this action. If this is done, a PLL start-up delay of 256 ms (programmed in the Clock Control register at Index 8Fh) must elapse before High-Speed PLL mode can be re-entered.

PMC pins may be used to control power to peripheral devices on a per-mode basis. Software may restore power to any peripherals whose power (controlled by their associated PMC pin states) was removed by a previous transition to a lower power mode.

1.1.1.3 Doze Mode

Doze mode is the second level of power conservation. The CPU, system, and DMA clocks, and the high-speed PLL are stopped. This mode is entered after a programmed time without activity has elapsed. For details, see “Low-Speed to Doze Timer Register (Index 84h)” on page 4-44.

By default, the CPU clock is stopped in Doze mode, along with the DMA clock and internal system clock (see Table 1-1 on page 1-6). The video, UART, and 8254 timer clocks are driven by the low-speed PLL, which is enabled by default. By allowing these clocks to run, it is possible for timer and keyboard interrupts to be generated.

Table 1-1 PMU Clock Speeds

Mode	High-Speed CPUCLK	Low-Speed CPUCLK	Video Clock	DMA Clock	SYSCLK	8254 Clock (Timer)	16450 Clock (UART)
High-Speed PLL	33/25/20 MHz	9.2 MHz	14.336 MHz	4.6 MHz	9.2 MHz	1.19 MHz	1.8432 MHz
Low-Speed PLL	9.2 MHz	4.608/2.304/ 1.152/0.56 MHz	14.336 MHz	2.3/1.2/0.58/ 0.29 MHz	9.2 MHz	1.19 MHz	1.8432 MHz
Doze	DC[1]	DC[1]	14.3 MHz/ DC[2]	DC[1]	9.2 MHz/ DC[2]	1.19 MHz/ DC[2]	1.8 MHz/DC[2]
Sleep	DC	9.2 MHz/ DC[4]	14.3 MHz/ DC[2]	4.6 MHz/ DC[4]	DC	1.19 MHz/ DC[2]	1.8 MHz/DC[2]
Suspend	DC	9.2 MHz/ DC[4]	14.3 MHz/ DC[2]	4.6 MHz/ DC[4]	DC	1.19 MHz/ DC[2]	1.8 MHz/DC[2]
Off	DC	9.2 MHz/ DC[4]	14.3 MHz/ DC[3]	4.6 MHz/ DC[4]	DC	1.19 MHz/ DC[3]	1.8 MHz/DC[3]

Notes:

The DMA clock can be stopped except during DMA transfers. The Function Enable 1 register at Index B0h controls this function.

The CPU clock speed in Low-Speed PLL mode is selectable. For additional information, see "PMU Control 3 Register (Index ADh)" on page 4-60.

1. *Can be programmed to run intermittently (on the IRQ0 pin) at 9.2 MHz.*
2. *Is a programmable option, but not on a per-clock basis—all clocks with this note are controlled by a single on/off select for that PMU mode.*
3. *Is a programmable option—reflects the setting in Suspend mode.*
4. *Can be programmed to run at 9.2 MHz during temporary-on NMI or SMI handlers.*

Through an option enabled by setting bit 3 of the MMSB Control register at Index 74h, the PMU can be programmed to periodically start the low-speed CPU clock when the IRQ0 pin (generated by the 8254 timer) is asserted. By default, the clock runs only while IRQ0 is active or the ISR0 bit is High, stopping on the next refresh after this condition is no longer valid. In this case, the CPU clock runs at 9.2 MHz. The 8259 interrupt controller must be programmed to unmask IRQ0. By setting bit 0 of the PMU Control 2 register at Index AFh, the run time may be extended for 64 refresh cycles after ISR0 goes Low.

By setting bit 7 of the Power Control 1 register at Index 80h, the low-speed PLL and video PLL can be shut down in Doze mode. Shutting down the low-speed PLL also shuts down the 8254 timer clock, the UART clock, and the keyboard clock; therefore, the IRQ0 wake-up cannot be used in this instance. The high-speed PLL is always shut down in this mode.

The PMC pins may be programmed to a specific state for this mode. An NMI or SMI may be generated upon entering Doze mode. The CPUCLK signal runs at 9.2 MHz during the NMI or SMI handler.

1.1.1.4 Sleep Mode

Sleep mode is the third level of power conservation. In addition to the clocks disabled in Doze mode, the keyboard clock (external SYSCLK signal) is disabled, regardless of whether the low-speed PLL is enabled. Sleep mode is entered after a programmable time without activity has elapsed. For details, see “Doze to Sleep Timer Register (Index 85h)” on page 4-44.

In this mode, the CPU, system, and DMA clocks are stopped and cannot be restarted unless one of the following events occurs:

- An SMI or NMI on $\overline{BL1}$ or $\overline{BL3}$ (when enabled)
- An SMI or NMI generated on a change to Suspend mode
- A wake-up event causes an exit from Sleep mode to High-Speed PLL mode

In the first two cases, the clock runs only during the SMI or NMI routine and then stops again. The keyboard clock is also shut down and can only be restarted by waking up to High-Speed PLL mode. The high-speed PLL is always shut down in Sleep mode.

By setting bit 3 of the Power Control 2 register at Index 81h, the low-speed PLL and video PLL also may be shut down in this mode. In this case, the low-speed and video PLL is restarted before responding to an SMI or NMI or changing to High-Speed PLL mode. When changing to High-Speed PLL mode, the high-speed PLL is also restarted. Note that the low-speed PLL is divided to generate the 8254 timer clock.

The PMC pins may be programmed to a specific state for Sleep mode. An NMI or SMI may be generated upon entering Sleep mode, in which case the handler runs at 9.2 MHz.

1.1.1.5 Suspend Mode

With regard to the clocks and PLLs, Suspend mode has the same functionality as Sleep mode. But bit 7 of the Power Control 2 register at Index 81h enables shutdown of the video and low-speed PLLs in Suspend mode. The distinction between Suspend and Sleep mode is in the way the external Power-Management Control (PMC) pins behave and may be programmed to behave. The PMC pins may be programmed to a specific state for this mode. An NMI or SMI may be generated upon entering Suspend mode, in which case the handler runs at 9.2 MHz.

1.1.1.6 Off Mode

Off is a powered-down mode in which the Programmable General-Purpose 2 and 3 (PGP2 and PGP3) pins are set to a predefined state, and memory refresh may be disabled. The state of the PGP pins is determined by the General-Purpose I/O 2 and 3 registers at Indexes 94h and 95h, and the General-Purpose I/O Control register at Index 91h.

The system cannot be programmed to enter Off mode directly. The only method of Off mode entry is by expiration of the Suspend to Off Mode Timer register at Index 87h. When this happens, the PMU state machine is left in Suspend mode, and an internal, nonreadable flip-flop is set, indicating Off mode. An NMI or SMI may be generated upon entering Off mode.

Refresh may be programmed to be disabled when the PMU is in Off mode. Setting bit 7 of the Memory Configuration 2 register at Index B9h causes the $\overline{RAS}$ and $\overline{CAS}$ outputs to be driven Low when the PMU is in Off mode. The system logic should power off the DRAM in this mode, or the Low $\overline{RAS}$ and $\overline{CAS}$ outputs may keep the row buffers enabled,

thus drawing additional power from the DRAM devices. DRAM content is invalid when exiting from Off mode when the disable-refresh feature is being used.

1.1.2 PMU Operating-Mode Transitions

Figure 1-1 on page 1-4 shows the ÉlanSC310 microcontroller's six operating modes and the transitions that can occur between them. In this diagram, the term *inactivity*, followed by a number from 1 to 5, refers to different timer intervals of inactivity, which are set by programming the Mode Timer registers at Indexes 83–87h. The term *activity* refers to the programmer-specified computing activities set in the following registers:

- Activity Mask 1 and 2 registers at Indexes 75–76h
- I/O Activity Address 0 and 1 registers at Indexes 8C–8Dh
- Memory Write Activity Lower and Upper Boundary registers at Indexes 9A–9Bh
- PMU Control 1–3 registers at Indexes A7h, AFh, and ADh

For more information, see the programming examples later in this chapter. The $\overline{\text{SUS}}$/$\overline{\text{RES}}$ pin is an external pin that can be triggered to cause a transition between modes.

Each power-management mode is characterized by a different clock-frequency pattern—the CPU, system bus, and many peripheral devices have their own clocks. Several clock-switching permutations are possible through the use of the PMU modes. The PLLs act as sources for other clocks as follows:

- **High-Speed PLL** Generates the high-speed CPU/memory clock only. It can be programmed to run at either 40, 50, or 66 MHz, yielding an internal CPU operating frequency of 20, 25, or 33 MHz, respectively. The PMU state machine has controls that can disable this clock to prevent it from being used or, in addition, turn off the PLL.
- **Low-Speed PLL** Is divided to generate the following clocks:
  - — Low-speed CPU/internal system clock
  - — 8254 timer clock
  - — 8250 UART clock
  - — 9.2-MHz keyboard-controller clock/external SYSCLK

  The PMU state machine has controls that can disable some of these clocks to prevent them from being used or, in addition, turn off the PLL entirely.
- **Video PLL** Generates the clock that for a 14.336 Mhz signal that can, under program control, be driven on the ÉlanSC310 microcontroller's X1OUT or X14OUT pins. This signal can be used as a clock for an LCD display. The disable for this clock is shared with the low-speed PLL. If the low-speed PLL is turned off, the video PLL is also turned off.

As shown in Table 1-1 on page 1-6, in High-Speed PLL mode, the CPUCLK signal can be programmed to run at 33 MHz, 25 MHz, 20 MHz, or 9.2 MHz during DRAM, local-bus, fast-ROM, and idle cycles. In all other cases, the CPUCLK signal runs at 9.2 MHz.

In Low-Speed PLL mode, the CPUCLK signal can be programmed to run at 4.6 MHz, 2.27 MHz, 1.13 MHz, or 0.568 MHz. The CPUCLK signal cannot run at 9.2 MHz in Low-Speed PLL mode, and it always runs at the same speed regardless of the type of cycle.

1.1.3 PMU Clock Sources

The ÉlanSC310 microcontroller's PMU uses the 32-kHz clock to derive its internal timing. This clock runs off the ÉlanSC310 microcontroller's internal oscillator, which cannot be disabled. Many events are synchronized with the internal refresh signal, which by default is derived from the 32-kHz clock. If the ÉlanSC310 microcontroller's PMU is being used in a system design, the refresh clock must not be set to Timer Channel 1 because the timer is disabled in some PMU modes.

The remainder of this section describes the functionality of the system during each of the PMU states. Later sections discuss the different ways the PMU can be caused to enter these states.

1.1.4 Reading the PMU Mode

The current PMU mode can be read from the CPU Status 1 register at Index A4h. If the PMU is in the Off mode, this register indicates Suspend mode. PMU mode changes always take effect on the next refresh after the mode change was registered. The mode that is read from the CPU Status 1 register at Index A4h is one refresh delay in advance of the internal signals that actually execute the functions of the PMU mode. Therefore, if it is necessary to know the exact mode of the PMU at a specific time (e.g., for the purpose of determining the state of the PMC signals), the software must read the CPU Status 1 register on two successive refreshes and verify that the mode has not changed.

1.1.5 Merging of PMU Modes

Although six PMU modes are defined, the system designer may reduce the effective number of PMU modes by defining identical functions for some of the modes. For example, assuming that Full-ISA or Local-Bus modes are being used, a three-mode system that effectively merges the Doze, Sleep, Suspend, and Off modes can be achieved. To merge the Doze, Sleep, Suspend, and Off modes, use the following procedure:

1. Set bit 7 of the Power Control 1 register at Index 80h and bits 3 and 7 of the Power Control 2 register at Index 81h to disable the low-speed and video PLLs in Doze, Sleep, and Suspend modes.
2. Set all the PMC bits to the same value for Doze, Sleep, and Suspend modes.
3. Do not enable the PGP2 and PGP3 pins to change in Off mode.

The net effect of this procedure is to create a three-mode system, effectively consisting of High-Speed PLL, Low-Speed PLL, and Suspend modes, where Doze, Sleep, Suspend, and Off modes have been merged into a single new *pseudo-Suspend* mode.

1.1.6 Programming Example: Power-Management Setup

Neglecting peripheral control for the moment, the first step in setting up power-management for a system is to define what constitutes *activity.* The following events constitute activity:

- DMA requests and interrupt requests (DRQ, IRQ)
- Keyboard, LPT, COM, and programmable I/O port accesses
- MMS, video memory, and programmable memory range accesses
- Hard disk drive and floppy disk drive accesses
- AC adapter active (rising edge of ACIN)

The second step is to define the absence of activity as *inactivity*. The third step is to define the time intervals of inactivity to be allowed before the system automatically shifts itself to lower levels of power consumption. For this example, the inactivity time intervals used are shown in Table 1-2 on page 1-10.

Table 1-2 Inactivity States and Transition Intervals

State	From This Mode	To This Mode	Interval
Inactivity 1	High-Speed PLL	Low-Speed PLL	1⁄16 s
Inactivity 2	Low-Speed PLL	Doze	10 s
Inactivity 3	Doze	Sleep	10 min
Inactivity 4	Sleep	Suspend	10 s
Inactivity 5	Suspend	Off	1 hr

The settings in Table 1-3 on page 1-11 implement these steps. In this case, activity is considered to include most conventional interrupts and peripheral accesses. However, accesses to special I/O port ranges or memory ranges are not considered activity. The inactivity timings shown are fairly conventional for notebook and palmtop computers.

Table 1-3 Power-Management Setup

Instruction	Ports	Index and Data	Comment
IOW IOW	022h 023h	75h 0100 0000	Activity: DMA requests, IRQ15–IRQ2 (except IRQ8 and IRQ4–IRQ3), keyboard, and MMS Inactivity: AC adapter
IOW IOW	022h 023h	76h 1011 0000	Activity: Hard disk drive, floppy disk drive, LPT, COM I/O, and video memory accesses Inactivity: Programmable I/O ports and memory ranges
IOW IOW	022h 023h	08h 1110 0011	Activity: IRQ8 and IRQ4–IRQ3 (these wake up the system from Sleep, Suspend, and Off modes)
IOW IOW	022h 023h	AFh 1100 0000	Set the granularity of the Low-Speed to Doze Timer register to 1⁄4 s and the High-Speed to Low-Speed Timer register to 1⁄16 s, respectively.
IOW IOW	022h 023h	83h 0000 0001	Set the High-Speed to Low-Speed Timer register to 1⁄16 s (High-Speed to Low-Speed mode transition).
IOW IOW	022h 023h	84h 0010 1000	Set the Low-Speed to Doze Timer register to 10 s (Low-Speed to Doze mode transition).
IOW IOW	022h 023h	85h 1001 0110	Set the Doze to Sleep Timer register to 10 min (Doze to Sleep mode transition).
IOW IOW	022h 023h	86h 1010 0000	Set the Sleep to Suspend Timer register to 10 s (Sleep to Suspend mode transition).
IOW IOW	022h 023h	87h 0011 1000	Set the Suspend to Off Timer register to 59 min, 44 s [1 hr] (Suspend to Off mode transition).

Note:

For this and subsequent examples, the index registers cannot be programmed directly, but must be accessed by writing the location of the index register to address 22h and the data for the index register to address 23h.

For example, the first I/O access in the above example can either be implemented using 8-bit accesses:

```
mov al,75h
out 22h,al
mov al,40h
out 23h,al
```

or 16-bit accesses:

```
mov ax,4075
out 22h,ax
```

1.1.7 Programming Example: Peripheral-Device Power

The simplest way to control peripheral devices is to integrate their control into the above mode-based system. The ÉlanSC310 microcontroller allows five external Power-Management Control (PMC) pins to be activated in such a way that they can indicate to external devices the mode of the system. Simple logic can be used to disable the peripheral when the system is in specific modes.

As a concrete example, consider the previous example, which has several peripheral devices (floppy disk drive, serial port, etc.). Suppose the following peripheral-power-management scheme is to be implemented:

- 10 s of inactivity—Turn off the floppy disk drive
- 10 min of inactivity—Turn off the serial port transceiver

This scheme can be implemented by using Doze mode to signal the floppy disk drive, and Sleep or Suspend mode to signal the transceiver. The PMC pins can be programmed to provide this control through the Power Control 1–4 registers at Indexes 80–81h and AB–ACh.

The settings shown in Table 1-4 on page 1-12 provide the following scheme:

- PMC1 is activated when the system is in High-Speed PLL or Low-Speed PLL mode
- PMC2 is activated when the system is in Doze mode
- PMC3 is activated when the system is in Sleep mode
- PMC4 is activated when the system is in Suspend mode

The designer can use the PMC4–PMC2 pins to disable the floppy disk drive and the transceiver.

Table 1-4 Power-Management Control Pin Settings

Instruction	Ports	Index and Data	Comment
IOW IOW	022h 023h	ACh 0001 0000	Activate PMC1 when in High-Speed PLL or Low-Speed PLL mode.
IOW IOW	022h 023h	80h 0100 0000	Activate PMC2 when in Doze mode.
IOW IOW	022h 023h	ABh 1000 1011	Activate PMC3 when in Sleep mode and activate PMC4 when in Suspend mode.

A more elaborate system permits the control of each peripheral on an individual basis. The settings shown in Table 1-5 on page 1-12 extend the PMC initialization example to show how to power down an external device after 8 s. The following example shows how the ÉlanSC310 microcontroller can be programmed to make the PMC1 pin change state after 8 s of system inactivity. The designer can use this signal to turn off power to the external device.

Table 1-5 PIO Timeout Settings

Instruction	Ports	Index and Data	Comment
IOW IOW	022h 023h	45h 1100 000	Set PIO address to the window at port 300h.
IOW IOW	022h 023h	46h 0100 0110	Set the I/O window size to 8 bytes and the timeout to 8 s.

If the time-out period expires, the PMC1 pin is pulled Low, which causes the external device's power to be turned off. If the program then attempts to access the external device via I/O addresses in the range 300–307h, then an SMI is generated. This procedure allows the program to turn on external device power before reissuing the access. The settings shown in Table 1-6 on page 1-13 enable SMI generation for PIO accesses to that address range.

Table 1-6 SMI-Generation Settings for PIO Accesses

Instruction	Ports	Index and Data	Comment
IOW IOW	022h 023h	A9h 1100 xxxx	Set the SMI Memory-Mapping System (MMS) (bits 23–22) to 00; enable SMI MMS.
IOW IOW	022h 023h	AAh 1000 0000	Set the SMI MMS to 200000h.
IOW IOW	022h 023h	43h 0000 0000	Reset the SMI Status register.
IOW IOW	022h 023h	41h 0000 0100	Enable SMI for PIO accesses.

Note:

In this and subsequent examples, the notation xxxx refers to a field of bits whose value must be preserved. That is, the programmer must execute an I/O-read–modify–I/O-write cycle to ensure that the current contents of this field are not changed.

An SMI causes the CPU to store its internal state at location 060000h, which in this example is mapped by the Memory Mapping System to 200000h. The CPU begins executing at the reset vector, and the code executes the SMI handler after it checks the SMI flag.

By checking the SMI Status register at Index 43h, the SMI handler determines that a PIO access caused the SMI. Then the handler can turn on the external device's power by setting bit 4 of the Power Control 4 register at Index ACh, which causes the PMC1 pin to be reset, thus restoring power to the device.

1.2 EXTERNAL-DEVICE CONTROL INTERFACE

The external-device control interface includes the following pins:

- Power-Management Control (PMC4–PMC0)
- Programmable General-Purpose (PGP3–PGP2)
- Latched Power ($\overline{\text{LPH}}$)

1.2.1 Power-Management Control Pins

The Power-Management Control (PMC4–PMC0) pins may be used for either or both of the following purposes:

- General-purpose control of external devices in conjunction with the PMU state machine
- Timer-controlled shutdown of a floppy disk drive (addresses 3F0–3F7h), hard disk drive (addresses 1F0–1F7h), or user-specified I/O device in conjunction with the ÉlanSC310 microcontroller's SMI interface.

1.2.1.1 General-Purpose Control Using the PMU State Machine

When using the PMC pins for general-purpose control, PMC4 and PMC2–PMC0 are noninverting and drive a 0 at reset. PMC3 is inverting and will drive a 1 at reset. Each pin may be programmed to drive a unique state in High-Speed PLL, Low-Speed PLL, Doze, Sleep, and Suspend modes. Pin state switching occurs on the next refresh cycle after a PMU state change. The signals are not synchronous and may glitch when changing to High-Speed PLL mode. Internal gray encoding prevents glitching when the PMU states are cycling down sequentially from High-Speed PLL mode. Table 1-7 on page 1-14 summarizes the functionality of these pins.

Table 1-7 PMC Pin Functionality

PMC Pin No.	Control Register	Output Sense	SMI Timer Function
0	Index ACh, bits 3–0	Noninverting	Floppy disk drive (3F0–3F7h)
1	Index ACh, bits 7–4	Noninverting	Programmable I/O address
2	Index 80h, bits 6 and 2 Index 81h, bits 6 and 2	Noninverting	(None)
3	Index ABh, bits 3–0	Inverting	(None)
4	Index ABh, bits 7–4	Noninverting	Hard disk drive (1F0–1F7h)

1.2.1.2 Timer-Controlled Shutdown Using the SMI Interface

When using PMC4 or PMC1–PMC0 for timer-controlled shutdown of devices, the SMI timer logic must be enabled through the SMI Enable register at Index 41h. In addition, the bits in the General-Purpose I/O 0–1 and 4 registers that correspond to the modes in which the devices normally run should be programmed to 1. A PMC output pin goes High to enable a device. The pin goes Low to disable a device under any of the following conditions:

- The SMI device timer expires.
- The PMU enters Sleep, Suspend, or Off mode.

- The PMU enters another mode for which the device is programmed to turn off.

It is also important to understand that an expiring SMI device timer clears all four of the PMC control bits for that device by a short pulse generated when the timer expires. In addition, the control-register bits for that device are held in reset when the PMU is in Sleep, Suspend, or Off mode. These conditions only apply to PMC pins that have an SMI device timer enabled by the SMI Enable register at Index 41h. For more information, see “Accesses to Powered-Down Device SMI” on page 1-29.

1.2.2 Programmable General-Purpose Pins 2 and 3

Through an option enabled via the General-Purpose I/O 2 and 3 registers at Indexes 94–95h and the General-Purpose I/O Control register at Index 91h, the PGP3–PGP2 pins can be enabled to switch from High to Low when the PMU enters Off mode. This option can be enabled as follows:

- **PGP2** Clear bits 4 and 5 of the General-Purpose I/O Control register at Index 91h, and set bit 7 of the General-Purpose I/O 2 register at Index 94h.
- **PGP3** Clear bits 6 and 7 of the General-Purpose I/O Control register, and set bit 7 of the General-Purpose I/O 3 register at Index 95h.

This setup causes the PGP pins to default to 1. When the PMU enters Off mode, the PGP2 or PGP3 pin is driven by the inverse of bit 7 of the General-Purpose I/O 2 register or the General-Purpose I/O 3 register, respectively. Because the PMU timer delay to Off mode can be set to as long as 256 min, this feature can be used to turn off a device after a prolonged period of inactivity.

1.2.3 Latched Power Pin

The Latched Power ($\overline{\text{LPH}}$) pin can be used to indicate a low battery. The default state of $\overline{\text{LPH}}$ is 0. When enabled by setting bit 7 of the MMSB Control register at Index 74h, a 0 on the Battery Level 4 ($\overline{\text{BL4}}$) input pin causes $\overline{\text{LPH}}$ to drive a 1, provided that the ACIN (AC Input Active) pin is also 0.

1.3 CLOCK-SWITCHING LOGIC

The ÉlanSC310 microcontroller's clock-switching logic handles the task of switching clock speeds as directed by the PMU or other input, and sequencing the shutdown and startup of the clocks and PLLs.

1.3.1 CPU/Memory Clock Switching

The CPU clock-switching circuit delivers a signal that switches cleanly between the high-speed PLL clock and a low-speed clock source in High-Speed PLL mode. The low-speed clock source switches cleanly between the low-speed PLL clock (9.2 MHz) and a selectable slow clock on a PMU state transition (see Table 1-1 on page 1-6).

The high-speed PLL clock is used only under narrowly defined conditions. The high-speed clock must be enabled by setting bit 6 of the I/O Wait State register at Index 61h. Doing this enables the use of the high-speed clock after the next refresh cycle if the high-speed PLL is already started.

In addition, the PMU must be in High-Speed PLL mode and the current bus cycle must be one of the following types of cycles:

- CPU idle
- Local DRAM
- Fast ROM
- Local bus

Use of the high-speed clock is disallowed when all of the following conditions are true:

- $\overline{BL1}$ is Low
- Bit 5 of the PMU Control 2 register at Index AFh has been set to enable this feature
- ACIN is Low

If the auto low-speed logic is enabled, use of the high-speed clock is disallowed periodically to conserve power. In cases where the use of the high-speed PLL clock is disallowed, the low-speed PLL's 9.2-MHz CPU clock is used.

When the PMU is in Low-Speed PLL mode, the CPU clock is generated from the low-speed clock source. This is a programmable divider chain controlled by bits 1–0 of the PMU Control 3 register at Index ADh which provides a clock frequency of 9.216 MHz that is divided by 2, 4, 8, or 16. These bits should be changed only when the PMU is not in Low-Speed PLL mode.

When a temporary-on condition occurs while the PMU is in Doze, Sleep, Suspend, or Off mode—and the CPU/memory clock is enabled to run—the 9.2-MHz CPU clock is used.

The high-speed PLL clock frequency may be selected by writing to bits 4–3 of the Function Enable 2 register at Index B1h. These bits should not be changed when the high-speed clock is enabled.

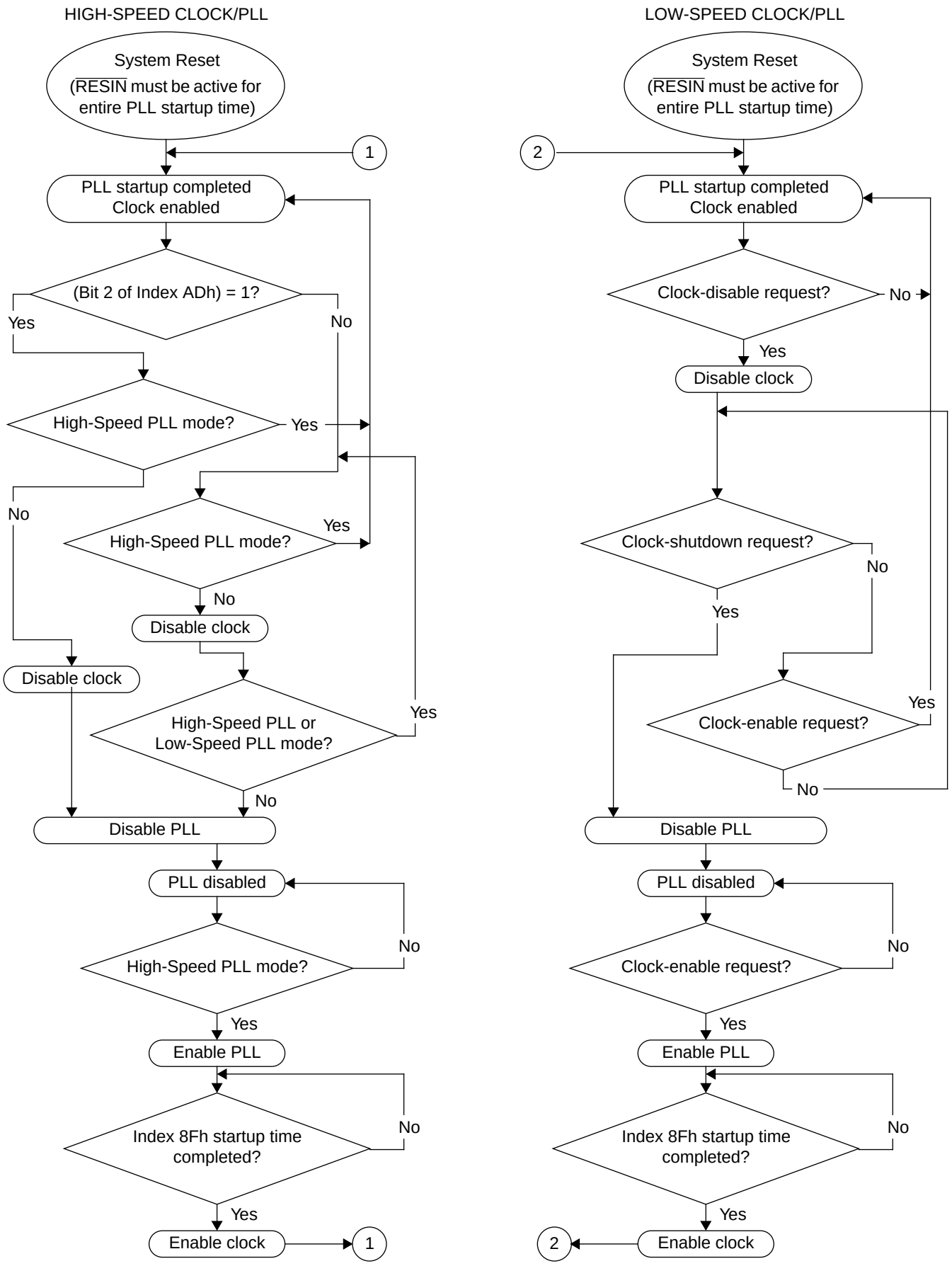
1.3.2 Clock Startup and Shutdown Logic

The clock startup and shutdown logic provides a mechanism for properly coordinating the activation of the ÉlanSC310 microcontroller's PLLs with the CPU's clock-enabling logic. The flowchart in Figure 1-2 on page 1-18 illustrates the logic flow. For the high-speed clock, there is a choice of turning off the PLL in either Low-Speed PLL mode or Doze

mode, selectable through bit 2 of the PMU Control 3 register at Index ADh. For the low-speed clock, there are a number of options with regard to when the PLL is turned off. These options are controlled using bits 7 and 3 of the Power Control 1 and 2 registers at Indexes 80–81h. The PLL startup time must be programmed in the Clock Control register at Index 8Fh to allow sufficient startup time for the PLL before the clocks are engaged. It is recommended that the startup time be programmed to at least 256 ms when starting from a PMU mode where the low-speed PLL is disabled. A startup time of 128 ms can be used if the low-speed PLL will never be disabled.

If the PMU is returning to High-Speed PLL mode from a mode where the high-speed PLL was disabled but the low-speed PLL was enabled, the CPU clock will begin operating at 9.2 MHz immediately. After the PLL startup time expires, the CPU clock switches to the high-speed PLL frequency if enabled to do so via bit 6 of the I/O Wait State register at Index 61h.

Figure 1-2 PLL Control Flowchart



1.4 ACTIVITY MONITORS

The activity-monitor logic keeps track of events that indicate that the CPU or peripherals are in demand. Examples of such events are I/O or memory decodes to certain addresses, DMA requests, interrupts, and changes in status signals. When an activity event is recognized, the PMU state machine immediately switches to High-Speed PLL mode on the next refresh cycle. The exception to this rule is that activities are not recognized during execution of SMIs or NMIs. All PMU activities are edge detected.

Activities that are referred to in Table 1-3 on page 1-11 can be classified as CPU-related and non-CPU-related. The CPU-related activities that the ÉlanSC310 microcontroller's PMU is able to count as events are selected via the Activity Mask 1 and 2 registers at Indexes 75–76h, with status read from the Activity Status 1 and 2 registers at Indexes A0–A1h.

The non-CPU-related activities that the PMU is able to count as events are selected using the Resume Mask register at Index 08h, with status read from the Resume Status register at Index 09h. Non-CPU-related activities are also classified as wake-up events. Note that it also is possible for some of the CPU-related activities to wake up the PMU. Both types of wake-ups are discussed in "Wake-Up Logic" on page 1-23.

Detected activities in the Activity Status 1 and 2 registers at Indexes A0–A1h and the Resume Status register at Index 09h are indicated by a 1 in the appropriate bit. After reading one of these registers, software should write any value to clear the register.

Figure 1-3 on page 1-20 shows how activities play a part in the power-management flow.

Figure 1-3 State Transition Flowchart

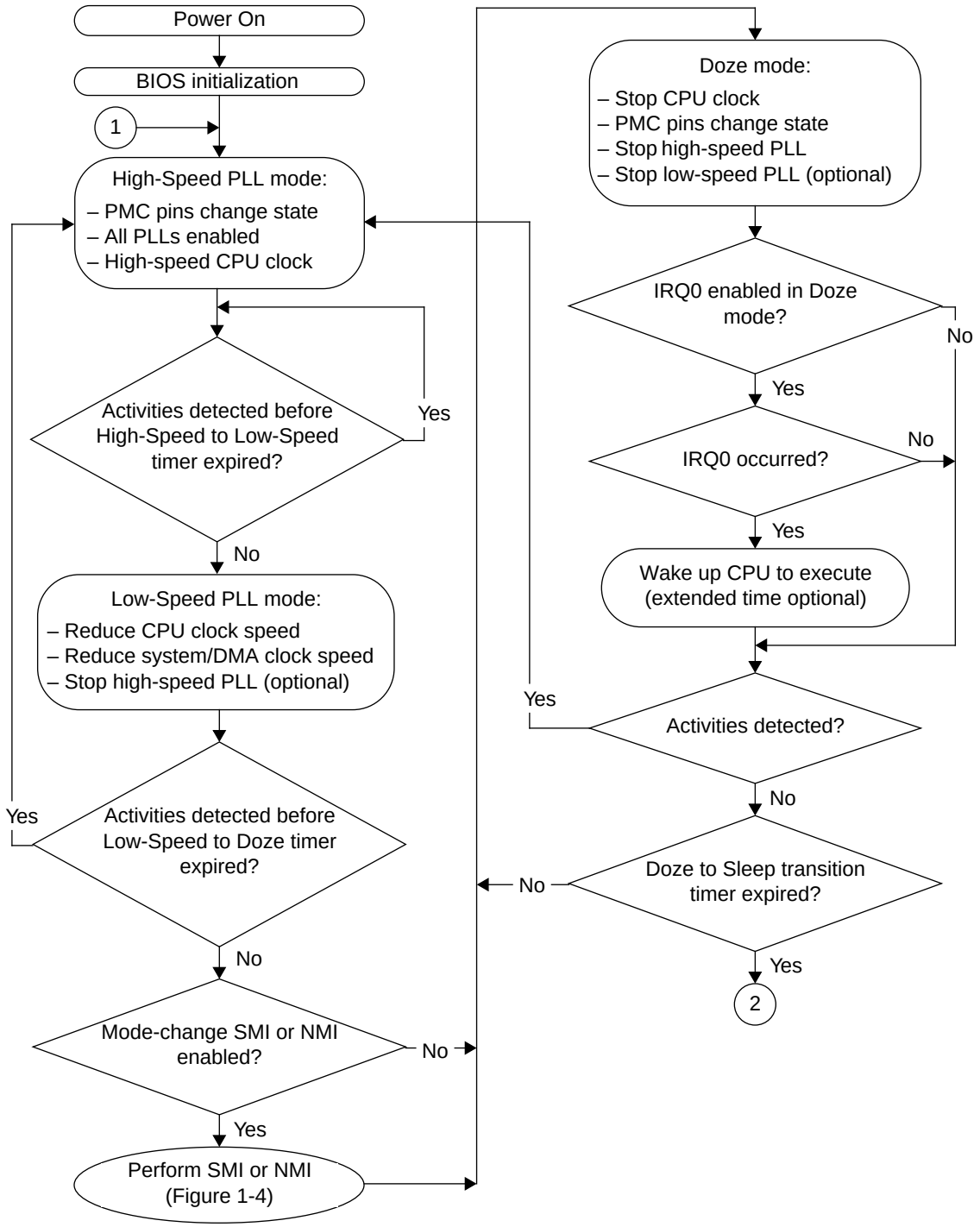
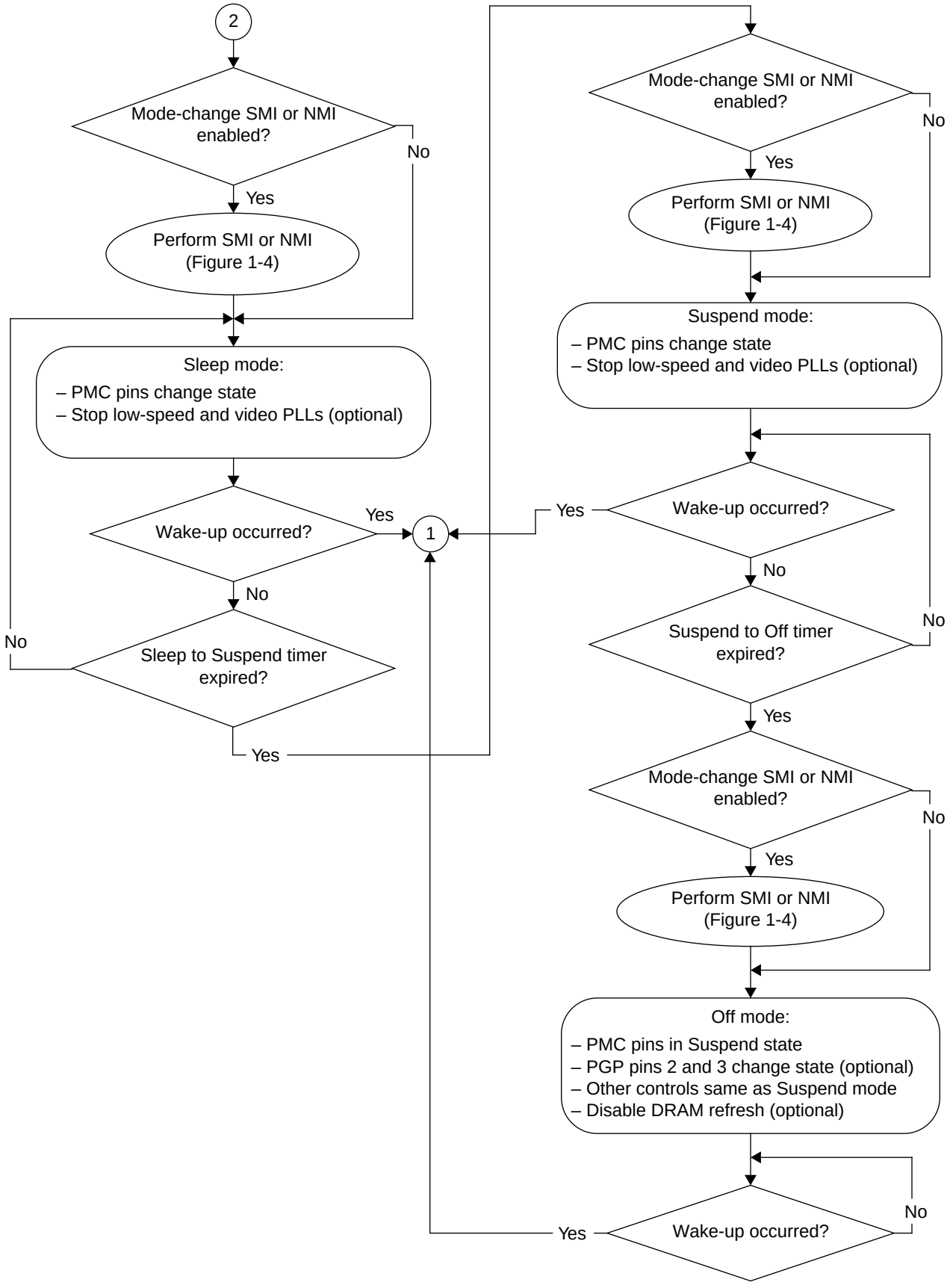


Figure 1-3 State Transition Flowchart (continued)



1.5 STATE-TRANSITION TIMER

The state-transition-timer logic allows the system designer to specify the amount of time that the PMU waits between state transitions when no activity is occurring. As shown in Table 1-3 on page 1-11, the time values are programmed through the Mode Timer registers at Indexes 83–87h.

