EXHIBIT 21

Intel® PXA27x Processor Family

Developer's Manual

April 2004

Order Number: 280000-001

Clocks and Power Manager 3

This chapter describes the clocks and power management unit and registers supported by the Intel® PXA27x Processor (PXA27x processor).

3.1 Overview

The clocks and power manager unit administers the processor resets, clocks, power management, and controls external power management ICs (PMIC). Control over these features allows optimization of the processor's overall power consumption and performance for an individual application. This chapter describes the following management and control units:

- Reset Manager (Section 3.4)

  The reset manager places the processor into one of five reset states: power-on, hardware, watchdog, GPIO, or sleep-exit reset.

- Clocks Manager (Section 3.5)

  The clocks manager contains all clock-generation, gating, and frequency controls for the processor.

- Power Manager (Section 3.6)

  The power manager controls the following:

  — All internal power domains and external power supply functionality

  — Entry and exit sequences for the different power modes: normal, idle, deep-idle, standby, sleep, and deep-sleep.

- Voltage Manager (Section 3.7)

  The voltage manager provides dynamic or static voltage management for the processor through the use of the power manager I^2C module, which is dedicated to communication with the external VCC_CORE regulator.

3.2 Features

Wireless Intel SpeedStep® technology includes the following features:

- Five reset sources: power-on, hardware, watchdog, GPIO, and exit from sleep and deep-sleep modes (sleep-exit)
- Multiple clock-speed controls to adjust frequency, including frequency change, turbo mode, half-turbo mode, fast-bus mode, memory clock, 13M mode, A-bit mode, and AC '97.
- Switchable clock source
- Functional-unit clock gating
- Programmable frequency-change capability
- One normal-operation power mode (also called run mode) and five low-power modes to control power consumption (idle, deep-idle, standby, sleep, and deep-sleep modes)

- Programmable I^2C-based external regulator interface to support changing dynamic core voltage, frequency change, and power mode coupling.

3.3 Signal Descriptions

The following signals are inputs or outputs from the clocks and power manager unit. The usage of some pins can be shared with general-purpose I/O (GPIO) functions. The GPIO functionality is described in Chapter 24, "General-Purpose I/O Controller".

Table 3-1. Clocks and Power Manager I/O Signal Descriptions (Sheet 1 of 2)

Name	Type	Definition
nRESET	Input	Signals the processor to enter hardware-reset state.
nRESET_OUT	Output	Signals the system that the processor is in any reset state (configurable for sleep- and deep-sleep-exit and GPIO resets).
GPIO<n>	Bidirectional	The GPIO<n> pins are used as standby and sleep wake-up sources. For the possible values of *n*, see Section 3.3.3.
GPIO<3>	Bidirectional	The GPIO<3> pin is used as a standby, sleep, and deep-sleep wake-up source.
GPIO<1:0>	Bidirectional	The GPIO<1:0> pins are used as standby and sleep/deep-sleep wake-up sources, and as deep-sleep wake-up sources after nBATT_FAULT or nVDD_FAULT is asserted.
PXTAL_IN	Input	Can be connected to an external 13-MHz crystal or to an external clock source. For more information, see Section 3.5.2.
PXTAL_OUT	Analog	Can be connected to an external 13-MHz crystal or to an external clock source. PXTAL_OUT must be either complementary to PXTAL_IN or floated.
CLK_PIO	Bidirectional	Can output a buffered version of the PXTAL_IN oscillator input or can be used as a clock-input alternative to PXTAL_IN.
TXTAL_IN	Input	Can be connected to an external 32.768-kHz crystal or to an external clock source. TXTAL_IN is distributed to the timekeeping control system and power-management unit. (See Section 3.5.3 for more information.)
TXTAL_OUT	Analog	Can be connected to an external 32.768-kHz crystal or to an external clock source. TXTAL_OUT must be either complimentary to TXTAL_IN or floated.
CLK_TOUT	Output	Drives a buffered and inverted version of the TXTAL_IN oscillator input.
CLK_REQ	Bidirectional	Input during power-on or hardware reset that indicates to the processor whether the processor oscillator clock input comes from PXTAL_IN (CLK_REQ low) or CLK_PIO (CLK_REQ floating). If CLK_PIO is the processor oscillator input, CLK_REQ becomes an output indicating when the processor oscillator is required. For more information, see Section 3.5.1.
CLK_EXT	Input	Can be used by the mobile scalable link (MSL) interface, SSP serial ports, or the operating system (OS) timer module as a clock input.
nBATT_FAULT	Input	Signals the processor that the main battery is low or has been removed from the system.
nVDD_FAULT	Input	Signals the processor that the main power supply is going out of regulation.
PWR_EN	Output	Enables the external low-voltage power domains: VCC_CORE, VCC_SRAM, VCC_PLL
SYS_EN	Output	Enables the external high-voltage power domains: VCC_IO, VCC_MEM, VCC_LCD, VCC_USB, VCC_USIM, VCC_BB
PWR_SCL	Input/Output	Power manager I^2C clock pin
PWR_SDA	Input/Output	Power manager I^2C data pin

Table 3-1. Clocks and Power Manager I/O Signal Descriptions (Sheet 2 of 2)

Name	Type	Definition
PWR_CAP<3:0>	Analog	The PWR_CAP pins connect to external capacitors that are used with on-chip DC-DC converter circuits to achieve very low power in sleep and deep-sleep modes.
PWR_OUT	Analog	Connects to an external isolated capacitor. See Section 3.6.2.3.
48_MHz	Output	48-MHz output clock—Divided down output generated from the 312-MHz peripheral clock. Generally used for board bring-up or debug purposes.

3.3.1 Hardware Reset (nRESET)

nRESET is an active-low input that signals the processor to enter hardware-reset state. The assertion of nRESET cannot be gated and causes the processor to enter a complete and unconditional reset state. While nRESET is asserted, the processor does not recognize any external events except CLK_REQ. (See Section 3.3.11 for more information.)

3.3.2 Internal Reset (nRESET_OUT)

nRESET_OUT is an active-low output that signals the system that the processor is in reset state. nRESET_OUT is asserted during power-on, hardware, watchdog, and sleep-exit resets.

If PSLR[SL_ROD] is clear (see Section 3.8.1.11), nRESET_OUT is also asserted during sleep and deep-sleep modes. See Section 3.4, Section 3.6.9, and Section 3.6.10 for descriptions of these modes.

If PCFR[GP_ROD] is clear (see Section 3.8.1.8), nRESET_OUT is also asserted during GPIO reset.

3.3.3 GPIO Wake-Up Sources

The GPIO<*n*> pins are used as standby and sleep wake-up sources. Possible values for *n* are:

116	113	102	101	100	99	98	97	96	95	94
93	91	90	83	53	40	39	38	37	36	35
34	31	17	16	15	14	13	12	11	10	9
4	3	1	0							

See Table 3-17 and Table 3-28 for details. These pins contain internal resistive pull-downs or pull-ups that are enabled during power-on, hardware, watchdog, and GPIO resets and disabled when PSSR[RDH] is clear.

The GPIO<3> pin is used as a wake-up source for standby, sleep, and deep-sleep modes. This pin has an internal resistive pull-up that is enabled during power-on, hardware, watchdog, and GPIO resets and disabled when PSSR[RDH] is clear.

The GPIO<1:0> pins are used as dedicated standby, sleep, or deep-sleep wake-up sources. If nVDD_FAULT or nBATT_FAULT caused entry or re-entry into deep-sleep mode, then these GPIO pins are the only allowed wake-up sources. Therefore, if using nBATT_FAULT or nVDD_FAULT to indicate power-supply health, at least one of these GPIO pins must be programmed as an input.

3.3.4 GPIO Reset (nRESET_GPIO/GPIO<1>)

The nRESET_GPIO signal is an alternate function to GPIO<1>. If PCFR[GPR_EN] is set, then GPIO<1> functions as nRESET_GPIO. nRESET_GPIO is an active-low input that signals the processor to enter GPIO reset state. The GPIO<1> functionality, as described in Chapter 24, "General-Purpose I/O Controller", is enabled unless PCFR[GPR_EN] is set.

For more information about the GPIO reset function, see Section 3.4.6.

3.3.5 Processor Oscillator Input (PXTAL_IN)

PXTAL_IN is a clock input that is distributed to the processor control system. PXTAL_IN can be connected to an external 13-MHz crystal or to an external clock source. If OSCC[CRI] is set, PXTAL_IN is ignored and must be grounded. See Section 3.8.2.3 for details.

3.3.6 Processor Oscillator Output (PXTAL_OUT)

PXTAL_OUT is the output of the processor clock-control crystal oscillator amplifier. If PXTAL_IN is connected to an external 13-MHz crystal, then PXTAL_OUT must be connected to the other terminal of the crystal. If PXTAL_IN is connected to an external clock source, PXTAL_OUT must be driven with a signal complementary to PXTAL_IN or left floating. Noise causes performance degradation if PXTAL_OUT is floated. For more information, see Section 3.5.2.

3.3.7 Processor Clock Input/Output (CLK_PIO/GPIO<9>)

The CLK_PIO signal can be either an output that is driven with a buffered version of the PXTAL_IN oscillator input or an input used as an alternative to PXTAL_IN, based on OSCC[CRI] (see Section 3.8.2.3). CLK_PIO can be shared with the GPIO<9> function if the buffered processor clock input/output function is not required. The GPIO<9> functionality, as described in Chapter 24, "General-Purpose I/O Controller", is enabled unless OSCC[PIO_EN] or OSCC[CRI] is set. For OSCC register details, see Section 3.8.2.3.

3.3.8 Timekeeping Oscillator Input (TXTAL_IN)

TXTAL_IN is a clock input that is distributed to the processor timekeeping control system, which includes the real-time clock (RTC) and power manager. TXTAL_IN can be connected to an external 32.768-kHz crystal or to an external clock source. If OSCC[OON] and OSCC[CRI] are both clear (see Section 3.8.2.3), TXTAL_IN can be left unconnected or grounded. For more information, see Section 3.5.3.

3.3.9 Timekeeping Oscillator Output (TXTAL_OUT)

TXTAL_OUT is the output of the timekeeping control system's crystal oscillator amplifier. If TXTAL_IN is connected to an external 32.768-kHz crystal, then TXTAL_OUT must be connected to the other terminal of the crystal. If TXTAL_IN is connected to an external clock source, then TXTAL_OUT must be driven with a signal complementary to TXTAL_IN or left floating. Noise causes performance degradation if TXTAL_OUT is floated. For more information, see Section 3.5.3.

3.3.10 Timekeeping Clock Output (CLK_TOUT/GPIO<10>)

The CLK_TOUT signal is an output that drives a buffered version of the TXTAL_IN oscillator input. CLK_TOUT can be shared with the GPIO<10> function if the buffered timekeeping-clock-output function is not required. The GPIO<10> functionality, as described in Chapter 24, "General-Purpose I/O Controller", is enabled unless OSCC[TOUT_EN] is set. During deep-sleep mode, CLK_TOUT is held low if OSCC[TOUT_EN] is set.

The CLK_TOUT pin has an internal resistive pull-down that is enabled during power-on, hardware, watchdog, and GPIO resets and is disabled when PSSR[RDH] is clear.

3.3.11 Clock Request (CLK_REQ)

CLK_REQ is an input during power-on or hardware reset that is used for external clock source selection. For details of operation, see Section 3.5.1.

3.3.12 External Clock (CLK_EXT)

CLK_EXT is an input that can be used by the mobile scalable link (MSL), operating system (OS) timer, or SSP modules as a possible clock source for those modules.

Note: CLK_EXT does not function during standby and sleep modes if any OS timer is on. For more information, see Chapter 22, "Operating System Timers".

3.3.13 Battery Fault and VDD Fault (nBATT_FAULT, nVDD_FAULT)

nBATT_FAULT is an active-low input indicating that the main battery is low or has been removed from the system. nVDD_FAULT is an active-low input indicating that the main power supply is going out of regulation (for example, when an overload occurs).

Assertion of either of these power faults causes the processor to enter deep-sleep mode if PMCR[xIDAE] is 0, as described in Section 3.6.4, Section 3.6.10, and Section 3.6.11.

Once nBATT_FAULT or nVDD_FAULT has been asserted, the processor recognizes only GPIO<1:0> as wake-up sources.

For more information, refer to the *Intel®* PXA27x *Processor Family Power Requirements Application Note*.

3.3.14 Power Enable (PWR_EN)

PWR_EN is an active-high output that enables the external low-voltage core power supplies. Deasserting PWR_EN informs the external regulator that the processor is entering sleep or deep-sleep mode and that the *external low-voltage power domains* (which include VCC_CORE, VCC_SRAM, and VCC_PLL) can be removed.

3.3.15 System Power Enable (SYS_EN)

SYS_EN is an active-high output that enables the external high-voltage system power supplies. Deasserting SYS_EN informs the power supplies that the processor is entering deep-sleep mode and that the *external high-voltage power domains* (which include VCC_IO, VCC_LCD, VCC_MEM, VCC_USIM, VCC_USB, and VCC_BB) can be removed.

3.3.16 Power Manager I^2C Clock (PWR_SCL/GPIO<3>)

PWR_SCL is the power manager I^2C clock pin (see Chapter 9, "I2C Bus Interface Unit" for details of the I^2C pins). PWR_SCL can be shared with the GPIO<3> function if the power manager I^2C feature is not required. The GPIO functionality, as described in Chapter 24, "General-Purpose I/O Controller", is enabled unless PCFR[PI^2C_EN] is set.

3.3.17 Power Manager I^2C Data (PWR_SDA/GPIO<4>)

PWR_SDA is the power manager I^2C data pin (see Chapter 9, "I2C Bus Interface Unit" for details of the I^2C pins). PWR_SDA can be shared with the GPIO<4> function if the power manager I^2C feature is not required. The GPIO functionality, as described in Chapter 24, "General-Purpose I/O Controller", is enabled unless PCFR[PI^2C_EN] is set.

3.3.18 Power Manager Capacitor Pins (PWR_CAP<3:0>)

The PWR_CAP signals connect to external capacitors, which are used with on-chip DC-DC converter circuitry to achieve very low power in sleep and deep-sleep modes.

3.3.19 Power Manager Supply Output (PWR_OUT)

The PXA27x processor requires an external 0.1-µF capacitor connected to the PWR_OUT pin. This connection is the only allowed connection or load on the PWR_OUT pin. This function is not optional. The pin **must** be connected to a capacitor for correct operation.

3.3.20 48-MHz Output Clock (48_MHz)

This output signal is the divided-down output generated from the 312-MHz peripheral clock. It is generally used for board bring-up or debug.