The High-Speed to Low-Speed Timer register at Index 83h is used in High-Speed PLL mode. By default, it is programmed in increments of 1⁄512 s, with a minimum time of 1⁄512 s when set to a value of 1 and a maximum time of 498 ms (1⁄512 • 255) when programmed to FFh. If bit 6 of the PMU Control 2 register at Index AFh is 1, the time increments for this register change to 1⁄16 s, for a minimum time of 1⁄16 s and a maximum time of 15.94 s. When this time expires, the PMU changes to Low-Speed PLL mode.

The Low-Speed to Doze Timer register at Index 84h is used in Low-Speed PLL mode. By default, it is programmed in increments of 1⁄16 s, with a minimum time of 1⁄16 s when set to a value of 1 and a maximum time of 15.94 s when programmed to FFh. If bit 7 of the PMU Control 2 register at Index AFh is 1, the time increments for this register change to 1⁄4 s, for a minimum time of 1⁄4 s and a maximum time of 63.75 s. When this time expires, the PMU changes to Doze mode.

The Doze to Sleep Timer register at Index 85h is used in Doze mode. It is always programmed in 4-s increments, with a minimum time of 4 s and a maximum time of 1020 s (4 • 255). When this time expires, the PMU changes to Sleep mode.

The Sleep to Suspend Timer register at Index 86h is used in Sleep mode. It is always programmed in 1⁄16-s increments, with a minimum time of 1⁄16 s and a maximum time of 15.94 s (1⁄16 • 255). When this time expires, the PMU changes to Suspend mode.

The Suspend to Off Timer register at Index 87h is used in Suspend mode. It is always programmed in 64-s increments, with a minimum time of 64 s and a maximum time of 16320 s (64 • 255), equivalent to 4 hr and 32 min. When this time expires, the PMU changes to Off mode. Since Off mode is the last state, there is no timer for it.

The State Transition Timer, also known as the PMU Timer, is an internal timer whose value cannot be read by the system. This timer counts up from 0 and is compared to the Transition Timer register associated with the current PMU mode. When the PMU Timer exceeds the time programmed into the appropriate Mode Timer register, the mode change is initiated and the PMU Timer is reset to 0.

When one of the above registers is cleared, upon reaching the PMU mode corresponding to that register, the PMU timer will reset and stop counting. The PMU will thus remain in that mode unless an activity is detected or the PMU is forced into another state via the $\overline{\text{SUS}}/\overline{\text{RES}}$ pin, $\overline{\text{BL2}}$ or $\overline{\text{BL4}}$ pin, or a write to the Software Mode Control register at Index 88h. Thus, for example, if the High-Speed to Low-Speed Timer register at Index 83h is cleared, the PMU never leaves High-Speed PLL mode unless one of the previously mentioned events occurs.

The PMU timer is also reset by any of the following events:

- An unmasked PMU activity event
- A write to the Software Mode Control register at Index 88h
- An SMI or NMI caused by the battery-level pins, a PMU mode change, or a $\overline{SUS}$/$\overline{RES}$ pin pulse (the timer is held in reset during one of the above SMIs until a write to the NMI/SMI Control register at Index A5h occurs)
- The high-speed or low-speed PLL is started (the timer is held in reset until startup is completed)

1.6 WAKE-UP LOGIC

The ÉlanSC310 microcontroller's wake-up logic provides a mechanism for allowing certain external events to bring the PMU out of Sleep, Suspend, or Off mode into High-Speed PLL mode. These events are defined in the Resume Mask register at Index 08h. See Table 1-8 on page 1-23 for a list of the wake-up signals and associated trigger mechanisms. If the PMU is in Low-Speed PLL or Doze mode, these wake-up events function as activities, returning the PMU to High-Speed PLL mode.

Table 1-8 Wake-Up Signal Descriptions

Signal	Trigger	Description
DRQ1	Rising edge	Active until DRQ1 goes Low. Not maskable at the DMA controller.
DRQ2	Rising edge	Active until DRQ2 goes Low. Not maskable at the DMA controller.
DRQ3	Rising edge	Active until DRQ3 goes Low. Not maskable at the DMA controller.
DRQ5	Rising edge	Active until DRQ5 goes Low. Not maskable at the DMA controller.
DRQ6	Rising edge	Active until DRQ6 goes Low. Not maskable at the DMA controller.
DRQ7	Rising edge	Active until DRQ7 goes Low. Not maskable at the DMA controller.
IRQ1	Rising edge	Active until the keyboard controller deasserts IRQ1. Not maskable at the PIC.
IRQ3	Rising edge	Active until IRQ3 goes Low or channel 3 receives INT/ACK and EOI. Maskable at the PIC.
IRQ4	Rising edge	Active until IRQ4 goes Low or channel 4 receives INT/ACK and EOI. Maskable at the PIC.
IRQ8	Rising edge	Active until IRQ8 goes Low or channel 8 receives INT/ACK and EOI. Maskable at the PIC.
RI	Falling edge	Active only until the next refresh. Clearing bit 5 of the Resume Status register at Index 09h prior to the next refresh may not clear the $\overline{RI}$ activity status.

The events that may be allowed are Ring-Indicate signals from the internal UART, as well as IRQ3 or IRQ4 (from the internal UART or an external pin, depending on the system configuration) and IRQ8 (from the internal RTC only). In addition to the events defined in the Resume Mask register, a rising edge on DRQ2 could be programmed as a wake-up event through bit 2 of the Activity Mask 1 register at Index 75h. A rising edge on ACIN may be enabled as a wake-up activity (bit 6 of the Activity Mask 1 register and bit 4 of the PMU Control 3 register at Index ADh). And a rising edge on IRQ1 (AT keyboard interrupt) could also be enabled as a wake-up activity (bit 4 of the Activity Mask 1 register). If the

system is configured in Maximum ISA mode, it is also possible to use DRQ1 or DRQ7–DRQ3 as rising-edge wake-up events (bits 0, 2, and 3 of the Activity Mask 1 register). If the system is configured in Local-Bus mode, the choice of DRQs is limited to DRQ1, DRQ2, or DRQ5.

When using IRQ or DRQ pins as wake-up signals, external hardware must hold the active state (High) until the PMU returns to High-Speed PLL mode. (This is not necessary for the Ring-Indicate wake-up signals because they are internally latched.) IRQ wake-up events (IRQ3, IRQ4, and IRQ8) are masked at the PIC and must be enabled in order to generate a wake-up. However, unlike the IRQs, the DRQ events are not masked at the DMA controller. Also, for IRQ wake-ups, care must be taken to ensure signal integrity on the IRQ lines. Glitches on the IRQ lines as small as 4 ns will cause a wake-up to occur. The PMU has no ability to filter glitches; therefore, the system could be returned to High-Speed PLL mode by what amounts to a spurious interrupt.

If using multiple DRQs and/or IRQs as wake-up signals, the system designer must ensure that one wake-up signal does not block the others by being held continuously in the active state while its corresponding PMU-activity-mask bit is set to 0. These wake-up sources are logically ORed before edge detection is performed. Wake-ups received during Temporary-On SMI or NMI routines (see “Temporary-On Mode” on page 1-26) require the following special attention:

1. The wake-up is delayed (not serviced by the PMU logic) until the SMI or NMI routine is completed.
2. The wake-up is serviced and cleared before exiting the SMI or NMI handler when all of the following conditions are true:
   — CPU interrupts are enabled by the routine (to handle other non-wake-up events)
   — There is an associated interrupt service routine (ISR) for the wake-up event
   — The wake-up is not masked at the PIC

Note that the interrupt handler does not have to explicitly clear the wake-up source. The CPU just has to acknowledge the active IRQ channel. This results in a lost wake-up event from the first case. Therefore, if CPU interrupts need to be serviced from within an SMI/NMI handler, the IRQs that are set up as wake-ups must be masked off at the PIC (disabled) at the beginning of the SMI/NMI routine, and unmasked (re-enabled) upon exiting the routine. Make sure CPU interrupts are disabled before re-enabling wake-ups at the PIC.

The ring-in wake-up ($\overline{RI}$) persists only until the next refresh. If software tries to clear the status bits in the Resume Status register prior to that subsequent refresh, the bits may remain set. Software must clear the Resume Status register on two successive refreshes.

An additional wake-up event is the $\overline{SUS}$/$\overline{RES}$ pin. This event is not maskable by the system. For more information, see “Suspend/Resume Pin Logic” on page 1-34.

Figure 1-3 on page 1-20 shows how wake-ups play a part in the power-management flow.

1.7 SMI AND NMI CONTROL

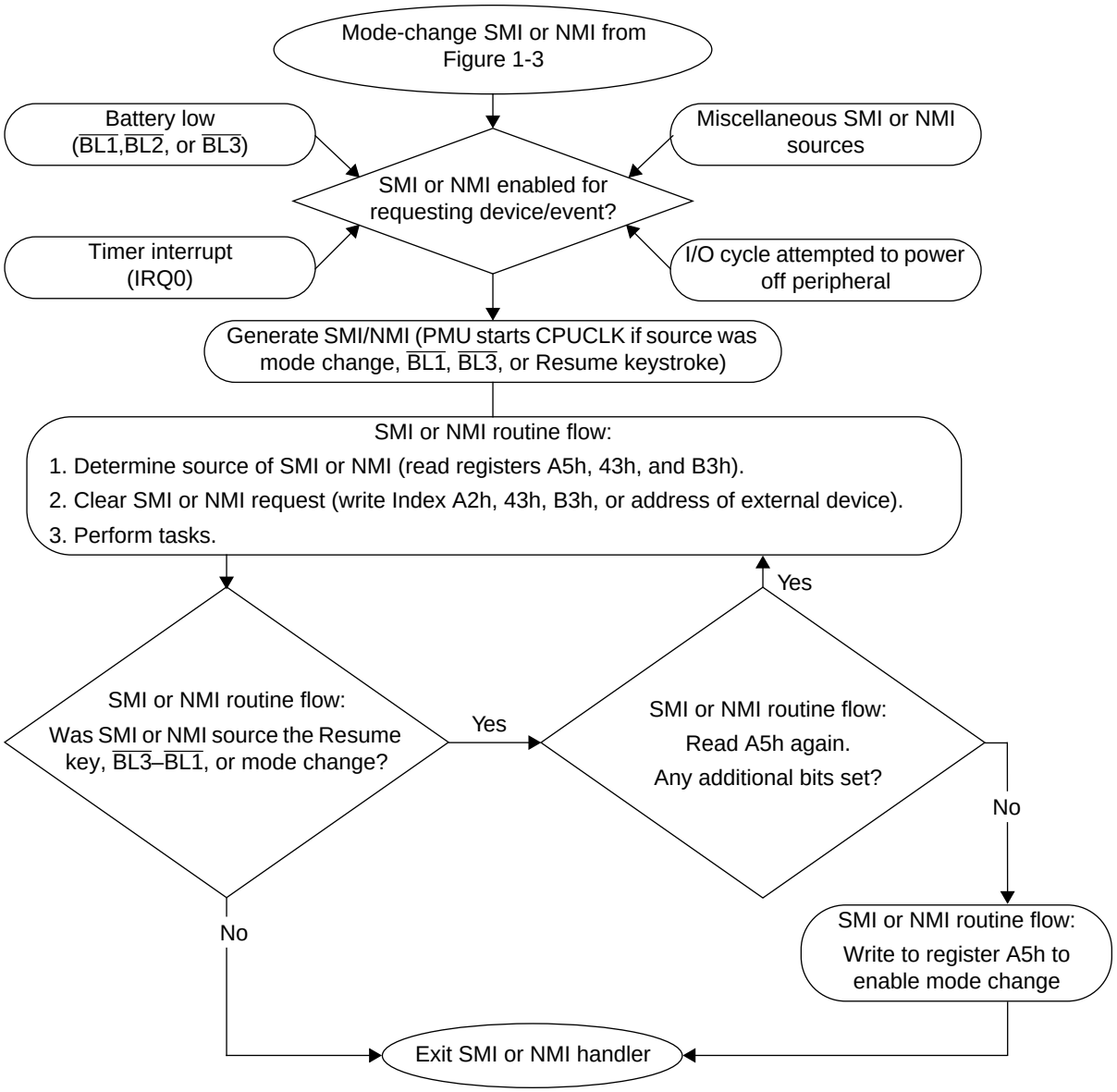
The use of SMIs gives the designer the ability to customize and extend the features of a system's power management. This is due to the transparency of SMIs relative to operating systems and application software. The ÉlanSC310 microcontroller may be programmed to generate SMIs from the following sources:

- PMU mode changes (e.g., on a change from Sleep mode to Suspend mode)
- Battery-level changes ($\overline{BL1}$, $\overline{BL2}$, and $\overline{BL3}$)
- $\overline{SUS}$/$\overline{RES}$ pin activation (only on wake-up)
- Attempted I/O accesses to devices that the PMU has powered down after a specified time-out period
- Interrupts from IRQ0 (Timer Channel 0 must be programmed and IRQ0 must be enabled at the 8259 PIC)
- System-management interrupts on the EXTSMI pin
- I/O accesses to the RTC or the external 8042 keyboard controller

Of the above SMI sources, the first three may be programmed to generate an NMI instead of an SMI.

Refer to Figure 1-4 on page 1-26 for the general flow of an SMI or NMI.

Figure 1-4 SMI Processing Flowchart



1.7.1 Temporary-On Mode

Only the first three SMI or NMI sources listed above will cause the CPU, DMA, and internal system clocks to start if they were stopped. If not in the High-Speed PLL or Low-Speed PLL mode, the clocks will run based on the 9.2-MHz frequency until the next refresh after an I/O write to the NMI/SMI Control register at Index A5h is performed. This is called Temporary-On mode.

This mode is useful when it is desired to have the CPU perform certain tasks before changing PMU modes, or when the CPU receives a low-battery indication or a resume keystroke, regardless of whether it is asleep or awake. For the first two sources, when in Temporary-On mode, the PMU remains in the same state it was in before the CPU clock started. It does not change to High-Speed PLL mode. For the third source, the PMU makes a transition to High-Speed PLL mode before the SMI is generated. Writing to the NMI/SMI Control register causes the PMU to leave Temporary-On mode and return to

normal operation for the mode it was previously in, except that for a mode-change SMI, the PMU will advance to the next PMU mode after leaving Temporary-On mode. Note that if the PMU was in High-Speed PLL mode when receiving a battery-level-change SMI or NMI, it does not slow down the clock, but continues running at High-Speed PLL mode clock rates. While the PMU is in Temporary-On mode, all activities, wake-up events, and PMU mode transition timers are disabled. However, activities and wake-up events are internally latched and cause the appropriate action to be taken when exiting Temporary-On mode.

When the system SMI handler determines that an SMI has been generated from the first three sources, the write to the NMI/SMI Control register at Index A5h should always be performed at the end of the SMI handler. The NMI/SMI Control register should be read just prior to writing to it. If any additional SMI flags are set, they should be processed prior to writing the NMI/SMI Control register.

In some cases, it may be important to ensure that the CPU clocks are stopped prior to exiting the SMI/NMI routine. This prevents the main code from accessing a device that was just powered down in the routine. This can be accomplished by polling the Port B register for a refresh after the write to the NMI/SMI Control register. The clocks are stopped when the refresh occurs. When the clocks start again, the SMI/NMI routine finishes executing.

1.7.2 Enabling SMIs

To enable SMI generation, bit 7 of the MMSB Socket register at Index A9h must be 1. If bit 7 is 0, the first three SMI and NMI sources generate NMIs instead of SMIs. When enabling SMIs, the SMI MMS Page register at Index AAh and bits 4 and 5 of the MMSB Socket register must be programmed to set the MMS page where SMI CPU core data will be preserved. Bit 0 of the Miscellaneous 2 register at Index 6Bh gives the programmer the option of forcing address bit 20 Low during SMI routines.

Table 1-9 on page 1-27 shows the registers that enable the various SMI sources.

Table 1-9 Registers that Enable SMI Sources

Register Name	Index
SMI Enable	41h
NMI/SMI Control	A5h
Function Enable 1	B0h
PMU Control 3	ADh
NMI/SMI Enable	82h

1.7.3 Processing NMI or SMI Source

On reset, firmware must check bit 7 of the Version register at Index 64h to determine if the reset was caused by an SMI. Upon receiving an SMI and entering the SMI handler, the CPU must then poll the various SMI status registers to determine the source of the SMI. The status registers are as follows:

- Miscellaneous 5 register at Index B3h for the EXTSMI pin
- SMI Status register at Index 43h for the following sources:
  - — Hard disk drive
  - — Floppy disk drive
  - — Programmable I/O
  - — 8042 accesses
  - — RTC accesses
  - — IRQ0 generation
  - — PMU status changes

If PMU status changes (bit 5) cause the SMI, the SMI handler must read the NMI/SMI Control register at Index A5h to determine the source of the PMU SMI.

If the SMI or NMI is caused by either PMU status changes or IRQ0 generation, the PMU Status 1 register at Index A2h must be written in order to reset the SMI generation logic. This write must occur after reading the SMI Status register at Index 43h and the NMI/SMI Control register at Index A5h. Failure to follow this order may result in additional SMIs being generated with incorrect status.

If the SMI or NMI was caused by a PMU status change, a write to the NMI/SMI Control register must occur near the end of the routine. Prior to performing this write, the NMI/SMI Control register should be read again to determine if additional PMU status changes need to be processed. If additional PMU status changes need to be processed, the write to the NMI/SMI Control register should be delayed, the PMU Status 1 register should again be written, and any code specific to the new status change should be executed.

A condition exists where the SMI or NMI state-transition status (as reflected in the NMI/SMI Control register at Index A5h) can be different than the PMU's current state. If a state-transition SMI or NMI is received (and subsequently queued by the PMU logic) while the system is executing another state-transition SMI or NMI routine, and the system is forced to another state via the Software Mode Control register at Index 88h, *before* exiting the routine but *after* clearing the NMI/SMI Control register (to allow the state transition), then an incoherency causing the difference can occur.

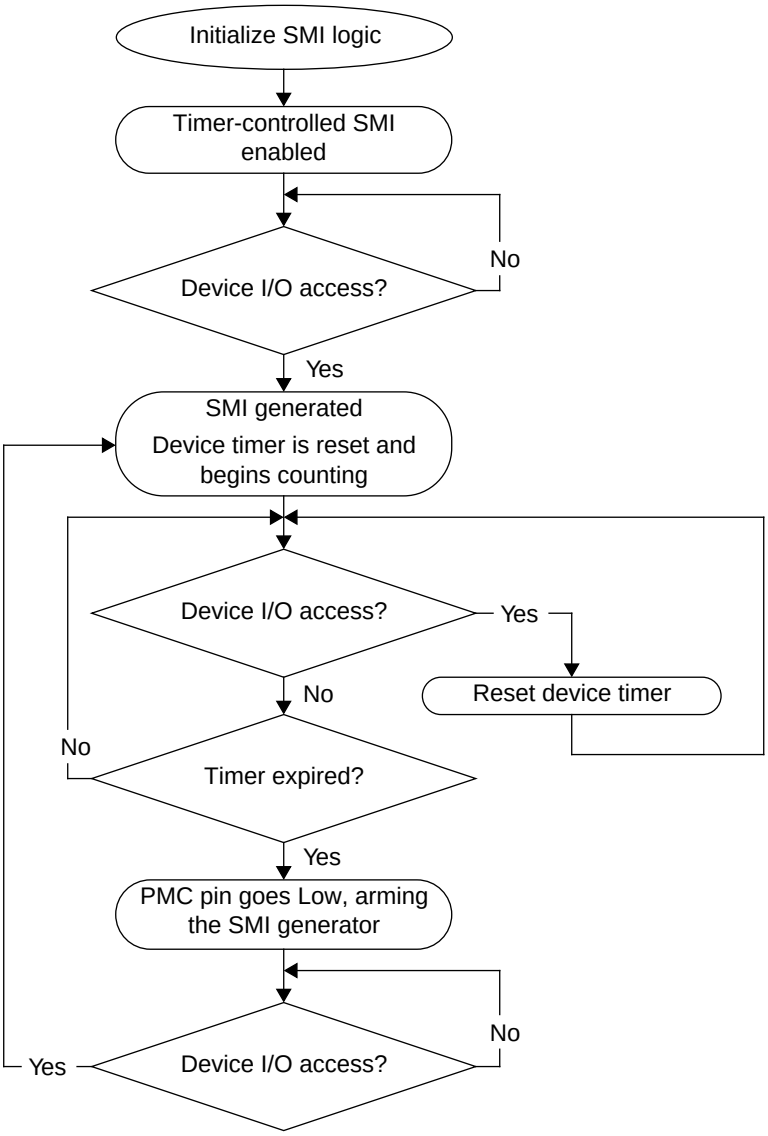
The source of the incoherency lies in the fact that as soon as the current SMI or NMI completes, the CPU begins processing the queued-up state-transition SMI or NMI. Upon entry to the handler, if the code reads the NMI/SMI Control register to determine the source of the SMI/NMI, the PMU reports a state transition based on the mode change *before* the write to the Software Mode Control register, while a read of the current state as shown in the CPU Status 1 register at Index A4h reports the mode that occurred *after* the write to the Software Mode Control register.

To prevent this incoherency from occurring, forced state transitions caused by writes to the Software Mode Control register should not be performed after a write to the NMI/SMI Control register from within a Temporary-On SMI or NMI routine.

1.7.4 Accesses to Powered-Down Device SMI

The ÉlanSC310 microcontroller's automatic device-power-down logic consists of a set of three timers, each dedicated to a specific device at a hardwired (floppy disk drive and hard disk drive) or programmed (PIO) address, and associated with a specific PMC pin (see Table 1-7 on page 1-14). The flowchart in Figure 1-5 on page 1-30 illustrates the operation of this logic. Each timer may be programmed to one of 16 possible time-out values. See "PIO Timer Register (Index 46h)" on page 4-16 and "Drive Timer Register (Index 47h)" on page 4-17. Each timer is cleared when an activity is detected on the assigned address. The addresses for floppy disk drives and hard disk drives are hardwired to the AT standard locations. The PIO Address register at Index 45h is used to set the PIO base address. Bits 6 and 7 of the PIO Timer register at Index 46h set the number of bytes decoded. Optionally, bit 0 (IRQEN) of the Miscellaneous 4 register at Index 44h may be set to enable a hard disk drive interrupt (IRQ14) and a floppy-disk-drive interrupt (IRQ6) to also be considered activities by their respective timers. However, these IRQs will not generate SMIs.

Figure 1-5 SMI Device-Powerdown Flowchart



SMIs from the timer-controlled devices are enabled by bits 2–0 of the SMI Enable register at Index 41h. Once a device is enabled, the first I/O access to that device after enabling SMIs generates an SMI. This initial SMI can be differentiated from subsequent SMIs by virtue of the fact that the PMC power pins have not been cleared. After this first SMI is processed, the timer is enabled. Any I/O access (or optional interrupts) to the device resets the timer. If no accesses occur before the timer expires, the SMI generator is armed and the next I/O access generates an SMI and resets the timer. Expiration of the timer also causes the appropriate General-Purpose I/O register to change to 0, which then causes the device to power down if the PMC pin is used to control device power. In this case, an SMI indicates that an I/O access was attempted to a powered-down device. In the ÉlanSC310 microcontroller, CPU I/O trapping is enabled, so it is possible to retry the attempted I/O instruction after powering up the device by reinitializing the appropriate PMC register and performing whatever device initialization commands are required. The states of address bit 0, $\overline{\text{IOR}}$, $\overline{\text{IOW}}$, and $\overline{\text{BHE}}$ are latched in the SMI I/O Status register at Index 42h at the end of the I/O command that caused the SMI. Note that this applies only

to the SMIs enabled at the SMI Enable register at Index 41h and the Function Enable 1 register at Index B0h. Optionally, firmware may enable the next I/O access to any of the timer-controlled devices to unconditionally cause an SMI by writing a 0 to the appropriate bit in the I/O Timeout register at Index 40h.

I/O trapping for powered-down devices is only applicable when the CPU is not servicing an SMI. Therefore, SMI handlers should poll the I/O Timeout register at Index 40h before accessing I/O devices that may be powered down. If any of bits 2–0 are 0, and the corresponding bits in the SMI Enable register are 1, then I/O accesses to those devices will generate an SMI.

1.7.5 Treatment of Pending SMIs

If an SMI is already in progress, the ÉlanSC310 microcontroller detects and holds as pending another incoming SMI from a different device. When the current SMI routine is finished executing, another SMI is immediately asserted if other SMIs have been generated and not cleared. The exceptions to this are SMIs that are generated from I/O decodes, which include the real-time clock (RTC), keyboard, floppy disk drive, hard disk drive, and PIO. In these cases, another SMI is not generated. However, the status bits in the SMI Status register at Index 43h are 1.

1.7.6 External SMI Pin

The External SMI (EXTSMI) pin is enabled by bit 4 of the Function Enable 1 register at Index B0h. This pin may be used by an external device to generate an SMI. It may be used for a single device or multiple devices as explained below. Pin polarity may be selected using bit 5 of this register.

1.7.7 External SMI with a Single Device

If the EXTSMI polarity bit (bit 5 of the Function Enable 1 register at Index B0h) is 1, a falling edge on EXTSMI causes an SMI request to be generated. EXTSMI should then be held Low by the external device until the SMI handler releases it via an I/O write to the external device. The state of the EXTSMI pin may be read at bit 1 of the Miscellaneous 5 register at Index B3h to determine that EXTSMI has been activated. If the EXTSMI polarity bit is 0, then all of the above polarities are reversed.

1.7.8 External SMI with Multiple Devices

The External SMI (EXTSMI) pin may be treated as an open-drain signal driven by multiple devices. As with a single device, each SMI-generating device should continue asserting EXTSMI until acknowledged by the SMI handler. After determining that the EXTSMI pin is asserted (bit 1 of the Miscellaneous 5 register at Index B3h), the SMI handler must poll each external device and clear its SMI-generating logic if needed. When all external devices are serviced, the EXTSMI pin should return to the inactive state. As long as the EXTSMI pin is active, SMIs to the CPU will be repeatedly generated.

1.8 BATTERY-MANAGEMENT LOGIC

The ÉlanSC310 microcontroller's battery-management logic allows the system designer to specify up to four battery-detection levels. The four battery-level-indication input pins are named $\overline{BL1}$, $\overline{BL2}$, $\overline{BL3}$, and $\overline{BL4}$. A 0 on one of these inputs indicates a low-battery condition for that level. Another pin named ACIN, when High, indicates that external non-battery power has been applied. The following paragraphs give a description of the functionality of each of these pins. Table 1-10 on page 1-32 provides a summary of the functionality of $\overline{BL4}$–$\overline{BL1}$.

Table 1-10 Battery-Level Management Functionality

Pin	Slow Clock	Level SMI	Edge SMI	Force to Sleep	Force to Suspend
$\overline{BL1}$	■	■			
$\overline{BL2}$			■	■	
$\overline{BL3}$		■			
$\overline{BL4}$					■

1.8.1 Battery Level 1

The Battery Level 1 ($\overline{BL1}$) pin is intended to be used as a first-line warning, indicating that battery power is low, but that enough power remains for continued use. The state of this pin may be read directly at the CPU Status 0 register at Index A3h.

Through an option enabled by bit 5 of the PMU Control 2 register at Index AFh, the CPU clock in High-Speed PLL mode can be forced to run at 9.2 MHz on all cycles when $\overline{BL1}$ and ACIN are Low, thus prolonging battery life.

$\overline{BL1}$ may also be programmed to generate SMIs. This feature is enabled by bit 5 of the NMI/SMI Enable register at Index 82h. This is a level-triggered SMI (i.e., the input should be held in the trigger state, not pulsed). After system reset, a level change on $\overline{BL1}$ from 1 to 0 generates an SMI. During the SMI service routine, the CPU must read the NMI/SMI Control register at Index A5h to reset the $\overline{BL1}$ SMI-generation logic. This read also sets the logic to look for the opposite state on $\overline{BL1}$ to trigger the next SMI. Therefore, any change of state on $\overline{BL1}$ generates an SMI, provided that a read from the NMI/SMI Control register is always performed in the SMI service routine. Because the NMI/SMI Control register is also one of the SMI source-polling registers, this read is needed anyway to determine the SMI source. The SMI handler must also write to the PMU Status 1 register at Index A2h to clear the PMU SMI request.

An SMI generated by $\overline{BL1}$ causes the CPU clock to be started, regardless of the PMU mode (see "Temporary-On Mode" on page 1-26). The clock remains running until the next refresh cycle following a write to the NMI/SMI Control register at Index A5h.

1.8.2 Battery Level 2

The Battery Level 2 ($\overline{BL2}$) pin is intended to be used as a second-line warning in a four-level battery-management system, or as a final warning in a two-level system. The state of this pin may be read directly at the CPU Status 0 register at Index A3h.

The PMU may be programmed to automatically enter Sleep mode after a 0 is detected on $\overline{BL2}$ while ACIN is also at 0. This feature is enabled by clearing bit 6 of the MMSB Control register at Index 74h, and is enabled by default. The PMU does not enter Sleep mode instantaneously, but steps down a mode level on each refresh until Sleep mode is reached. Once the PMU is in Sleep mode, the PMU state-transition timers control transitions to Suspend and Off modes. Enabled wake-up/activity events will cause the PMU to change to High-Speed PLL mode. However, the PMU will step down a mode level on each refresh if the $\overline{BL2}$ pin is Low. PMU state transition to Sleep mode can be temporarily interrupted by enabling Low-to-Doze or Doze-to-Sleep mode-change NMI/SMI events in the NMI/SMI Enable register at Index 82h.

SMIs may be generated by $\overline{BL2}$. This feature is enabled by bit 6 of the NMI/SMI Enable register. There are important differences between the SMI behavior of $\overline{BL2}$ relative to $\overline{BL1}$ and $\overline{BL3}$. SMIs are generated only on falling edges of $\overline{BL2}$. During the SMI handler, the CPU must read the NMI/SMI Control register at Index A5h to reset the $\overline{BL2}$ SMI generation logic. The SMI service routine must also write to the PMU Status 1 register at Index A2h to clear the PMU SMI request.

SMIs from $\overline{BL2}$ are masked in Sleep, Suspend, and Off modes and when pin $\overline{BL4}$ is Low. If an SMI from $\overline{BL2}$ occurs in Doze mode and the CPU clock is stopped, the clock is started to execute the SMI. The clock remains running until the next refresh cycle following a write to the NMI/SMI Control register.

1.8.3 Battery Level 3

The Battery Level 3 ($\overline{BL3}$) pin is intended to be used as a third-line warning, indicating that battery power is low, but that enough power remains for limited use. The state of this pin may be read directly at the CPU Status 0 register at Index A3h.

$\overline{BL3}$ may also be programmed to generate SMIs. This feature is enabled by bit 7 of the NMI/SMI Enable register at Index 82h. The functionality of this SMI is analogous to that of the $\overline{BL1}$ SMI.

An SMI generated by $\overline{BL3}$ causes the CPU clock to be started, regardless of the PMU mode (see “Temporary-On Mode” on page 1-26). The clock remains running until the next refresh cycle following a write of 0 to the NMI/SMI Control register at Index A5h.

1.8.4 Battery Level 4

The Battery Level 4 ($\overline{BL4}$) pin is intended to be used as an indication of the end of useful battery life. It cannot be programmed to generate an SMI, and the state of $\overline{BL4}$ cannot be read.

The PMU may be programmed to automatically enter Suspend mode after a 0 is detected on $\overline{BL4}$ while ACIN is also 0. This feature is enabled by clearing bit 7 of the MMSB Control register at Index 74h and is enabled by default. The PMU does not enter Suspend mode instantaneously, but steps down a mode level on each refresh until Suspend mode is reached. Once the PMU is in Suspend mode, the PMU state-transition timers control when (or if) there will be a transition to Off mode. After the PMU has been forced into this condition, enabled wake-up/activity events will cause the PMU to transition to High-Speed PLL mode. However, the PMU steps down a mode level on each refresh if the $\overline{BL4}$ pin is Low. PMU state transition to Suspend mode can be temporarily interrupted by enabling Low-to-Doze, Doze-to-Sleep, or Sleep-to-Suspend mode-change NMI/SMI events in the NMI/SMI Enable register at Index 82h.

Even if the automatic Suspend-mode feature is disabled, a 0 on $\overline{BL4}$ always forces the PMU into Suspend mode on the next refresh after entering Sleep mode. However, system wake-up is allowed.

1.8.5 AC Input Status Pin

The AC Input Status (ACIN) pin is used to indicate that a permanent power source is driving the system. When ACIN is High (indicating that external power is applied), the PMU state-transition timers are disabled for High-Speed PLL to Low-Speed PLL, Low-Speed PLL to Doze, and Doze-to-Sleep mode transitions. This allows the system to remain in High-Speed PLL mode when AC power is available. Sleep-to-Suspend and Suspend-to-Off mode-transition timers are still enabled. This allows the PMU to change all the way to Off mode after the PMU changes to Sleep mode due to the $\overline{SUS}/\overline{RES}$ pin logic.

Setting bit 5 of the Miscellaneous 6 register at Index 70h has the same effect as asserting the ACIN pin, with the exception that the Force-to-Sleep and Force-to-Suspend functions of $\overline{BL2}$ and $\overline{BL4}$ are disabled only by asserting the external ACIN pin.

The ACIN pin can also be programmed to generate a PMU-activity event. Doing this ensures that if external power is connected to a system with dead batteries, the PMU will automatically wake up. The activity is enabled by setting bit 4 of the PMU Control 3 register at Index ADh and clearing bit 6 of the Activity Mask 1 register at Index 75h. The PMU event is generated by detecting a rising edge on an internal signal that is the logical OR of the external ACIN pin and the internal software SACIN bit (in the Miscellaneous 6 register at Index 70h).

1.9 SUSPEND/RESUME PIN LOGIC

The Suspend/Resume ($\overline{SUS}/\overline{RES}$) pin allows the system designer to provide an end-user mechanism for either waking up the system (i.e., forcing it into High-Speed PLL mode) or putting it into Sleep mode. In a typical system design, this is realized by way of a special key on the system that is logically connected to this pin.

The suspend/resume logic is triggered by a rising edge on the external $\overline{SUS}/\overline{RES}$ pin. If the PMU is in High-Speed PLL, Low-Speed PLL, or Doze mode, such an event causes the PMU to immediately step down through the PMU states (one step per refresh cycle) until it reaches Sleep mode, at which point control passes back to the PMU state machine. If the PMU is in Sleep, Suspend, or Off mode, and it receives a rising edge on the same pin, it will jump directly to High-Speed PLL mode on the next refresh cycle.

The information in this section should be used as a guide to understanding many of the controls and caveats of implementing an SMI as it pertains to a suspend/resume operation. The pseudocode that is provided only addresses handling of $\overline{SUS}/\overline{RES}$ activity. Therefore, it represents only a portion of a potentially more complex SMI handler. Code designed to manage a platform that utilizes SMIs to restart instructions after reinitializing powered-down devices, handling battery-low warnings, waking up as a result of an activity other than the $\overline{SUS}/\overline{RES}$ activity, and so forth requires additional event handling that is beyond the scope of this section.

In addition, this section assumes that you understand the System Management mode (SMM) capabilities of the ÉlanSC310 microcontroller's CPU core. For more information on SMM and SMI functionality for the ÉlanSC310 microcontroller's CPU core, see the *3-Volt System Logic for Personal Computers* manual*, PID 17028.*

This section deals with handling a suspend/resume operation using the SMI capability of the ÉlanSC310 microcontroller. The PMU is designed to be able to perform a minimal suspend/resume operation without any software intervention. However, software may be required to handle platform-specific code that takes care of peripheral-device state save, device power-down, or general cleanup prior to suspend. If this is done, software may be required to perform device-state restoration or other tasks upon resume so that the entire process is transparent to the application code.

These tasks may be handled by either an NMI or an SMI. This is a system-design decision. The use of an SMI to handle suspend/resume operations is shown here for demonstration purposes.

The ÉlanSC310 microcontroller's PMU has the following capabilities that are critical to implementing a suspend/resume capability using SMIs:

1. The PMU, if it is in Sleep, Suspend, or Off mode and upon detecting a rising edge on the $\overline{SUS}/\overline{RES}$ pin, may generate a processor SMI request (bit 0 of the NMI/SMI Enable register at Index 82h).
2. The PMU, if it is in Sleep, Suspend, or Off mode and upon detecting a rising edge on the $\overline{SUS}/\overline{RES}$ pin, wakes up the processor into High-Speed PLL mode.
3. The PMU, if it is in High-Speed PLL, Low-Speed PLL, or Doze mode, and upon detecting a rising edge on the $\overline{SUS}/\overline{RES}$ pin, steps down one PMU state per refresh until it reaches Sleep mode.
4. The PMU may generate a processor SMI request when the Sleep-to-Suspend mode timer expires (Sleep to Suspend Timer register at Index 86h and bit 3 of the NMI/SMI Enable register at Index 82h).

Features 1 and 4 must be enabled individually to implement an SMI-based suspend/resume capability (see the indicated configuration registers). Features 2 and 3 are automatic and require no setup. Note that there is no capability for the PMU to directly generate an SMI request as a result of activity on the $\overline{SUS}/\overline{RES}$ pin that would result in a transition to Sleep mode. For the purpose of this documentation, assume that the suspend SMI request is generated by utilizing feature 4.

The ÉlanSC310 microcontroller's $\overline{SUS}/\overline{RES}$ pin serves both as a go-to-sleep (suspend) and a wake-up (resume) signal input. In either case, a rising edge on the $\overline{SUS}/\overline{RES}$ pin is considered activity. This manual refers to such activity on the $\overline{SUS}/\overline{RES}$ input when the PMU is in Sleep, Suspend, or Off modes as a *resume input*. The same activity, if it occurs when the PMU is in High-Speed PLL, Low-Speed PLL, or Doze mode is referred to as a *suspend input*.

Other issues that should be considered are as follows:

- While processing a suspend input, an SMI occurs as a result of the Sleep-to-Suspend transition (in this example). In order to process the SMI, the PMU enters Temporary-On mode. Near the end of the SMI handler, a write of any value should be made to the NMI/SMI Control register at Index A5h. This causes the PMU to transition into Suspend mode on the refresh following the write to the NMI/SMI Control register.
- When the PMU generates an SMI, a bit is set in the NMI/SMI Control register to indicate to software which PMU event generated the SMI. Writing any value to the NMI/SMI Control register clears this bit.
- Prior to writing the NMI/SMI Control register, the register should be read again to determine if any additional SMI events have occurred during the time it took to process the previous event. If additional SMI events have occurred, they should also be processed starting with a write to the PMU Status 1 register at Index A2h. Otherwise, these events may be lost.
- If in Temporary-On mode, activity on the $\overline{\text{SUS}}$/$\overline{\text{RES}}$ pin is delayed until a write to the NMI/SMI Control register occurs.
- Temporary-On mode is not a PMU mode in the normal sense. Its state cannot be read at the CPU Status 1 register at Index A4h like other PMU modes. It is really a special-case state that turns on system clocks to allow the CPU to temporarily process instructions. This allows servicing an event that may occur while the PMU is in a clock-stopped mode.
- A suspend input causes further activity on the $\overline{\text{SUS}}$/$\overline{\text{RES}}$ pin to be ignored until the PMU has finished transitioning to Sleep mode.
- The PMU can request that the processor perform an SMI. This request is in the form of holding the CPU's $\overline{\text{SMI}}$ signal active. The CPU ignores this request if it is already processing an SMI, and it continues to do so until the SMI handler executes the RES3 instruction. The RES3 instruction signals the end of the SMI. If the PMU is asserting an SMI request when RES3 is executed, another $\overline{\text{SMI}}$ signal is generated by the CPU. Otherwise, control returns to the software that was initially interrupted by the SMI.
- Once an SMI handler is entered, it must clear the PMU's SMI request to the CPU. This is done by writing any value to the PMU Status 1 register at Index A2h. The $\overline{\text{SMI}}$ CPU input is level sensitive. As stated above, if a PMU SMI request is pending upon execution of RES3, another SMI is generated immediately by the CPU. This write to the PMU Status 1 register should happen after the SMI Status register at Index 43h and the NMI/SMI Control register at Index A5h are read to determine the cause of the SMI, but before the SMI Status register and the NMI/SMI Control register are written in order to clear their status.
- When a resume input is serviced by the PMU, it places the PMU into Full-On mode and then generates the resume SMI. In other words, the PMU is *not* in a Temporary-On mode during this SMI. A suspend input at this point in the code will immediately start transitioning the PMU towards Sleep mode.
- The SMM memory area can be redirected through an MMS page at 060000h to point to any location in system memory.

1.9.1 Required Initialization

For the following $\overline{SUS}/\overline{RES}$ SMI-handler design to work, the following configuration-register initializations must be performed:

- Initialize the Sleep-to-Suspend time-out to a nonzero value, such as writing 01h to the Sleep to Suspend Timer register at Index 86h.
- Allow the resume input to generate an SMI. This is done via bit 0 of the NMI/SMI Enable register at Index 82h.
- Set bit 7 of the SMI MMS Upper Page register at Index A9h to select an SMI instead of an NMI. This choice applies to all PMU events that are unmasked via the NMI/SMI Enable register at Index 82h.
- Set up the SMM memory area to the desired system memory location (bits 5–4 of the SMI MMS Upper Page register at Index A9h and the SMI MMS Page register at Index AAh).
- Mask all wake-up events (via the Resume Mask register at Index 08h) and activities that can cause the system to wake up (bits 6 and 4–0 of the Activity Mask 1 register at Index 75h). This means only the resume input can wake up the system from Sleep or Suspend modes, ensuring the resume SMI is always executed.

1.9.2 Start of SMI Handler

SMI handler code must determine why the handler has been entered. Since all SMIs cause the processor to enter Real mode and jump to the reset vector, it must be determined early in the boot code whether an SMI handler is being executed (bit 7 of the Version register at Index 64h). If so, the handler must determine whether a suspend or a resume is in progress. This may be accomplished by reading the NMI/SMI Control register at Index A5h and checking to see whether the SMI was caused by a Sleep-to-Suspend mode transition (bit 3) or by a resume input (bit 0).

1.9.3 Suspend Input Caused the SMI

If a suspend input caused the SMI, the following events occurred:

1. The system was running in High-Speed PLL, Low-Speed PLL, or Doze mode.
2. A rising edge was detected on the $\overline{SUS}/\overline{RES}$ input (a suspend input).
3. The PMU started to ignore further activity on the $\overline{SUS}/\overline{RES}$ input ($\overline{SUS}/\overline{RES}$ activity).
4. The PMU stepped down through the PMU modes, one per refresh, until Sleep mode was entered. (Note that the PMU would wake up to High-Speed PLL mode if a resume input had occurred at this point.)
5. The Sleep-to-Suspend timer expired.
6. The PMU mode started to change from Sleep to Suspend, causing an SMI. At the same time, the PMU became capable of buffering (delaying) one subsequent $\overline{SUS}/\overline{RES}$ activity. (If this activity occurred at this point, it would be delayed until Index A5h was next written. The PMU would then process the activity as if it had just occurred.) In addition, the PMU entered Temporary-On mode, causing the CPU to start running at 9.2 MHz.
7. When the SMI handler was entered, SMIs were automatically masked by the CPU until the next execution of the RES3 instruction.

1.9.4 Suspend Pseudocode

1. The PMU Status 1 register at Index A2h is written in order to clear the PMU SMI request to the CPU.
2. Platform-dependent code prepares the system for the suspend operation.
3. The bit location in the SMM RAM state-save area that contains the value of bit 12 of DR7 (the processor's state) must be cleared. This handles an ÉlanSC310 microcontroller CPU errata where bit 12 of DR7 is automatically set prior to the SMM state save. If this bit is not cleared in the SMM state-save area prior to the CPU executing a RES3 instruction, the erroneous bit is reloaded into DR7 and the trace opcode (F1h) is redefined as a *soft* SMI. The next trace instruction causes a soft SMI to occur.
4. The NMI/SMI Control register at Index A5h is read to determine if any additional SMI events have occurred while processing the suspend SMI. These additional events should also be processed or they will be lost.
5. The NMI/SMI Control register is written, which causes the PMU to exit Temporary-On mode on the next refresh and transition from Sleep to Suspend mode. At the same time, the ability of the next rising edge on the $\overline{SUS}/\overline{RES}$ input to generate a wake-up into High-Speed PLL mode is converted to immediate instead of buffered.
6. At this point, the SMI handler polls bit 4 of the AT-Compatible B port (61h), looking for at least one refresh. Each time the bit toggles, a refresh has occurred. This must be done to ensure that the PMU transition into Suspend mode occurs before any more of the suspend SMI handler is executed. The CPU clock thus stops before the RES3 instruction is executed. This is done to prevent control from being returned to the application that was interrupted by the SMI. This should be avoided because the system has been prepared for Suspend mode. Peripheral devices may have been turned off, which causes errors if accessed in this state.

The CPU clock is now stopped and the system is suspended. From this point on, an edge on the $\overline{SUS}/\overline{RES}$ pin is considered a resume input. Note that although RES3 has not been executed by the CPU (thus ending the suspended SMI), all remaining suspend-SMI code after this is processed as the result of a resume input. The effect is that the last few instructions of the suspend handler (everything before RES3) are really the first few instructions of the resume code. This is not to say that the resume SMI entry point is anywhere in the suspend SMI handler. The remainder of the handler execution follows:

7. A resume input is detected. This transitions the PMU into High-Speed PLL mode and causes it to assert an SMI request to the CPU. Note that the SMI request is not seen by the CPU at this point because RES3 has not been generated for the suspend SMI.
8. The CPU RES3 instruction is executed to signal the end of the SMI routine. This allows the CPU to detect pending SMIs asserted by the PMU. As a result of the pending SMI request, an SMI is generated immediately, and the SMI routine is entered to process the resume.

1.9.5 Resume Input Caused the SMI

If a resume input caused the SMI, the following events occurred:

1. The CPU was in the Sleep, Suspend, or Off mode.
2. A rising edge was detected by the $\overline{SUS}/\overline{RES}$ input.
3. The PMU jumped directly into High-Speed PLL mode on the first refresh following the resume input.

Note: *At this point, the next $\overline{SUS}/\overline{RES}$ edge is considered a suspend input.*

4. The resume input caused an SMI.
5. The SMI handler was entered and SMIs were automatically masked by the CPU until a RES3 is executed.

1.9.6 Resume Pseudocode

1. The PMU Status 1 register at Index A2h is written in order to clear the PMU SMI request to the CPU.
2. Platform-dependent code prepares the system for the resume operation.
3. The NMI/SMI Control register at Index A5h is read to determine if any additional SMI events have occurred while processing the suspended SMI. These additional events should also be processed or they will be lost.
4. When the PMU generates an SMI, a bit is set in the NMI/SMI Control register to indicate to software the PMU event that generated the SMI. The NMI/SMI Control register is written to clear this bit.
5. The bit location in the SMM RAM state-save area that contains bit 12 of DR7 (the processor's state) must be cleared. This handles an ÉlanSC310 microcontroller CPU errata where bit 12 of DR7 is automatically set prior to the SMM state save. If this bit is not cleared in the SMM state-save area prior to the CPU executing a RES3 instruction, the erroneous bit will be reloaded into DR7. The trace opcode (F1h) is redefined as a *soft* SMI. The next trace instruction then causes a soft SMI to occur.
6. The CPU RES3 instruction is executed to signal the end of the SMI routine and to arm further SMIs.
7. The application program regains control.

1.9.7 Things to Avoid

Do not allow the suspend handler to execute RES3 until the resume keystroke occurs. Failure to adhere to this results in control returning to the application program with the CPU clock running for a short period of time. If application code is executed during this time, there is a chance that powered-down peripherals will be accessed. Even DRAM may be powered down at this point as a result of the preparation for Suspend mode.

1.10 AUTO LOW-SPEED LOGIC

The ÉlanSC310 microcontroller's auto low-speed logic provides a way for the system designer to fine-tune the system's current consumption in High-Speed PLL mode. Auto Low-Speed mode is enabled by setting bit 3 of the Control B register at Index 77h. Auto Low-Speed mode has no effect unless the high-speed clock is enabled at bit 6 of the I/O Wait State register at Index 61h. When enabled, the auto low-speed logic causes the use of the high-speed clock to be disallowed at periodic intervals when in High-Speed PLL mode. During the active trigger period, all bus cycles are performed at 9.2 MHz in High-Speed PLL mode. There is no effect in any other PMU mode. The start and end of the low-speed interval is synchronized with refresh. To program the trigger and duration periods, see "Auto Low-Speed Control Register (Index 9Fh)" on page 4-51.

1.11 MICRO POWER-OFF MODE

The Micro Power-Off feature of the ÉlanSC310 microcontroller should not be confused with a PMU mode or state such as High-Speed PLL mode, Low-Speed PLL mode, or Suspend mode.

There is no software processing required or available to enter the Micro Power-Off mode. For most applications, Micro Power-Off mode is like completely turning off the power to the system while maintaining real-time clock operation and CMOS contents. The system enters Micro Power-Off mode immediately when the $\overline{\text{IORESET}}$ pin is sampled Low. There is no option to generate an SMI or NMI to save the system state upon detection of $\overline{\text{IORESET}}$ being asserted.

The type of micro power-off DRAM refresh performed (CAS-before-RAS, or self-refresh) is the same as that for which the part was configured (via bit 3 of the Miscellaneous 5 register at Index B3h) prior to sampling the $\overline{\text{IORESET}}$ pin Low.

Exiting Micro Power-Off mode is analogous to a power-up cold boot, with the exception that the bits shown below remain set to their previously programmed values. The configuration-register bits that are not reset when exiting Micro Power-Off mode are as follows:

- Bits 1–0 of the Version register at Index 64h
- Bits 7–0 of the Clock Control register at Index 8Fh
- Bits 7–4 and 2–0 of the Reserved register at Index 9Dh
- Bit 1 of the PMU Control 1 register at Index A7h
- Bit 3 of the Miscellaneous 5 register at Index B3h
- Bits 4–0 of the Miscellaneous 3 register at Index BAh

The one software option that relates to Micro Power-Off mode is whether the DRAM contents will be maintained along with the CMOS/RTC functions mentioned in the preceding paragraph. This option is enabled by setting bit 2 of the Miscellaneous 3 register at Index BAh. This bit is cleared upon core reset, using the $\overline{\text{RESIN}}$ pin. $\overline{\text{IORESET}}$ has no effect on this bit. The bit may be used by system firmware to determine whether or not DRAM has been retained after an $\overline{\text{IORESET}}$ has occurred.

For more information on the Micro Power-Off mode, see the *ÉlanSC310 Microcontroller Data Sheet, PID 20668*.

1.12 OTHER POWER-SAVING FEATURES

The ÉlanSC310 microcontroller has additional features that provide reduced power consumption. These features can be enabled at the system designer's discretion.

1.12.1 DMA Clock Stop

This feature is enabled by bit 3 of the Function Enable 1 register at Index B0h. It causes the clock to the 8237 DMA controller to stop except when actually needed to perform a DMA transfer. This feature operates independently of the PMU mode. If clocks are not disabled by the PMU, the DMA clock starts after one of the DRQs goes active High, and the DMA clock continues running until AEN is deasserted.

1.12.2 Data-Path Disabling Logic

The following bits help limit the amount of data-bus toggling in the ÉlanSC310 microcontroller to peripheral devices which are not in use:

- Bits 4 and 5 of the PMU Control 1 register at Index A7h can be set to disable data-bus toggling to the internal UART and PMU blocks during memory cycles.
- Bit 4 of the Miscellaneous 4 register at Index 44h, if 1, disables data-bus toggling to most of the ÉlanSC310 microcontroller (except between the CPU and the external data-bus interface) during internal DRAM cycles.

1.12.3 Slow Refresh

For systems using DRAM that supports slow refresh, the ÉlanSC310 microcontroller provides five user-selectable refresh rates. The default refresh rate is the slowest: 8192 refreshes per second. If bits 0 and 1 of PMU Control 1 register at Index A7h are 0, the refresh rate is controlled by bits 1 and 0 of the Version register at Index 64h. Supported rates are 32768, 16384, 10922, and 8192 refreshes per second. If bit 0 of the PMU Control 1 register at Index A7h is 0 and bit 1 is 1, the refresh rate becomes 65536 refreshes per second; if bit 0 is 1, the 8254 becomes the refresh source.

1.12.4 Quiet Bus

Setting bit 2 of the Control A register at Index 48h disables $\overline{\text{MEMR}}$ and $\overline{\text{MEMW}}$ on the ISA bus from toggling during internal memory cycles.

CHAPTER

2 MEMORY MANAGEMENT

From the programmer's perspective, the ÉlanSC310 microcontroller manages three separate 16-Mbyte memory-address spaces:

— System memory

— ROM BIOS

— ROM DOS

In addition, the ÉlanSC310 microcontroller manages a single 64-Kbyte I/O address space.

Only the system-memory-address space is directly accessible in its entirety; the ROM-BIOS and ROM-DOS address spaces are directly accessible in part. The bulk of the ROM-BIOS and ROM-DOS address spaces are accessible only via a memory-mapping system.

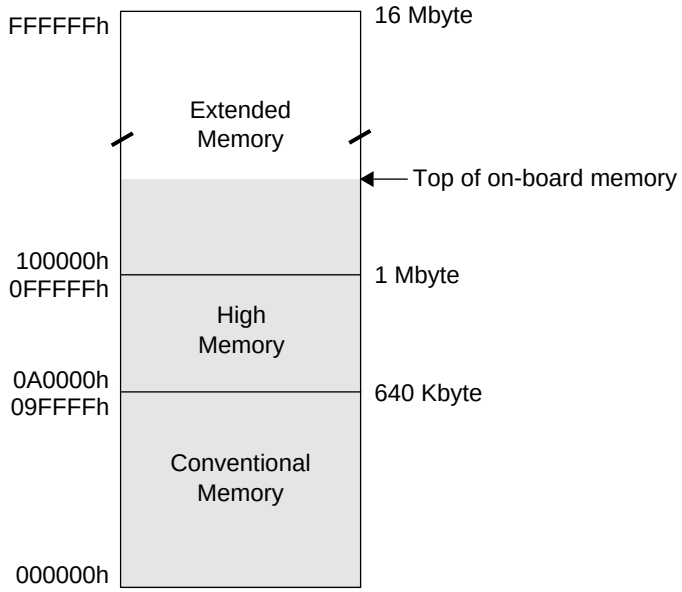
The actual memory of a system based on the ÉlanSC310 microcontroller consists of memory devices—DRAM and EPROMs or flash memory. Usually, only a subset of each address space corresponds with any device, and many addresses are not associated with any device. For example, a typical configuration might have only 8 Mbyte of DRAM system memory, 128 Kbyte of BIOS ROM, and 128 Kbyte of DOS ROM.

The ÉlanSC310 microcontroller's Memory Controller Unit (MCU) is responsible for managing access to the address spaces and the associated physical memory.

2.1 SYSTEM MEMORY: DRAM AND BUS

Figure 2-1 on page 2-2 shows a typical system address space for the ÉlanSC310 microcontroller.

Figure 2-1 Typical AT Address Space



The three regions of system-address space are conventional memory (the first 640 Kbyte), high memory (the next 384 Kbyte), and extended memory (the remaining 15 Mbyte). Only the first Mbyte (conventional and high memory)—along with the first 64 Kbyte of extended memory—can be accessed while the ÉlanSC310 microcontroller is in Real mode without using the MMS. All 16 Mbyte of system-address space may be addressed while the ÉlanSC310 microcontroller is in Protected mode. For more information on Real and Protected modes, see the *Am486® Microprocessor Software User's Manual*, PID 18497.

In Figure 2-1 on page 2-2, the shaded portion of the address space represents on-board memory. That is, it represents the portion of the system address space that is currently associated with physical memory chips attached to the ÉlanSC310 microcontroller's system board (e.g., in SIMMs). The unshaded portion can be associated with memory devices connected to the ISA or local bus. These addresses will be empty if no ISA or local-bus memory devices are attached.

For on-board memory, the ÉlanSC310 microcontroller supports only DRAM.

The ÉlanSC310 microcontroller supports two different bus configurations:

- Local bus with a partial ISA bus
- Maximum ISA bus

The selection of one of these two options must be made when the system is designed and may not be changed dynamically. This choice also affects the meaning and behavior of several pins. For information on exactly which pins are affected, refer to the *ÉlanSC310 Microcontroller Data Sheet, PID 20668*.

2.1.1 DRAM Configurations

The ÉlanSC310 microcontroller supports up to 16 Mbyte of on-board DRAM, organized as one or two memory banks (16-bit data path to memory). The $\overline{RAS0}$ and $\overline{RAS1}$ signals enable the banks, while the $\overline{CAS0L}$, $\overline{CAS0H}$, $\overline{CAS1L}$, and $\overline{CAS1H}$ signals are used to select the upper and lower sections of each bank. Each bank is 16 bits wide and contains 1, 2, or 4 chips depending on the bit architecture of the memory chips. The following bits are used to specify the memory size and architecture:

- Bits 4–2 of the Memory Configuration 1 register at Index 66h
- Bits 7–6 of the Function Enable 2 register at Index B1h
- Bit 7 of the Function Enable 3 register at Index B4h

Insight into the mapping of both DRAM configurations can be gained by examining Table 4-23 on page 4-28 and Table 4-38 on page 4-63. As an example of how to program the memory configuration, the settings in Table 2-1 on page 2-3 select a 4-Mbyte memory that is organized into two banks and configured for Enhanced-Page-mode operation.

Table 2-1 Memory Initialization Example

Instruction	Ports	Data	Comment
IOW IOW	022h 023h	B4h 0100 0000	Enable the Memory Configuration 1 register at Index 66h to select memory configuration.
IOW IOW	022h 023h	66h xxx1 0011	Select 4-Mbyte memory configuration (Enhanced Page mode). Note that bits 6–5 are read only.

2.1.2 Refresh and Wait States

The ÉlanSC310 microcontroller supports two memory refresh modes:

- 8254-based DRAM refresh
- Slow-refresh DRAM

The 8254-based DRAM refresh cannot be selected if power management is performed because the 8254 clock will be turned off in Suspend mode. Memory refresh needs to remain enabled since the PMU uses the refresh pulse to synchronize events. The refresh frequency is under programmer control. See “PMU Control 1 Register (Index A7h)” on page 4-56 and “Version Register (Index 64h)” on page 4-26.

For example, the settings shown in Table 2-2 on page 2-3 perform the following operations:

- Select Slow-Refresh mode for DRAM refreshes. The DRAMs are refreshed at the rate selected by bits 1 and 0 of the Version register at Index 64h.
- Select the 32-kHz clock input as the refresh source.

Table 2-2 Refresh Initialization Example

Instruction	Ports	Data	Comment
IOW IOW	022h 023h	A7h xxxx xx00	Enable slow refresh; select 32-kHz clock for refresh source; disable refresh during Sleep mode.

The ÉlanSC310 microcontroller also supports a variety of speeds of ROM. Some speeds require wait states to be inserted in memory accesses between the point in a cycle when the read or write signal is asserted and the point where the data is transferred to or from the memory device. The number of wait states to be inserted under different conditions is under programmer control.

Page-mode DRAMs are organized so that successive locations are along the same row, or *page*. This allows the memory controller to generate a single-row address if successive accesses happen to lie in the same page (a *page hit*). When a page hit occurs, the memory cycle can be shortened by eliminating the row-address portion of the cycle. At the programmer's discretion, wait states may be inserted into the Page-mode cycle. The programmer has the option to set the number of wait states after a page miss and the number of first-cycle wait states. The number of wait states for a page-hit read cycle and a page-hit write cycle are fixed at 0 and 1, respectively.

The settings in Table 2-3 on page 2-4 set up Page-mode DRAM accesses using the MOD select bit of the Memory Configuration 1 register at Index 66h. They set the first-cycle wait state to 3 cycles (the default) and the bank-miss wait states to 5 by setting the following bits:

- Bits 5 and 6 of the Wait State Control register at Index 63h
- Bit 5 of the ROM Configuration 1 register at Index 65h
- Bit 4 of the MMS Memory Wait State 1 register at Index 62h

Table 2-3 Memory-Speed Initialization Example

Instruction	Ports	Data	Comment
IOW IOW	022h 023h	66h xxxx xx10	Set memory mode to Page-mode DRAMs. Note that bits 7–5 of this register are read only because they affect the choice of bus.
IOW IOW	022h 023h	63h x11x xxxx	Set top bits of first-cycle wait state to 3 and bank-miss wait state to 5.
IOW IOW	022h 023h	65h xx1x xxxx	Set low bit of first-cycle wait to 3.
IOW IOW	022h 023h	62h xxx1 xxxx	Set low bit of bank-miss waits to 5.

The ÉlanSC310 microcontroller also supports Enhanced Page mode when both banks of DRAM are used. It may be selected using bit 0 of the Memory Configuration 1 register at Index 66h. This mode effectively doubles the page size by arranging the DRAM address lines such that one page is spread across both DRAM banks. This avoids the precharge penalty that would otherwise occur when incrementing across the bank-section boundary. For more information on the Enhanced-Page-mode address translation, see the *ÉlanSC310 Microcontroller Data Sheet, PID 20668*.

Table 2-4 on page 2-5 shows the number of wait states required in a 33-MHz ÉlanSC310 microcontroller configuration for 70-ns Page-mode DRAMs.

Table 2-4 33-MHz Wait States

Cycle Type	Wait States
Read cycle, page hit	0
Write cycle, page hit	1
First cycle	3
Page miss	5

2.1.3 Maximum ISA and Local Bus Configurations

The ÉlanSC310 microcontroller is designed as a single-chip integration of all the principal components needed for an embedded PC. The ÉlanSC310 microcontroller supports the following strategies for video control:

- A local-bus interface
- The ÉlanSC310 microcontroller's Maximum ISA bus interface

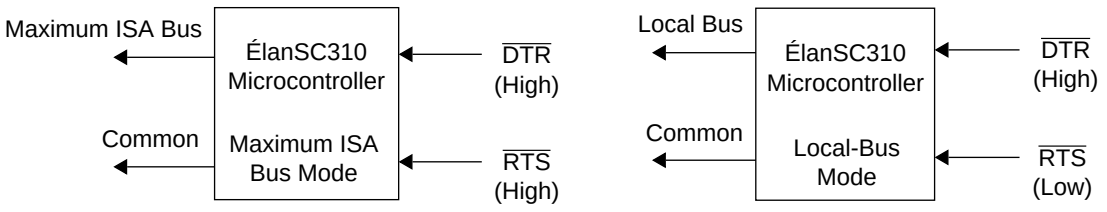
The local-bus interface is integrated with the memory controller and the ISA-bus controller, and it permits fast transfers to and from external local-bus peripherals such as video controllers.

These two strategies are alternatives because they share the same set of pins. The configuration employed (local bus or Maximum ISA bus) depends on the initial values of bits 5 and 6 in the Memory Configuration 1 register at Index 66h at system reset. These initial values are determined by the values of the $\overline{DTR}$ and $\overline{RTS}$ pins, which are established by pull-up or pull-down resistors attached to the pins. They are not set at run time by programming the registers. Table 2-5 on page 2-5 shows the meaning of these pins and bits.

Table 2-5 Bus Configuration

Bus Selected	$\overline{DTR}$ (CFG1)	$\overline{RTS}$ (CFG0)
Local Bus	1	0
Maximum ISA	x	1

Figure 2-2 ÉlanSC310 Microcontroller Bus Configurations



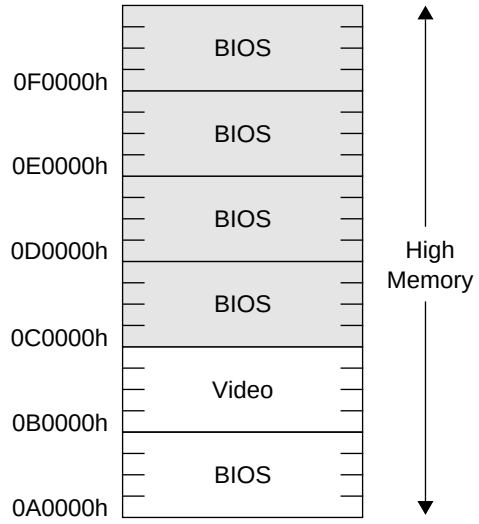
2.2 ROM-BIOS MEMORY

ROM-BIOS memory is accessed when the ÉlanSC310 microcontroller activates the BIOS chip-select signal ($\overline{\text{ROMCS}}$). The combination of this signal and the 24 address lines defines a separate 16-Mbyte ROM-BIOS address space. However, usually only a few hundred Kbyte of this space are occupied by physical ROM devices. ROM BIOS may be accessed directly or as a function of the MMS.

With the ÉlanSC310 microcontroller, up to 320 Kbyte of non-MMS ROM BIOS is supported, and addresses in the ranges 0A0000–0AFFFFh and 0C0000–0FFFFFh can be supported. The address range for the ROM BIOS is set via the ROM Configuration 1 register at Index 65h and the ROM Configuration 2 register at Index 51h. For additional information, see Chapter 3, “PC/AT Peripheral Registers.” Figure 2-3 on page 2-6 shows a diagram of high memory with all the address ranges indicated that may be programmed in the ÉlanSC310 microcontroller to generate $\overline{\text{ROMCS}}$ (indicated by the word *BIOS*). This figure also shows the address ranges that may be shadowed. Accesses to the boot-vector location, FF0000–FFFFFFh, always assert the $\overline{\text{ROMCS}}$ signal.

ROM BIOS is accessed as an 8-bit device unless the system hardware drives $\overline{\text{MCS16}}$ active, which results in a 16-bit access. There is no programmable option to set whether the ROM BIOS is an 8-bit or 16-bit device.

Figure 2-3 High Memory



Note:

The shaded area (0C0000–0F0000h) is shadowable.

As an example, the settings shown in Table 2-6 on page 2-6 implement a 128-Kbyte ROM BIOS located at 0E0000–0FFFFFh.

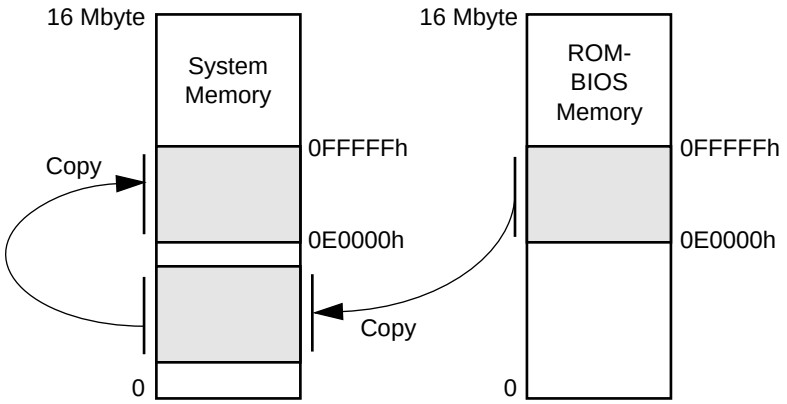
Table 2-6 ROM-BIOS Address Initialization

Instruction	Ports	Data	Comment
IOW IOW	022h 023h	51h xxxx xxx0	Disable $\overline{\text{ROMCS}}$ for 0A0000–0AFFFFh.
IOW IOW	022h 023h	65h xxxx 0010	Enable $\overline{\text{ROMCS}}$ for 0E0000–0FFFFFh; disable it for 0C0000–0DFFFFh.

In order to speed up BIOS accesses, many PC systems copy the ROM-BIOS contents into on-board DRAM at startup. Then the system is set to send ROM-BIOS accesses to the DRAM instead of the ROM. This technique is called *ROM shadowing*.

The usual implementation is to copy the BIOS image stored in ROM to the same addresses in system RAM, and then transfer program control to the BIOS image in system RAM (see Figure 2-4 on page 2-7).

Figure 2-4 Copying ROM Contents



For example, to shadow ROM to RAM in a system where the BIOS ROM is a 128-Kbyte section located from 0E0000h to 0FFFFFh in CPU address space, the following algorithm can be implemented:

1. Copy the ROM-based image into a 128-Kbyte buffer in conventional RAM.
2. Jump to the exact same place in the conventional RAM-based BIOS image that was being executed in the ROM-based BIOS image.
3. Enable the ÉlanSC310 microcontroller's shadow-RAM function and disable the shadow RAM's write-protect feature by setting bits 4 and 7 of the ROM Configuration 1 register at Index 65h.
4. Set the shadow-RAM address range to match the BIOS ROM's range of 0E0000h to 0FFFFFh by writing FFh to the Shadow RAM Enable 2 register at Index 69h.
5. Disable $\overline{\text{ROMCS}}$ for the range 0E0000–0FFFFFh by setting bit 0 and clearing bit 1 of the ROM Configuration 1 register.
6. Copy the conventional RAM-based BIOS image to shadow RAM from 0E0000h to 0FFFFFh. Once this is done, shadow RAM may optionally be write protected by clearing bit 7 of the ROM Configuration 1 register.
7. Jump to the same place in the shadow-RAM-based BIOS image that was being executed in the conventional RAM-based BIOS image.

Note: *Use caution when performing read-modify-write sequences of the ROM Configuration 1 register. Bit 0 reads the inverse of what was last written; therefore, it must be flipped again prior to write-back to keep $\overline{\text{ROMCS}}$ settings unchanged.*

2.3 ROM-DOS MEMORY

ROM-DOS memory is accessed whenever the ÉlanSC310 microcontroller activates the $\overline{\text{DOSCS}}$ chip-select signal. The combination of this signal and the 24 address lines defines a separate 16-Mbyte ROM-DOS address space. This implementation is particularly well suited for accessing ROMs that contain a disk image loaded by DOS. Hence the name ROM DOS.

The ROM-DOS address space can be configured as either an 8-bit or 16-bit device. Bit 1 of the ROM Configuration 2 register at Index 51h controls this. The default is an 8-bit device.

The ROM-DOS address space is typically accessed using the ÉlanSC310 microcontroller's MMS, which is described in "Memory Mapping" on page 2-9. In addition to using MMS, up to 15 Mbyte minus 64 Kbyte of ROM-DOS address space can also be accessed as direct system memory using linear address decodes. Bits 3–0 of the ROM Configuration 3 register at Index B8h determine the address range used for the ROM-DOS decode. If these bits are 0, the ROM-DOS address space is accessible only through MMS. Table 4-42 on page 4-68 provides the bit settings for the allowable address ranges. Note that the top of the address range is fixed at FEFFFFh. The address range FF0000–FFFFFFh is reserved for ROM BIOS.

As the size of the ROM-DOS linear-decode address space increases, the lower address boundary decreases. Care must be taken to ensure that the ROM-DOS linear-decode address range does not overlap the on-board DRAM range. Bits 7–4 of the Miscellaneous 1 register at Index 6Fh can be used to prevent the upper portion of system DRAM from being accessed as direct system memory, thereby eliminating the conflict.

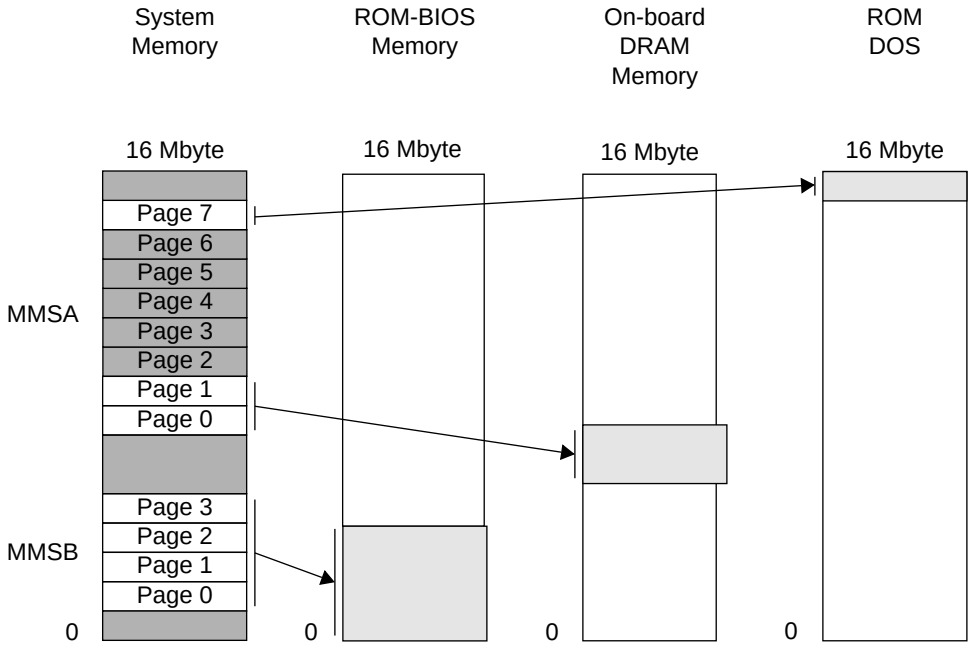
2.4 MEMORY MAPPING

The ÉlanSC310 microcontroller's memory controller supports a Memory Mapping System (MMS). The MMS defines two windows in the first Mbyte of system address space:

- MMSA, with eight 16-Kbyte pages
- MMSB, with four 16-Kbyte pages

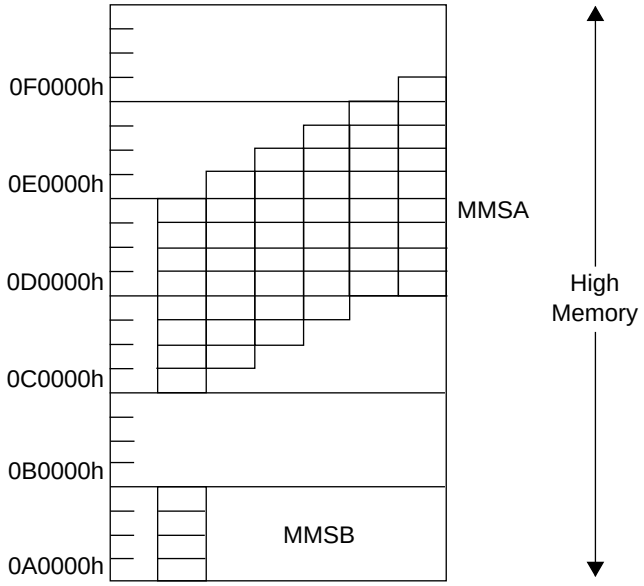
The programmer can set these pages to point to any similarly sized page of ROM DOS, ROM BIOS, or on-board DRAM. Once these pages are programmed, an address generated in a system-address page is automatically translated into an access to the mapped page of the other address space. Figure 2-5 on page 2-9 shows the basic idea. Using the ÉlanSC310 microcontroller's MMS windows, it is possible to access every addressable location in each of the five address spaces listed.