3.4 Reset Manager Operation

3.4.1 Reset Types

The reset manager can place the PXA27x processor into one of five resets:

- Power-On Reset (Section 3.4.3 —An uncompromised, ungated, total and complete reset that is used when power is first applied to the VCC_BATT pin

- Hardware Reset (nRESET asserted) (Section 3.4.4)—An uncompromised, ungated, total, and complete reset that is used when absolutely no system information requires preservation
- Watchdog Reset (Section 3.4.5)—Enabled through the OS timer; resets all registers except those listed in Table 3-2.
- GPIO Reset (GPIO<1>) (Section 3.4.6)—Enabled with PCFR[GPR_EN], GPIO reset is an alternative to hardware reset. An external source can reset the processor while preserving the registers listed in Table 3-2.
- Sleep-Exit Reset (Section 3.6.9 and Section 3.6.10)—Provides resets to the modules that have been powered off during sleep or deep-sleep mode so that they recover properly when power is reapplied.

See Table 2-6, "Effect of Each Type of Reset on Internal Register State" on page 2-9 for the states of all processor modules during all resets.

3.4.2 Boot Sequences After Reset

The required boot sequence begins. All units in the processor (except those listed in Table 3-2) start with their predefined reset conditions. Software must examine the Reset Controller Status register (RCSR—see Table 3-23) to determine the reset source.

Each type of reset requires a boot sequence tailored to the machine states that are lost or retained during the reset. These reset states are defined in the discussion of each reset.

3.4.3 Power-On Reset

Power-on reset is invoked when a positive power supply is detected on the backup battery pin, VCC_BATT, and the nRESET pin is deasserted.

3.4.3.1 Behavior During Power-On Reset

During power-on reset, the following conditions occur:

- All internal power domains except RTC remain powered off. (See Figure 3-2 for information on the power domains and what units are in each domain.)
- All internal registers and processes are held at their defined reset conditions.
- All clock sources are disabled; there is no activity inside the processor.
- The internal clocks are stopped and the chip is fully static.
- All pins assume their reset conditions.
- The nBATT_FAULT and nVDD_FAULT pins are ignored.
- nRESET_OUT pin is asserted when the power-on reset state occurs. For the states of processor pins during power-on reset, see the Pin Usage table in the *Intel® PXA270 Processor Electrical, Mechanical, and Thermal Specification* and *Intel® PXA27x Processor Family Electrical, Mechanical, and Thermal Specification* (*Intel® PXA27x Processor Family EMTS*).

3.4.3.2 Invoking Power-On Reset

Power-on reset is invoked when a positive transition is detected on the backup battery pin VCC_BATT followed by the deassertion of nRESET (within <10 ms). VCC_BATT has to be completely powered off for more than 10 µs prior to invoking power-on reset. Typically, this occurs when VCC_BATT is asserted (the back-up battery is initially inserted into the system), before the initial system configuration. Power-on reset cannot be entered from any other mode, since the processor must be completely powered off first. See the *Intel® PXA27x Processor Family EMTS* for more details.

The following sequence occurs during power-on reset:

1. A positive transition is detected on VCC_BATT. Power must be applied in the following order:
   a. nRESET is asserted before or at the same time as VCC_BATT is asserted.
   b. VCC_BATT followed by the deassertion of nRESET (initiates the power-on reset).
   c. The high-voltage supplies VCC_IO, VCC_MEM, VCC_LCD, VCC_USB, VCC_USIM, and VCC_BB in any order except VCC_IO, which must be first.

      After the high-voltage supplies have been applied, VCC_IO must be maintained at a voltage as high as or higher than the other high-voltage supplies (*except* VCC_BATT and VCC_USB).
   d. VCC_CORE, VCC_SRAM, and VCC_PLL in any order.
2. The internal power domains are powered on.
3. The 13-MHz processor oscillator and internal PLL clock generators are enabled and wait for stabilization.
4. nRESET_OUT is deasserted.

Caution: The *Intel® PXA27x Processor Family EMTS* describes the power-supply ramp times, timing specifications and sequences and the delay between the power-supply transitions and the deassertion of nRESET_OUT. Any other supply power-on sequence might cause permanent damage to the processor.

Note: Deassertion of the nRESET pin must follow VCC_BATT detection (within specification) to initiate a power-on-reset. Removing and then re-applying VCC_BATT without the use of nRESET puts the processor into an undefined state.

3.4.4 Hardware Reset

Hardware reset is invoked when the nRESET pin is asserted. All units in the processor are reset to known states. Hardware reset is complete and total.

3.4.4.1 Behavior During Hardware Reset

During hardware reset, the following conditions occur:

- All internal registers and processes are held at their defined reset conditions.
- While nRESET is asserted, the only activity inside the processor is the stabilization of the 13-MHz processor oscillator and phase-locked loops.
- The remaining internal clocks are stopped and the chip is fully static.

- All pins assume their reset conditions, and the nBATT_FAULT and nVDD_FAULT pins are ignored.
- The nRESET_OUT pin is asserted when nRESET is asserted.

For the states of all processor pins during hardware reset, refer to the *Intel® PXA27x Processor Family EMTS*.

3.4.4.2 Invoking Hardware Reset

Hardware reset is invoked when an external source drives the nRESET input pin low. nRESET is unmaskable and is always enabled. Upon assertion of nRESET, hardware reset is entered regardless of the previous mode. For more information, see the *Intel® PXA27x Processor Family EMTS*.

The following sequence occurs during hardware reset:

1. The nRESET input is asserted for the period of time described in the *Intel® PXA27x Processor Family EMTS*. The nRESET_OUT signal is also asserted.
2. nRESET is deasserted.

Caution: VCC_BATT must be stable prior to the deassertion of nRESET. Otherwise, permanent damage to the PXA27x processor might result, and operation is undefined.

3. The 13-MHz processor oscillator and internal PLL clock generators wait for stabilization, if not stabilized already.
4. nRESET_OUT is deasserted.

3.4.5 Watchdog Reset

Watchdog reset is invoked when software fails to prevent the watchdog timer from expiring. In watchdog reset, all units in the PXA27x processor (except those listed in Table 3-2) are reset to their predefined reset states.

3.4.5.1 Behavior During Watchdog Reset

During watchdog reset, the following conditions occur:

- All units except those listed in Table 3-2 are held at their defined reset conditions.
- All pins assume their reset conditions, and the nBATT_FAULT and nVDD_FAULT pins are ignored.
- The nRESET_OUT pin is asserted during watchdog reset.

3.4.5.2 Invoking Watchdog Reset

Watchdog reset is invoked when OWER[WME] is set and OSMR3 matches the OS timer counter (see Chapter 22, “Operating System Timers”). When this occurs, watchdog reset is entered regardless of the previous mode. When watchdog reset is invoked, the nRESET_OUT pin is asserted.

The following sequence occurs during watchdog reset:

1. The watchdog reset source asserts. When this happens, OSMR3 matches the OS timer counter (see Section 22.4.2, "Compares and Matches" on page 22-4). The nRESET_OUT signal is asserted.
2. The watchdog reset source is reset automatically (OWER[WME] is reset (see Table 22-6, "OWER Bit Definitions" on page 22-16) as a result of propagation of the internal reset state.
3. The 13-MHz processor oscillator and internal PLL clock generators wait for stabilization, regardless of the state of the clocks at watchdog reset assertion.
4. The nRESET_OUT signal is deasserted. See the *Intel® PXA27x Processor Family EMTS* for timing details.

Note: If a watchdog reset occurs while the processor is in sleep mode, both the sleep-reset and the watchdog-reset indicator bits are set in the RCSR.

3.4.6 GPIO Reset

GPIO reset is invoked when PCFR[GPR_EN] is set (see Section 3.8.1.8) and the GPIO<1> pin is asserted low for more than 240 ns in normal and idle modes of operation.