Figure 2-5 Memory Mapping System



The beginning of the MMSA window may be located at one of six starting addresses between 0C0000h and 0D4000h. The MMSB window always starts at 0A0000h. Figure 2-6 on page 2-10 shows how high memory contains the MMSA and MMSB windows. Notice that MMSA can have six different positions, depending on how the MMS Address register at Index 6Dh is programmed. The MMSA window size is fixed at 128 Kbyte (8 pages • 16 Kbyte). The MMSB window size is fixed at 64 Kbyte (4 pages • 16 Kbyte).

Figure 2-6 MMSA and MMSB



The MMS windows and their corresponding pages are set up by programming several configuration registers in the ÉlanSC310 microcontroller. In particular, control is exercised through the following registers:

- ROM Configuration 1 register at Index 65h
- MMSA Address Extension 1 register at Index 67h
- MMS Address Extension 1 and 2 registers at Indexes 6Ch and 6Eh
- MMS Address register at Index 6Dh
- MMSA Device 1 and 2 registers at Indexes 71h and 72h
- MMSB Device register at Index 73h
- MMSB Control register at Index 74h

All the index registers used in this section can be referenced in Appendix A, “Configuration Index Register Reference,” as MMS registers except for the ROM Configuration 1 register. Bits 3–0 of the MMS Address register at Index 6Dh define the location in I/O space of the eight MMSA and four MMSB page registers. Bit 1 of the MMSB Control register at Index 74h determines whether the page-register I/O locations are accessing MMSA page registers or MMSB page registers.

Each MMS window contains a global switch that enables or disables all the pages within that window. Bit 5 of the ROM Configuration 1 register at Index 65h is the switch bit for MMSA. Bit 0 of the MMSB Control register at Index 74h is the switch bit for MMSB.

As stated earlier, each MMS page is a fixed size of 16 Kbyte. Each page is mappable to an equal-size page located on a 16-Kbyte boundary in one of the three address spaces: ROM DOS, ROM BIOS, or on-board DRAM. Each individual page within an MMS window can be enabled or disabled by setting or clearing a bit in its corresponding page register.

System RAM is not the default device that is accessed if an MMS window or a page of an MMS window is disabled. This allows externally decoded, memory-mapped devices on

the ISA bus to exist in the address spaces that are normally decoded by the MMS windows. Thus, if more than 640 Kbyte of system RAM is desired for use by a DOS memory extender (for example), the MMS page should be programmed to point to system RAM explicitly. This feature also allows a block of system address space in high memory to be left open for use by the EMM386 memory manager to map in pages of extended memory using capabilities provided by 386-class and higher processors.

Another implication of system RAM not being the default device is that if the base address for MMSA is not set to 0C0000h, only externally decoded devices can utilize the address space abandoned by MMSA in this example. More specifically, if MMSA is made to start at 0C8000h, the 32 Kbyte of address space from 0C0000h to 0C7FFFh is usable only if decoded by an external ISA memory-mapped device (e.g., a VGA ROM).

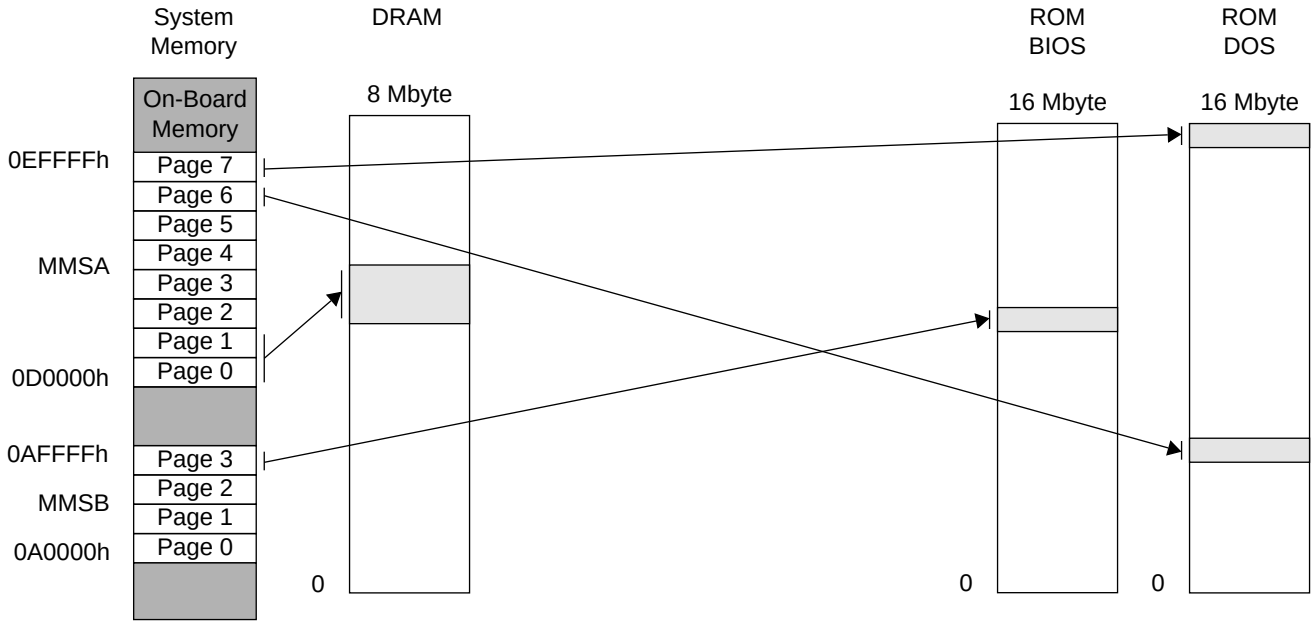
The address spaces—ROM DOS and ROM BIOS—are implemented as chip-select signals in the ÉlanSC310 microcontroller to eliminate the need for external address decoding. Cycles to these devices use the ISA command signals, MEMR and MEMW. Thus, any external device that does not have access to the chip signals generated by the ÉlanSC310 microcontroller (such as ISA devices) may have address conflicts with cycles that use the MMS mapping.

There is no separate command-signal option for interfacing to the DOS ROM. It is suggested that the DOS or application ROM be made to physically reside above 1 Mbyte in the system-address space to avoid conflicts with devices whose decoding is fixed by the PC/AT architecture.

It is invalid to have ROM BIOS shadowing enabled for a region that contains an enabled MMS page. If this is done, conflicts will result when accessing the overlapping areas. For example, if ROM BIOS shadowing is enabled for the address range 0E0000–0EFFFFh and the MMSA starting address is set to 0C4000h, then MMSA page 7 (0E0000–0E3FFFh) should always remain disabled.

Figure 2-7 on page 2-12 shows an example of MMS mapping between system-address space and all three address spaces. In this example, there are 8 Mbyte of on-board memory. One page of ROM-BIOS space, two pages of ROM-DOS space, and two pages of on-board memory are being mapped. The settings in Table 2-7 on page 2-13 define the MMS mapping shown in Figure 2-7 on page 2-12.

Figure 2-7 MMS Mapping Example



Notes:

The configuration is summarized as follows:

MMSA base address: 0D0000h

Page 0: On-board (5FC000–5FFFFFh)
Page 1: On-board (600000–603FFFh)
Page 2: (not mapped)
Page 3: (not mapped)
Page 4: (not mapped)
Page 5: (not mapped)
Page 6: ROM DOS (220000–223FFFh)
Page 7: ROM DOS (FFC000–FFFFFFh)

MMSB base address: 0A0000h

Page 0: (not mapped)
Page 1: (not mapped)
Page 2: (Not mapped)
Page 3: ROM BIOS (400000–403FFFh)

Table 2-7 MMS Mapping Example Settings

Instruction	Ports	Data	Comment
IOW IOW	022h 023h	66h xxx1 011x	Set on-board memory size to 8 Mbyte. Note that bits 7–5 are read only because they affect the choice of bus.
IOW IOW	022h 023h	74h xxxx xx1x	Select MMSA for programming.
IOW IOW	022h 023h	6Dh 0100 0000	Set MMSA base address to D0000h and I/O base address to 0208h.
IOW IOW	022h 023h	6Ch 1000 0000	Set all translated MMSA A23s (page 7 = 1, others = 0).
IOW IOW	022h 023h	6Eh 0000 1110	Set A22 and A21 for MMSA pages 0–3.
IOW IOW	022h 023h	67h 1101 0000	Set A22 and A21 for MMSA pages 4–7.
IOW IOW IOW IOW IOW IOW IOW IOW	0208h 2208h 4208h 6208h 8208h A208h C208h E208h	1111 1111 1000 0000 0xxx xxxx 0xxx xxxx 0xxx xxxx 0xxx xxxx 1000 1000 1111 1111	Translate A20–A14 of MMSA page 0; enable the page. Translate A20–A14 of MMSA page 1; enable the page. Disable translation for MMSA page 2. Disable translation for MMSA page 3. Disable translation for MMSA page 4. Disable translation for MMSA page 5. Translate A20–A14 of MMSA page 6; enable the page. Translate A20–A14 of MMSA page 7; enable the page.
IOW IOW	022h 023h	71h xxxx 0101	Set memory type for MMSA pages: 0–1 on board (pages 2-3 not mapped).
IOW IOW	022h 023h	72h 0000 xxxx	Set memory type for MMSA pages: 6–7 ROM DOS (pages 4-5 not mapped).
IOW IOW	022h 023h	65h x1xx xxxx	Enable MMSA.
IOW IOW	022h 023h	74h xxxx xx01	Enable MMSB and select MMSB for programming.
IOW IOW	022h 023h	6Ch xxxx 0xxx	Set all translated MMSB A23s (pages 0,1, 2 not translated).
IOW IOW	022h 023h	6Eh 10xx xxxx	Set A22 and A21 for MMSB page 3 (pages 0,1, 2 not translated).
IOW IOW IOW IOW	0208h 2208h 4208h 6208h	0xxx xxxx 0xxx xxxx 0xxx xxxx 1000 0000	Disable translation for MMSB page 0. Disable translation for MMSB page 1. Disable translation for MMSB page 2. Translate A20–A14 of MMSB page 3; enable the page.
IOW IOW	022h 023h	73h 11xx xxxx	Set memory type for MMSB pages: 0–2 (unmapped), 3 (ROM BIOS).

2.5 OTHER MEMORY CONTROLLER INFORMATION

2.5.1 ROM Chip-Select Command Gating

The $\overline{\text{DOSCS}}$ and $\overline{\text{ROMCS}}$ chip-select outputs of the ÉlanSC310 microcontroller are, by default, internally gated with the memory-read command ($\overline{\text{MEMR}}$) or the memory-write command ($\overline{\text{MEMW}}$). In the ÉlanSC310 microcontroller, the following configuration register bits may be used to disable the command gating and allow the $\overline{\text{DOSCS}}$ and $\overline{\text{ROMCS}}$ signals to be available as address decodes only:

- **Bit 2 of the Miscellaneous 5 register at Index B3h** Enables the $\overline{\text{ROMCS}}$ signal as an address decode as follows:
  - — 0 Address decode with command gating
  - — 1 Address decode only
- **Bit 4 of the ROM Configuration 3 register at Index B8h** Enables the $\overline{\text{DOSCS}}$ signal as an address decode as follows:
  - — 0 Address decode with command gating
  - — 1 Address decode only

When the CPU clock is stopped, the $\overline{\text{ROMCS}}$ and $\overline{\text{DOSCS}}$ chip-selects are forced High.

2.5.2 Wait States and Command Delays

The ÉlanSC310 microcontroller provides several programmable options for controlling the number of wait states and command delays inserted into a cycle. This section covers wait states and command delays for ROM-BIOS, ROM-DOS, and ISA cycles.

A command delay is inserted between the point in a memory or I/O cycle where the address is placed on the bus and the point where the memory or I/O, read or write, command signal is asserted. This delay gives slower devices extra time to decode the address. Command delays do not lengthen the overall cycle time. That is, the command signal is deasserted at the same time it normally is (determined by the number of wait states), without the command delay. The net effect is a shortened command time.

Table 2-8 on page 2-14 documents the duration of the command delay for the various cycles.

Table 2-8 Command Delay Duration for Various Cycles

Cycle Type	Command Delay Duration (in SYSCLK Cycles)
8-bit ISA memory	1 or 0.5 (determined by bit 2 of the Command Delay register at Index 60h)
8-bit ROM-DOS memory	0, 0.5, or 1 (determined by bit 2 of the Command Delay register and bits 2 and 6 of the MMS Memory Wait State 2 register)
8-bit ROM-BIOS memory	0.5
16-bit memory (all)	0
8-bit external I/O (0100–03FFh)	0.5, 1, 2 (determined by bits 0 and 1 of the Command Delay register)
8-bit internal I/O (000–0FFh)	0.5
16-bit external I/O (0100–03FFh)	0.5

Wait states extend the amount of time the read- or write-command signal is asserted for memory or I/O accesses. This has the effect of increasing the total length of the cycle. Table 2-9 on page 2-15 documents the number of wait states for the various ROM-DOS, ROM-BIOS and ISA cycles.

Table 2-9 Wait States for Various Cycles

Cycle Type	Number of Wait States (in SYSCLK Cycles)
8-bit ISA memory 8-bit ROM BIOS (MMS-accessed)	2, 3, 4, or 5 (determined by bits 0 and 1 of the MMS Memory Wait State 1 register at Index 62h)
8-bit ROM DOS (MMS accessed and linearly decoded)	1, 2, 3, 4, or ISA setting (determined by bits 2–0 of the MMS Memory Wait State 2 register)
8-bit ROM BIOS (not MMS accessed)	2 or 3 (determined by bits 7 and 4 of the Command Delay register at Index 60h)
16-bit ROM-DOS, ROM-BIOS and ISA memory	1, 2, 3, or 4 (determined by bits 3 and 2 of the MMS Memory Wait State 1 register)
8-bit internal I/O (000–0FFh)	2 or 4 (determined by bit 3 of the Wait State Control register at Index 63h)
8-bit floppy-disk-drive I/O (3F0–3F7h)	2, 3, 4, or 5 (determined by bits 1 and 0 of the I/O Wait State register at Index 61h)
8-bit hard-disk-drive I/O (1F0–1F7h)	2, 3, 4, or 5 (determined by bits 3 and 2 of the I/O Wait State register)
Other 8-bit I/O (100–3FFh)	2, 3, 4, or 5 (determined by bits 5 and 4 of the I/O Wait State register)
16-bit I/O	3 or 4 (determined by bit 2 of the Wait State Control register)

2.5.3 High-Speed Clock ROM Cycles

To improve the ROM-access times, an option is provided so that accesses using the $\overline{\text{ROMCS}}$ or $\overline{\text{DOSCS}}$ chip selects may run at the high-speed CPU clock rate rather than the low-speed CPU clock rate of 9.2 MHz.

2.5.4 ROM Chip-Select Signal

The high-speed CPU clock rate is enabled for the ROM Chip-Select ($\overline{\text{ROMCS}}$) signal and unique wait-state controls for each ROM chip-select may be programmed through the Miscellaneous 5 register at Index B3h.

- **Bit 6 of the Miscellaneous 5 register at Index B3h** Enables $\overline{\text{ROMCS}}$ ROM accesses to run at the high-speed CPU clock rate as follows:
  - — 0 Disabled (default)
  - — 1 Enabled
- **Bits 5 and 4 of the Miscellaneous 5 register at Index B3h** Control the number of wait states for fast $\overline{\text{ROMCS}}$ cycles (see Table 2-10 on page 2-16).

Table 2-10 $\overline{\text{ROMCS}}$ Wait-State Control-Bit Logic

Index B3h Bit 5	Index B3h Bit 4	Wait States
0	0	4
0	1	3
1	0	2
1	1	1

Note that if the ÉlanSC310 microcontroller is in its Maximum ISA Bus mode, the BALE output will not be generated for high-speed $\overline{\text{ROMCS}}$ cycles. It is recommended that fast $\overline{\text{ROMCS}}$ cycles not be enabled in systems where the target $\overline{\text{ROMCS}}$ device is an 8-bit device and other devices in the system assert $\overline{\text{MCS16}}$. This is because the $\overline{\text{MCS16}}$ timing is violated when the ISA bus is running at the high-speed PLL frequency. For more details, refer to the *ÉlanSC310 Microcontroller ISA Bus Anomalies Application Note*, available from the Applications Hotline.

2.5.5 DOS Chip-Select Signal

The CPU clock rate is enabled for the DOS Chip-Select ($\overline{\text{DOSCS}}$) signal, and unique wait-state controls for each ROM chip select may be programmed through the ROM Configuration 3 register at Index B8h.

- **Bit 7 of the ROM Configuration 3 register at Index B8h** Enables $\overline{\text{DOSCS}}$ ROM accesses to run at the high-speed CPU clock rate as follows:
  - — 0 Disabled (default)
  - — 1 Enabled
- **Bits 6 and 5 of the ROM Configuration 3 register at Index B8h** Control the number of wait states for fast $\overline{\text{DOSCS}}$ cycles (see Table 2-11 on page 2-16).

Table 2-11 $\overline{\text{DOSCS}}$ Wait-State Control-Bit Logic

Index B8h Bit 6	Index B8h Bit 5	Wait States
0	0	4
0	1	3
1	0	2
1	1	1

Note that if the ÉlanSC310 microcontroller is in its Maximum-ISA-Bus mode, the BALE output will not be generated for high-speed $\overline{\text{DOSCS}}$ cycles.

Bit 1 of the ROM Configuration 2 register at Index 51h should be 1 when enabling fast 16-bit $\overline{\text{DOSCS}}$ cycles. It is recommended that fast $\overline{\text{DOSCS}}$ cycles not be enabled in systems where the target $\overline{\text{DOSCS}}$ device is an 8-bit device and other devices in the system assert $\overline{\text{MCS16}}$. This is because the $\overline{\text{MCS16}}$ timing is violated when the ISA bus is running at the high-speed PLL frequency. For more details, see the *ÉlanSC310 Microcontroller ISA Bus Anomalies Application Note*, available from the Applications Hotline.

2.5.6 Self-Refresh DRAMs

Self-refreshing DRAMs are supported in the ÉlanSC310 microcontroller as follows:

- Bit 3 of the Miscellaneous 5 register at Index B3h enables Self-Refresh mode when the PMU changes to a mode that causes the CPU clock to stop as follows:
  - — 0 Disabled (default)
  - — 1 Enabled
- Upon exiting the stop clock, the system logic forces one CAS-before-RAS refresh cycle before the normal CAS-before-RAS refresh logic takes control.
- If a complete burst row refresh is required by the DRAM, the ÉlanSC310 microcontroller will not directly support this.

2.5.7 80-ns DRAM Support

The ÉlanSC310 microcontroller supports 80-ns DRAMs when running at 25 MHz, and 70-ns DRAMs when running at 33 MHz.

CHAPTER

3 PC/AT PERIPHERAL REGISTERS

The ÉlanSC310 microcontroller contains internal registers used to display the status of various ÉlanSC310 microcontroller internal states, to serve as the target for software commands, to act as data paths to external peripherals, and to access other registers.

All ÉlanSC310 microcontroller registers have an address. A small number of registers have explicit I/O addresses—that is, their location is fixed in the I/O address space of the Am386SXLV processor. However, most of the registers are addressed via an indirect addressing scheme in which a few registers with actual I/O addresses are used to point to the others. This system is explained below in the sections where it is relevant.

This chapter only describes the peripheral registers; the configuration registers are described in the following chapter. For purposes of this manual, the ÉlanSC310 microcontroller's register set is divided into two groups:

- Core AT-compatible peripheral registers (and miscellaneous AT-compatible registers)
- Configuration registers

Registers in the first group have explicit I/O addresses, and are directly addressed in I/O space. Registers in the second group are addressed indirectly, and have an index value instead of an I/O address.

Notes:

1. *When using a logic analyzer to probe the address, data, and control lines of the ÉlanSC310 microcontroller, accesses to the internal registers can be captured, with the exception of reads from even addresses. However, when internal registers are accessed, $\overline{DBUFOE}$ is not asserted.*
2. *In general, the ÉlanSC310 microcontroller's cores decode only up to address bit A9 for I/O accesses. What this means is that the I/O address space is mirrored every 400h bytes. Care should be taken when assigning peripherals I/O addresses above 3FFh so that conflicts with the mirrored I/O registers are avoided.*

3.1 BUS INITIALIZATION REGISTERS

The ÉlanSC310 microcontroller contains a small number of registers that are indexed off I/O address 3D4h and 3D5h. These registers must be initialized before any Local Bus or ISA bus accesses can be done. Refer to Table 4-2, "Mandatory Configuration Bit Settings," on page 4-3 for a typical initialization programming sequence. Once the initialization sequence is complete, these registers can no longer be accessed and any accesses to these I/O addresses will go external on the Local Bus or ISA Bus as appropriate.

Table 3-1 Bus Initialization Index and Data Registers

Port	R/W	Register Name	Description
3D4h	W	Index Address	Index (pointer) to a specific register in this register set
3D5h	R/W	Index Data	Data for the specific register in this register set

3.1.1 Bus Initialization Enable Register (Port 3D4h, Index 12h)

This index must be read before any of the other port 3D4h index registers can be accessed.

Table 3-2 Enabling the Bus Initialization Registers

Perform this Access	To or From this Port	With this Data
Write	3D4h	12h (address value)
Read	3D5h	

3.1.2 Bus Initialization Register (Port 3D4h, Index 19h)

This register must be initialized before any Local Bus or ISA bus accesses can be done. Once this register is written, no other registers in the Port 3D4h register set can be accessed. Thus, if you wish to access the Parallel Port Address Select Register (Port 3D4h, Index 20h) to change the parallel port base address, you must do so before writing Index 19h.

	7							0
Field								
Bit	(Reserved)							
Reset	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7–6		W	(Reserved—must be 0)
5		W	(Reserved—must be 1)
4-0			(Reserved)

3.1.3 Parallel Port Address Select Register (Port 3D4h, Index 20h)

This register controls the base I/O port address of the internal parallel port.

	7							0
Field							Parallel Port I/O Select	
Bit	(Reserved)						PRNPRT1	PRNPRT0
Reset	0	0	0	0	0	0	0	1

Bit	Name	R/W	Function
7–2		W	(Reserved—must be 0)
1–0	PRNPRT1–PRNPRT0	W	Parallel Port I/O Select Printer I/O port select (see Table 3-3 on page 3-3).

Table 3-3 Parallel Port I/O Select

Bit 1	Bit 0	Parallel Port I/O Base
0	0	(Disabled)
0	1	(default) 3Bxh
1	0	37xh
1	1	27xh

3.2 PC/AT-COMPATIBLE PORT REGISTERS

The core peripherals of the ÉlanSC310 microcontroller are four standard integrated circuits and logical components compatible with the IBM PC/AT motherboard:

- 8259A Programmable Interrupt Controller
- 8254 Counter-Timer
- 8237A DMA Controller and DMA Page
- PC/AT Standard Parallel Port
- UART
- Real-Time Clock

In addition, one other logical component is often grouped with the core peripherals:

- Miscellaneous PC/AT-Compatible Registers

In this section and the next, these components are described. The miscellaneous PC/AT-compatible registers are given their own section because these registers are discussed in detail. Only the I/O addresses are shown for the first three components, and the reader is directed to standard data sheets for more details.

3.2.1 Interrupt Controller Registers

The ÉlanSC310 microcontroller has two integrated interrupt controllers. These registers are PC/AT-compatible versions of the industry-standard 8259A programmable interrupt controller. I/O addresses for the interrupt controller are shown in Table 3-4 and in Table 3-5 on page 3-4.

Table 3-4 Interrupt Controller 1

Port	R/W	Description
020h	W	Initialization Control Word 1, IC1 (D4 =1)
021h	W	Initialization Control Word 3, Master, IC1
021h	W	Initialization Control Word 3, Slave, IC1
021h	W	Initialization Control Word 4, IC1
021h	R/W	Operation Control Word 1, (IMR), IC1
020h	W	Operation Control Word 2, IC1 (D5,4 = 00)
020h	W	Operation Control Word 3, IC1 (D5,4 = 01)
020h	R	Interrupt Request Register (IRR), IC1
020h	R	In Service Register (ISR), IC1; State Register, IC1

Note:
Refer to the 8259A Data Sheet for an explanation of the control words.

Table 3-5 Interrupt Controller 2

Port	R/W	Description
0A0h	W	Initialization Control Word 1, IC2 (D4=1)
0A1h	W	Initialization Control Word 2, IC2
0A1h	W	Initialization Control Word 3, Master, IC2
0A1h	W	Initialization Control Word 3, Slave, IC2
0A1h	W	Initialization Control Word 4, IC2
0A1h	R/W	Operation Control Word 1, (IMR), IC2
0A0h	W	Operation Control Word 2, IC2 (D5,4=00)
0A0h	W	Operation Control Word 3, IC2 (D5,4=01)
0A0h	R	Interrupt Request Register (IRR), IC2
0A0h	R	In Service Register (ISR), IC2; State Register, IC2

Note:
Refer to the 8259A Data Sheet for an explanation of the control words.

3.2.2 Programmable Interval Timer Registers

The timer that is integrated into the ÉlanSC310 microcontroller is a PC/AT-compatible version of the industry-standard 8254 programmable interval timer. This counter occupies I/O addresses 040–043h. The timer register definition and locations are shown in Table 3-6 on page 3-5.

Table 3-6 System Timer Registers

Port	R/W	Description
040h	R/W	Timer/Counter 0 Count Register
041h	R/W	Timer/Counter 1 Count Register
042h	R/W	Timer/Counter 2 Count Register
043h	R/W	Timer Control Word Register

Note:
Refer to the 8254 Data Sheet for an explanation of the control words.

The input clock to the ÉlanSC310 microcontroller's 8254 timer runs at 1.1892 MHz. This deviates from the AT-compatible standard of 1.19318 MHz. This 0.4% difference can result in incorrect DOS clock readings over time if the difference is not taken into account.

In AT-compatible systems, the BIOS loads a value of 0FFFFh into Timer 0, resulting in a timer interrupt every 54.93 ms. To program the ÉlanSC310 microcontroller to generate the Timer-0 interrupt at the same rate, load Timer 0 with a value of 0FF23h.

A second option is to write the BIOS timer tick handler to reload the DOS clock at regular intervals from the real-time clock (RTC), which maintains accurate time.

3.2.3 DMA Controller Registers

The two DMA controllers that are integrated in the ÉlanSC310 microcontroller are PC/AT-compatible versions of the industry-standard 8237A DMA controller. DMA controller addresses are shown in Table 3-7 on page 3-6 and in Table 3-8 on page 3-7.

Table 3-7 DMA Controller 1

Port	R/W	Description
000h	R/W	CH0 Base Address
001h	R/W	CH0 Base Word Count
002h	R/W	CH1 Base Address
003h	R/W	CH1 Base Word Count
004h	R/W	CH2 Base Address
005h	R/W	CH2 Base Word Count
006h	R/W	CH3 Base Address
007h	R/W	CH3 Base Word Count
008h	R/W	CH3–0 Read Status Register/Write Command Register
009h	R/W	CH3–0 Write Request Register
00Ah	W	CH3–0 Write Single Mask Register Bit
00Bh	W	CH3–0 Write Mode Register
00Ch	W	CH3–0 Clear Byte Pointer Flip-Flop
00Dh	R/W	CH3–0 Read Temporary Register/Write Master Clear
00Eh	R/W	CH3–0 Clear Mask Register
00Fh	W	CH3–0 Write All Mask Register Bits

Note:

Refer to the 8237A Data Sheet for an explanation of these registers.

Table 3-8 DMA Controller 2

Port	R/W	Description
0C0h	R/W	CH4 Base Address
0C2h	R/W	CH4 Base Word Count
0C4h	R/W	CH5 Base Address
0C6h	R/W	CH5 Base Word Count
0C8h	R/W	CH6 Base Address
0CAh	R/W	CH6 Base Word Count
0CCh	R/W	CH7 Base Address
0CEh	R/W	CH7 Base Word Count
0D0h	R/W	Read Status Register/Write Command Register
0D2h	W	Write Request Register
0D4h	W	Write Single Mask Register Bit
0D6h	W	Write Mode Register
0D8h	W	Clear Byte Pointer Flip-Flop
0DAh	R/W	Read Temporary Register/Write Master Clear
0DCh	W	Clear Mask Register
0DEh	W	Write All Mask Register Bits

Note:

Refer to the 8237A Data Sheet for an explanation of these registers.

3.2.4 DMA Page Registers

DMA page registers provide the upper address bits during DMA transfers. The processor writes the page registers before enabling DMA transfers. Addresses for the page registers are shown in Table 3-9 on page 3-8.

Table 3-9 DMA Page Registers

Address	R/W	Description
0080h	R/W	General Register
0081h	R/W	Channel 2 Page Register
0082h	R/W	Channel 3 Page Register
0083h	R/W	Channel 1 Page Register
0084h	R/W	General Register
0085h	R/W	General Register
0086h	R/W	General Register
0087h	R/W	Channel 0 Page Register
0088h	R/W	General Register
0089h	R/W	Channel 6 Page Register
008Ah	R/W	Channel 7 Page Register
008Bh	R/W	Channel 5 Page Register
008Ch	R/W	General Register
008Dh	R/W	General Register
008Eh	R/W	General Register
008Fh	R/W	General Register

3.2.5 Parallel Port Interface Registers

The parallel port is register-compatible with the industry-standard, AT-compatible, EPP-compliant parallel port. The parallel port can be set up to have a base I/O address of 3BCh, 278h, or 378h. EPP support is only possible when the base I/O address is set to 278h or 378h. EPP-mode enable and bidirectional enable/control are set by bits 2–0 of the Function Enable 1 register at Index B0h. The parallel port interrupt is fixed at IRQ7.

The base I/O address is set using bits 1–0 of the Parallel Port Address Select Register (Port 3D4h, Index 20h).The setting of the parallel port base I/O address can only be done during the processor initialization sequence. Refer to Section 3.1.2, "Bus Initialization Register (Port 3D4h, Index 19h)," on page 3-2 and to Table 4-2, "Mandatory Configuration Bit Settings," on page 4-3 for details on this initialization sequence.

After the parallel port I/O address has been set and, optionally, the EPP support or bidirectional support has been enabled, then the parallel port registers may be accessed as I/O ports. The parallel port register fields are different for AT-compatible mode vs. EPP-compliant mode. Section 3.2.5.1, "AT-Compatible Mode," describes the parallel port registers when the parallel port is configured for AT-compatible mode. Section 3.2.5.2, "EPP-Compliant Mode," describes the parallel port registers when the parallel port is configured for EPP-compliant mode.

3.2.5.1 AT-Compatible Mode

3.2.5.1.1 Parallel Data Port (Ports 278h, 378h, & 3BCh)

	7	0
Bit	Data Bits 7–0	
Default		

Bit	Name	R/W	Function
7–0		R/W	Parallel Port Data Register

3.2.5.1.2 Parallel Status Port (Ports 279h, 379h, & 3BDh)

	7							0
Bit	$\overline{\text{BUSY}}$	$\overline{\text{ACK}}$	PE	SLCT	$\overline{\text{ERR}}$	(Reserved)		
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7	$\overline{\text{BUSY}}$	R	0 = Printer Busy (active Low)
6	$\overline{\text{ACK}}$	R	0 = Printer Acknowledge. The printer pulses this line Low when it has received a byte of data (active Low)
5	PE	R	1 = Paper End
4	SLCT	R	1 = Printer Selected
3	$\overline{\text{ERR}}$	R	0 = Printer Error (active Low)
2–0		R	(Reserved)

3.2.5.1.3 Parallel Control Port (Ports 27Ah, 37Ah, & 37Eh)

	7							0
Bit	(Reserved)		DIR	IRQEN	SLCTIN	$\overline{\text{INIT}}$	AUTOFDXT	STROBE
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7–6		R	(Reserved)
5	DIR	R/W	Bidirectional parallel port data direction: 0 = Out (normal printer) 1 = In
4	IRQEN	R/W	1 = Printer IRQ enable. Clearing this bit clears any pending interrupts.
3	SLCTIN	R/W	1 = Printer selected 0 = Not selected
2	$\overline{\text{INIT}}$	R/W	Printer reset signal control: 0 = Hold printer in reset 1 = Release printer from reset
1	AUTOFDXT	R/W	Auto-line feed signal control: 1 = $\overline{\text{AUTOFDXT}}$ pin active
0	STROBE	R/W	Printer port strobe signal control: 1 = $\overline{\text{STROBE}}$ pin active

3.2.5.2 EPP-Compliant Mode

3.2.5.2.1 Parallel Data Port (Ports 278h & 378h)

	7	0
Bit	Data Bits 7–0	
Default	(Undefined)	

Bit	Name	R/W	Function
7–0		R/W	Parallel Port Data Register

3.2.5.2.2 Parallel Status Port (Ports 279h & 379h)

	7							0
Bit	$\overline{\text{BUSY}}$	$\overline{\text{ACK}}$	PE	SLCT	$\overline{\text{ERR}}$	ACKSTAT	(Reserved)	EPPTO
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7	$\overline{\text{BUSY}}$	R	0 = Printer busy (active Low)
6	$\overline{\text{ACK}}$	R	0 = Printer acknowledge. The printer pulses this line Low when it has received a byte of data (active Low).
5	PE	R	1 = Paper end
4	SLCT	R	1 = Printer selected
3	$\overline{\text{ERR}}$	R	0 = Printer error (active Low)
2	ACKSTAT	R	This bit is latched Low when ACK transitions Low to High. Reading this bit sets it to 1.
1		R	(Reserved)
0	EPPTO	R	EPP time-out status: 0 = No time-out 1 = EPP cycle time-out occurred This bit is reset when either the status register is read or when EPP mode is enabled.

3.2.5.2.3 Parallel Control Port (Ports 27Ah & 37Ah)

	7							0
Bit	(Reserved)		DIR	IRQEN	(Reserved)	$\overline{\text{INIT}}$	(Reserved)	
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7–6		R	(Reserved)
5	DIR	R/W	Bidirectional parallel port data direction: 0 = Out (normal printer) 1 = In
4	IRQEN	R/W	1 = Printer IRQ enable. Clearing this bit clears any pending interrupts.
3		R	(Reserved)
2	$\overline{\text{INIT}}$	R/W	Printer reset signal control: 0 = Hold printer in reset 1 = Release printer from reset
1–0		R	(Reserved)

3.2.5.2.4 Parallel EPP Address Port (Ports 27Bh & 37Bh)

	7							0
Bit	Address Bits 7–0							
Default	(Undefined)							

Bit	Name	R/W	Function
7–0		R/W	Address Register

3.2.5.2.5 Parallel EPP 32-Bit Data Register (Ports 27C–27Fh & 37C–37Fh)

A 32-bit I/O write to 27Ch or 37Ch causes four back-to-back 8-bit bus cycles to occur to the four EPP data registers. An EPP data strobe is generated for all four bus cycles. A 16-bit I/O write to 27Ch or 37Ch causes two back-to-back 8-bit bus cycles to occur to the EPP data registers 27Ch or 37Ch and 27Dh or 37Dh. In common practice, all write accesses to the parallel port EPP data register (8-, 16-, or 32-bit I/O instructions) should be directed to port 27Ch or 37Ch.

	7							0
Bit	Address Bits 7–0							
Default	(Undefined)							

Bit	Name	R/W	Function
7–0		R/W	Address Register

3.2.6 UART Registers

The UART implemented in the ÉlanSC310 microcontroller is register-compatible with the industry-standard 16450 universal asynchronous receiver/transmitter.

3.2.6.1 Transmitter Holding Register (Ports 2F8h & 3F8h)

This write-only register contains the data waiting to be transferred. Bit 7 is the most significant bit; bit 0 is the least significant bit.

	7							0
Field	Data to be Transmitted							
Default	0	0	0	0	0	0	0	0

3.2.6.2 Receiver Buffer Register (Ports 2F8h & 3F8h)

This read-only register contains the data received. Bit 7 is the most significant bit; bit 0 is the least significant bit.

	7							0
Field	Data to be Received							
Default	0	0	0	0	0	0	0	0

3.2.6.3 Divisor Latch Lower Byte (Ports 2F8h & 3F8h)

This register controls the programmable baud-rate generator that is capable of dividing 1.843 MHz by values from 1 to (2^{16} – 1). Two 8-bit registers store the divisor; this read/write register contains the lower byte.

	7							0
Field	Baud-Rate Divisor, Lower Byte							
Default	0	0	0	0	0	0	0	0

3.2.6.4 Divisor Latch Upper Byte (Ports 2F8h & 3F9h)

This register controls the programmable baud-rate generator that is capable of dividing 1.843 MHz (I/P OSC) by values from 1 to (2^{16} – 1). Two 8-bit registers store the divisor; this read/write register contains the upper byte.

	7							0
Field	Baud-Rate Divisor, Upper Byte							
Default	0	0	0	0	0	0	0	0

3.2.6.5 Interrupt Enable Register (Ports 2F9h & 3F9h)

This register is used to enable UART functions.

	7							0
Bit	(Reserved)				EMSI	ELSI	ETDEI	ERDI
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7–4		W	(Reserved)
3	EMSI	W	Enable modem status interrupt
2	ELSI	W	Enable receiver line status interrupt
1	ETDEI	W	Enable transmitter holding register empty interrupt
0	ERDI	W	Enable receiver data ready interrupt

3.2.6.6 Interrupt Identification Register (Ports 2FAh & 3FAh)

This register is used to identify the nature of an interrupt.

	7							0
Field						Interrupt ID		
Bit	(Reserved)					ID1	ID0	IP
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7–3		R	(Reserved)
2	ID1	R	Interrupt ID, bit 1
1	ID0	R	Interrupt ID, bit 0
0	IP	R	0 = Interrupt pending

Table 3-10 Interrupt ID Bit Logic

ID1	ID0	Priority	Interrupt Type
1	1	1	Receiver line status
1	0	2	Received data ready
0	1	3	Transmitter holding register empty
0	0	4	Modem status

3.2.6.7 Line Control Register (Ports 2FBh & 3FBh)

This register controls certain line characteristics.

7 0

Field							Word Length	
Bit	DLAB	SB	SP	EPS	PE	STP	WLB1	WLB0
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7	DLAB	R/W	Divisor latch access bit 0: 1 = Enable divisor programming 0 = Disable divisor programming
6	SB	R/W	Set break bit
5	SP	R/W	Stuck parity: 1 = Enable
4	EPS	R/W	Even parity: 1 = Enable
3	PE	R/W	Parity enable: 1 = Enable
2	STP	R/W	Stop bit: 0 = 1 stop bit 1 = 1.5 stop bits if word length is 5 (2 stop bits if word length is 6, 7, or 8)
1	WLB1	R/W	Word length, bit 1
0	WLB0	R/W	Word length, bit 0

Table 3-11 Word Length Bit Logic

Bit 1	Bit 0	Word Length (Bits)
0	0	5
0	1	6
1	0	7
1	1	8

3.2.6.8 Modem Control Register (Ports 2FCh & 3FCh)

This register governs certain modem characteristics.

	7							0
Bit	(Reserved)			LOOP	$\overline{OUT2}$	OUT1	RTS	DTR
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7–5		R/W	(Reserved)
4	LOOP	R/W	Local loop-back diagnostic test: 1 = Enable loop 0 = Normal operation
3	$\overline{OUT2}$	R/W	$\overline{OUT2}$ (must be 1 to enable interrupts)
2	$\overline{OUT1}$	R/W	OUT1
1	RTS	R/W	$\overline{RTS}$: 1 = Enable $\overline{RTS}$
0	DTR	R/W	$\overline{DTR}$: 1 = Enable $\overline{DTR}$

Note:

If loop-back is enabled, interrupts from the UART are disabled and the four modem control signals ($\overline{DTR}$, $\overline{RTS}$, $\overline{OUT1}$, and $\overline{OUT2}$) are internally connected to $\overline{DSR}$, $\overline{CTS}$, $\overline{RI}$, and $\overline{DCD}$, respectively.

3.2.6.9 Line Status Register (Ports 2FDh & 3FDh)

This register reports the line status. This is a read-only register unless bit 1 of the Control A register at Index 48h is 1, in which case it becomes read/write.

	7							0
Bit	(Reserved)	TEMT	THRE	BI	FE	PE	OE	DR
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7		R	(Reserved)
6	TEMT	R	1 = Transmitter empty
5	THRE	R	1 = Transmitter holding register empty
4	BI	R	1 = Break interrupt
3	FE	R	1 = Framing error
2	PE	R	1 = Parity error
1	OE	R	1 = Overrun error
0	DR	R	1 = Data ready

3.2.6.10 Modem Status Register (Ports 2FEh & 3FEh)

This register reports more status information.

	7							0
Bit	RLSD	RI	DSR	CTS	DRLSD	TERI	DDSR	DCTS
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7	RLSD	R	Status of $\overline{\text{DCD}}$ from modem
6	RI	R	Status of $\overline{\text{RI}}$ from modem
5	DSR	R	Status of $\overline{\text{DSR}}$ from modem
4	CTS	R	Status of $\overline{\text{CTS}}$ from modem
3	DRLSD	R	Delta data carrier detect
2	TERI	R	Trailing edge of ring indicator from modem
1	DDSR	R	Delta data set ready
0	DCTS	R	Delta clear to send

3.2.6.11 Scratch Pad Register (Ports 2FFh & 3FFh)

This is a general-purpose register.

	7							0
Field	I/O Port Number							
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7–0		R/W	Scratch data

3.2.7 Real-Time Clock

- Counts seconds, minutes, and hours of the day
- Counts days of the week, date, month, and year
- Binary or BCD representation of time, calendar, and alarm
- 12–24-hr clock with A.M. and P.M. in 12-hr mode
- Daylight-savings-time option
- Automatic end-of-month recognition
- Automatic leap-year compensation
- 14 bytes of clock and control registers
- 114 bytes of general-purpose RAM
- Three interrupts are separately software maskable and testable
- Time-of-day alarm (once per second to once per day)
- Periodic interrupt rates from 122 ms to 500 ms
- End-of-clock update cycle

The real-time clock (RTC) block in the ÉlanSC310 microcontroller is compatible with the Motorola MC146818A device used in PC/AT systems. This block consists of a time-of-day clock with an alarm and a 100-yr calendar. The clock/calendar can be represented in binary or BCD, has a programmable periodic interrupt, and has 114 bytes of static user RAM. The RTC block is powered from the same power as the core logic.

The block also has 10 registers for time, calendar, and alarm data and four general-purpose registers. The interrupt signal from the RTC is an active-Low signal and is routed to interrupt-request line 8.

3.2.7.1 Addressing

The RTC is accessed using 8-bit-wide I/O cycles at addresses 70h and 71h. The contents of the RTC registers and its 114 bytes of RAM are accessed in an indexed fashion.

Table 3-12 Real-Time Clock

Port	R/W	Register Name	Description
070h	W	RTC Index Address	Index (pointer) to the data in the RTC
071h	R/W	RTC Index Data	Data stored in the RTC

Note:

Bit 7 of the NMI/RTC Index register, I/O address 70h, is the ENMI control bit. If this bit is set, NMIs are masked. Bits 6–0 of this register are used to access the RTC registers and RAM.

3.2.7.2 RTC Registers

The RTC registers and RAM are indexed with the address map shown in Table 3-13 on page 3-17.

Table 3-13 RTC Register Summary

RTC Index	Function
0	Seconds
1	Seconds Alarm
2	Minutes
3	Minutes Alarm
4	Hours
5	Hours Alarm
6	Day of Week
7	Date of Month
8	Month
9	Year
10	Register A
11	Register B
12	Register C
13	Register D
14–127	User-Static RAM

3.2.7.2.1 Register A (RTC Index 0Ah)

	7							0
Bit	UIP	DV2	DV1	DV0	RS3	RS2	RS1	RS0
Default	(Undefined)							

Bit	Name	Function
7	UIP	Update in Progress: This bit is a status bit that can be polled by a program to indicate that the clock/calendar is currently being updated.
6–4	DV2–DV0	Divider Chain: These bits select the time-base frequency for the clock. (Table 3-15 on page 3-18 describes the operation of these bits.)
3–0	RS3–RS0	Rate Selection: These bits select the periodic interrupt rate. (Table 3-14 on page 3-18 describes the operation of these bits.)

Table 3-14 Register A Periodic-Interrupt Rate-Selection Bits (32.768 kHz)

RS3	RS2	RS1	RS0	Periodic Interrupt Rate
0	0	0	0	(None)
0	0	0	1	(Reserved)
0	0	1	0	(Reserved)
0	0	1	1	122.070 µs
0	1	0	0	244.141 µs
0	1	0	1	488.281 µs
0	1	1	0	976.562 µs
0	1	1	1	1.953125 ms
1	0	0	0	3.90625 ms
1	0	0	1	7.8125 ms
1	0	1	0	15.625 ms
1	0	1	1	31.25 ms
1	1	0	0	62.5 ms
1	1	0	1	125 ms
1	1	1	0	250 ms
1	1	1	1	500 ms

Table 3-15 Register A Time-Base Divider-Chain Bits

DV2	DV1	DV0	Time Base Frequency
0	1	0	32.768 kHz (AT-compatible setting)

Note:

Other combinations of divider bits are used for test purposes only.

3.2.7.2.2 Register B (RTC Index 0Bh)

	7							0
Bit	SET	PIE	AIE	UIE	SQWE	DM	24/12	DSE
Default	(Undefined)							

Bit	Name	Function
7	SET	Set Time: This bit prevents the time calendar from being incremented.
6	PIE	Programmable Interrupt Enable: This bit enables the periodic interrupt flag (PF) in Register C to generate an interrupt request.
5	AIE	Alarm Interrupt Enable: This bit enables the alarm flag (AF) in Register C to generate an interrupt request.
4	UIE	Update-Ended Enable: This bit enables the update-ended flag (UF) in Register C to generate an interrupt request.
3	SQWE	Square Wave Enable: This bit is not used.
2	DM	Data Mode: This bit selects between BCD and binary formats for the time and calendar: 1 = BCD 0 = Binary
1	24/12	Hour Format: This bit establishes the format of the hours byte for 24- or 12-hr time: 1 = 24-hr clock 0 = 12-hr clock
0	DSE	Daylight Savings Time: This bit enables the clock to automatically account for daylight savings time adjustments: 1 = Enabled 0 = Disabled

3.2.7.2.3 Register C (RTC Index 0Ch)

	7							0
Bit	IRQF	PF	AF	UF	(Reserved)			
Default					0	0	0	0

Bit	Name	Function
7	IRQF	Interrupt Request Flag: This bit is read only and is set when any of these Register C bits are set: PF, AF, or UF.
6	PF	Periodic Interrupt Flag: This bit is read only and is set according to the rate set by the RS3–RS0 bits in Register A.
5	AF	Alarm Flag: This bit is read only and is set when the current item matches the alarm time.
4	UF	Update-Ended Flag: This bit is read only and is set at the end of each time-update cycle.
3–0		(Reserved)

3.2.7.2.4 Register D (RTC Index 0Dh)

	7							0
Bit	VRT	(Reserved)						
Default		0	0	0	0	0	0	0

Bit	Name	Function
7	VRT	Valid RAM and Time: This bit is read only and is reset when the $\overline{\text{RESIN}}$ pin is asserted Low. This bit is set by reading Register D.
6–0		(Reserved)

3.3 MISCELLANEOUS PC/AT-COMPATIBLE PORT REGISTERS

3.3.1 XT Keyboard Data Register (Port 060h)

The XT Keyboard Data register is located at I/O address 060h. This read-only, 8-bit register holds the data from the XT keyboard. The XT keyboard is enabled by setting bit 3 (XTKBDEN) of the PMU Control 3 register at Index ADh. While the XT keyboard is enabled, this register is cleared by setting bit 7 of the Port B register.

	7							0
Field	Data Received from the XT Keyboard							
Default	0	0	0	0	0	0	0	0

3.3.2 Port B Register (Port 061h)

Port B is an AT-standard, miscellaneous-feature control register which is located at I/O address 061h. The lower 4 bits of the 8-bit register are read/write control bits that enable or disable NMI check-condition sources and sound-generation features. The most significant 4 bits are read-only bits that return status and diagnostic information. Bit 4 (RFD) toggles state with every refresh. Bit 5 (T2OUT) follows the state of the Timer 2 output. Bit 6 (IOCHCK) is set upon detection of a channel check and is cleared upon the reading of this register.

Bits 6 and 7 of this register have alternate functions when the XT keyboard feature is enabled by setting bit 3 of Index ADh.

	7							0
Bit	(Reserved)	IOCHCK	T2OUT	RFD	EIC	(Reserved)	SPKD	T2G
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7		W	(Reserved)
(Alt.)		R/W	1 = Clear XT Keyboard Data register
6	IOCHCK	R	I/O channel check
(Alt.)	KBCLKEN	R/W	1 = Enable KBCLK as an output
5	T2OUT	R	Timer 2 output
4	RFD	R	Refresh detected
3	EIC	W/R	Enable I/O channel check: 0 = Enable
2			(Reserved)
1	SPKD	W/R	Speaker data: 1 = Enable speaker output
0	T2G	W/R	Timer 2 gate (speaker): 1 = Speaker gated on

3.3.3 NMI/RTC Index Address Register (Port 070h)

Both the PMU and the I/O channel check are possible sources for the generation of an NMI to the internal CPU. The master NMI enable function can inhibit any NMIs from reaching the CPU regardless of the state of the individual source enables. The NMI enable bit in this register is a write-only bit and it has an active-0 sense. The default value for the NMI enable bit is 1, which inhibits NMI generation. A write to this I/O address must always be followed by a write to—or read from—address 71h to ensure proper operation of the RTC.

	7							0
Bit	ENMI	RTC Index Address Bits 6–0						
Default	1	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7	ENMI	W	0 = Enable NMI 1 = Disable NMI (default)
6–0		W	RTC index address

Note:

For a definition of the index values, see Real-Time Clock, on page 3-16.

3.3.4 RTC Index Data Register (Port 071h)

This register is used to read data from, or write data to, the real-time clock indexed by bits 6–0 in I/O register 070h.

	7							0
Bit	RTC Data Bits 7–0							
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7–0		R/W	RTC data bits 7–0

3.3.5 Port 92 (Port 092h)

This is the System control Port A Register and is for AT compatibility.

	7							0
Bit	(Reserved)						ALTA20	HOTRST
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7–2		R	(Reserved)
1	ALTA20	W	A20 gate control: 1 = A20 is CPUA20 0 = A20 is 0 if bit 0 of Index 6Fh is 0 and the A20GATE pin is 0
0	HOTRST	W	A Low-to-High transition in this bit causes CPURST to be asserted for 16 CPU clock cycles.

CHAPTER

4 CONFIGURATION REGISTERS

The ÉlanSC310 microcontroller's configuration registers are defined as those registers that control the four central functions:

- General Control
- ISA Bus Control
- Memory Control Unit (MCU)
- Power Management Unit (PMU)

These registers provide a uniform method of accessing the device's control and configuration parameters. The parameters are mapped into bits and bit fields contained in logical groupings of 8-bit registers.

Most of the configuration registers are set to a default value by a system reset condition. The default values have been specified to allow the device to correctly execute BIOS code out of the local Flash memory or ROM upon exiting the reset state.

4.1 CONFIGURATION REGISTER OVERVIEW

There are two basic types of registers that are used to configure the ÉlanSC310 microcontroller:

- Port registers
- Index registers

4.1.1 Port Registers

The configuration port registers are located in I/O space. They are addressed directly by the ÉlanSC310 microcontroller. The configuration port registers are listed in Table 4-1 on page 4-2. For a description of each port register, see “Configuration Port Registers” on page 4-11.

Table 4-1 Configuration Port Registers

Port	Register
022h	Configuration Address
023h	Configuration Data

The Configuration Address register points to the index register to be read or written when a data-register I/O cycle occurs. The Configuration Data register is used to read or write the index register.

Reading the Configuration Data register reads the contents of the index register. Writing a value to the Configuration Data register writes that value to the index register.

The contents of the Configuration Address register is static relative to multiple data register reads and writes. This allows for read-modify-write operations. The address of the index register is written to the Configuration Address register, and then the data is read, modified, and written back to the Configuration Data register without rewriting the Configuration Address register.

4.1.2 Index Registers

The configuration index registers are located in a separate internal memory space. They are addressed indirectly by the ÉlanSC310 microcontroller. For a description of each index register, see “Configuration Index Registers” on page 4-11.

4.1.2.1 Mandatory Settings

Default settings of the microcontroller guarantee that code can be fetched and executed out of the ROM BIOS address space and that the configuration index registers can be accessed upon power up. Several of the configuration index registers contain bits that must be written soon after reset with their specified values in order for the device to operate as specified. These mandatory bits control things such as chip-test modes, interrupt mapping, SMI generation, and cycle timings.

Setting of these mandatory bits must be done prior to enabling any other functions of the device. This chapter describes the configuration registers that are indexed off I/O Port 22h. However, Table 4.2, “Mandatory Configuration Bit Settings” on page 4-3 shows a few registers that are accessed through port 3D4h and 3D5h. Port 3D4h registers are described in the section “Bus Initialization Registers” on page 3-2. The “Mandatory Configuration Bit Settings” table shows the following:

- Mandatory bit settings indexed off port 3D4h for any speed operation
- Mandatory bit settings indexed off port 22h for any speed operation
- Additional mandatory bit settings indexed off port 22h for 33-MHz system speed
- Mandatory bit settings indexed off port 22h for 33-MHz and 25-MHz system speed

Table 4-2 Mandatory Configuration Bit Settings

Mandatory Bit Settings Accessed Through Ports 3D4h and 3D5h	
Index	**Bit Settings (x = Don’t Care)**
12h	read (see Note 1)
20h	0 0 0 0 0 0 x x (see Note 2)
19h	0 0 1 x x x x x
1. Port 3D4h, Index 12h, must be read before Indexes 20h and 19h are written. *2. Note: Port 3D4h, Index 20h, selects the parallel port base I/O address through bits 0 and 1. This selection can only be made before Index 19h is written.*	
Mandatory Bit Settings Indexed off Port 22h	
Index	**Bit Settings (x = Don’t Care)**
0Fh	1 1 1 1 1 1 1 1
44h	1 0 x x 0 0 0 x
51h	x x x x 0 0 x x
60h	x x 0 x 0 x x x
62h	0 x x x x x x x
63h	x x x x x x 0 x
64h	1 0 0 x 1 1 x x
66h	x x x x x x 1 x
6Ah	0 0 0 0 0 0 0 0
6Bh	0 x x 1 x x x x
70h	x x x x x x x 1
74h	x x 0 0 x x x x
80h	x x x x 0 x x x
8Fh	1 x x 0 x x x x
93h	0 0 0 0 0 0 0 0
9Dh	0 1 0 0 0 0 0 0

Table 4-2 Mandatory Configuration Bit Settings (continued)

Mandatory Bit Settings Indexed off Port 22h	
Index	**Bit Settings (x = Don't Care)**
B1h	x x 0 x x x x x
B4h	x 1 x x x x x x
BAh	x x x 1 x x x x
Additional 33-MHz Bit Settings Indexed off Port 22h	
Index	**Bit Settings (x = Don't Care)**
60h	x 0 0 x 0 x x x
62h	0 x x 1 x x x x
63h	x 1 1 x x x 0 x
65h	x x 1 x x x x x
Additional 33-MHz and 25-MHz Bit Settings Indexed off Port 22h	
Index	**Bit Settings (x = Don't Care)**
6Bh	0 x x 1 x 0 x x

4.1.2.2 Recommended Settings

Table 4-3 on page 4-4 contains recommended bit settings that improve the performance of the microcontroller under certain conditions. These bits should be set in addition to the bits listed in Table 4-2 on page 4-3.

Table 4-3 Recommended Configuration Bit Settings

Index	Bit Settings (x = Don't Care)
44h	x x x 1 x x x x
63h	x x x x 1 1 x x
A7h	x x 1 1 x x x 0

4.1.3 Configuring the ÉlanSC310 Microcontroller

When you are ready to configure the ÉlanSC310 microcontroller, you program its configuration index registers according to your system's requirements. The following sections identify the index registers—and the bits within them—that you use to control different aspects of the ÉlanSC310 microcontroller.

4.1.3.1 To Control the PC/AT Bus and Its Timing

See	And program bits
“Control A Register (Index 48h)” on page 4-18	2
“Command Delay Register (Index 60h)” on page 4-20	2–0
“I/O Wait State Register (Index 61h)” on page 4-22	5–0
“MMS Memory Wait State 1 Register (Index 62h)” on page 4-23	3–0
“Wait State Control Register (Index 63h)” on page 4-25	3–2
“Miscellaneous 5 Register (Index B3h)” on page 4-65	6, 5–4, and 2
“ROM Configuration 3 Register (Index B8h)” on page 4-67	7–0

4.1.3.2 To Determine the Bus Configuration

See	And program bits
“Memory Configuration 1 Register (Index 66h)” on page 4-28	6–5

4.1.3.3 To Control CPU and PC/AT Compatibility

See	And program bits
“Parallel Port Address Select Register (Port 3D4h, Index 20h)” on page 3-3	1–0
“Port 92 (Port 092h)” on page 3-21	1–0
“Control A Register (Index 48h)” on page 4-18	1
“Miscellaneous 2 Register (Index 6Bh)” on page 4-30	0
“Miscellaneous 1 Register (Index 6Fh)” on page 4-34	1–0
“Control B Register (Index 77h)” on page 4-42	7–4
“UART Clock Enable Register (Index 92h)” on page 4-49	0

4.1.3.4 To Control the Speed of the CPU

See	And program bits
“I/O Wait State Register (Index 61h)” on page 4-22	6
“Miscellaneous 2 Register (Index 6Bh)” on page 4-30	2
“Control B Register (Index 77h)” on page 4-42	3
“Auto Low-Speed Control Register (Index 9Fh)” on page 4-51	3–0
“PMU Control 3 Register (Index ADh)” on page 4-60	1–0
“PMU Control 2 Register (Index AFh)” on page 4-61	5
“Function Enable 2 Register (Index B1h)” on page 4-62	4–3

4.1.3.5 To Control Direct Memory Accesses

See	And program bits
“I/O Wait State Register (Index 61h)” on page 4-22	7
“Miscellaneous 1 Register (Index 6Fh)” on page 4-34	3–2
“Function Enable 1 Register (Index B0h)” on page 4-61	3

4.1.3.6 To Enable Interrupts and Specify How They Are Mapped

See	And program bits
“Control B Register (Index 77h)” on page 4-42	7–6
“NMI/SMI Enable Register (Index 82h)” on page 4-43	7–0
“SMI MMS Upper Page Register (Index A9h)” on page 4-57	7
“PMU Control 3 Register (Index ADh)” on page 4-60	6
“Function Enable 1 Register (Index B0h)” on page 4-61	5–4
“PIRQ Configuration Register (Index B2h)” on page 4-64	7–0

4.1.3.7 To Set Up Memory Mapping (MMS Windows)

See	And program bits
“ROM Configuration 1 Register (Index 65h)” on page 4-27	6
“MMSA Address Extension 1 Register (Index 67h)” on page 4-29	7–0
“MMS Address Extension 1 Register (Index 6Ch)” on page 4-31	7–0
“MMS Address Register (Index 6Dh)” on page 4-31	7–0
“MMS Address Extension 2 Register (Index 6Eh)” on page 4-33	7–0
“MMSA Device 1 Register (Index 71h)” on page 4-36	7–0
“MMSA Device 2 Register (Index 72h)” on page 4-37	7–0
“MMSB Device Register (Index 73h)” on page 4-38	7–0
“MMSB Control Register (Index 74h)” on page 4-39	1–0

4.1.3.8 To Set Up the System DRAM

See	And program bits
“Miscellaneous 4 Register (Index 44h)” on page 4-15	4
“Command Delay Register (Index 60h)” on page 4-20	6
“MMS Memory Wait State 1 Register (Index 62h)” on page 4-23	4
“Wait State Control Register (Index 63h)” on page 4-25	6–4
“Version Register (Index 64h)” on page 4-26	4 and 1–0
“ROM Configuration 1 Register (Index 65h)” on page 4-27	7 and 5–4
“Memory Configuration 1 Register (Index 66h)” on page 4-28	4–0
“Shadow RAM Enable 1 Register (Index 68h)” on page 4-29	7–0
“Shadow RAM Enable 2 Register (Index 69h)” on page 4-30	7–0
“Miscellaneous 1 Register (Index 6Fh)” on page 4-34	7–4
“Miscellaneous 6 Register (Index 70h)” on page 4-35	0
“PMU Control 1 Register (Index A7h)” on page 4-56	1–0
“Function Enable 2 Register (Index B1h)” on page 4-62	7–6
“Miscellaneous 5 Register (Index B3h)” on page 4-65	3
“Function Enable 3 Register (Index B4h)” on page 4-66	7
“Memory Configuration 2 Register (Index B9h)” on page 4-69	7–0
“Miscellaneous 3 Register (Index BAh)” on page 4-70	2

4.1.3.9 To Set Up the Parallel Port

See	And program bits
“Parallel Port Address Select Register (Port 3D4h, Index 20h)” on page 3-3	1–0
“Function Enable 1 Register (Index B0h)” on page 4-61	2–0
“Miscellaneous 3 Register (Index BAh)” on page 4-70	4–3

4.1.3.10 To Set Up the UART

See	And program bits
“Control A Register (Index 48h)” on page 4-18	1
“Control B Register (Index 77h)” on page 4-42	7–4
“UART Clock Enable Register (Index 92h)” on page 4-49	0

4.1.3.11 To Set Up the General-Purpose and PMC Pins

See	And program bits
“Miscellaneous 6 Register (Index 70h)” on page 4-35	6
“MMSB Control Register (Index 74h)” on page 4-39	2
“Power Control 1 Register (Index 80h)” on page 4-42	6 and 2
“Power Control 2 Register (Index 81h)” on page 4-43	6 and 2
“General-Purpose I/O 0 Register (Index 89h)” on page 4-46	7–0
“General-Purpose I/O Control Register (Index 91h)” on page 4-47	7–0
“General-Purpose I/O 2 Register (Index 94h)” on page 4-49	7–0
“General-Purpose I/O 3 Register (Index 95h)” on page 4-49	7–0
“General-Purpose I/O 1 Register (Index 9Ch)” on page 4-51	7–0
“CPU Status 0 Register (Index A3h)” on page 4-54	0
“CPU Status 1 Register (Index A4h)” on page 4-55	3
“Power Control 3 Register (Index ABh)” on page 4-58	7–0
“Power Control 4 Register (Index ACh)” on page 4-59	7–0
“PMU Control 3 Register (Index ADh)” on page 4-60	3
“PIRQ Configuration Register (Index B2h)” on page 4-64	7–4 and 3–0
“Miscellaneous 3 Register (Index BAh)” on page 4-70	4–3

4.1.3.12 To Control the Clocks (Phase-Locked Loops)

See	And program bits
“I/O Wait State Register (Index 61h)” on page 4-22	6
“Miscellaneous 2 Register (Index 6Bh)” on page 4-30	2
“MMSB Control Register (Index 74h)” on page 4-39	3
“Control B Register (Index 77h)” on page 4-42	3
“Power Control 1 Register (Index 80h)” on page 4-42	7
“Power Control 2 Register (Index 81h)” on page 4-43	7 and 3
“Clock Control Register (Index 8Fh)” on page 4-47	2–0
“UART Clock Enable Register (Index 92h)” on page 4-49	0
“Auto Low-Speed Control Register (Index 9Fh)” on page 4-51	3–0
“PMU Control 3 Register (Index ADh)” on page 4-60	3–0
“PMU Control 2 Register (Index AFh)” on page 4-61	5 and 0
“Function Enable 1 Register (Index B0h)” on page 4-61	6 and 3
“Function Enable 2 Register (Index B1h)” on page 4-62	4–2
“Miscellaneous 3 Register (Index BAh)” on page 4-70	3

4.1.3.13 To Control Power Management Activities and Events

See	And program bits
“Miscellaneous 4 Register (Index 44h)” on page 4-15	0
“PIO Address Register (Index 45h)” on page 4-15	7–0
“PIO Timer Register (Index 46h)” on page 4-16	7–6 and 3–0
“Drive Timer Register (Index 47h)” on page 4-17	7–0
“Miscellaneous 6 Register (Index 70h)” on page 4-35	5
“MMSB Control Register (Index 74h)” on page 4-39	7–6 and 3
“Activity Mask 1 Register (Index 75h)” on page 4-40	7–0
“Activity Mask 2 Register (Index 76h)” on page 4-41	7–0
“Power Control 1 Register (Index 80h)” on page 4-42	7–6 and 2
“Power Control 2 Register (Index 81h)” on page 4-43	7–6 and 3–2
“Software Mode Control Register (Index 88h)” on page 4-45	2–0
“I/O Activity Address 0 Register (Index 8Ch)” on page 4-46	7–0
“I/O Activity Address 1 Register (Index 8Dh)” on page 4-46	7–0
“Memory Write Activity Lower Boundary Register (Index 9Ah)” on page 4-50	7–2 and 0
“Memory Write Activity Upper Boundary Register (Index 9Bh)” on page 4-50	7–4 and 2–0
“Power Control 3 Register (Index ABh)” on page 4-58	7–0
“Power Control 4 Register (Index ACh)” on page 4-59	7–0
“PMU Control 3 Register (Index ADh)” on page 4-60	4
“PMU Control 2 Register (Index AFh)” on page 4-61	0

4.1.3.14 To Determine Power Management Status

See	And program bits
“Resume Status Register (Index 09h)” on page 4-12	5–0
“Activity Status 1 Register (Index A0h)” on page 4-53	7–0
“Activity Status 2 Register (Index A1h)” on page 4-53	7–0
“PMU Status 1 Register (Index A2h)” on page 4-54	7
“CPU Status 0 Register (Index A3h)” on page 4-54	6–1
“CPU Status 1 Register (Index A4h)” on page 4-55	7 and 2–0
“NMI/SMI Control Register (Index A5h)” on page 4-56	7–0
“Miscellaneous 5 Register (Index B3h)” on page 4-65	1–0

4.1.3.15 To Control the Power Management State Timers

See	And program bits
“High-Speed to Low-Speed Timer Register (Index 83h)” on page 4-44	7–0
“Low-Speed to Doze Timer Register (Index 84h)” on page 4-44	7–0
“Doze to Sleep Timer Register (Index 85h)” on page 4-44	7–0
“Sleep to Suspend Timer Register (Index 86h)” on page 4-44	7–0
“Suspend to Off Timer Register (Index 87h)” on page 4-45	7–0
“PMU Control 2 Register (Index AFh)” on page 4-61	7–6

4.1.3.16 To Map ROM Accesses and Control ROM Cycles

See	And program bits
“MMS Memory Wait State 2 Register (Index 50h)” on page 4-19	6 and 2–0
“ROM Configuration 2 Register (Index 51h)” on page 4-20	1–0
“Command Delay Register (Index 60h)” on page 4-20	7 and 4
“MMS Memory Wait State 1 Register (Index 62h)” on page 4-23	6–5
“ROM Configuration 1 Register (Index 65h)” on page 4-27	3–0
“Miscellaneous 5 Register (Index B3h)” on page 4-65	6–4 and 2
“ROM Configuration 3 Register (Index B8h)” on page 4-67	7–0

4.1.3.17 To Control SMIs and Determine Status

See	And program bits
“I/O Timeout Register (Index 40h)” on page 4-12	2–0
“SMI Enable Register (Index 41h)” on page 4-13	4–0
“SMI I/O Status Register (Index 42h)” on page 4-14	3–0
“SMI Status Register (Index 43h)” on page 4-14	6–0
“Wait State Control Register (Index 63h)” on page 4-25	7
“Version Register (Index 64h)” on page 4-26	7
“Miscellaneous 2 Register (Index 6Bh)” on page 4-30	0
“NMI/SMI Enable Register (Index 82h)” on page 4-43	7–0
“PMU Status 1 Register (Index A2h)” on page 4-54	7–0
“NMI/SMI Control Register (Index A5h)” on page 4-56	7–0
“SMI MMS Upper Page Register (Index A9h)” on page 4-57	7–4
“SMI MMS Page Register (Index AAh)” on page 4-57	7–0
“PMU Control 3 Register (Index ADh)” on page 4-60	6
“Function Enable 1 Register (Index B0h)” on page 4-61	5–4
“Miscellaneous 5 Register (Index B3h)” on page 4-65	1–0

4.2 CONFIGURATION PORT REGISTERS

4.2.1 Configuration Address Register (Port 022h)

This register contains the index of the configuration index register being programmed.

	7							0
Field	Index of Configuration Index Register							
Default	0	0	0	0	0	0	0	0

4.2.2 Configuration Data Register (Port 023h)

This register is used to read data from, or write data to, the register pointed to by the Configuration Address register.

	7							0
Field	Data Transferred to/from Configuration Index Register							
Default	0	0	0	0	0	0	0	0

4.3 CONFIGURATION INDEX REGISTERS

These registers are listed in Index numerical order. See Appendix A for an alphabetical list by register name.

4.3.1 Reserved Registers (Index 00-07h)

These index locations are reserved.

4.3.2 Resume Mask Register (Index 08h)

When set, bits 5–2 of this register mask the corresponding function.

	7							0
Bit	(Reserved)		RIMSK	IRQ8MSK	IRQ4MSK	IRQ3MSK	(Reserved)	
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7-6		R/W	(Reserved)
5	RIMSK	R/W	If this bit is 0, a ring-in input can wake up the system.
4	IRQ8MSK	R/W	If this bit is 0, IRQ8 can wake up the system.
3	IRQ4MSK	R/W	If this bit is 0, IRQ4 can wake up the system.
2	IRQ3MSK	R/W	If this bit is 0, IRQ3 can wake up the system.
1-0		R/W	(Reserved)

Bit 5 Clearing this bit allows the internal UART Ring Indicate to wake up the system.

Bit 4 Clearing this bit allows the internal RTC to wake up the system.

Bit 3 Clearing this bit allows a rising edge on IRQ4 (whether internally or externally generated) to wake up the system. IRQ4 should be held at 1 until the wake-up is complete.

Bit 2 Clearing this bit allows a rising edge on IRQ3 (whether internally or externally generated) to wake up the system. IRQ3 should be held at 1 until the wake-up is complete.

4.3.3 Resume Status Register (Index 09h)

A 1 in any bit in this register indicates that the corresponding function caused the system to wake up. 00h must be written to this register to clear it.

	7							0
Bit	(Reserved)		PRIM_RI	IRQ8	IRQ4	IRQ3	(Reserved)	
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7–6		R/W	(Reserved)
5	PRIM_RI	R/W	System was awakened by a Ring-In from the internal UART
4	IRQ8	R/W	System was awakened by IRQ8
3	IRQ4	R/W	System was awakened by IRQ4
2	IRQ3	R/W	System was awakened by IRQ3
1-0		R/W	(Reserved)

4.3.4 Reserved Registers (Indexes 0A-0Eh)

These index locations are reserved.

4.3.5 Reserved Register (Index 0Fh)

This index location is reserved and must be initialized to FFh at boot time.

4.3.6 Reserved Registers (Indexes 10–39h)

These index locations are reserved.

4.3.7 I/O Timeout Register (Index 40h)

This register may be used with the SMI Enable register at Index 41h to determine if an I/O-device access generates an SMI. If a bit in this register is 0 and the corresponding bit in the SMI Enable register is 1, the next I/O access to that device causes an SMI.

In addition, writing a 0 to a bit enables an SMI to occur on the next I/O access to that device. Writing a 1 has no effect. For more information, see “Accesses to Powered-Down Device SMI” on page 1-29.

	7							0
Bit	(Reserved)					PIOTOLTCH	FDTOLTCH	HDTOLTCH
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7–3		R/W	(Reserved)
2	PIOTOLTCH	R/W	1 = PIO access caused the SMI; PIO SMIs are masked until this bit is 0
1	FDTOLTCH	R/W	1 = Floppy disk drive access caused the SMI; floppy-disk-drive SMIs are masked until this bit is 0
0	HDTOLTCH	R/W	1 = Hard disk drive access caused the SMI; hard-disk-drive SMIs are masked until this bit is 0

4.3.8 SMI Enable Register (Index 41h)

This register is used to control the generation of individual SMIs upon access to a specific I/O location. A value of 1 in each bit enables an SMI to be generated when the corresponding device is accessed. A value of 0 disables SMI generation for the corresponding device. For bits 2–0, an SMI is generated on the first access to the I/O address after one of the following events occur:

- The SMI is enabled
- The timer expires
- The corresponding bit in the I/O Timeout Register at Index 40h is 0
- The PMU enters Sleep or Suspend mode

I/O accesses that occur before the timer expires cause the timer to be reloaded. The hard-disk-drive and floppy-disk-drive timers are also reloaded when bit 0 of the Miscellaneous 4 register at Index 44h is 1 and IRQ14 (hard disk drive) or IRQ6 (floppy disk drive) is asserted. For more information, see “Accesses to Powered-Down Device SMI” on page 1-29.

	7							0
Bit	(Reserved)			RTCSMIEN	KBSMIEN	ENPIO	ENFD	ENHD
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7–5		R/W	(Reserved)
4	RTCSMIEN	R/W	1 = Enable SMI generation on access to I/O address 07xh
3	KBSMIEN	R/W	1 = Enable SMI generation on 8042 access
2	ENPIO	R/W	1 = Enable SMI generation on PIO access
1	ENFD	R/W	1 = Enable SMI generation on floppy-disk-drive (3F0–3F7h) access
0	ENHD	R/W	1 = Enable SMI generation on hard-disk-drive (1F0–1F7h) access

4.3.9 SMI I/O Status Register (Index 42h)

This register contains the states of the CPU bus when an SMI is generated. Software can read this register to determine the type of bus cycle executed. This register is updated when an I/O access generates an SMI that corresponds to bits 4–0 of the SMI Enable register at Index 41h or an EXTSMI pin event.