In standby, sleep, and deep-sleep modes, GPIO reset is first treated as a wake-up event. It is then recognized as a GPIO reset if held asserted for more than 1 ms after all power supplies are stable.

In GPIO reset, all units in the PXA27x processor (except those listed in Table 3-2) are reset to their predefined reset states.

Note: If the flash reset signal is connected to the processor nRESET_OUT signal, then software must disable GPIO reset before putting the flash into a program or erase cycle. This ensures that a GPIO reset occurs only when the flash is in a read cycle and, if the GPIO reset occurs, then nRESET_OUT is held for at least 230 ns, which meets the flash requirements.

3.4.6.1 Enabling GPIO Reset

The nRESET_GPIO pin (GPIO<1>) has an internal pull-up that is active following any reset (except sleep- or deep-sleep-exit) until PSSR[RDH] (see Section 3.8.1.2) is cleared. Thus, PCFR[GPR_EN] must be written **before** PSSR[RDH] is cleared.

To enable the GPIO reset function, set PCFR[GPR_EN] (see Section 3.8.1.8). When PCFR[GPR_EN] is set, the GPIO reset function overrides GPIO<1> settings, regardless of its state. PCFR[GPR_EN] is not affected during standby, sleep, and deep-sleep modes, allowing use of the GPIO reset function in these modes. PCFR[GPR_EN] is not affected by GPIO reset.

GPIO reset is ignored while a frequency-change operation is in progress. If GPIO<1> remains asserted low for 240 ns after the frequency change completes, it is recognized as a GPIO reset.

3.4.6.2 Behavior During GPIO Reset

During GPIO reset, the following conditions occur:

- All units (except those listed in Table 3-2) are held at their predefined reset states.
- All pins assume their reset conditions, and the nBATT_FAULT and nVDD_FAULT pins are ignored. See the Pin Usage table in the *Intel® PXA27x Processor Family EMTS* for the states of all processor pins during GPIO reset.

- If PCFR[GP_ROD] is clear, the nRESET_OUT pin is asserted during GPIO reset.

3.4.6.3 Invoking GPIO Reset

3.4.6.3.1 Normal Power Mode

In normal power mode, GPIO reset is invoked when the function is properly configured (see Section 3.4.6.1) and nRESET_GPIO is asserted low for the time specified in the *Intel® PXA27x Processor Family EMTS*. When GPIO reset is invoked, nRESET_OUT is asserted if PCFR[GP_ROD] is clear.

In normal mode, the following sequence occurs during GPIO reset:

1. The GPIO reset source asserts the GPIO reset pin (GPIO<1>). The processor's GPIO reset pin is edge-triggered. nRESET_OUT asserts if PCFR[GP_ROD] is clear.
2. If PCFR[GPR_EN] is set, the external GPIO reset source must be deasserted.
3. If GPIO reset was entered while the core PLL was disabled or unlocked (such as during a power-mode or clock change), the core PLL clock generator waits for stabilization.
4. nRESET_OUT is deasserted if PCFR[GP_ROD] is clear. See the *Intel® PXA27x Processor Family EMTS* for timing details.

3.4.6.3.2 Low-Power Modes

In the low-power modes (standby, sleep, and deep-sleep), if PCFR[GPR_EN] is set, the following GPIO reset sequence occurs:

1. The GPIO reset must be asserted for 100 µs to be treated first as a wake-up event.
2. Then, if held asserted for more than 1 ms after all power supplies are stable, it is recognized as a GPIO reset.
3. nRESET_OUT asserts if PCFR[GP_ROD] is clear.
4. If PCFR[GPR_EN] is set, the external GPIO reset source must be deasserted. (Failure to deassert the reset source would force the GPR_EN bit to be cleared thus preventing GPIO<1> to be configured as the GPIO Reset pin).
5. If GPIO reset was entered while the core PLL was disabled or unlocked (such as during a power-mode or clock change), the core PLL clock generator waits for stabilization.
6. nRESET_OUT is deasserted if PCFR[GP_ROD] is clear. See the *Intel® PXA27x Processor Family EMTS* for timing details.

Note: If a GPIO reset occurs while the processor is in sleep or deep-sleep mode, both the corresponding sleep-reset and the GPIO-reset indicator bits are set in the Reset Controller Status register (RCSR).

Note: GPIO<1> is not recognized as a reset source again until configured to do so in software. Software must check the state of GPIO<1> before configuring as a reset to ensure that no spurious reset is generated.

3.4.7 Summary of Module Reset Sensitivity

The registers and functions of most modules assume their default (reset) values when entering any of the five reset states. Table 3-2 describes which modules are reset for the various reset modes.

Table 3-2. Summary of Module Reset Functions

Module	Register Name or Function	Sleep-Exit	GPIO	Watchdog	Hardware	Power-On
OS Timers	All registers and functions	x[1]	x	x	x	x
PWR_I^{2}C	All registers and functions	x[2]	x	x	x	x
Clocks and Power Manager	PGSR0, PGSR1, PGSR2, PGSR3	x	x	x	x	x
	PVCR, PCMD	x[2]	x	x	x	x
	PSSR, PSPR, PWER, PRER, PFER, PEDR, PKWR, PKSR		x	x	x	x
	All bits except GP_ROD in PCFR		x	x	x	x
	GP_ROD bit in PCFR			x	x	x
	CCCR, CKEN, CCSR			x	x	x
	OSCC				x	x
RTC	RTTR				x	x
RTC	Other registers and functions	x	x	x	x	x
I/O Pins	See the Pin Usage table in the *Intel® PXA27x Processor Family EMTS*.		x	x	x	x

NOTES:

x: Register takes its reset value when the corresponding reset is asserted

x[1]: Register takes its reset value in during sleep- or deep-sleep-exit unless the pwr_I^{2}C island retains state, which occurs if PSLR[SL_PI] is set or nTRST is asserted before sleep or deep-sleep entry.

x[2]: Register takes its reset value in during sleep- or deep-sleep-exit unless the pwr_I^{2}C island retains state, which occurs if PCFR[PI^{2}C_EN] is set or nTRST is asserted before sleep or deep-sleep entry.

Any module not listed takes the reset value for all of its registers.

3.4.8 Summary of Reset Sequences

All of the processor resets follow similar sequences. Table 3-3 summarizes the sequence for each type of reset and the differences between them.

Table 3-3. Summary of Reset Sequences (Sheet 1 of 2)

Description of Action	Clock Source	Latency (Cycles)	Sleep-Exit	GPIO	Watchdog	Hardware	Power-On
Reset source asserted	-		x	x	x	x	x
Reset source synchronized to clocks/power manager state machine	13M	3		x			
Reset asynchronously distributed; nRESET_OUT asserted	-		x[1]	x[3]	x	x	x
All I/O, CPU, peripheral modules take reset state (except as noted in the Pin Usage table in the *Intel® PXA27x Processor Family EMTS* and text in this chapter). All clock sources are driven from the 13-MHz processor oscillator.	-		x	x	x	x	x
PLLs, 13-MHz processor oscillator disabled	-				x	x	x
13-MHz processor oscillator re-enabled, self-biases and stabilizes	13M	64k			x	x	x

Table 3-3. Summary of Reset Sequences (Sheet 2 of 2)

Description of Action	Clock Source	Latency (Cycles)	Sleep-Exit	GPIO	Watchdog	Hardware	Power-On
Core PLL (CPLL) and peripheral PLL (PPLL) enabled	13M	3	x	x[2]	x	x	x
Core PLL and peripheral PLL self-bias	-	<100 µs	x	x[2]	x	x	x
Core PLL and peripheral PLL stabilize	13M	512	x	x[2]	x	x	x
All clock sources switched to functional source (PPLL, CPLL, processor oscillator)	13M	3	x	x	x	x	x
Final clock stabilization	CPLL	512	x	x	x	x	x
nRESET_OUT deasserted, boot sequence begins; RCSR written	-		x[1]	x[3]	x	x	x

NOTES:

x: These steps apply if the corresponding reset has been asserted.

x[1]: If PSLR[SL_ROD] is set, do not assert nRESET_OUT.

x[2]: If PLL was disabled when GPIO reset was asserted, these steps apply.

x[3]: If PCFR[GP_ROD] is set, do not assert nRESET_OUT.

3.5 Clocks Manager Operation

The clocks manager contains all clock generation, gating, and frequency controls for the PXA27x processor. Clock generation and distribution consists of the following sources:

- 13-MHz processor oscillator—Creates the PLL reference clock and the functional clock for several units.
- 32.768-kHz timekeeping oscillator—Creates the low-power, low-frequency clock for timekeeping functions during low-power modes.
- Peripheral phase-locked loop (312 MHz)—Creates the fixed-frequency clocks for the peripheral bus and the peripheral units listed in Table 3-6.
- Core phase-locked loop (26–624 MHz)—Creates the programmable frequency clocks for the core, LCD controller, memory controller, and the system bus, although the core can run at 13 MHz in some modes.

Note: 625 MHz is available on the PXA270 processor only. See the *Intel® PXA270 Processor Electrical, Mechanical, and Thermal Specification* and *Intel® PXA27x Processor Family Electrical, Mechanical, and Thermal Specification* for supported frequency product points.

- Memory controller clock input—Sets the memory controller to run at the same frequency as the system bus. (CCCR[A] bit).
- Functional-unit clock gates—Enables and disables clocks to the functional units.

The clocks manager adjusts the frequencies of the internal clocks. Adjusting the frequency of the processor and peripheral system is one of the most effective methods of optimizing power consumption for an application's performance requirements. Applications might also require frequency changes for system reasons. The primary means of changing frequency are:

- Turbo/run selection—Changes the processor frequency relative to the system frequency. It is enabled by setting CLKCFG[T] (see Section 3.8.3.1).
- Fast-bus mode—Changes the ratio of the system bus to the CPU run-mode frequency from 1:2 (reset value) to 1:1. It is enabled by setting CLKCFG[B].

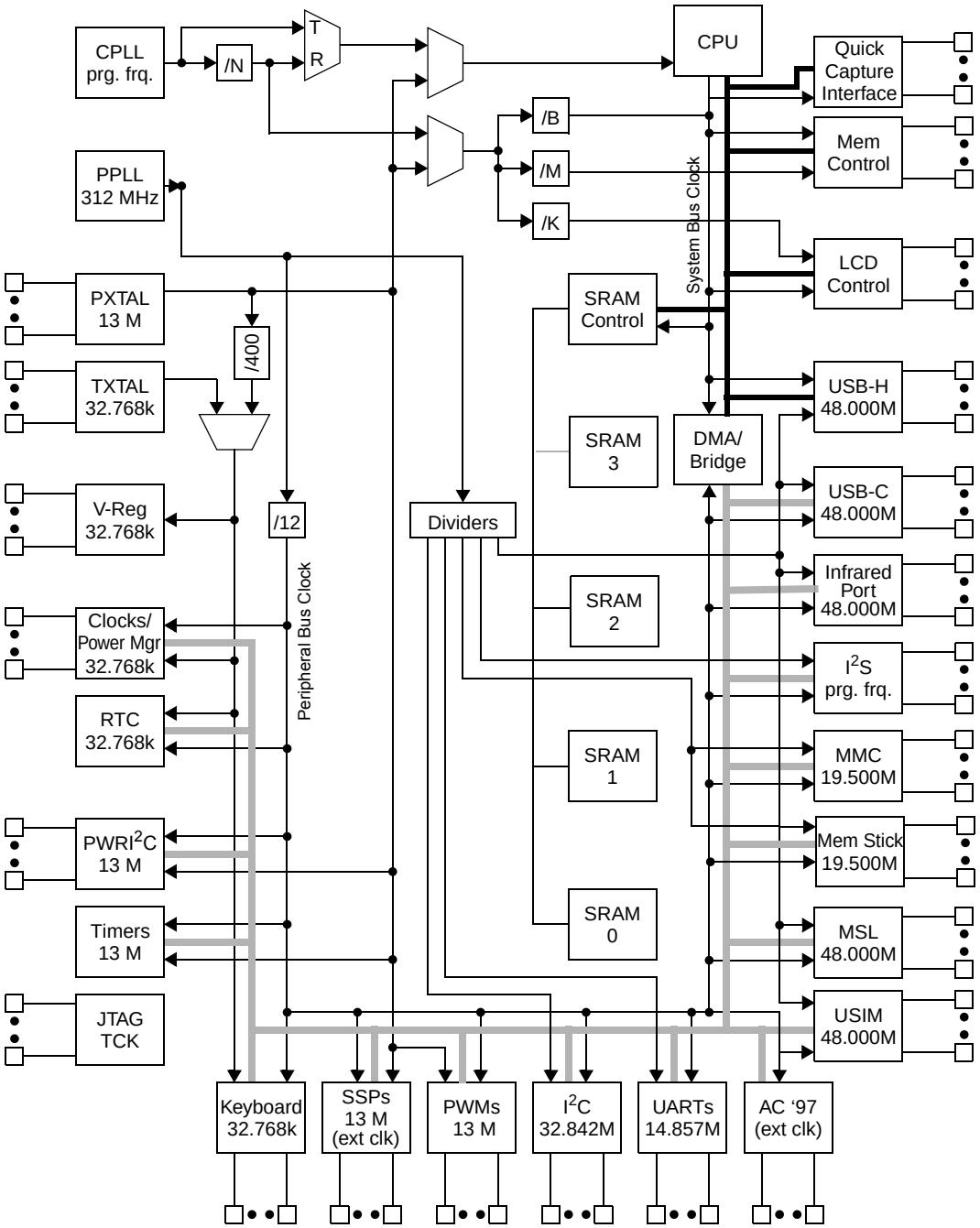
- Frequency change—Changes the frequency of the entire system to a new, unrelated frequency with some interruption in system operation. It is enabled by setting CLKCFG[F].

Note: Refer to Section 3.5.7 for more information.

Figure 3-1 illustrates the clock distribution within the processor. The diagram uses the following definitions, where L, A, and N are programmed in the Core Clock Configuration register (see Table 3-31) and B is programmed in the Clock Configuration register (see Section 3.8.3.1):

```
    Turbo-mode frequency (T) = 13-MHz processor-oscillator frequency * L * N

    Run-mode frequency (R) = 13-MHz processor-oscillator frequency * L

    System-bus frequency = 13-MHz processor-oscillator frequency * L / B,
        where B = 1 (when in fast-bus mode) or B = 2 (when not in fast-bus mode)

For CCCR[A] = 0 (see Table 3-7):

    Memory-controller frequency = 13-MHz processor-oscillator frequency * L / M,
        where M = 1 (L = 2-10), M = 2 (L = 11-20), or M = 4 (L = 21-31)

    LCD frequency = 13-MHz processor-oscillator frequency * L / K,
        where K = 1 (L = 2-7), K = 2 (L = 8-16), or K = 4 (L = 17-31)

For CLKCFG[B] = 0 and CCCR[A] = 1 (see Table 3-7):

    Memory-controller frequency = 13-MHz processor-oscillator frequency * L / 2

    LCD frequency = 13-MHz processor-oscillator frequency * L / K,
        where K = 1 (L = 2-7), K = 2 (L = 8-16), or K = 4 (L = 17-31)

For CLKCFG[B] = 1 and CCCR[A] = 1 (see Table 3-7):

    Memory-controller frequency = 13-MHz processor-oscillator frequency * L

    LCD frequency = 13-MHz processor-oscillator frequency * L / K,
        where K = 1 (L = 2-7), K = 2 (L = 8-16), or K = 4 (L = 17-31)
```

Figure 3-1. Clocks Manager and Clocks Distribution Block Diagram



NOTE: Denotes system bus
Denotes peripheral bus

3.5.1 External Clock Source Selection (CLK_REQ)

CLK_REQ is an input during power-on or hardware reset that sets or clears OSCC[CRI] (see Section 3.8.2.3). Table 3-4 summarizes the external clock selection as a function of OSCC[CRI], OSCC[OON], and CLK_REQ.