	7							0
Bit	(Reserved)				IOW	IOR	BHE	A0
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7–4			(Reserved)
3	IOW	R	State of $\overline{IOW}$ when SMI was generated: 1 = $\overline{IOW}$ was active
2	IOR	R	State of $\overline{IOR}$ when SMI was generated: 1 = $\overline{IOR}$ was active
1	BHE	R	State of $\overline{BHE}$ when SMI was generated: 0 = $\overline{BHE}$ was active
0	A0	R	State of address bit 0 when SMI was generated

4.3.10 SMI Status Register (Index 43h)

This register contains the status of SMI sources. A 1 in any of these bits indicates that the corresponding device generated an SMI. The programmer may clear this register by writing to it; the data written is irrelevant. For information on the SMI enables that correspond to the bits in this register, see “SMI Enable Register (Index 41h)” on page 4-13 and “PMU Control 3 Register (Index ADh)” on page 4-60.

	7							0
Bit	(Reserved)	IRQ0SMI	PMCSMI	RTCSMIEN	KBSMIEN	PIOSMI	FDSMI	HDSMI
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7		R/W	(Reserved, must be 0)
6	IRQ0SMI	R/W	1 = IRQ0 requested an SMI
5	PMCSMI	R/W	1 = PMU mode change, $\overline{BL}$ event, or $\overline{SUS}$/$\overline{RES}$ event requested an SMI
4	RTCSMIEN	R/W	1 = SMI generation on access to address 07xh
3	KBSMIEN	R/W	1 = SMI generation on 8042 access
2	PIOSMI	R/W	1 = PIO requested an SMI
1	FDSMI	R/W	1 = Floppy disk drive requested an SMI
0	HDSMI	R/W	1 = Hard disk drive requested an SMI

4.3.11 Miscellaneous 4 Register (Index 44h)

This register is used to clear the hard-disk-drive timer and control data propagation through the ÉlanSC310 microcontroller.

	7							0
Bit	(Reserved)			DISDEN	(Reserved)			IRQEN
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7		W	(Reserved—must be 1)
6		W	(Reserved—must be 0)
5		W	(Reserved)
4	DISDEN	W	1 = Data read from DRAM via a CPU cycle is propagated only through the ÉlanSC310 microcontroller's CPU core 0 = The DRAM data is propagated to all internal ÉlanSC310 microcontroller cores
3–1		W	(Reserved—must be 0)
0	IRQEN	W	If 1, IRQ14 (hard disk drive) or IRQ6 (floppy disk drive) causes the hard-disk-drive timer or floppy-disk-drive timer to reload (see Index 47h)

4.3.12 PIO Address Register (Index 45h)

This register is used to program PIO address bits A9–A2. The PIO base address can be anywhere in the range 000–3FCh and should be programmed on an address boundary that corresponds to the address-decode range specified in bits 6 and 7 of the PIO Timer register at Index 46h.

	7							0
Field	I/O Address Bits 9–2							
Bit	A9	A8	A7	A6	A5	A4	A3	A2
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7	A9	R/W	PIO address bit A9
6	A8	R/W	PIO address bit A8
5	A7	R/W	PIO address bit A7
4	A6	R/W	PIO address bit A6
3	A5	R/W	PIO address bit A5
2	A4	R/W	PIO address bit A4
1	A3	R/W	PIO address bit A3
0	A2	R/W	PIO address bit A2

4.3.13 PIO Timer Register (Index 46h)

This register is used to program the PIO time-out period. If no accesses are made to the I/O range specified by the PIO Address register at Index 45h in the time-out period specified here, PMC1 is driven Low. This line may be used to control power to an external device. Bits 6 and 7 of this register may be used to mask bits 2–0 of the PIO Address register. For more information, see “SMI Enable Register (Index 41h)” on page 4-13.

7							0
Field	Decode Range			Timer Setting			
Bit	RA1	RA0	(Reserved)	PIOT3	PIOT2	PIOT1	PIOT0
Default	0	0	0 0	0	0	0	0

Bit	Name	R/W	Function
7	RA1	R/W	PIO decode range bit 1
6	RA0	R/W	PIO decode range bit 0
5–4		R/W	(Reserved)
3	PIOT3	R/W	PIO timer setting bit 3
2	PIOT2	R/W	PIO timer setting bit 2
1	PIOT1	R/W	PIO timer setting bit 1
0	PIOT0	R/W	PIO timer setting bit 0

Table 4-4 PIO Timer Setting Logic

PIOT3	PIOT2	PIOT1	PIOT0	Period
0	0	0	0	128 ms
0	0	0	1	256 ms
0	0	1	0	512 ms
0	0	1	1	1 s
0	1	0	0	2 s
0	1	0	1	4 s
0	1	1	0	8 s
0	1	1	1	16 s
1	0	0	0	32 s
1	0	0	1	64 s
1	0	1	0	s 128
1	0	1	1	s 256
1	1	0	0	s 512
1	1	0	1	1024 s
1	1	1	0	2048 s
1	1	1	1	4096 s

Table 4-5 PIO Address Range Decode Logic

RA1	RA0	PIO Address Range
0	0	4 bytes
0	1	8 bytes
1	0	16 bytes
1	1	32 bytes

4.3.14 Drive Timer Register (Index 47h)

This register is used to program the hard-disk-drive and floppy-disk-drive time-out periods. If no accesses are made to the hard-disk drive in the time-out period specified here, PMC4 is driven Low. If no accesses are made to the floppy disk drive in the time-out period specified here, PMC0 is driven Low. These lines may be used to control power to those drives. For more details concerning the use of this register, see “SMI Enable Register (Index 41h)” on page 4-13.

	7							0
Field	Floppy-Disk-Drive Timer Setting				Hard-Disk-Drive Timer Setting			
Bit	FDT3	FDT2	FDT1	FDT0	HDT3	HDT2	HDT1	HDT0
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7	FDT3	W	Floppy-disk-drive timer setting bit 3
6	FDT2	W	Floppy-disk-drive timer setting bit 2
5	FDT1	W	Floppy-disk-drive timer setting bit 1
4	FDT0	W	Floppy-disk-drive timer setting bit 0
3	HDT3	W	Hard-disk-drive timer setting bit 3
2	HDT2	W	Hard-disk-drive timer setting bit 2
1	HDT1	W	Hard-disk-drive timer setting bit 1
0	HDT0	W	Hard-disk-drive timer setting bit 0

Table 4-6 Hard-Disk-Drive and Floppy-Disk-Drive Timer Setting Bit Logic

HDT3 or FDT3	HDT2 or FDT2	HDT1 or FDT1	HDT0 or FDT0	Period
0	0	0	0	128 ms
0	0	0	1	256 ms
0	0	1	0	512 ms
0	0	1	1	1 s
0	1	0	0	2 s
0	1	0	1	4 s
0	1	1	0	8 s
0	1	1	1	16 s
1	0	0	0	32 s
1	0	0	1	64 s
1	0	1	0	128 s
1	0	1	1	256 s
1	1	0	0	512 s
1	1	0	1	1024 s
1	1	1	0	2048 s
1	1	1	1	4096 s

4.3.15 Control A Register (Index 48h)

This register contains miscellaneous control functions. Bit 1 of this register converts the UART Control register at Port 3FDh from read only (the default) to read/write, which is compatible with the 16450. Bit 2 causes the MEMR and MEMW signals to be disabled during on-board memory cycles.

	7							0
Bit	(Reserved)					DISCMD	LSRWCNTL	(Reserved)
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7–3		W	(Reserved)
2	DISCMD	W	1 = Disable $\overline{\text{MEMR}}$ and $\overline{\text{MEMW}}$ signals on the bus during on-board memory cycles
1	LSRWCNTL	W	1 = Enable UART Line Status register (Index 3FDh) for writes (retains 16450 compatibility)
0		W	(Reserved)

4.3.16 Reserved Registers (Indexes 49–4Fh)

These index locations are reserved.

4.3.17 MMS Memory Wait State 2 Register (Index 50h)

This register is used to specify the command delay and the number of wait states used with 8-bit accesses to ROM DOS. The reference clock is the internal version of SYSCLK. For more information, see “Wait States and Command Delays” on page 2-14.

	7							0
Field							RDOSWS	
Bit	(Reserved)	RDOSCMDL	(Reserved)			RDOSWSEN	RDOSWS1	RDOSWS0
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7		W	Reserved
6	RDOSCMDL	W	ROMDOS (ROM accessed by $\overline{\text{DOSCS}}$) command delay
5-3		W	Reserved
2	RDOSWSEN	W	8-bit ROM DOS (ROM accessed by $\overline{\text{DOSCS}}$) wait state enable
1	RDOSWS1	W	8-bit ROM DOS (ROM accessed by $\overline{\text{DOSCS}}$) wait states, bit 1
0	RDOSWS0	W	8-bit ROM DOS (ROM accessed by $\overline{\text{DOSCS}}$) wait states, bit 0

Table 4-7 on page 4-19 and Table 4-8 on page 4-19 describe the wait states and command delays for 8-bit ROM DOS accesses (ROM accessed via $\overline{\text{DOSCS}}$).

Table 4-7 ROM-DOS Wait-State Select Logic

ENFSTRDOS Bit 7 of Index B8h	RDOSWSEN	RDOSWS1	RDOSWS0	ROM Wait States
0	1	0	0	4
0	1	0	1	3
0	1	1	0	2
0	1	1	1	1
0	0	x	x	Controlled by bits 1–0 in Index 62h
1	x	x	x	Controlled by bits 6–5 in Index B8h

Table 4-8 ROM-DOS Command-Delay Select Logic

RDOSWSEN Bit 2 of Index 50h	RDOSCMDL Bit 6 of Index 50h	8MCD Bit 2 of Index 60h	ROM-DOS Memory-Cycle Command Delay
0	x	0	1 SYSCLK cycle
0	x	1	0.5 SYSCLK cycle
1	1	x	0 delay

4.3.18 ROM Configuration 2 Register (Index 51h)

This register is used to enable ROM BIOS accesses and to set the ROM-DOS accesses as 16-bit transfers. For more information on configuration bits for ROM-BIOS accesses, see “ROM Configuration 1 Register (Index 65h)” on page 4-27.

	7							0
Bit	(Reserved)						ROMDOS16	ENROMA
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7–2		W	(Reserved—must be 0)
1	ROMDOS16	W	1 = Select 16-bit ROM-DOS configuration
0	ENROMA	W	1 = $\overline{\text{ROMCS}}$ is active when the address is within 0A0000–0AFFFFh

4.3.19 Reserved Registers (Indexes 52–5Fh)

These index locations are reserved.

4.3.20 Command Delay Register (Index 60h)

This register is used to select different command delays for ISA-bus I/O cycles and wait states for both MMS and non-MMS memory cycles.

	7							0
Field							IOCD	
Bit	ROMWS1	REFWS	(Reserved)	ROMWS0	(Reserved)	8MCD	IOCD1	IOCD0
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7	ROMWS1	W	ROM-BIOS, non-MMS-cycle wait states, bit 1
6	REFWS	W	Refresh wait states. Must be 0 for 33-MHz operation.
5		W	(Reserved—must be 0)
4	ROMWS0	W	ROM-BIOS, non-MMS-cycle wait states, bit 0
3		W	(Reserved—must be 0)
2	8MCD	W	8-bit ISA memory cycle command delay
1	IOCD1	W	8-bit ISA I/O cycle command delay, bit 1
0	IOCD0	W	8-bit ISA I/O cycle command delay, bit 0

Table 4-9 8-Bit ISA I/O Access Command Delay

IOCD1	IOCD0	SYSCLK Cycles Delayed
0	0	1 (default)
0	1	2
1	x	0.5

Note:

Acts on addresses from 100h to 3FFh.

Table 4-10 8-Bit ISA Memory Access Command Delay

8MCD	SYSCLK Cycles Delayed
0	1 (default)
1	0.5

Table 4-11 ROM-BIOS Wait States

ROMWS1	ROMWS0	Number of Wait Cycles
1	0	3
0	1	2
1	1	(Invalid combination)
0	0	3 (default)

Note:

These bits control the number of wait cycles for ROM-BIOS accesses only during non-MMS cycles. During MMS cycles to ROM BIOS, the number of wait cycles is controlled by the settings for ISA memory cycles found in the MMS Memory Wait State Select 1 register, Index 62h.

Table 4-12 Refresh-Cycle Wait States

REFWS	SYSCLK Cycles in Refresh Delay
1	2
0	3 (default)

Note:

For 33-MHz operation, this bit must be 0.

4.3.21 I/O Wait State Register (Index 61h)

This register defines the number of wait states for I/O cycles to different I/O addresses. It also contains the control for forcing the CPU clock to run at 9.2 MHz from the low-speed PLL during High-Speed PLL mode.

	7							0
Field			IOWS		HDWS		FDWS	
Bit	DMAMMS	SPEED	IOWS1	IOWS0	HDWS1	HDWS0	FDWS1	FDWS0
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7	DMAMMS	W	0 = Disable MMS during DMA 1 = Enable MMS in DMA cycle
6	SPEED	W	CPU clock speed select: 0 = Force all operations to be low speed (9.2 MHz) 1 = High speed
5	IOWS1	W	Other bus I/O wait states, bit 1
4	IOWS0	W	Other bus I/O wait states, bit 0
3	HDWS1	W	Hard-disk-drive wait states, bit 1
2	HDWS0	W	Hard-disk-drive wait states, bit 0
1	FDWS1	W	Floppy-disk-drive wait states, bit 1
0	FDWS0	W	Floppy-disk-drive wait states, bit 0

The number of wait states selected in this register must be greater than the bus I/O command delay specified in the Command Delay register at Index 60h.

After reset, the CPU clock runs at low speed (9.2 MHz). Write a 1 to bit 6 of this register to enable the CPU to run at high speed (the speed set by bits 4–3 of the Function Enable 2 register at Index B1h). The CPU only runs in high speed during DRAM accesses, local bus accesses, fast ROM accesses, and while idle. ISA accesses cause the CPU clock to switch to 9.2 MHz, thus saving power.

Table 4-13 Floppy-Disk-Drive Wait States

FDWS1	FDWS0	SYSCLK Cycles Delayed
0	0	5 (default)
0	1	4
1	0	3
1	1	2

Note:
Acts on addresses from 3F0–3F7h.

Table 4-14 Hard-Disk-Drive Wait States

HDWS1	HDWS0	SYSCLK Cycles Delayed
0	0	5 (default)
0	1	4
1	0	3
1	1	2

Note:
Acts on addresses from 1F0–1F7h.

Table 4-15 General Bus I/O Wait States

IOWS1	IOWS0	SYSCLK Cycles Delayed
0	0	5 (default)
0	1	4
1	0	3
1	1	2

Note:
Acts on addresses from 100h to 3FFh, except 3F0–3F7h, 1F0–1F7h, and the MMS I/O base address.

4.3.22 MMS Memory Wait State 1 Register (Index 62h)

This register defines the wait states for different memory accesses.

	7							0
Field					16BMWS		8BMWS	
Bit	(Reserved)	NFRDOSEN	NFROMEN	MISOUT	16BMWS1	16BMWS0	8BMWS1	8BMWS0
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7		W	(Reserved—must be 0)
6	NFRDOSEN	W	1 = Enable writes to devices selected by $\overline{\text{DOSCS}}$
5	NFROMEN	W	1 = Enable writes to devices selected by $\overline{\text{ROMCS}}$ pin
4	MISOUT	W	Page mode bank miss and time-out wait state select (see Table 4-19 on page 4-25): 0 = 3 wait states 1 = 5 wait states This bit must be 1 for 33-MHz operation.
3	16BMWS1	W	Wait states bit 1 for 16-bit ISA memory cycles
2	16BMWS0	W	Wait states bit 0 for 16-bit ISA memory cycles
1	8BMWS1	W	Wait states bit 1 for 8-bit ISA memory cycles
0	8BMWS0	W	Wait states bit 0 for 8-bit ISA memory cycles

Table 4-16 on page 4-24 describes 8-bit ISA memory-cycle wait states. These bits also apply for any 8-bit cycles to ROM BIOS addressed through the MMS map where $\overline{\text{MEMCS16}}$ is not asserted.

Table 4-16 8-Bit ISA Memory-Cycle Wait States

8BMWS1	8BMWS0	SYSCLK Cycles Delayed
0	0	(default) 5
0	1	4
1	0	3
1	1	2

Table 4-17 on page 4-24 describes 16-bit ISA memory-cycle wait states. The following applies for cycles when either $\overline{\text{MEMCS16}}$ is asserted or ROM DOS is set up for 16-bit cycles.

Table 4-17 16-Bit ISA Memory-Cycle Wait States

16BMWS1	16BMWS0	SYSCLK Cycles Delayed
0	0	(default) 4
0	1	3
1	0	2
1	1	1

4.3.23 Wait State Control Register (Index 63h)

This register defines the wait states for miscellaneous accesses.

	7							0
Bit	SMMSIZE	BKMISS	FCYCWAIT1	FCYCWAIT0	INTIOWAIT	16IOWAIT	(Reserved)	SHUTD
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7	SMMSIZE	W	SMM memory range select: 0 = 16 Kbyte 1 = 64 Kbyte
6	BKMISS	W	Set DRAM bank-miss wait state in Page mode
5	FCYCWAIT1	W	Set DRAM first-cycle wait state in Page mode
4	FCYCWAIT0	W	Set DRAM first-cycle wait state in Page mode. See Table 4-18 on page 4-25.
3	INTIOWAIT	W	This bit controls the number of wait states for I/O addresses to internal cores: 0 = 4 wait states 1 = 2 wait states
2	16IOWAIT	W	Wait state select for 16-bit I/O cycles 0 = 4 wait cycles 1 = 3 wait cycles
1		W	(Reserved—must be 0)
0	SHUTD	W	Wait state select for shutdown cycle: 0 = 16 cycles 1 = 32 cycles

Note:

Bits 5 and 6 of this register must be set for 33-MHz operation. See also bit 5 of Index 65h.

Bit 7 This bit controls the size of the MMS page that is mapped to address 60000h when an SMI occurs. Refer to “SMI MMS Page Register (Index AAh)” on page 4-57.

Table 4-18 DRAM First Cycle Wait State Select Logic

PFWS Bit 5 of Index 65h	FCYCWAIT1 Bit 5	FCYCWAIT0 Bit 4	DRAM First-Cycle Wait States in Page Mode
0	x	0	(default) 1
0	x	1	2
1	0	x	2
1	1	x	(33 MHz) 3

Table 4-19 DRAM Bank Miss Wait State Select Logic

BKMISS Bit 6	MISOUT Bit 4 of Index 62h	DRAM Bank-Miss Wait States in Page Mode
x	0	(default) 3
0	1	4
1	1	(33 MHz) 5

4.3.24 Version Register (Index 64h)

This register displays whether SMI is active. In addition, bits 2–0 of this register can be read to determine the major stepping level information (processor version). Bits 6–3 hold the minor stepping (additional revision) information. Writing to this register selects the refresh rate for DRAM and enables Enhanced Page Mode when using 512-Kbit × 8-bit DRAMs.

4.3.24.1 Read Functions

	7							0
Field		Minor Level				Major Level		
Bit	RSMI							
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7	RSMI	R	SMI active
6–3		R	Minor step level (revision)
2–0		R	Major step level (revision)

Table 4-20 Read Version Stepping Level Decode

Revision	Minor Level Bits 6 5 4 3	Major Level Bits 2 1 0
A	0 0 0 0	0 0 1
B0–B2	0 0 0 0	0 1 0
B3	0 0 0 1	0 1 0

4.3.24.2 Write Functions

	7							0
Bit	(Reserved)			EPMODE	(Reserved)		REFSEL1	REFSEL0
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7		W	(Reserved—must be 1)
6–5		W	(Reserved—must be 0)
4	EPMODE	W	Enable Enhanced Page mode when the value of bits 4–2 of Index 66h is 010b and bit 7 of Index B4h is 0. Bit 0 of Index 66h must also be 1.
3–2		W	(Reserved—must be 1)
1	REFSEL1	W	Refresh interval select, bit 1
0	REFSEL0	W	Refresh interval select, bit 0

Bits 1–0 These bits are not reset when exiting Micro Power Off mode.

The refresh interval shown in Table 4-21 on page 4-27 is selected only if bits 1–0 of the PMU Control 1 register at Index A7h are both 0.

Table 4-21 Refresh Interval Select Logic

REFSEL1	REFSEL0	Refresh Interval (cycles per second)
0	0	8192
0	1	10922
1	0	16384
1	1	32768

4.3.25 ROM Configuration 1 Register (Index 65h)

This register is used to configure ROM accesses. Bits 3–0 are used to set the size of the BIOS ROM that is connected to the $\overline{\text{ROMCS}}$ signal. When each bit is 1, the $\overline{\text{ROMCS}}$ signal goes active during accesses within the corresponding address range. $\overline{\text{ROMCS}}$ must be disabled for regions that are shadowed. Bit 4 enables shadow RAM and bit 7 is the shadow RAM write protect. This bit must be 1 to allow writes to the shadow RAM.

	7							0
Bit	DISW	ENMMSA	PFWS	SHADOW	ENROMC	ENROMD	ENROME	ENROMF
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7	DISW	R/W	0 = Shadow RAM write protect
6	ENMMSA	R/W	1 = MMSA enable
5	PFWS	R/W	Page mode first-cycle wait state select (see Index 63h, bit 5, and Table 4-18 on page 4-25)
4	SHADOW	R/W	1 = Shadow RAM enable
3	ENROMC	R/W	1 = $\overline{\text{ROMCS}}$ is active when address is within range 0C0000–0CFFFFh.
2	ENROMD	R/W	1 = $\overline{\text{ROMCS}}$ is active when address is within range 0D0000–0DFFFFh.
1	ENROME	R/W	1 = $\overline{\text{ROMCS}}$ is active when address is within range 0E0000–0EFFFFh.
0	ENROMF	R/W	0 = $\overline{\text{ROMCS}}$ is active when address is within range 0F0000–0FFFFFh. This bit reads back the inverse of what was written to it. Writing a 0 enables access.

Bit 6 This bit enables/disables all windows in MMSA. If this bit is 0, all windows are disabled. If this bit is 1, each window can be individually enabled/disabled via bit 7 of the appropriate Page register.

Bit 5 This bit must be 1 for 33-MHz operation.

Bit 0 Read-modify-write operations must invert bit 0 if bit 0 is not to be changed by the operation.

4.3.26 Memory Configuration 1 Register (Index 66h)

This register controls memory size and operating mode. Bus configuration is determined by the state of the $\overline{DTR}$ and $\overline{RTS}$ pins at reset. Software may read the latched state of SOUT as it was sampled at reset. SOUT may be used as a general-purpose latched input.

	7							0
Field				MS				
Bit	SOUTL	CFG0	CFG1	MS2	MS1	MS0	(Reserved)	MOD0
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7	SOUTL	R	Latched state of SOUT at reset
6	CFG0	R	Latched state of $\overline{RTS}$ at reset
5	CFG1	R	Latched state of $\overline{DTR}$ at reset
4	MS2	R/W	Memory bank configuration, bit 2
3	MS1	R/W	Memory bank configuration, bit 1
2	MS0	R/W	Memory bank configuration, bit 0
1		R/W	(Reserved—must be 1)
0	MOD0	R/W	DRAM Enhanced Page mode is enabled when this bit is set (see Index 64h, bit 4)

Table 4-22 Bus Option Status Table

Bus Selected	CFG1	CFG0
Reserved	0	x
2x Clock Local Bus	1	0
Maximum ISA	1	1

Table 4-23 Memory Configuration (DRAM)

Bit 7 of Index B4h	MS2	MS1	MS0	Total Memory	Bank 0	Bank 1
0	0	0	1	1 Mbyte	1 Mbyte	–
0	0	1	0	2 Mbyte	1 Mbyte	1 Mbyte
0	0	1	1	2 Mbyte	2 Mbyte	–
0	1	0	0	4 Mbyte	2 Mbyte	2 Mbyte
0	1	0	1	8 Mbyte	8 Mbyte	–
0	1	1	0	16 Mbyte	8 Mbyte	8 Mbyte

Note:

See Indexes B4h and B1h for additional DRAM configurations.

4.3.27 MMSA Address Extension 1 Register (Index 67h)

This is the MMSA register containing address-extension bits 21–22 for pages 4–7. Bit 1 of the MMSB Control register at Index 74h must be 1 prior to writing this register.

	7							0
Field	Page 7		Page 6		Page 5		Page 4	
Bit	E7A22	E7A21	E6A22	E6A21	E5A22	E5A21	E4A22	E4A21
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7	E7A22	R/W	MMSA page 7 address extension bit 22
6	E7A21	R/W	MMSA page 7 address extension bit 21
5	E6A22	R/W	MMSA page 6 address extension bit 22
4	E6A21	R/W	MMSA page 6 address extension bit 21
3	E5A22	R/W	MMSA page 5 address extension bit 22
2	E5A21	R/W	MMSA page 5 address extension bit 21
1	E4A22	R/W	MMSA page 4 address extension bit 22
0	E4A21	R/W	MMSA page 4 address extension bit 21

4.3.28 Shadow RAM Enable 1 Register (Index 68h)

This register controls the shadow-RAM mapping range. Mapping is disabled by default.

	7							0
Bit	SDCF	SD8B	SD47	SD03	SCCF	SC8B	SC47	SC03
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7	SDCF	R/W	Enable shadow RAM at range 0DC000–0DFFFFh
6	SD8B	R/W	Enable shadow RAM at range 0D8000–0DBFFFh
5	SD47	R/W	Enable shadow RAM at range 0D4000–0D7FFFh
4	SD03	R/W	Enable shadow RAM at range 0D0000–0D3FFFh
3	SCCF	R/W	Enable shadow RAM at range 0CC000–0CFFFFh
2	SC8B	R/W	Enable shadow RAM at range 0C8000–0CBFFFh
1	SC47	R/W	Enable shadow RAM at range 0C4000–0C7FFFh
0	SC03	R/W	Enable shadow RAM at range 0C0000–0C3FFFh

4.3.29 Shadow RAM Enable 2 Register (Index 69h)

This register controls the shadow-RAM mapping range. Mapping is disabled by default.

	7							0
Bit	SFCF	SF8B	SF47	SF03	SECF	SE8B	SE47	SE03
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7	SFCF	R/W	Enable shadow RAM at range 0FC000–0FFFFFh
6	SF8B	R/W	Enable shadow RAM at range 0F8000–0FBFFFh
5	SF47	R/W	Enable shadow RAM at range 0F4000–0F4000h
4	SF03	R/W	Enable shadow RAM at range 0F0000–0F3FFFh
3	SECF	R/W	Enable shadow RAM at range 0EC000–0EFFFFh
2	SE8B	R/W	Enable shadow RAM at range 0E8000–0EBFFFh
1	SE47	R/W	Enable shadow RAM at range 0E4000–0E7FFFh
0	SE03	R/W	Enable shadow RAM at range 0E0000–0E3FFFh

4.3.30 Reserved Register (Index 6Ah)

This register is reserved and must be 00h.

4.3.31 Miscellaneous 2 Register (Index 6Bh)

This register contains several miscellaneous control bits.

	7							0
Bit	(Reserved)					CPU_IDLE	(Reserved)	A20SMI
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7		W	(Reserved—must be 0)
6–5		R	(Reserved)
4		R/W	(Reserved—must be 1)
3		R	(Reserved)
2	CPU_IDLE	W	Selects whether CPU idle cycles are dynamically switched to use the high-speed clock frequency: 0 = High speed 1 = Low speed (9.2 MHz) This bit must be 0 for 33-MHz and 25-MHz operation.
1			(Reserved)
0	A20SMI	W	Controls gate A20 during SMI access: 1 = A20 propagates 0 = A20 Low

4.3.32 MMS Address Extension 1 Register (Index 6Ch)

This is the MMS register that contains address-extension bit 23 for all pages.

	7							0
Bit	E7A23	E6A23	E5A23	E4A23	E3A23	E2A23	E1A23	E0A23
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7	E7A23	R/W	MMSA page 7 address extension bit 23
6	E6A23	R/W	MMSA page 6 address extension bit 23
5	E5A23	R/W	MMSA page 5 address extension bit 23
4	E4A23	R/W	MMSA page 4 address extension bit 23
3	E3A23	R/W	MMSA/MMSB page 3 address extension bit 23
2	E2A23	R/W	MMSA/MMSB page 2 address extension bit 23
1	E1A23	R/W	MMSA/MMSB page 1 address extension bit 23
0	E0A23	R/W	MMSA/MMSB page 0 address extension bit 23

Bits 3–0 Provide mapped address-extension bit 23 for both MMSA and MMSB (pages 0–3). Before programming this register, software must select the region to program (i.e., MMSA or MMSB). This selection is performed via bit 1 of the MMSB Control register at Index 74h.

4.3.33 MMS Address Register (Index 6Dh)

This register selects the base I/O addresses and page addresses.

	7							0
Field	Base Address				Page Address			
Bit	EMBA3	EMBA2	EMBA1	EMBA0	EMIO3	EMIO2	EMIO1	EMIO0
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7	EMBA3	R/W	MMSA base address, bit 3
6	EMBA2	R/W	MMSA base address, bit 2
5	EMBA1	R/W	MMSA base address, bit 1
4	EMBA0	R/W	MMSA base address, bit 0
3	EMIO3	R/W	MMSA/B page register(s) I/O address, bit 3
2	EMIO2	R/W	MMSA/B page register(s) I/O address, bit 2
1	EMIO1	R/W	MMSA/B page register(s) I/O address, bit 1
0	EMIO0	R/W	MMSA/B page register(s) I/O address, bit 0

Bits 3–0 Provide selection of page-register I/O addresses for both MMSA (pages 0–7) and MMSB (pages 0–3). These page registers get mapped at different I/O locations as listed in Table 4-24 on page 4-32. Once the I/O address spaces for these registers are programmed, writes to these address spaces store mapped address-extension bits 20–14 for the windows in MMSA and MMSB as indicated in Table 4-25 on page 4-32.

Table 4-24 MMSA/B Page Register I/O Addresses

Bit				Page Register I/O Address							
3	2	1	0	Page 0	Page 1	Page 2	Page 3	Page 4	Page 5	Page 6	Page 7
0	0	0	0	208h	2208h	4208h	6208h	8208h	A208h	C208h	E208h
0	0	0	1	218h	2218h	4218h	6218h	8218h	A218h	C218h	E218h
0	1	0	1	258h	2258h	4258h	6258h	8258h	A258h	C258h	E258h
0	1	1	0	268h	2268h	4268h	6268h	8268h	A268h	C268h	E268h
1	0	1	0	2A8h	22A8h	42A8h	62A8h	82A8h	A2A8h	C2A8h	E2A8h
1	0	1	1	2B8h	22B8h	42B8h	62B8h	82B8h	A2B8h	C2B8h	E2B8h
1	1	1	0	2E8h	22E8h	42E8h	62E8h	82E8h	A2E8h	C2E8h	E2E8h

Table 4-25 Page Register Contents Description

Bit	Name	R/W	Function
7	PAGEEN	R/W	0 = Page disable 1 = Page enable
6	EA20	R/W	MMSA/B translate address bit A20
5	EA19	R/W	MMSA/B translate address bit A19
4	EA18	R/W	MMSA/B translate address bit A18
3	EA17	R/W	MMSA/B translate address bit A17
2	EA16	R/W	MMSA/B translate address bit A16
1	EA15	R/W	MMSA/B translate address bit A15
0	EA14	R/W	MMSA/B translate address bit A14

Bits 7–4 Provide selection of starting addresses of memory windows in MMSA (pages 0–7). These windows get mapped at different system-memory address locations as listed in Table 4-26 on page 4-33. Once the system-memory address spaces for these windows are programmed (i.e., the base address is selected) and the page register I/O addresses are selected, the software can program the page registers. Software must also program the other address extension registers (MMS Address Extension 2 register at Index 6Eh, MMSA Address Extension 1 register at Index 67h, and MMS Address Extension 1 register at Index 6Ch) before a page in the MMSA or the MMSB is enabled. See the ROM Configuration 1 register at Index 65h and the MMSB Control Register at Index 74h to enable the MMSA and the MMSB. Note that the MMSB base address is fixed at 0A0000h and is not under software control.

Table 4-26 MMSA Base Addresses

Bit				Base Address							
7	6	5	4	Page 0	Page 1	Page 2	Page 3	Page 4	Page 5	Page 6	Page 7
0	0	0	0	C0000h	C4000h	C8000h	CC000h	D0000h	D4000h	D8000h	DC000h
0	0	0	1	C4000h	C8000h	CC000h	D0000h	D4000h	D8000h	DC000h	E0000h
0	0	1	0	C8000h	CC000h	D0000h	D4000h	D8000h	DC000h	E0000h	E4000h
0	0	1	1	CC000h	D0000h	D4000h	D8000h	DC000h	E0000h	E4000h	E8000h
0	1	0	0	D0000h	D4000h	D8000h	DC000h	E0000h	E4000h	E8000h	EC000h
0	1	0	1	D4000h	D8000h	DC000h	E0000h	E4000h	E8000h	EC8000h	F0000h

4.3.34 MMS Address Extension 2 Register (Index 6Eh)

This is the MMS register containing address-extension bits 21–22 for pages 0–3.

	7							0
Field	Page 3		Page 2		Page 1		Page 0	
Bit	E3A22	E3A21	E2A22	E2A21	E1A22	E1A21	E0A22	E0A21
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7	E3A22	R/W	MMSA/MMSB page 3 address extension bit 22
6	E3A21	R/W	MMSA/MMSB page 3 address extension bit 21
5	E2A22	R/W	MMSA/MMSB page 2 address extension bit 22
4	E2A21	R/W	MMSA/MMSB page 2 address extension bit 21
3	E1A22	R/W	MMSA/MMSB page 1 address extension bit 22
2	E1A21	R/W	MMSA/MMSB page 1 address extension bit 21
1	E0A22	R/W	MMSA/MMSB page 0 address extension bit 22
0	E0A21	R/W	MMSA/MMSB page 0 address extension bit 21

Bits 7–0 Provide mapped address-extension bits 22–21 for both the MMSA and the MMSB (pages 0–3).

Before programming this register, software must select the region to program (i.e., MMSA or MMSB). This selection is performed via bit 1 of the MMSB Control register at Index 74h.

4.3.35 Miscellaneous 1 Register (Index 6Fh)

This register is used to control ISA and MCU functions.

	7							0
Field	MMSZ							
Bit	MMSZ3	MMSZ2	MMSZ1	MMSZ0	MRDLY	DMWS	RESCPU	GATEA20
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7	MMSZ3	R/W	MMS memory range, bit 3
6	MMSZ2	R/W	MMS memory range, bit 2
5	MMSZ1	R/W	MMS memory range, bit 1
4	MMSZ0	R/W	MMS memory range, bit 0
3	MRDLY		MEMR delay: 1 = Delay MEMR by 1 DMACK cycle 0 = No delay for MEMR
2	DMWS	R/W	DMA wait states: 1 = 2 wait cycles 0 = 1 wait cycle
1	RESCPU	R/W	CPU reset: A Low-to-High transition in this bit automatically resets the CPU. The reset lasts for 16 PROCLK cycles.
0	GATEA20	R/W	A20 gate control: 1 = A20 is CPUA20 0 = A20 is 0 if bit 0 of Port 92h and the A20GATE pin are also 0

Table 4-27 MMS Memory Range Select Logic

Bit 7 6 5 4	MMS Memory Range	Bit 7 6 5 4	MMS Memory Range
0 0 0 0	No MMS	1 0 0 0	8 Mbyte
0 0 0 1	1 Mbyte	1 0 0 1	9 Mbyte
0 0 1 0	2 Mbyte	1 0 1 0	10 Mbyte
0 0 1 1	3 Mbyte	1 0 1 1	11 Mbyte
0 1 0 0	4 Mbyte	1 1 0 0	12 Mbyte
0 1 0 1	5 Mbyte	1 1 0 1	13 Mbyte
0 1 1 0	6 Mbyte	1 1 1 0	14 Mbyte
0 1 1 1	7 Mbyte	1 1 1 1	15 Mbyte

MMS memory range provides a method for disabling on-board memory accesses when directly accessing memory (non-MMS cycles). The programmed MMS memory range is subtracted from the amount of on-board memory configured by bits 4–2 of the Memory Configuration 1 register at Index 66h or bits 7–6 of the Function Enable 2 register at Index B1h. If the programmed MMS memory range is equal to or greater than the configured amount of on-board memory, then on-board memory is disabled and all memory cycles are transferred on the ISA bus.

For example, assume that 8 Mbyte of DRAM have been configured as on-board main memory. If the user desires to have 10 Mbyte of linearly addressed DOS ROM, then bits 7–4 of the Miscellaneous 1 register at Index 6Fh should be programmed to 0010b. This allows all accesses in the range 000000–5FFFFFh to transfer to DRAM and all accesses in the range 600000–FFFFFFh to be ISA bus transfers. For more information, see “ROM-DOS Memory” on page 2-8.

4.3.36 Miscellaneous 6 Register (Index 70h)

This register is used to control MCU and PMU functions.

	7							0
Bit	(Reserved)	PGP0DIR	SACIN	(Reserved)				
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7		R/W	(Reserved)
6	PGP0DIR	R/W	PGP0 pin direction: 0 = Input 1 = Output
5	SACIN	R/W	Software ACIN input: 1 = PMU behaves as if ACIN pin was asserted
4–1		R/W	(Reserved)
0		R/W	(Reserved—must be 0)

Bit 5 Setting this bit is equivalent to asserting the ACIN pin, except that the $\overline{BL4}$–$\overline{BL0}$ pins are not gated with this bit. Therefore, the $\overline{BL4}$–$\overline{BL0}$ pins can still change the state of the PMU.

4.3.37 MMSA Device 1 Register (Index 71h)

This register selects the peripheral device that each MMSA page controls.