If CLK_REQ is driven low during power-on or hardware reset, OSCC[CRI] is cleared, and:

- The 13-MHz processor oscillator input is taken from PXTAL_IN.
- The 32.768-kHz timekeeping oscillator is software-enabled.
- CLK_PIO can be used as an output.
- CLK_REQ is an input.

If CLK_REQ is floated during power-on or hardware reset, OSCC[CRI] is set, and:

- The 13-MHz processor oscillator input is taken from the CLK_PIO pin.
- The 32.768-kHz timekeeping oscillator is enabled and driven out onto CLK_TOUT.
- PXTAL_IN must be grounded.
- CLK_REQ is an output indicating that the processor oscillator is requested from the external source.

If OSCC[CRI] is set, CLK_REQ is driven high after nRESET is deasserted until the processor enters deep-sleep mode. During deep-sleep mode, CLK_REQ is driven low. After leaving deep-sleep mode, CLK_REQ is not resampled.

Note: Do not drive CLK_REQ high externally, since the processor drives it low during deep-sleep mode.

If OSCC[OON] and OSCC[TOUT_EN] are set, CLK_TOUT does not start until the 32.768-kHz timekeeping oscillator is stable, which can take up to two seconds after power-on or hardware reset is asserted. For more information, see Section 3.5.2.

Refer to the *Intel® PXA27x Processor Family EMTS* for timing specifications.

Table 3-4. External Clock Source Selection

CLK_REQ†	OSCC[CRI]	OSCC[OON]	32.768-kHz Source	13-MHz Source
float	1	1	TXTAL_IN	CLK_PIO
0	0	0	Internally generated as PXTAL_IN / 400 = 32.5 kHz	PXTAL_IN
0	0	1††	TXTAL_IN	PXTAL_IN

NOTES:
† During power-on or hardware reset
†† Software-programmed

3.5.2 13-MHz Processor Oscillator

The 13-MHz processor oscillator provides the primary clock source for the PXA27x processor. The on-chip PLL frequency multipliers and several peripheral modules use the processor oscillator as a reference. If the application has not enabled the 32.768-kHz timekeeping oscillator, the processor oscillator also drives the real-time clock (RTC) and power manager.

The processor oscillator can be disabled during standby, sleep, and deep-sleep modes by setting PCFR[OPDE] (see Section 3.8.1.8), but only if the timekeeping oscillator is enabled and stabilized (OSCC[OOK] is set—see Section 3.8.2.3). If the timekeeping oscillator has not stabilized, the processor oscillator remains enabled for the duration of the low-power mode.

The processor oscillator can also serve as the clock source for the core, system bus, memory controller, and LCD controller by setting CCCR[CPDIS]. The peripheral PLL can be enabled by clearing CCCR[PPDIS] bit. The PPDIS bit can be independently set or cleared, regardless of the CPDIS setting. In this mode, the system bus frequency is 13 MHz regardless of CLKCFG[B] (see Section 3.8.3.1). When selected, the processor oscillator is used with the same programming model as the core PLL, with the exception that the programming values are fixed, as shown in Table 3-5.

Table 3-5. Processor Oscillator Output Frequencies for 13-MHz Crystal

Run Mode Frequency (MHz)	System Bus Frequency B = 1 (MHz)	System Bus Frequency B = 0 (MHz)	CLK_MEM Frequency A = 1 (MHz)	CLK_MEM Frequency A = 0 (MHz)	LCD Frequency (MHz)
13	13	13	13	13	13 or 26†

NOTE:
† CCCR[LCD_26] determines the LCD frequency.

The 13-MHz processor oscillator input can be generated by using either:

- An external crystal between the PXTAL_IN and PXTAL_OUT pins (lowest power consumption)
- An external clock source on the PXTAL_IN pin, or
- An external clock source on the CLK_PIO pin.

When driving the PXTAL_IN pin, observe the following precautions:

- OSCC[CRI] must be clear (see Section 3.8.2.3), and the CLK_REQ pin must be driven low during power-on and hardware reset.
- For best results, drive PXTAL_IN and PXTAL_OUT with complementary, terminated signals. Noise causes performance degradation if PXTAL_OUT is floated.
- Drive PXTAL_IN and PXTAL_OUT with a VSS-to-VCC_PLL rail-to-rail signal. If more than 10% overshoot or undershoot occurs, use a 1.5-kΩ series resistor or a voltage divider to reduce the level of VSS-to-VCC_PLL.

When driving the CLK_PIO pin, observe the following precautions:

- OSCC[CRI] must be set (the CLK_REQ pin must be floated during power-on and hardware reset).
- Drive CLK_PIO with a VSS-to-VCC_IO rail-to-rail signal. If more than 10% overshoot or undershoot occurs, use a 1.5-kΩ series resistor or a voltage divider to reduce the level of VSS-to-VCC_IO.
- Ground the PXTAL_IN pin, and leave the PXTAL_OUT pin floating.

If OSCC[CRI] is clear, setting OSCC[PIO_EN] drives a buffered version of the PXTAL_IN signal out to the external system on the CLK_PIO pin.

Note: The two pairs of crystal pins are located close to each other on the processor package. This arrangement is advantageous when there are crystals connected to the pins because the low

amplitudes and slow slew rates reduce noise coupling between the pins. If one of the crystals is replaced by an external signal source and the other is not, some degradation of the remaining crystal oscillator can result due to increased noise coupling. This effect can be reduced by limiting the slew rate on the pin(s) driven by the external source.

3.5.3 32.768-kHz Timekeeping Oscillator

The 32.768-kHz timekeeping oscillator is a low-power, low-frequency oscillator that clocks the real-time clock (RTC) and power manager. If OSCC[CRI] is set, the timekeeping oscillator is always enabled, and the CLK_TOUT pin drives a buffered version of the TXTAL_IN signal after the deassertion of nRESET (for a period of up to 2.048 seconds (64,000 x 32-kHz clock cycles), this buffered signal might not be present. This is due to the unknown stabilization time of the externally supplied clock source). This output can then be disabled by clearing OSCC[TOUT_EN].

If OSCC[CRI] is clear, the timekeeping oscillator and CLK_TOUT are disabled upon exit from power-on reset or hardware reset, and the RTC and power manager blocks use the 13-MHz processor oscillator divided by 400. Setting OSCC[OON] enables the timekeeping oscillator and configures the RTC and power manager to use the timekeeping oscillator after it has stabilized. CLK_TOUT is enabled by setting OSCC[TOUT_EN]. Doing so eliminates the need for the external (TXTAL) crystal oscillator, for cost savings in less power-sensitive applications.

To use the timekeeping oscillator and CLK_TOUT function, follow these steps:

1. Enable the timekeeping oscillator by setting OSCC[OON]. It is automatically set if OSCC[CRI] is set.

Note: OSCC[OON] cannot be cleared once it has been set, and the timekeeping oscillator cannot be disabled until a power-on or hardware reset occurs.

2. Wait for OSCC[OOK] to be automatically set. Do not attempt to enter standby, sleep, or deep-sleep modes with PCFR[OPDE] set until OSCC[OOK] is set. Otherwise, the 13-MHz processor oscillator is used and is not powered off.
3. Enable or disable the CLK_TOUT output as needed by setting or clearing OSCC[TOUT_EN], respectively. If OSCC[CRI] is set, OSCC[TOUT_EN] is set automatically after power-on reset or hardware reset.

For lowest power consumption, connect a 32.768-kHz crystal between the TXTAL_IN and TXTAL_OUT pins. However, some systems might have other clock sources of the same frequency, so the overall system cost is reduced by driving the TXTAL crystal pins with one of those clock sources.

When driving the TXTAL_IN and TXTAL_OUT pins, observe the following precautions:

- For best results, drive TXTAL_IN and TXTAL_OUT with complementary, terminated signals. Noise causes performance degradation if TXTAL_OUT is floated.
- Drive TXTAL_IN and TXTAL_OUT with VSS-to-VCC_PLL rail-to-rail signals. If more than 10% overshoot or undershoot is experienced, use a 1.5-kΩ series resistor or a voltage divider to reduce the level to VSS-to-VCC_PLL.

3.5.4 Peripheral Phase-Locked Loop (312 MHz)

When the CCCR[PPDIS] bit is clear (see Section 3.8.2.1), the peripheral PLL generates a fixed-frequency clock source (312 MHz) that is used in several peripherals, as shown in Table 3-6.

To save power, the peripheral PLL can be disabled by initiating a frequency change with CCCR[PPDIS] set. The frequencies of the peripheral clocks then derive from the 13-MHz processor oscillator. However, doing so causes those peripheral modules with strict bandwidth or protocol-related frequency requirements not to work. When the peripheral PLL is re-enabled, the exit sequence from the frequency change lengthens for up to 150 µs.

The generated PLL frequencies, listed in Table 3-6, are not exact, because of the choice of crystal frequency and the lack of a least common multiple between peripheral units. The 13-MHz crystal maintains the clock frequency for each unit within the unit's clock tolerance.

Note: The following configuration is not supported:

— CCCR[CPDIS] = 0 and CCCR[PPDIS] = 1

Table 3-6. Peripheral PLL Output Frequencies for 13-MHz Crystal

Units	Divide Ratio	Required Frequency (MHz)	Actual Frequency (MHz)	Systemic Error Due to Divide (%)	Total Error with PLL/ Osc Jitter (%)	Frequency, PPDIS = 1 (MHz)
Peripheral Bus	24/2	26.000	26.000	0.000	±0.025	13
USB-H, USB-C, Infrared Port, USIM	13/2	48.000	48.000	0.000	±0.025	13
MSL	13/2	48.000	48.000	0.000	±0.025	13
I^2C	19/2	33.333	32.842	–1.474	–1.500 to –1.450	13
MMC	32/2	20.000	19.500	–0.500	–2.525 to –2.475	13
UARTs	42/2	14.746	14.857	+0.754	+0.725 to +0.780	13
I^2S (48.000k)	51/2	12.288	12.235	–0.429	–0.455 to –0.400	13
I^2S (44.100k)	55/2	11.290	11.346	+0.493	+0.465 to +0.520	13
I^2S (22.050k)	(37*3)/2	5.645	5.622	–0.411	–0.440 to –0.385	13
I^2S (16.000k)	(38*4)/2	4.096	4.105	+0.226	+0.200 to +0.255	13
I^2S (11.025k)	(37*6)/2	2.822	2.811	–0.411	–0.440 to –0.385	13
I^2S (8.000k)	(38*8)/2	2.048	2.053	+0.226	+0.200 to +0.255	13
SD/SDIO (8.000k)	32/2	25.000	19.500	–5.5	–2.525 to –2.475	13

3.5.5 Core Phase-Locked Loop (Programmable)

When the CCCR[CPDIS] bit is clear (see Section 3.8.2.1), the core PLL serves as the clock source for the core, system bus, memory controller, and LCD controller. The core PLL generates three output frequencies:

- Run-mode frequency = 13-MHz processor oscillator * L
- Turbo-mode frequency = run-mode frequency * N
- Memory/LCD controller frequencies = predefined divisor of run-mode frequency based on L, where Table 3-7 gives the values of L and N along with the output frequencies. See Section 3.8.2.1 for information on programming the L and N factors.

Note: Do not exceed the maximum specified frequency for the applied VCC_CORE voltage. Observe the T_{CASE} specification. See the *Intel® PXA27x Processor Family EMTS* for details.

To reduce power, the core PLL can be disabled by performing a frequency change with CCCR[CPDIS] set. Refer to Table 3-5 for details. Setting the PLL early enable bit, CCCR[PLL_EARLY_EN], prior to a frequency change reduces the frequency change time. Refer to the *Intel® PXA27x Processor Family EMTS* for timing details.

Note: The following configuration is not supported:

— CCCR[CPDIS] = 0 and CCCR[PPDIS] = 1

Table 3-7. Clock Frequencies

Core Run Freq (MHz)	CLKCFG[T]	Core Turbo Freq (MHz)	CLKCFG[T]	CLKCFG[HT]	CCCR[L]	CCCR[2N]	System Bus (MHz)	CLKCFG[B]	CLK_MEM (MHz)	CCCR[A]	SDCLK<2:1> SDRAM Clocks (MHz)	MDREFR[KxDB2]††††	Synchronous Flash (MHz)	MDREFR[K0DB4]	MDREFR[K0DB2]	LCD (MHz)
13†	X	—	X	X	X	X	13	X	13	X	13	X	13	X	X	13 or 26††
91†††	0	—	—	0	7	2	45	0	91	0	45	1	22.5	1	1	91
104	0	104	1	0	8	2	104	1	104	1	104	0	52	0	1	52
156	0	156	1	1	8	6	104	1	104	1	104	0	52	0	1	52
104	0	312	1	0	8	6	104	1	104	1	104	0	52	0	1	52
208	0	208	1	0	16	2	208	1	208	1	104	1	52	1	X	104
208	0	312	1	0	16	3	208	1	208	1	104	1	52	1	X	104
208	0	416	1	0	16	4	208	1	208	1	104	1	52	1	X	104
208	0	520	1	0	16	5	208	1	208	1	104	1	52	1	X	104
208	0	624†††††	1	0	16	6	208	1	208	1	104	1	52	1	X	104

NOTES:

† Not a PLL clock frequency. Refer to Section 3.5.7.7.

†† Use CCCR[LCD_26] to control this setting. See Table 3-31.

††† L = 7 (Core = 91.0 MHz) is used for hardware boot-up frequency only and must not be used for normal operation.

†††† KxDB2 represents K1DB2 and K2DB2

††††† 624 MHz is available on the PXA270 processor only. See the *Intel® PXA270 Processor Electrical, Mechanical, and Thermal Specification* and *Intel® PXA27x Processor Family Electrical, Mechanical, and Thermal Specification* for supported frequency product points.

3.5.6 Functional-Unit Clock Gating

Functional-unit clock gating allows software to enable and disable the clock to individual modules (peripherals) within the processor. This provides an advantage for low-power system designs.