	7							0
Field	Page 3 Device		Page 2 Device		Page 1 Device		Page 0 Device	
Bit	EMDP31	EMDP30	EMDP21	EMDP20	EMDP11	EMDP10	EMDP01	EMDP00
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7	EMDP31	R/W	MMSA page 3 device type, bit 1
6	EMDP30	R/W	MMSA page 3 device type, bit 0
5	EMDP21	R/W	MMSA page 2 device type, bit 1
4	EMDP20	R/W	MMSA page 2 device type, bit 0
3	EMDP11	R/W	MMSA page 1 device type, bit 1
2	EMDP10	R/W	MMSA page 1 device type, bit 0
1	EMDP01	R/W	MMSA page 0 device type, bit 1
0	EMDP00	R/W	MMSA page 0 device type, bit 0

Page 3 Device Select		**Page 2 Device Select**	
Bit 7 6	**Selected Device**	**Bit 5 4**	**Selected Device**
0 0	MMS ROM DOS	0 0	MMS ROM DOS
0 1	MMS on-board main memory	0 1	MMS on-board main memory
1 0	Reserved	1 0	Reserved
1 1	MMS BIOS	1 1	MMS BIOS

Page 1 Device Select		**Page 0 Device Select**	
Bit 3 2	**Selected Device**	**Bit 1 0**	**Selected Device**
0 0	MMS ROM DOS	0 0	MMS ROM DOS
0 1	MMS on-board main memory	0 1	MMS on-board main memory
1 0	Reserved	1 0	Reserved
1 1	MMS BIOS	1 1	MMS BIOS

4.3.38 MMSA Device 2 Register (Index 72h)

Bits of this register select the peripheral device that each MMSA page controls.

	7							0
Field	Page 7 Device		Page 6 Device		Page 5 Device		Page 4 Device	
Bit	EMDP71	EMDP70	EMDP61	EMDP60	EMDP51	EMDP50	EMDP41	EMDP40
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7	EMDP71	R/W	MMSA page 7 device type, bit 1
6	EMDP70	R/W	MMSA page 7 device type, bit 0
5	EMDP61	R/W	MMSA page 6 device type, bit 1
4	EMDP60	R/W	MMSA page 6 device type, bit 0
3	EMDP51	R/W	MMSA page 5 device type, bit 1
2	EMDP50	R/W	MMSA page 5 device type, bit 0
1	EMDP41	R/W	MMSA page 4 device type, bit 1
0	EMDP40	R/W	MMSA page 4 device type, bit 0

Page 7 Device Select		**Page 6 Device Select**	
Bit 7 6	**Selected Device**	**Bit 5 4**	**Selected Device**
0 0	MMS ROM DOS	0 0	MMS ROM DOS
0 1	MMS on-board main memory	0 1	MMS on-board main memory
1 0	Reserved	1 0	Reserved
1 1	MMS BIOS	1 1	MMS BIOS

Page 5 Device Select		**Page 4 Device Select**	
Bit 3 2	**Selected Device**	**Bit 1 0**	**Selected Device**
0 0	MMS ROM DOS	0 0	MMS ROM DOS
0 1	MMS on-board main memory	0 1	MMS on-board main memory
1 0	Reserved	1 0	Reserved
1 1	MMS BIOS	1 1	MMS BIOS

4.3.39 MMSB Device Register (Index 73h)

Bits of this register select the peripheral device that each MMSB page controls.

	7							0
Field	Page 3 Device		Page 2 Device		Page 1 Device		Page 0 Device	
Bit	EMDP31	EMDP30	EMDP21	EMDP20	EMDP11	EMDP10	EMDP01	EMDP00
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7	EMDP31	R/W	MMSB page 3 device type, bit 1
6	EMDP30	R/W	MMSB page 3 device type, bit 0
5	EMDP21	R/W	MMSB page 2 device type, bit 1
4	EMDP20	R/W	MMSB page 2 device type, bit 0
3	EMDP11	R/W	MMSB page 1 device type, bit 1
2	EMDP10	R/W	MMSB page 1 device type, bit 0
1	EMDP01	R/W	MMSB page 0 device type, bit 1
0	EMDP00	R/W	MMSB page 0 device type, bit 0

Page 3 Device Select		**Page 2 Device Select**	
Bit 7 6	**Selected Device**	**Bit 5 4**	**Selected Device**
0 0	MMS ROM DOS	0 0	MMS ROM DOS
0 1	MMS on-board main memory	0 1	MMS on-board main memory
1 0	Reserved	1 0	Reserved
1 1	MMS BIOS	1 1	MMS BIOS

Page 1 Device Select		**Page 0 Device Select**	
Bit 3 2	**Selected Device**	**Bit 1 0**	**Selected Device**
0 0	MMS ROM DOS	0 0	MMS ROM DOS
0 1	MMS on-board main memory	0 1	MMS on-board main memory
1 0	Reserved	1 0	Reserved
1 1	MMS BIOS	1 1	MMS BIOS

4.3.40 MMSB Control Register (Index 74h)

This register controls MCU and PMU functions.

	7							0
Bit	NENLB4	NENLB2	(Reserved)		ENPMCIRQ0	PGP1DIR	MMSABSEL	ENMMSB
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7	NENLB4	R/W	1 = $\overline{BL4}$ going active does not cause the PMU to transition to Suspend mode.
6	NENLB2	R/W	1 = $\overline{BL2}$ going active does not cause the PMU to transition to Sleep mode.
5		R/W	(Reserved—must be 0)
4		R/W	(Reserved—must be 0)
3	ENPMCIRQ0	R/W	Enable IRQ0 active in Doze mode: 1 = Enabled. If this bit is 1 and bit 0 of the PMU Control 2 register at Index AFh is 0, the CPUCLK signal is active while IRQ0 is High in Doze mode. If this bit is 1 and bit 0 of the PMU Control 2 register is 1, the CPUCLK signal remains active for an additional 64 refresh cycles after IRQ0 is deasserted.
2	PGP1DIR	R/W	PGP1 pin direction: 0 = Input 1 = Output
1	MMSABSEL	R/W	MMSA and MMSB select bit: 0 = MMSB 1 = MMSA
0	ENMMSB	R/W	Enable MMSB: 1 = Enabled

Bit 7 If bit 7 = 0 and ACIN = 0, the PMU transitions to Suspend mode when $\overline{BL4}$ = 0.

Bit 6 If bit 6 = 0 and ACIN = 0, the PMU transitions to Sleep mode when $\overline{BL2}$ = 0. This bit has no effect on whether an SMI for $\overline{BL2}$ is generated. It also does not have an effect on the status read at the CPU Status 0 register at Index A3h.

Bit 1 Because the MMSA and the MMSB use the same I/O address for the page registers and address extension registers for pages 0–3, this bit selects either the MMSA or the MMSB for programming. In other words, this bit directs the I/O address to either the MMSA or the MMSB. If this bit is 1, an I/O cycle accesses the MMSA; otherwise, an I/O cycle accesses the MMSB.

Bit 0 This bit enables/disables all windows in the MMSB. If this bit is 0, all windows are disabled. If this bit is 1, each window can be individually enabled/disabled via bit 7 of the appropriate page register.

4.3.41 Activity Mask 1 Register (Index 75h)

This register is used in conjunction with the Activity Mask 2 register at Index 76h and the Resume Mask register at Index 08h to enable which activities are detected by the PMU. Each of these bits masks out the corresponding activity. A 1 means the activity is masked; a 0 means it is counted. For information on status and enabling ACIN activity, see “Activity Status 1 Register (Index A0h)” on page 4-53 and “PMU Control 3 Register (Index ADh)” on page 4-60.

	7							0
Bit	INT	ACIN	MMS	KB	DRQ3	DRQ2	DRQ1	DRQ7–DRQ5
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7	INT	R/W	Interrupt recognition: 1 = Disabled 0 = Enabled
6	ACIN	R/W	AC adapter input or Low-to-High transition of bit 5 of the Miscellaneous 6 register at Index 70h recognition: 1 = Disabled 0 = Enabled
5	MMS	R/W	MMS cycle activity recognition: 1 = Disabled 0 = Enabled
4	KB	R/W	Keyboard interrupt (IRQ1) activity recognition: 1 = Disabled 0 = Enabled
3	DRQ3	R/W	DRQ3 activity recognition: 1 = Disabled 0 = Enabled
2	DRQ2	R/W	DRQ2 activity recognition: 1 = Disabled 0 = Enabled
1	DRQ1	R/W	DRQ1 activity recognition: 1 = Disabled 0 = Enabled
0	DRQ7–DRQ5	R/W	DRQ7–DRQ5 activity recognition: 1 = Disabled 0 = Enabled

Note:

Activities are not detected during the execution of SMIs or NMIs.

Bit 7 INT means that all interrupts from IRQ2 to IRQ15 can serve as PMU activity that causes a PMU transition from either Low-Speed PLL or Doze mode to High-Speed PLL mode. This bit does not allow the above stated IRQ levels to wake up the processor from Sleep, Suspend, or Off mode.

IRQ3, IRQ4, and IRQ8 can be programmed individually by the Resume Mask register at Index 08h to act as wake-up events. Unmasking these events allows their occurrence to wake up the system from Sleep, Suspend, or Off mode into High-Speed PLL mode.

Bit 6 Bit 4 of the PMU Control 3 register at Index ADh must also be set to permit ACIN going active to count as activity.

Bits 6 and 3–0 These activities also wake up the system from Sleep, Suspend, or Off mode into High-Speed PLL mode.

Bit 4 Unlike other IRQs, IRQ1 activity cannot be masked by the 8259 PIC.

4.3.42 Activity Mask 2 Register (Index 76h)

This register is used in conjunction with the Activity Mask 1 register at Index 75h and the Resume Mask register at Index 08h to enable which activities are detected by the PMU. Each of these bits masks the corresponding activity when set. For more information on activity status reporting, see “Activity Status 2 Register (Index A1h)” on page 4-53.

	7							0
Bit	PMW	VD	PIO1	PIO0	COM	HD	FD	LPT
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7	PMW	R/W	Memory address range (defined in the Memory Write Activity Lower and Upper Boundary registers at Indexes 9Ah and 9Bh) recognition: 1 = Disabled 0 = Enabled
6	VD	R/W	Video memory write recognition: 1 = Disabled 0 = Enabled
5	PIO1	R/W	I/O address range (defined in the I/O Activity Address 1 register at Index 8Dh) recognition: 1 = Disabled 0 = Enabled
4	PIO0	R/W	I/O address range (defined in the I/O Activity Address 0 register at Index 8Ch) recognition: 1 = Disabled 0 = Enabled
3	COM	R/W	COM1–COM2 read/write recognition: 1 = Disabled 0 = Enabled
2	HD	R/W	Hard disk drive read/write recognition: 1 = Disabled 0 = Enabled
1	FD	R/W	Floppy disk drive read/write recognition: 1 = Disabled 0 = Enabled
0	LPT	R/W	LPT1–LPT3 read/write recognition: 1 = Disabled 0 = Enabled

Note:

Activities are not detected during SMI or NMI execution.

4.3.43 Control B Register (Index 77h)

This register controls various general functions.

	7							0
Bit	UART_IR4	UART_IR3	UART_IOP	UART_EN	AUTLOW	(Reserved)		
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7	UART_IR4	R/W	Set internal UART IRQ to IRQ4
6	UART_IR3	R/W	Set internal UART IRQ to IRQ3
5	UART_IOP	R/W	0 = Set internal UART I/O address to 3F8–3FFh 1 = Set internal UART I/O address to 2F8–2FFh
4	UART_EN	R/W	1 = Enable internal UART
3	AUTLOW	R/W	1 = Enable Auto Low-Speed
2–0		R/W	(Reserved)

Bit 6 If the ÉlanSC310 microcontroller is in Local Bus mode, program the PIRQ Configuration register at Index B2h so it does not conflict with the IRQ selection for the internal UART.

Bit 3 This bit is only useful if bit 6 of the I/O Wait State register at Index 61h is 1, which puts the CPU clock into High-Speed PLL mode. Otherwise, the CPU clock is always operating at the low-speed PLL frequency. This function is not dependent on any activity (see “Auto Low-Speed Control Register (Index 9Fh)” on page 4-51).

4.3.44 Reserved Registers (Indexes 78–7Fh)

These index locations are reserved.

4.3.45 Power Control 1 Register (Index 80h)

This register controls the PMC2 output pin in High-Speed PLL mode, Low-Speed PLL mode, and Doze mode. It also enables/disables the low-speed PLL and video PLL in Doze mode.

	7							0
Bit	0CLK_DOZ	DZ2	(Reserved)			FO2	(Reserved)	
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7	0CLK_DOZ	R/W	1 = Low-speed PLL and video PLL are shut down in Doze mode
6	DZ2	R/W	1 = State of PMC2 pin in Doze mode
5–4		R/W	(Reserved)
3		R/W	(Reserved—must be 0)
2	FO2	R/W	1 = State of PMC2 pin in High-Speed PLL and Low-Speed PLL modes
1–0		R/W	(Reserved)

Note:

The state of PMC2 after power-on is Low. When the bit is 0, the corresponding PMC output is Low. For details, see Chapter 1, “Power Management.”

4.3.46 Power Control 2 Register (Index 81h)

This register activates the PMC2 output pin in Sleep, Suspend, and Off modes. It also enables/disables the low-speed PLL and video PLL in Sleep, Suspend, and Off modes.

	7							0
Bit	0CLK_SUS	SU2			0CLK_SLP	SP2		
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7	0CLK_SUS	R/W	1 = Low-speed PLL and Video PLL are shut down in Suspend and Off modes
6	SU2	R/W	1 = State of PMC2 pin in Suspend and Off modes
5–4		R/W	(Reserved)
3	0CLK_SLP	R/W	1 = Low-speed PLL and video PLL are shut down in Sleep mode
2	SP2	R/W	1 = State of PMC2 pin in Sleep mode
1–0		R/W	(Reserved)

Note:

The state of PMC2 after power-on is Low. When the bit is 0, the corresponding PMC output is Low. For details, see Chapter 1, "Power Management."

4.3.47 NMI/SMI Enable Register (Index 82h)

This register is used to enable the generation of NMIs or SMIs during certain conditions, such as mode changes or battery-low conditions. By default, NMIs and SMIs are disabled. The choice of SMIs or NMIs is selected by bit 7 of the SMI MMS Upper Page register at Index A9h.

	7							0
Bit	BL3	BL2	BL1	SUS	SLP	DZ	ON	RESU
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7	BL3	R/W	1 = Battery low warning 3 generates an NMI or SMI
6	BL2	R/W	1 = Battery low warning 2 generates an NMI or SMI
5	BL1	R/W	1 = Battery low warning 1 generates an NMI or SMI
4	SUS	R/W	1 = PMU generates NMI or SMI before entering Off mode from Suspend mode
3	SLP	R/W	1 = PMU generates NMI or SMI before entering Suspend mode from Sleep mode
2	DZ	R/W	1 = PMU generates NMI or SMI before entering Sleep mode from Doze mode
1	ON	R/W	1 = PMU generates NMI or SMI before entering Doze mode from Low-Speed PLL mode
0	RESU	R/W	1 = $\overline{\text{SUS}}$/$\overline{\text{RES}}$ pin will generate NMI or SMI

Note:

Bit 7 of AT-compatible port 70h must be 0 for NMIs to occur.

4.3.48 High-Speed to Low-Speed Timer Register (Index 83h)

This read/write register is used to program the time-out period from High-Speed PLL mode to Low-Speed PLL mode. The minimum period is 1⁄512 s. The maximum period is 15.94 s. A value of 00h disables the PMU timer. If any activities are detected during the timer counting period, the PMU timer is reset to 00h automatically.

	7							0
Field	Time-Out Period in Multiples of 1⁄512 s or 1⁄16 s							
Default	0	0	0	0	0	0	0	0

Note: *The timer granularity can be changed to 1⁄16 s by setting bit 6 of the PMU Control 2 register at Index AFh.*

4.3.49 Low-Speed to Doze Timer Register (Index 84h)

This read/write register is used to program the time-out period from Low-Speed PLL mode to Doze mode. The minimum period is 1⁄16 s; the maximum is 63.75 s. A value of 00h disables the PMU timer. If any activities are detected during the timer counting period, the PMU timer is reset to 00h automatically and returns to High-Speed PLL mode.

	7							0
Field	Time-Out Period in Multiples of 1⁄16 s or 1⁄4 s							
Default	0	0	0	0	0	0	0	0

Note: *The timer granularity can be changed to 1⁄4 s by setting bit 7 of the PMU Control 2 register at Index AFh.*

4.3.50 Doze to Sleep Timer Register (Index 85h)

This read/write register is used to program the time-out period from Doze mode to Sleep mode. The minimum period is 4 s; the maximum is 1024 s. A value of 00h disables the PMU timer. If any activities are detected during the timer counting period, the PMU timer is reset to 00h automatically and returns to High-Speed PLL mode.

	7							0
Field	Time-Out Period in Multiples of 4 s							
Default	0	0	0	0	0	0	0	0

4.3.51 Sleep to Suspend Timer Register (Index 86h)

This read/write register is used to program the time-out period from Sleep mode to Suspend mode. The minimum period is 1⁄16 s; the maximum is 16 s. A value of 00h disables the PMU timer. If a wake-up or the Resume key is detected during the timer counting period, the PMU timer is reset to 00h automatically and the system returns to High-Speed PLL mode.

	7							0
Field	Time-Out Period in Multiples of 1⁄16 s							
Default	0	0	0	0	0	0	0	0

4.3.52 Suspend to Off Timer Register (Index 87h)

This read/write register is used to program the time-out period from Suspend mode to Off mode. The minimum period is 1 min; the maximum is 256 min. A value of 00h disables the PMU timer. If a wake-up or the Resume key is detected during the timer counting period, the counter is reset to 00h automatically and the system returns to High-Speed PLL mode.

	7							0
Field	Time-Out Period in Multiples of 1 min							
Default	0	0	0	0	0	0	0	0

4.3.53 Software Mode Control Register (Index 88h)

This register allows software to force the system into a particular mode.

	7							0
Field						SPC		
Bit						SPC2	SPC1	SPC0
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7–3		W	(Reserved)
2	SPC2	W	Software command control bit 2
1	SPC1	W	Software command control bit 1
0	SPC0	W	Software command control bit 0

Note:

Do not force the PMU from a clock-stopped state (Sleep, Suspend, or Off mode) into the Low-Speed PLL mode or Doze mode. Instead, force the PMU into High-Speed PLL mode. This ensures that internal flags are properly reset for future PMU transitions.

Table 4-28 PMU Mode Select Logic

SPC2	SPC1	SPC0	Mode
0	0	0	High-Speed PLL
1	0	0	Low-Speed PLL
0	0	1	Doze
0	1	1	Sleep
0	1	0	Suspend

4.3.54 General-Purpose I/O 0 Register (Index 89h)

This is a write-only register. This register is used to control the PGP0 pin in either direct-control mode or address-decode mode when PGP0 is configured as an output. In direct-control mode, the state of PGP0 is controlled by bit 7. When bit 7 is 1, the PGP0 output is Low. When bit 7 is 0, PGP0 is High. In address-decode mode, PGP0 functions as a simple address decode. PGP0 is High until the SA9–SA3 signals match bits 6–0 of this register, at which time PGP0 goes Low for as long as the signals match. PGP0 can also be gated internally with the I/O Write command signal. The General-Purpose I/O Control register at Index 91h is used to select how PGP0 operates as an output. Bit 6 of the Miscellaneous 6 register at Index 70h is used for PGP0 direction control.

	7							0
Field	Address Bits 9–3							
Bit	DX	A9	A8	A7	A6	A5	A4	A3
Default	0	0	0	0	0	0	0	0

4.3.55 Reserved Registers (Indexes 8A-8Bh)

These index locations are reserved.

4.3.56 I/O Activity Address 0 Register (Index 8Ch)

This register is used by the PMU software to program the I/O address that the activity monitor checks (bit 4 of the Activity Mask 2 register at Index 76h is the mask). This is a write-only register. Status is read from bit 4 of the Activity Status 2 register at Index A1h.

	7							0
Field	I/O Address, Bits 9–3							
Bit	(Reserved)	A9	A8	A7	A6	A5	A4	A3
Default	0	0	0	0	0	0	0	0

4.3.57 I/O Activity Address 1 Register (Index 8Dh)

This register is used by the PMU software to program the I/O address that the activity monitor checks (bit 5 of the Activity Mask 2 register at Index 76h is the mask). This is a write-only register. Status is read from bit 5 of the Activity Status 2 register at Index A1h.

	7							0
Field		I/O Address, Bits 9–3						
Bit	(Reserved)	A9	A8	A7	A6	A5	A4	A3
Default	0	0	0	0	0	0	0	0

4.3.58 Reserved Register (Index 8Eh)

This index location is reserved.

4.3.59 Clock Control Register (Index 8Fh)

This register is used to program the crystal restart-delay time and CPU restart-delay time. Bits 7–0 of this register are not reset when exiting Micro Power Off mode.

	7							0
Bit	(Reserved)					XST2	XST1	XST0
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7		W	(Reserved—must be 1)
6–5		W	(Reserved)
4		W	(Reserved—must be 0)
3		W	(Reserved)
2	XST2	W	PLL restart delay time control bit 2
1	XST1	W	PLL restart delay time control bit 1
0	XST0	W	PLL restart delay time control bit 0

Table 4-29 PLL Restart Time Select Logic

XST2	XST1	XST0	PLL Restart Time
0	0	0	4 ms
0	0	1	8 ms
0	1	0	16 ms
0	1	1	32 ms
1	0	0	64 ms
1	0	1	128 ms
1	1	0	256 ms
1	1	1	1 s

Note:

A 256-ms restart time is recommended when clocks are started from a PMU state where the low-speed PLL is disabled. A 128-ms restart time is recommended when clocks are started from a PMU state where the low-speed PLL is enabled. Results are not guaranteed if values less than these are used. This restart value applies to both High-Speed PLL and Low-Speed PLL mode restarts.

4.3.60 Reserved Register (Index 90h)

This index location is reserved.

4.3.61 General-Purpose I/O Control Register (Index 91h)

This register is used to control the PGP3–PGP0 pins. PGP0 and PGP1 can be driven directly, can be gated by I/O commands, or can be driven by simple address decodes. PGP2 and PGP3 can be automatically switched to a programmable level when the Power Management Unit switches to Off mode, can be gated by I/O commands, or can be driven by simple address decodes (i.e., each pair of pins has three ways it can be controlled). When implemented as simple-address decodes, the PGP pin goes active when the address bus matches the address bits (bits 9–3) programmed into the General-Purpose

I/O register for that PGP pin. All remaining address bits are Don't Cares. It is up to the system designer to externally qualify this pin with the $\overline{IOR}$, $\overline{IOW}$, $\overline{MEMR}$, or $\overline{MEMW}$ command signal. For more information, see the descriptions for each of the General-Purpose I/O registers in this chapter (Indexes 89h, 94h, 95h, and 9Ch).

	7							0
Field	PGP Pin 3		PGP Pin 2		PGP Pin1		PGP Pin 0	
Bit	PG3IO1	PG3IO0	PG2IO1	PG2IO0	PG1IO1	PG1IO0	PG0IO1	PG0IO0
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7	PG3IO1	W	Program general-purpose pin 3 gate control bit 1
6	PG3IO0	W	Program general-purpose pin 3 gate control bit 0
5	PG2IO1	W	Program general-purpose pin 2 gate control bit 1
4	PG2IO0	W	Program general-purpose pin 2 gate control bit 0
3	PG1IO1	W	Program general-purpose pin 1 gate control bit 1
2	PG1IO0	W	Program general-purpose pin 1 gate control bit 0
1	PG0IO1	W	Program general-purpose pin 0 gate control bit 1
0	PG0IO0	W	Program general-purpose pin 0 gate control bit 0

PG0IO1	PG0IO0	PGP0 Output
0	0	PGP0 direct control mode
1	0	PGP0 gates with I/O Write command
0	1	PGP0 acts as an address decode only
1	1	PGP0 acts as an address decode only

PG1IO1	PG1IO0	PGP1 Output
0	0	PGP1 direct control mode
1	0	PGP1 acts as an address decode only
0	1	PGP1 gates with I/O Read command
1	1	PGP1 acts as an address decode only

PG2IO1	PG2IO0	PGP2 Output
0	0	PGP2 automatically switches to the inverse of bit 7 of the General-Purpose I/O 2 register at Index 94h when the PMU is switched to Off mode
1	0	PGP2 gates with I/O Write command
0	1	PGP2 acts as an address decode only
1	1	PGP2 acts as an address decode only

PG3IO1	PG3IO0	PGP3 Output
0	0	PGP3 automatically switches to the inverse of bit 7 of the General-Purpose I/O 3 register at Index 95h when the PMU is switched to Off mode
1	0	PGP3 acts as an address decode only
0	1	PGP3 gates with I/O Read command
1	1	PGP3 acts as an address decode only

4.3.62 UART Clock Enable Register (Index 92h)

This register is used to control the UART clock.

	7							0
Bit	(Reserved)							ENCLK
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7–1		W	(Reserved)
0	ENCLK	W	1 = Enable clock to internal 16450 UART

4.3.63 Reserved Register (Index 93h)

This index location is reserved and must be 0.

4.3.64 General-Purpose I/O 2 Register (Index 94h)

This is a write-only register. This register is used to control the PGP2 pin in either power-management mode or address-decode mode. In power-management mode, the state of PGP2 is High when the PMU is not in Off mode. When the PMU transitions to Off mode, the state of PGP2 is determined by bit 7. When bit 7 is 1, PGP2 is Low. When bit 7 is 0, PGP2 is High. In address-decode mode, PGP2 functions as a simple address decode. PGP2 is High until the SA9–SA3 signals match bits 6–0 of this register, at which time PGP2 goes Low for as long as the signals match. PGP2 can also be gated internally with the I/O Write command signal. The General-Purpose I/O Control register at Index 91h is used to select how PGP2 operates as an output. PGP2 cannot operate as an input.

	7							0
Field	Address Bits 9–3							
Bit	DX	A9	A8	A7	A6	A5	A4	A3
Default	0	0	0	0	0	0	0	0

4.3.65 General-Purpose I/O 3 Register (Index 95h)

This is a write-only register. This register is used to control the PGP3 pin in either power-management mode or address-decode mode. In power-management mode, the state of PGP3 is High when the PMU is not in Off mode. When the PMU transitions to Off mode, the state of PGP3 is determined by bit 7. When bit 7 is 1, PGP3 is Low. When bit 7 is 0, PGP3 is High. In address-decode mode, PGP3 functions as a simple address decode. PGP3 is High until the SA9–SA3 signals match bits 6–0 of this register, at which time PGP3 goes Low for as long as the signals match. PGP3 can also be gated internally with the I/O Read command signal. The General-Purpose I/O Control register at Index 91h is used to select how PGP3 operates as an output. PGP3 cannot operate as an input.

	7							0
Field	Address Bits 9–3							
Bit	DX	A9	A8	A7	A6	A5	A4	A3
Default	0	0	0	0	0	0	0	0

4.3.66 Reserved Registers (Indexes 96–99h)

These index locations are reserved.

4.3.67 Memory Write Activity Lower Boundary Register (Index 9Ah)

This register specifies the low memory-address boundary that the activity monitor counts as activity (addresses A23–A20 are all 0). The high memory address is defined by the Programmable Memory Write Activity Upper Boundary register at Index 9Bh. This activity is enabled/masked by bit 7 of the Activity Mask 2 register at Index 76h. This is a write-only register.

An activity is generated when a memory write to any address greater than the low address and less than the high address occurs. Address bits A13–A0 are Don't Cares when determining if the memory address is within the programmable range.

	7							0
Bit	LSA19	LSA18	LSA17	LSA16	LSA15	LSA14	(Reserved)	ENHIT
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7	LSA19	W	Lower boundary memory address SA19
6	LSA18	W	Lower boundary memory address SA18
5	LSA17	W	Lower boundary memory address SA17
4	LSA16	W	Lower boundary memory address SA16
3	LSA15	W	Lower boundary memory address SA15
2	LSA14	W	Lower boundary memory address SA14
1		W	(Reserved)
0	ENHIT	W	Hit count function enable

4.3.68 Memory Write Activity Upper Boundary Register (Index 9Bh)

This register specifies the high memory-address boundary that the activity monitor counts as activity (addresses A23–A20 are all 0). The low memory-address boundary is specified by the Programmable Memory Write Activity Lower Boundary register at Index 9Ah. This activity is enabled/masked by bit 7 of the Activity Mask 2 register at Index 76h. This is a write-only register.

An activity is generated when a memory write to any address greater than the low address and less than the high address occurs.

	7							0
Field						Hit Count Limit		
Bit	HSA19	HSA18	HSA17	HSA16	(Reserved)	SC2	SC1	SC0
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7	HSA19	W	Upper boundary memory address SA19
6	HSA18	W	Upper boundary memory address SA18
5	HSA17	W	Upper boundary memory address SA17
4	HSA16	W	Upper boundary memory address SA16
3		W	(Reserved)
2–0	SC2–SC0	W	Hit count limit, bits 2–0

Table 4-30 Hit-Count Limit Bit Logic

Bit 2	1	0	Hit-Count Limit
0	0	0	1
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

Notes:

The hit count is cleared when a PMU state-transition counter expires or any other activity occurs. Thus, all memory writes must occur in the same PMU state, without other activity, before they count as activity.

If the hit-count limit is 2 and two or more memory write cycles fall into the range specified by the address range, it counts as activity. If bit 0 of the Memory Write Activity Lower Boundary register at Index 9Ah is not set, then this function is disabled.

4.3.69 General-Purpose I/O 1 Register (Index 9Ch)

This is a write-only register. It is used to control the PGP1 pin in either direct-control mode or address-decode mode when PGP1 is configured as an output. In direct-control mode, the state of PGP1 is controlled by bit 7. When bit 7 is 1, PGP1 is Low. When bit 7 is 0, PGP1 is High. In address-decode mode, PGP1 functions as a simple address decode. PGP1 is High until the SA9–SA3 signals match bits 6–0 of this register, at which time PGP1 goes Low for as long as the signals match. PGP1 can also be gated internally with the I/O Read command signal. The General-Purpose I/O Control register at Index 91h is used to select how PGP1 operates as an output. Bit 2 of the MMSB Control register at Index 74h is used for PGP1 direction control.

	7							0
Field		Address Bits 9–3						
Bit	DX	A9	A8	A7	A6	A5	A4	A3
Default	0	0	0	0	0	0	0	0

4.3.70 Reserved Register (Index 9Dh)

This index location is reserved and must be set to 40h.

4.3.71 Auto Low-Speed Control Register (Index 9Fh)

This register controls the auto low-speed trigger and duration period. Bit 3 of the Control B register at Index 77h disables or enables the trigger; bit 6 of the I/O Wait State register at Index 61h enables the high-speed CPU clock.

	7							0
Field					Low Speed		Trigger Period	
Bit	(Reserved)				LOW1	LOW0	T1	T0
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7–4			(Reserved)
3	LOW1	W	Low-speed duration bit 1
2	LOW0	W	Low-speed duration bit 0
1	T1	W	Trigger period bit 1
0	T0	W	Trigger period bit 0

Table 4-31 Trigger Period Select Logic

T1	T0	Trigger Period
0	0	4 s
0	1	8 s
1	0	16 s
1	1	32 s

Table 4-32 Low-Speed Duration Period Select Logic

LOW1	LOW0	Low-Speed Period
0	0	0.25 s
0	1	0.5 s
1	0	1 s
1	1	2 s

4.3.72 Activity Status 1 Register (Index A0h)

This register contains the activity status of system peripherals and signals. A 1 indicates activity. Software can clear this register by writing any data to it.

	7							0
Bit	INT	ACIN	MMS	KB	DRQ3	DRQ2	DRQ1	DRQ0
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7	INT	R/W	1 = Interrupt was detected active
6	ACIN	R/W	1 = AC adapter input or bit 5 of the Miscellaneous 6 register at Index 70h was detected active
5	MMS	R/W	1 = MMS was detected active
4	KB	R/W	1 = Keyboard was detected active
3	DRQ3	R/W	1 = DRQ3 was detected active
2	DRQ2	R/W	1 = DRQ2 was detected active
1	DRQ1	R/W	1 = DRQ1 was detected active
0	DRQ0	R/W	1 = DRQ5, DRQ6, or DRQ7 was detected active

Note:

INT includes all interrupts from IRQ2 to IRQ15.

4.3.73 Activity Status 2 Register (Index A1h)

This register contains the activity status of system peripherals and signals. A 1 indicates activity. Software can clear this register by writing any data to it.

	7							0
Bit	PMW	VD	PIO1	PIO0	COM	HD	FD	LPT
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7	PMW	R	1 = Programmable memory range write access was detected active
6	VD	R	1 = Video memory write (0B0000–0BFFFFh) was detected active
5	PIO1	R	1 = General-Purpose I/O 1 register was detected active
4	PIO0	R	1 = General-Purpose I/O 0 register was detected active
3	COM	R/W	1 = COM1 or COM2 was detected active
2	HD	R/W	1 = Hard disk drive was detected active
1	FD	R/W	1 = Floppy disk drive was detected active
0	LPT	R/W	1 = LPT1, LPT2, or LPT3 was detected active

4.3.74 PMU Status 1 Register (Index A2h)

This register contains the status of the $\overline{\text{LPH}}$ pin.

	7							0
Bit	LPH1	(Reserved)						
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7	LPH1	R	Status of $\overline{\text{LPH}}$ pin
6-0		R	(Reserved)

4.3.75 CPU Status 0 Register (Index A3h)

	7							0
Field		Last Mode			Battery Low			
Bit	(Reserved)	LIND2	LIND1	LIND0	BL3IN	BL2IN	BL1IN	PG0IN
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7		R	(Reserved)
6	LIND2	R	Last PMU state indicator 2
5	LIND1	R	Last PMU state indicator 1
4	LIND0	R	Last PMU state indicator 0
3	BL3IN	R	Battery low detect pin 3 ($\overline{\text{BL3}}$) input data
2	BL2IN	R	Battery low detect pin 2 ($\overline{\text{BL2}}$) input data
1	BL1IN	R	Battery low detect pin 1 ($\overline{\text{BL1}}$) input data
0	PG0IN	R	Programmable general-purpose I/O pin 0 (PGP0) input data

Note:

The last PMU mode indicator bits are not updated for PMU state transitions caused by writes to the Software Mode Control register at Index 88h.

Table 4-33 Last PMU Mode Indicator Bits

LIND2	LIND1	LIND0	Last Mode
0	0	0	High-Speed PLL
1	0	0	Low-Speed PLL
0	0	1	Doze
0	1	1	Sleep
0	1	0	Suspend

4.3.76 CPU Status 1 Register (Index A4h)

The last mode status is accurate only between the time when an event transitions the PMU to High-Speed PLL mode and when the timer expires in High-Speed PLL mode. This function is intended to allow the system to know what state the PMU was in when an event brought it out of a lower-power mode than High-Speed PLL mode. This register must be read before the timer expires in High-Speed PLL mode.

	7							0
Field						Present State		
Bit	ACIN	(Reserved)			PG1IN	PIND2	PIND1	PIND0
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7	ACIN	R	ACIN input status
6–4		R	(Reserved)
3	PG1IN	R	Status of the PGP1 pin
2	PIND2	R	PMU Indicator bit 2
1	PIND1	R	PMU Indicator bit 1
0	PIND0	R	PMU Indicator bit 0

Table 4-34 Present PMU Mode Indicator Bits

PIND2	PIND1	PIND0	Present Mode
0	0	0	High-Speed PLL
1	0	0	Low-Speed PLL
0	0	1	Doze
0	1	1	Sleep
0	1	0	Suspend

4.3.77 NMI/SMI Control Register (Index A5h)

Reading this register returns the status information on the source of an NMI or SMI. Writing this register allows the mode change to occur on the next refresh cycle after the write. For example, if the ÉlanSC310 microcontroller is programmed to generate an NMI or SMI when the mode changes from Low-Speed PLL mode to Doze mode, then the PMU stays in Low-Speed PLL mode after the generation of an NMI or SMI until software writes to this register to enable the mode change.

	7							0
Bit	BL3	BL2	BL1	SU	SP	DZ	ON	RESUME
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7	BL3	R/W	1 = The $\overline{BL3}$ (battery low third warning) generated an NMI or SMI
6	BL2	R/W	1 = The $\overline{BL2}$ (battery low second warning) generated an NMI or SMI
5	BL1	R/W	1 = The $\overline{BL1}$ (battery low first warning) generated an NMI or SMI
4	SU	R/W	1 = The NMI or SMI was generated from Suspend mode to Off mode
3	SP	R/W	1 = The NMI or SMI was generated from Sleep mode to Suspend mode
2	DZ	R/W	1 = The NMI or SMI was generated from Doze mode to Sleep mode
1	ON	R/W	1 = The NMI or SMI was generated from Low-Speed PLL mode to Doze mode
0	RESUME	R/W	1 = The NMI or SMI was generated by the $\overline{SUS}$/$\overline{RES}$ pin

4.3.78 Reserved Register (Index A6h)

This index location is reserved.

4.3.79 PMU Control 1 Register (Index A7h)

This register is used to control various PMU functions.

	7							0
Bit	(Reserved)		ENADIN2	ENADIN1	(Reserved)		SLREF	REFSEL
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7–6		R/W	(Reserved)
5	ENADIN2	R/W	1 = ÉlanSC310 microcontroller disables data propagation to UART and PMU controller in memory cycle
4	ENADIN1	R/W	1 = ÉlanSC310 microcontroller disables data propagation to display controller in memory cycle
3–2		R/W	(Reserved)
1	SLREF	R/W	Enable slow refresh for DRAM (if bit 0 is set, this bit's setting has no meaning): 0 = Enable slow refresh. When slow refresh is enabled, the Version register at Index 64h programs the 32-kHz divisor for the source. 1 = Disable slow refresh. The refresh rate is 65536's unless bit 0 is set. This bit is not reset when exiting Micro Power Off mode.
0	REFSEL	R/W	Select DRAM refresh source (this bit setting overrides any setting for bit 1): 0 = Use 32-kHz clock multiplied by 2 (65536's) as refresh source 1 = Use 8254 as refresh source

Note:

If the PMU is enabled to stop the 8254 clock, setting bit 0 will cause the DRAM refresh to be lost.

4.3.80 Reserved Register (Index A8h)

This index location is reserved.

4.3.81 SMI MMS Upper Page Register (Index A9h)

This register, together with the “SMI MMS Page Register (Index AAh)” on page 4-57 is used to control the MMS page used during an SMI.

	7							0
Bit	GENSMI	(Reserved)	SMIA23	SMIA22	(Reserved)			
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7	GENSMI	R/W	1 = ÉlanSC310 microcontroller generates an SMI instead of an NMI for the sources enabled in the NMI/SMI Enable register at Index 82h. (Bit 7 of the AT-Compatible Port 70h must be 0 for NMIs to occur.)
6		R/W	(Reserved)
5	SMIA23	R/W	SMI MMS translate address A23 for SMI only. See “SMI MMS Page Register (Index AAh)” on page 4-57.
4	SMIA22	R/W	SMI MMS translate address A22 for SMI only. See “SMI MMS Page Register (Index AAh)” on page 4-57.
3-0		R/W	(Reserved)

4.3.82 SMI MMS Page Register (Index AAh)

This register is used to control the SMI MMS page register.

	7							0
Bit	SMIA21	SMIA20	SMIA19	SMIA18	SMIA17	SMIA16	SMIA15	SMIA14
Default	0	0	0	0	0	0	0	0

Note:

SMI MMS is active only in System Management Mode (SMM).

Bit	Name	R/W	Function
7	SMIA21	R/W	SMI MMS translate address bit A21
6	SMIA20	R/W	SMI MMS translate address bit A20
5	SMIA19	R/W	SMI MMS translate address bit A19
4	SMIA18	R/W	SMI MMS translate address bit A18
3	SMIA17	R/W	SMI MMS translate address bit A17
2	SMIA16	R/W	SMI MMS translate address bit A16
1	SMIA15	R/W	SMI MMS translate address bit A15
0	SMIA14	R/W	SMI MMS translate address bit A14

Note:

A special MMS with page address 060000h is used for the SMI function. SMI accesses that use the MMS always map to on-board memory.

4.3.83 Power Control 3 Register (Index ABh)

This register activates the PMC3 and PMC4 output pins in High-Speed PLL mode, Low-Speed PLL mode, Doze mode, Sleep mode, and Suspend mode. The PMC4 pin can be programmed to toggle inactive when the hard-disk-drive timer expires. For more information on using PMC4 with the hard-disk-drive timer, see “Accesses to Powered-Down Device SMI” on page 1-29.

	7							0
Bit	SU4	SP4	DZ4	FO4	SU3	SP3	DZ3	FO3
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7	SU4	R/W	State of PMC4 pin in Suspend mode
6	SP4	R/W	State of PMC4 pin in Sleep mode
5	DZ4	R/W	State of PMC4 pin in Doze mode
4	FO4	R/W	State of PMC4 pin in High-Speed PLL and Low-Speed PLL modes
3	SU3	R/W	Inverse state of PMC3 pin in Suspend mode
2	SP3	R/W	Inverse state of PMC3 pin in Sleep mode
1	DZ3	R/W	Inverse state of PMC3 pin in Doze mode
0	FO3	R/W	Inverse state of PMC3 pin in High-Speed PLL and Low-Speed PLL modes

Notes:

The initial state of PMC4 after power-on is Low. When the bit is 0, the corresponding PMC output for that PMU mode is Low.

The initial state of PMC3 after power-on is High. When the bit is 0, the corresponding PMC output for that PMU mode is High.

PMC4 is the pin associated with the dedicated hard-disk-drive address activity decode and timer. If PMC4 is not used for the hard disk drive, the hard-disk-drive timer and associated control must be disabled.

Bits 7–4 of this register are cleared when the hard-disk-drive timer expires.

4.3.84 Power Control 4 Register (Index ACh)

This register activates the PMC0 and PMC1 output pins in High-Speed PLL mode, Low-Speed PLL mode, Doze mode, Sleep mode, and Suspend mode. The PMC0 pin can be programmed to toggle inactive when the floppy-disk-drive timer expires. The PMC1 pin can be programmed to toggle inactive when the PIO timer expires. For more information on using the PMC pins with the timers, see “Accesses to Powered-Down Device SMI” on page 1-29.

	7							0
Bit	SU1	SP1	DZ1	FO1	SU0	SP0	DZ0	FO0
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7	SU1	R/W	State of PMC1 pin in Suspend mode
6	SP1	R/W	State of PMC1 pin in Sleep mode
5	DZ1	R/W	State of PMC1 pin in Doze mode
4	FO1	R/W	State of PMC1 pin in High-Speed PLL and Low-Speed PLL modes
3	SU0	R/W	State of PMC0 pin in Suspend mode
2	SP0	R/W	State of PMC0 pin in Sleep mode
1	DZ0	R/W	State of PMC0 pin in Doze mode
0	FO0	R/W	State of PMC0 pin in High-Speed PLL and Low-Speed PLL modes

Notes:

The initial state of PMC0 and PMC1 after power-on is Low. When the bit is 0, the corresponding PMC output for that PMU mode is Low.

PMC0 is associated with the dedicated floppy-disk-drive address activity decode and timer. If PMC0 is not used for the floppy disk drive, the floppy-disk-drive timer and associated control must be disabled.

PMC1 is associated with the dedicated PIO address activity decode and timer. If PMC1 is not used for the PIO, the PIO timer and associated control must be disabled.

Bits 7–4 These bits are cleared as a result of the PIO timer expiring.

Bits 3–0 These bits are cleared as a result of the floppy-disk-drive timer expiring.

4.3.85 PMU Control 3 Register (Index ADh)

This register controls several PMU functions. When switching the low-speed frequency, the low-speed PLL is divided. This divided frequency is selected for use when the PMU is not in the Low-Speed PLL mode. The Low-Speed Clock Select bits (bits 1 and 0 of the PMU Control 3 register) should be modified only when the PMU is not in Low-Speed PLL mode.

	7							0
Field							Clock Select	
Bit	(Reserved)	IRQ0SMIEN	(Reserved)	ENACIN	XTKBDEN	MAINOFF	ONCLK1	ONCLK0
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7		R/W	(Reserved)
6	IRQ0SMIEN	R/W	1 = 8254 channel 0 generates an SMI instead of the normal IRQ0. Note that the IRQ0 in the PIC must be enabled to allow this condition.
5		R/W	(Reserved)
4	ENACIN	R/W	1 = ACIN is treated as activity
3	XTKBDEN	R/W	1 = Allows the 8042CS and SYSCLK pins to become three-stated so they may be used as inputs, all other requirements being met. It also qualifies an internal decode so that port 60h is read as an internal port. It further switches a multiplexer to vector IRQ1 from the external pin to the output of this circuitry.
2	MAINOFF	R/W	1 = ÉlanSC310 microcontroller turns off the high-speed PLL in Low-Speed PLL mode; otherwise, the high-speed PLL is turned off in Doze mode.
1	ONCLK1	R/W	Low-Speed PLL mode CPU clock bit 1
0	ONCLK0	R/W	Low-Speed PLL mode CPU clock bit 0

Bit 4 This bit is used in conjunction with bit 6 of the Activity Mask 1 register at Index 75h. If bit 4 of the PMU Control 3 register is 0, bit 6 of the Activity Mask 1 register has no function.

Bits 1–0 These bits only have an effect in Low-Speed PLL mode.

Table 4-35 Low-Speed PLL Mode CPU Clock Speed Select

ONCLK1	ONCLK0	Clock Frequency to CPU	Internal CPU Operation Speed
0	0	9.216 MHz	4.608 MHz
0	1	4.608 MHz	2.304 MHz
1	0	2.304 MHz	1.152 MHz
1	1	1.152 MHz	0.576 MHz

4.3.86 Reserved Register (Index AEh)

This register is reserved.

4.3.87 PMU Control 2 Register (Index AFh)

This register controls several PMU operations.

	7							0
Bit	CHGON	CHGFUSET	BL1LOWSP	(Reserved)				EXTIR0ACT
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7	CHGON	R/W	Low-Speed to Doze Mode Timer register (Index 84h) unit value: 1 = 1⁄4 s 0 = 1⁄16 s
6	CHGFUSET	R/W	High-Speed to Low-Speed Mode Timer register (Index 83h) unit value: 1 = 1⁄16 s 0 = 1⁄512 s
5	BL1LOWSP	R/W	1 = Set CPU clock speed to low speed (9.2 MHz) in High-Speed PLL mode if $\overline{BL1}$ is Low and ACIN is Low.
4–1		R/W	(Reserved)
0	EXTIR0ACT	R/W	1 = Extend CPU run time for an additional 64 refresh cycles following IRQ0 while in Doze mode. This bit is effective only if bit 3 of the MMSB Control register at Index 74h is 1.

4.3.88 Function Enable 1 Register (Index B0h)

	7							0
Bit	(Reserved)	X1SEL	EXTSMIEDG	EXTSMIEN	DMASTCLK	EPPMODE	PPISBI	PPBIENB
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7		R/W	(Reserved)
6	X1SEL	W	X1OUT clock select: 0 = X1OUT determined by bit 2 of Index B1h 1 = X1OUT driven by BAUDOUT
5	EXTSMIEDG	R/W	External SMI active edge 1 = Active-Low external SMI 0 = Active-High external SMI
4	EXTSMIEN	R/W	1 = Enable external SMI
3	DMASTCLK	R/W	1 = DMA stop clock (Power-Save mode) enable
2	EPPMODE	R/W	EPP mode enable for parallel port
1	PPISBI	R/W	Bidirectional configuration enable for parallel port
0	PPBIENB	R/W	Bidirectional enable for parallel port

Bit 3 When this bit is set, the DMA clock runs only when a DMA access is happening; it is stopped between DMA transfers.

Bit 1 This bit is used to configure the ÉlanSC310 microcontroller's parallel-port control outputs for the parallel-port hardware interface implemented on the system board. When bit 1 is 0, $\overline{DBUFOE}$ is not generated during parallel-port accesses. This bit should only be 0 if the system is implemented without a system buffer. This bit must be 1 for systems that have a system buffer and implement a parallel port with either an input buffer and output latch, or just an output latch.

Bit 0 This bit is used to enable the bidirectional control for the parallel port. This bit functions only if bit 1 is 1. When bit 0 is 0, the parallel-port data-latch output-enable signal from the ÉlanSC310 microcontroller's CPU, $\overline{\text{PPOEN}}$, is forced Low, causing the parallel-port data-latch to drive the parallel-port data bus. When bit 0 is 1, the $\overline{\text{PPOEN}}$ signal is controlled by bit 5 of the Parallel Port Control register.

Table 4-36 Latch and Buffer Logic

PPBIENB	PPISBI	EPPMODE	Mode
0	0	0	Output latch, no system buffer, no input buffer, PP output only
0	0	1	(Invalid combination)
0	1	0	Output latch, system buffer, no input buffer, PP output only
0	1	1	Output latch, system buffer, no input buffer, EPP mode
1	0	0	(Invalid combination)
1	0	1	(Invalid combination)
1	1	0	Output latch, system buffer, input buffer, PP is bidirectional
1	1	1	Output latch, system buffer, input buffer, EPP mode

4.3.89 Function Enable 2 Register (Index B1h)

	7							0
Bit	EB_RMMD1	EB_RMMD0	(Reserved)	HSPLLFQ1	HSPLLFQ0	XTALUSE	(Reserved)	
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7	EB_RMMD1	R/W	RAM mode select bit 0
6	EB_RMMD0	R/W	RAM mode select bit 1
5		R/W	(Reserved—must be 0)
4	HSPLLFQ1	R/W	High-speed PLL frequency select, bit 1
3	HSPLLFQ0	R/W	High-speed PLL frequency select, bit 0
2	XTALUSE	R/W	Crystal interface: 1 = 14.336 MHz out on X1OUT pin 0 = Three-state X1OUT pin (see bit 6 of the Function Enable 1 register at Index B0h)
1-0			(Reserved)

Bit 7–6 When bit 7 of the Function Enable 3 register at Index B4h is 1, the values of bits 6 and 7 of this register determine the DRAM mode as shown in Table 4-38 on page 4-63.

Bit 4–3 Do not change these bits while running in the High-Speed PLL mode. To change these bits, first clear bit 6 of the I/O Wait State register at Index 61h, then wait for the next refresh to occur. Then change the high-speed PLL frequency and return to using the high-speed PLL.

Table 4-37 High-Speed PLL Frequency Select

HSPLLFQ 1 0	High-Speed PLL (CLK2) Frequency
0 0	40 MHz
0 1	50 MHz
1 0	66 MHz
1 1	(Reserved)

Note:

In order to meet DRAM timing at 33 MHz, the following registers need to be programmed as indicated:

Clear bit 6 of the Command Delay register at Index 60h.

Set bit 4 of the MMS Memory Wait State 1 register at Index 62h.

Set bit 5 of the ROM Configuration 1 register at Index 65h.

Set bits 5 and 6 of the Wait State Control register at Index 63h.

Table 4-38 RAM Mode Decode Logic

RAM Mode	Index B4h Bit 7	Index B1h Bits 7–6	Total Memory	Bank 0	Bank 1
256-Kbit × 4-bit DRAM	1	0 0	512 Kbyte	512 Kbyte	–
256-Kbit × 4-bit DRAM	1	0 1	1 Mbyte	512 Kbyte	512 Kbyte
Asymmetric 1-Mbit × 16-bit DRAM	1	1 0	2 Mbyte	2 Mbyte	–
Asymmetric 1-Mbit × 16-bit DRAM	1	1 1	4 Mbyte	2 Mbyte	2 Mbyte
(See Table 4-23 on page 4-28)	0	x x	Controlled by bits 4–2 of Index 66h		

Notes:

If bit 7 of the Function Enable 3 register at Index B4h is 1, bits 4–2 of the Memory Configuration 1 register at Index 66h are disabled. Bit 0 of the Memory Configuration 1 register determines Page mode or Enhanced Page mode.

If the 4-Mbyte memory configuration is selected, the Enhanced Page mode must be selected by setting bit 0 of the Memory Configuration 1 register. Page mode is illegal for this configuration.

4.3.90 PIRQ Configuration Register (Index B2h)

This register selects the IRQ level to which the PIRQ0 and PIRQ1 pins are connected. See Table 4-39 on page 4-65 for the valid IRQ selections. In Full ISA Bus mode, this register has no effect. This register is valid for Local Bus mode only. In Full ISA Bus mode, PIRQ0 is connected to IRQ3, and PIRQ1 is connected to IRQ6.

	7							0
Field	PIRQ1 Steering				PIRQ0 Steering			
Bit	PIRQ1SL3	PIRQ1SL2	PIRQ1SL1	PIRQ1SL0	PIRQ0SL3	PIRQ0SL2	PIRQ0SL1	PIRQ0SL0
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7	PIRQ1SL3	R/W	PIRQ1 IRQ select, bit 3
6	PIRQ1SL2	R/W	PIRQ1 IRQ select, bit 2
5	PIRQ1SL1	R/W	PIRQ1 IRQ select, bit 1
4	PIRQ1SL0	R/W	PIRQ1 IRQ select, bit 0
3	PIRQ0SL3	R/W	PIRQ0 IRQ select, bit 3
2	PIRQ0SL2	R/W	PIRQ0 IRQ select, bit 2
1	PIRQ0SL1	R/W	PIRQ0 IRQ select, bit 1
0	PIRQ0SL0	R/W	PIRQ0 IRQ select, bit 0

Note:

Do not program either of the PIRQ pins such that they conflict with other IRQs. For example, do not program both PIRQ pins to the same level or to the IRQ level used by the internal UART. Interrupt sharing is not supported.

Table 4-39 Interrupt Redirect Logic

PIRQ1SL3–PIRQ1SL0 or PIRQ0SL3–PIRQ0SL0				PIRQ1 or PIRQ0
3	**2**	**1**	**0**	**IRQ Selected**
0	0	0	0	(None)
0	0	0	1	(Reserved)
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	(Reserved)
1	0	0	1	IRQ9
1	0	1	0	IRQ10
1	0	1	1	IRQ11
1	1	0	0	IRQ12
1	1	0	1	(Reserved)
1	1	1	0	IRQ14
1	1	1	1	IRQ15

4.3.91 Miscellaneous 5 Register (Index B3h)

This register is used to obtain miscellaneous status information.

	7							0
Bit	32KHZSTE	ENFSTROMCS	FSTROMWS1	FSTROMWS0	ENSELFREF	ENBROMCS	EXTSMISTE	(Reserved)
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7	32KHZSTE	R	State of the 32-kHz clock
6	ENFSTROMCS	R/W	1 = $\overline{\text{ROMCS}}$ ROM accesses enabled to run at the high-speed clock rate
5	FSTROMWS1	R/W	Fast BIOS ROM wait-state select 1
4	FSTROMWS0	R/W	Fast BIOS ROM wait-state select 0
3	ENSELFREF	R/W	Self-refresh DRAM mode when CPUCLK is halted: 0 = CAS-before-RAS refresh (default) 1 = Self-refresh This bit is not reset when exiting Micro Power Off mode.
2	ENBROMCS	R/W	1 = $\overline{\text{ROMCS}}$ enabled as an address decode, but not qualified with $\overline{\text{MEMR}}$ or $\overline{\text{MEMW}}$
1	EXTSMISTE	R	State of external SMI pin
0			(Reserved)

Bit 6 This bit should not be set for systems that assert $\overline{\text{MCS16}}$ because running the ISA bus at the high-speed PLL frequency violates $\overline{\text{MCS16}}$ timing. When bit 6 is set, the Maximum ISA Bus signal, BALE, is not asserted for cycles to the $\overline{\text{ROMCS}}$ decode space. This may cause ISA decode conflicts for devices that use BALE to catch LA23–LA17.

Bits 5–4 When the $\overline{\text{ROMCS}}$ ROM accesses are enabled to run at high speed (bit 6), these bits control the number of wait states for these cycles as shown in Table 4-40 on page 4-66.

Bit 1 This bit indicates the state of the external SMI pin; it should be read to determine if an external SMI is being generated. The SMI handler should poll this bit and verify that it goes inactive prior to exiting the handler.

Table 4-40 ROM-BIOS Enable and Wait-State Select Logic

ENFSTROMCS	FSTROMWS1–FSTROMWS0	Number of Wait States for $\overline{\text{ROMCS}}$ Cycle
0	x x	(Fast $\overline{\text{ROMCS}}$ disabled)
1	0 0	4
1	0 1	3
1	1 0	2
1	1 1	1

4.3.92 Function Enable 3 Register (Index B4h)

This register is used to enable additional DRAM modes.

	7							0
Bit	ENRAME2	(Reserved)						
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7	ENRAME2	R/W	1 = Enable additional DRAM modes corresponding to bits 7–6 of the Function Enable 2 register at Index B1h
6			(Reserved—must be 1)
5–0			(Reserved)

Bit 7 When this bit is 0, RAM mode is determined by bits 4–2 of the Memory Configuration 1 register at Index 66h. Setting this bit enables additional DRAM modes configured via bits 7–6 of the Function Enable 2 register at Index B1h.

4.3.93 Reserved Registers (Indexes B5h-B7h)

These index locations are reserved.

4.3.94 ROM Configuration 3 Register (Index B8h)

This register controls the size of ROM DOS and the number of wait states for a $\overline{\text{DOSCS}}$ cycle.

	7							0
Bit	ENFSTRDOS	FRDOSWS1	FRDOSWS0	ENRDOSCS	RDOSSIZ3	RDOSSIZ2	RDOSSIZ1	RDOSSIZ0
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7	ENFSTRDOS	R/W	1 = Enable $\overline{\text{DOSCS}}$ accesses to run at the high-speed CPU clock rate
6	FRDOSWS1	R/W	$\overline{\text{DOSCS}}$ wait state select bit 0
5	FRDOSWS0	R/W	$\overline{\text{DOSCS}}$ wait state select bit 1
4	ENRDOSCS	R/W	1 = Enable $\overline{\text{DOSCS}}$ as an address decode not qualified with $\overline{\text{MEMR}}$ or $\overline{\text{MEMW}}$
3	RDOSSIZ3	R/W	$\overline{\text{DOSCS}}$ size select bit 3
2	RDOSSIZ2	R/W	$\overline{\text{DOSCS}}$ size select bit 2
1	RDOSSIZ1	R/W	$\overline{\text{DOSCS}}$ size select bit 1
0	RDOSSIZ0	R/W	$\overline{\text{DOSCS}}$ size select bit 0

Bit 7 This bit should not be set when decoding an 8-bit device in systems that assert $\overline{\text{MCS16}}$ because running the ISA bus at the high-speed PLL frequency violates $\overline{\text{MCS16}}$ timing. Bit 7 can be used if $\overline{\text{DOSCS}}$ accesses a 16-bit device by setting bit 1 of the ROM Configuration 2 register at Index 51h. Also, when bit 7 is 1, the Maximum ISA Bus signal, BALE, is not asserted for cycles to the $\overline{\text{DOSCS}}$ decode space. This may cause ISA decode conflicts for devices that use BALE to latch LA23–LA17.

Bits 6–5 When the $\overline{\text{DOSCS}}$ ROM accesses are enabled to run at high speed (bit 7), these bits control the number of wait states for these cycles as shown in Table 4-41 on page 4-67.

Table 4-41 ROM-DOS Enable and Wait-State Select Logic

ENFSTRDOS	FRDOSWS1–FRDOSWS0	Number of Wait States for $\overline{\text{DOSCS}}$ Cycle
0	x x	(See Index 50h)
1	0 0	4
1	0 1	3
1	1 0	2
1	1 1	1

Bits 3–0 These bits decode to the sizes shown in Table 4-42 on page 4-68.

Table 4-42 ROM-DOS Linear-Address Decode-Size Select Logic

RDOSSIZ3–RDOSSIZ0	$\overline{\text{DOSCS}}$ Address Decode	DOS ROM Size
0 0 0 0	($\overline{\text{DOSCS}}$ uses MMS mapping)	(NA)
0 0 0 1	F00000–FEFFFFh	1 Mbyte–64 Kbyte
0 0 1 0	E00000–FEFFFFh	2 Mbyte–64 Kbyte
0 0 1 1	D00000–FEFFFFh	3 Mbyte–64 Kbyte
0 1 0 0	C00000–FEFFFFh	4 Mbyte–64 Kbyte
0 1 0 1	B00000–FEFFFFh	5 Mbyte–64 Kbyte
0 1 1 0	A00000–FEFFFFh	6 Mbyte–64 Kbyte
0 1 1 1	900000–FEFFFFh	7 Mbyte–64 Kbyte
1 0 0 0	800000–FEFFFFh	8 Mbyte–64 Kbyte
1 0 0 1	700000–FEFFFFh	9 Mbyte–64 Kbyte
1 0 1 0	600000–FEFFFFh	10 Mbyte–64 Kbyte
1 0 1 1	500000–FEFFFFh	11 Mbyte–64 Kbyte
1 1 0 0	400000–FEFFFFh	12 Mbyte–64 Kbyte
1 1 0 1	300000–FEFFFFh	13 Mbyte–64 Kbyte
1 1 1 0	200000–FEFFFFh	14 Mbyte–64 Kbyte
1 1 1 1	100000–FEFFFFh	15 Mbyte–64 Kbyte

Note:

The linear $\overline{\text{DOSCS}}$ decode range must not overlap the on-board memory decode range unless the on-board memory in the overlapping range is disabled via bits 7–4 of the Miscellaneous 1 register at Index 6Fh.

4.3.95 Memory Configuration 2 Register (Index B9h)

This register disables refresh in Off mode and selects programmable drive strengths, as shown in Table 4-43 on page 4-69 and Table 4-44 on page 4-69.

	7							0
Bit	DISREFOFF	(Reserved)	MEMDATS1	MEMDATS0	MEMADRS1	MEMADRS0	MEMCTLS1	MEMCTLS0
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7	DISREFOFF	R/W	Disable refresh in the PMU Off mode
6		R/W	(Reserved)
5	MEMDATS1	R/W	D15–D0 drive strength select 1
4	MEMDATS0	R/W	D15–D0 drive strength select 0
3	MEMADRS1	R/W	MA10–MA0/SA23–SA13 and $\overline{\text{MWE}}$ drive strength select 1
2	MEMADRS0	R/W	MA10–MA0/SA23–SA13 and $\overline{\text{MWE}}$ drive strength select 0
1	MEMCTLS1	R/W	$\overline{\text{RAS0}}$, $\overline{\text{RAS1}}$ drive strength select 1
0	MEMCTLS0	R/W	$\overline{\text{RAS0}}$, $\overline{\text{RAS1}}$ drive strength select 0

Table 4-43 Output Drive Strength Select Logic

Drive Strength Select Bit Value 1 0	Output Drive Strength
0 0	E (default)
0 1	C
1 0	D
1 1	Three-state output

Table 4-44 I/O Drive Type Description

Drive Type	VCCIO (mA)	IOL_{TTL} (mA)	IOH_{TTL} (mA)
C	3.0–4.5	7.7–10.8	–8.6 to –34.2
D	3.0–4.5	7.7–10.8	–10.3 to –40.8
E	3.0–4.5	10.2–14.1	–13.6 to –53.9

Note:

Current out of a pin is given as a negative value.

4.3.96 Miscellaneous 3 Register (Index BAh)

This register enables the 14-MHz clock output on $\overline{\text{AFDT}}$ and preserves DRAM during Micro Power Off mode.

	7							0
Bit	(Reserved)				EN_14M_PP	EN_MPOM	(Reserved)	
Default	0	0	0	0	0	0	0	0

Bit	Name	R/W	Function
7–5		R/W	(Reserved)
4		R/W	(Reserved—must be 0)
3	EN_14M_PP	R/W	1 = Enable 14.336-MHz clock output on $\overline{\text{AFDT}}$
2	EN_MPOM	R/W	1 = Preserve DRAM during Micro Power Off mode
1–0		R/W	(Reserved)

Bit 3 When this bit is 1, the 14.336-MHz clock output is enabled on pin 80.

Bit 2 When this bit is 1, DRAM timing is maintained when entering Micro Power Off mode. It should be set if the contents of DRAM are to be preserved during Micro Power Off mode.

Bits 1–0 These bits are not reset when exiting Micro Power Off mode.

Table 4-45 Parallel Port Pin Redefinition

Pin No.	Name	New Function
80	AFDT	14.336-MHz clock
84	SLCTIN	PCMCOE
89	INIT	PCMCWE

APPENDIX

A CONFIGURATION INDEX REGISTER REFERENCE

Configuration index registers are used to set up and monitor the system configuration of the ÉlanSC310 microcontroller. The following table lists the configuration index registers alphabetically by name. Use it when you know the name of a configuration index register but not its index.

For a complete description of each of these registers, see “Configuration Index Registers” on page 4-11. For information on how to access these registers, see “Configuration Register Overview” on page 4-2.

Configuration Index Register	Index	Function
Activity Mask 1	75h	PMU
Activity Mask 2	76h	PMU
Activity Status 1	A0h	PMU
Activity Status 2	A1h	PMU
Auto Low-Speed Control	9Fh	PMU
Clock Control	8Fh	PMU
Command Delay	60h	ISA/MCU
Control A	48h	Control
Control B	77h	Control
CPU Status 0	A3h	PMU
CPU Status 1	A4h	PMU
Doze to Sleep Timer	85h	PMU
Drive Timer	47h	PMU
Function Enable 1	B0h	PMU
Function Enable 2	B1h	PMU
Function Enable 3	B4h	MCU
General-Purpose I/O 0	89h	PMU
General-Purpose I/O 1	9Ch	PMU
General-Purpose I/O 2	94h	PMU
General-Purpose I/O 3	95h	PMU
General-Purpose I/O Control	91h	PMU
High-Speed to Low-Speed Timer	83h	PMU
I/O Activity Address 0	8Ch	PMU
I/O Activity Address 1	8Dh	PMU
I/O Timeout	40h	PMU
I/O Wait State	61h	ISA/MCU
Low-Speed to Doze Timer	84h	PMU
Memory Configuration 1	66h	MCU
Memory Configuration 2	B9h	MMU
Memory Write Activity Lower Boundary	9Ah	PMU
Memory Write Activity Upper Boundary	9Bh	PMU
Miscellaneous 1	6Fh	ISA/MCU

Configuration Index Register	Index	Function
Miscellaneous 2	6Bh	MCU
Miscellaneous 3	BAh	MMU/PMU
Miscellaneous 4	44h	PMU
Miscellaneous 5	B3h	PMU
Miscellaneous 6	70h	PMU/MCU
MMS Address	6Dh	MCU
MMS Address Extension 1	6Ch	MCU
MMS Address Extension 2	6Eh	MCU
MMS Memory Wait State 1	62h	MCU
MMS Memory Wait State 2	50h	MCU
MMSA Address Extension 1	67h	MCU
MMSA Device 1	71h	MCU
MMSA Device 2	72h	MCU
MMSB Control	74h	PMU/MCU
MMSB Device	73h	MCU
NMI/SMI Control	A5h	PMU
NMI/SMI Enable	82h	PMU
PIO Address	45h	PMU
PIO Timer	46h	PMU
PIRQ Configuration	B2h	PMU
PMU Control 1	A7h	PMU
PMU Control 2	AFh	PMU
PMU Control 3	ADh	PMU
PMU Status 1	A2h	PMU
Power Control 1	80h	PMU
Power Control 2	81h	PMU
Power Control 3	ABh	PMU
Power Control 4	ACh	PMU
Resume Mask	08h	PMU
Resume Status	09h	PMU
ROM Configuration 1	65h	MCU
ROM Configuration 2	51h	MCU
ROM Configuration 3	B8h	MMU
Shadow RAM Enable 1	68h	MCU
Shadow RAM Enable 2	69h	MCU
Sleep to Suspend Timer	86h	PMU
SMI Enable	41h	PMU
SMI I/O Status	42h	PMU
SMI MMS Page	AAh	PMU
SMI MMS Upper Page	A9h	PMU
SMI Status	43h	PMU
Software Mode Control	88h	PMU
Suspend to Off Timer	87h	PMU
UART Clock Enable	92h	PMU
Version	64h	PMU
Wait State Control	63h	MCU

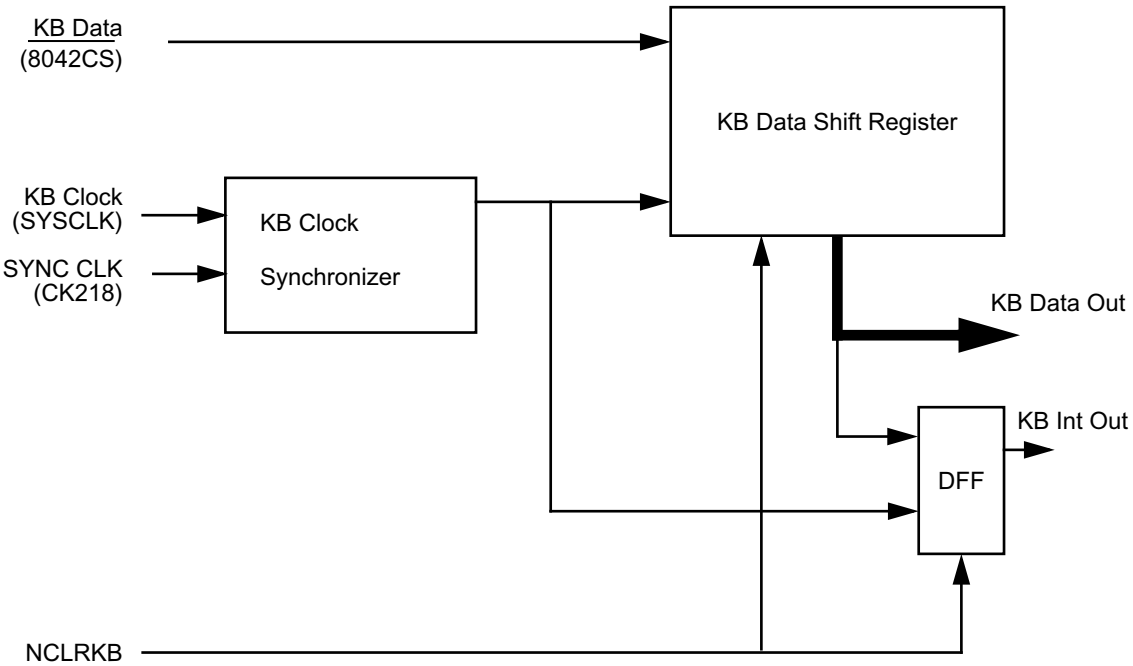
APPENDIX

B XT-KEYBOARD INTERFACE

The XT-keyboard interface consists of clock and data inputs to the ElanSC310 microcontroller and is compatible with the IBM PC/XT keyboard. One of the ElanSC310 microcontroller's output pins, such as a Programmable General-Purpose (PGP) pin, may be used to drive an actual XT-keyboard reset input.

The KB clock input is synchronized by two serial flip-flops, which are clocked by the CK218 signal. This signal is actually the CPU clock signal divided by 6. The (thus delayed) KB clock signal is then used to clock the data into the KB Data Shift register. The first bit to be clocked in is the start bit, and must be a logic 1 (High). Eight more bits are then shifted in, beginning with the least significant bit. On the ninth clock, the start bit is shifted into the DFF, which drives the KB interrupt output and is connected to IRQ1. At this same time, the KB Data I/O pad changes directions to become an output and is driven Low as a busy indication to the keyboard or other driving device. At this time, the host should respond to the interrupt and read the byte assembled in the KB Data Shift register at port 060h. The host also has the option of driving the clock pin Low as an additional handshake indication. After the host has read the byte from port 060h, the host clears the KB Data Shift register and the interrupt flip-flop by writing a 1 and then a 0 to bit 7 of port 061h. This action not only clears the shift register and interrupt, but also releases the data line to function as an input again.

Figure B-1 XT Keyboard Block Diagram



B.1 XT KEYBOARD ENABLE

The ÉlanSC310 microcontroller's XT keyboard circuitry is enabled by bit 3 of Index 0ADh. Setting this bit

- Allows the $\overline{8042CS}$ and SYSCLK pins to become three-stated so they may be used as inputs (all other requirements being met).
- Qualifies an internal decode so that port 060h is read as an internal port.
- Switches a multiplexer to vector IRQ1 from the external pin to the output of this circuitry.

B.2 KEYBOARD INTERFACE CONTROL

Two bits are provided for control of the XT keyboard interface. These bits are located at port 061h. Bit 7 is used to clear the keyboard interrupt and shift register, a flip-flop that enables the data line as an output (to act as a busy signal to the keyboard when Low). Two writes are required for the proper operation of this bit—one to set, and another to clear it. If bit 7 is not cleared, the shift register is held in a *clear* configuration. Bit 6, when High, makes the KBCLK line an output (driving it Low), which can also be used as a busy signal to the keyboard.

B.3 KEYBOARD DATA PORT

Once a serial keyboard byte has been assembled, it can be read at port 060h.

B.4 I/O MAP SUMMARY

- Setting bit 3 of Index ADh enables the XT keyboard.
- Port 060h contains the keyboard data.
- Setting bit 7 of Port 061h clears the keyboard shift register and interrupt.
- Setting bit 6 of Port 061h forces the keyboard clock Low.

B.5 PINS USED

The XT keyboard option uses the following pins:

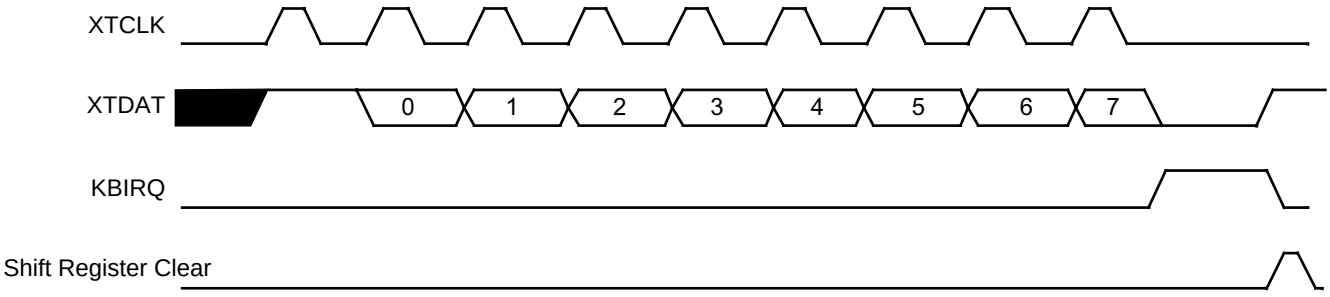
- **SYSCLK** Becomes the keyboard clock (XTCLK).
- **8042CS** Becomes the keyboard data (XTDAT).

Because both pins can also act as outputs and the data line is driven as an output at the end of each transferred byte, the pins must be driven by open-drain or open-collector drivers with external pull-ups.

B.6 TIMING

The XT-keyboard clock runs at approximately 100 kHz, or 10 µs per bit. The falling edge of the XTCLK input clocks the shift register, but it is delayed by two CPUCLK periods divided by 6. Therefore, XTDAT should be changed on the rising edge of the XTCLK signal. The XT-keyboard interface runs at speeds as high as 250 kHz.

Figure B-2 XT Keyboard Timing Diagram



INDEX

Numbers

A

B

C

D

E

F

G

H

K

L

M

N

O

P

Q

R

S

T

U