The Clock Enable register (CKEN—see Section 3.8.2.2) enables and disables the clocks to individual modules. The CKEN register values override the clock-gating functionality that might be present in some modules. These values are set only when entire modules are not being used. After power-on reset or hardware reset, software must disable the clocks to inactive modules.

If a module is temporarily quiescent but does not have clock gating functionality, use CKEN to disable the module's clock. With its clock disabled, register reads from that module return undefined values, and register writes to that module have no effect.

3.5.7 Modifying Clock Frequencies

3.5.7.1 General Procedure for Clock-Frequency Changes

Note: In this document, the term *CPU clock* refers to both the Intel XScale® microarchitecture and the Wireless MMX coprocessor clocks.

Modifying any of the processor clock frequencies requires programming two registers, in the following order:

1. The Core Clock Configuration register (CCCR—see Section 3.8.2.1) contains the clock configuration information. For details of the frequency changes, see the following sections:

   Section 3.5.7.3 — Changing Core Frequency

   Section 3.5.7.4 — Turbo Mode

   Section 3.5.7.6 — Fast-Bus Mode

2. Coprocessor 14, register C6 (CLKCFG—see Section 3.8.3.1) initiates the changes programmed in CCCR when any of the following bits are written to CLKCFG. It is legal to change the settings of any of these bits when writing to CLKCFG, in which case all of the changes in CCCR are implemented.

   CLKCFG[B]—Fast-Bus Mode. When B is set, the system-bus frequency is equal to the run-mode frequency indicated in CCCR. When B is clear, the system-bus frequency is equal to half the run-mode frequency indicated in the CCCR. It is illegal to set B if CCCR[CPDIS] is set. When B is modified, the core PLL is stopped, and then restarted with the new CCCR settings. See Section 3.5.7.6 for details.

   CLKCFG[F] —Core Frequency Change. When F is set and CCCR[xPDIS] are clear, the core PLL is stopped, and then restarted with the new CCCR settings. If the core PLL is disabled (CCCR[CPDIS] set), the PLL is not restarted.

   F remains set after the frequency change, and there is no need to clear it. Clearing F does not initiate a frequency change. To initiate a new frequency change, set F again. See Section 3.5.7.3 for details.

   CLKCFG[T]—Turbo Mode. When T is set, the CPU operates at the turbo frequency; when clear, the CPU operates at the run-mode frequency. If only T is set, F is clear, and B is not altered, then the core PLL is not stopped. See Section 3.5.7.4 for details.

   CLKCFG[HT]—Half-Turbo Mode. When HT is set, whether T is set or clear, the CPU operates at the turbo frequency divided by two; when HT is clear, and T is clear, the CPU operates at the run-mode frequency; when HT is clear, and T is set, the CPU operates at the turbo frequency. If only HT is set, F is clear, and B is not altered, then the core PLL is not stopped. See Section 3.5.7.4 for details.

3.5.7.2 Special Considerations for Clock-Frequency Changes

Changing a clock frequency generally incurs the following stoppages, latencies, and special requirements (except half-turbo and turbo mode changes). Table 3-8 summarizes the latencies in terms of clock cycles.

- All interrupts to the CPU are held, causing latencies for the peripherals.

- All current instructions, including incomplete fetches, are completed.
- All outstanding stores are completed.
- The CPU clock stops. The stoppage time incurred by each type of frequency change is given in the “Preparations” subsection for the specific change.
- Depending on the requested change sequence, the system-bus clock can be halted.
- Depending on system usage, some peripherals might need reconfiguration to account for the new frequencies (depending on system configuration and peripheral unit usage).

Note: Each of the clock-frequency change sequences can impose additional limitations and requirements. For details of a specific type of clock-frequency change, see the subsections below.

Note: When a write to CLKCFG initiates more than one clock-frequency change at the same time,

- The total latency time is the maximum required by any of the changes initiated. In other words, the change sequences run concurrently.
- Preparations for **all** requested changes must be completed before writing to CLKCFG. See the corresponding “Preparations” section for each type of frequency change in the following subsections.

Table 3-8 summarizes the clock-frequency change sequences. The total latency for each sequence is the sum of the latencies at each step.

Table 3-8. Summary of Clock-Frequency Change Sequences

Description of Action	Unit	Latency (cycles)	Type of Change		
			Turbo	Fast Bus	Core Freq.
Write to CP14 CLKCFG register (software); interrupts are held.	CPU	1 CPU	X	X	X
All current instructions, including incomplete fetches, are completed.	CPU	? CPU	X	X	X
All outstanding stores are completed.	CPU	? SB	X	X	X
Wait for synchronization. The CPU clock halts.	CPU	<20 CPU	X	X	X
Deny all bus requests from LCD, USB-H, DMA, CPU, and memory controllers.	PM	1 SB	—	X	X
Complete all memory control transactions (allow memory controller bus requests). Place SDRAM in self-refresh mode.	MEM	? SB	—	—	X
Synchronize clocks and power manager. Switch clock sources (done in hardware).	PM	<4 13M <8 SB	—	—	X
Core-frequency change sequence splits					
Disable PLLs if appropriate (xPDIS set).	PM	2 13M	—	—	X
Enable PLLs if appropriate (xPDIS clear). Wait for PLLs to lock.	PM	2000 13M	—	—	X
Synchronize clocks and power manager.	PM	<2 13M <4 SB	—	—	X
Core-frequency change sequence merges					
Set PLL and/or SB dividers.	PM	2 SB	—	X	X
Release system bus for all transactions.	PM	1 SB	—	X	X
Wait for synchronization.	PM	<20 CPU	X	X	X
Enable clocks, enable interrupts to the CPU, and begin execution.	PM	2 SB	X	X	X

NOTES:

X Step is followed in the corresponding clock mode.
— Not applicable
? Variable—depends on system/software configuration.
13M 13-MHz processor oscillator
CPU CPU
MEM Memory controller
PM Power manager
SB System bus

3.5.7.3 Changing Core Frequency

The core clock frequency can be changed in several ways:

- Selecting the 13-MHz clock source
- Changing the core PLL frequency
- Enabling turbo or half-turbo mode

The following parameters in the Core Clock Configuration register (CCCR) control the core clock frequency (see Section 3.8.2.1 for register details):

- The run-mode-to-oscillator ratio bit, CCCR[L], determines the run frequency by multiplying the external crystal oscillator input by L.
- The core PLL disable bit, CCCR[CPDIS=1], turns off the core PLL to reduce power consumption. The peripheral PLL can remain enabled if required (CCCR[PPDIS=0]). If the core PLL is disabled, all internal processor core units derive their clocks from the 13-MHz processor oscillator. Disabling the peripheral PLL, CCCR[PPDIS=0], forces all peripherals to derive their clocks from the 13-MHz processor oscillator. The PLLs can be independently enabled or disabled by the CCCR[xPDIS] bits.
- The turbo mode bit, CLKCFG[T], selects either turbo mode or run mode.
- The turbo-mode-to-run-mode ratio bit, CCCR[2N], determines the turbo frequency by multiplying the run frequency by N.
- The half-turbo mode bit, CLKCFG[HT], selects whether the core frequency is equal to the run or turbo frequency (dependent on the setting of CLKCFG[T]).

Table 3-7 summarizes the core PLL frequencies as functions of L and N.

3.5.7.3.1 Preparations for Core-Frequency Change

Before performing a core-frequency change, review Section 3.5.7.1 and Section 3.5.7.2.

Table 3-8 summarizes the specific steps that take place during a core-frequency change. The frequency-change sequence imposes the following limitations and requirements:

- All interrupts to the CPU are held, causing latencies for the peripherals.
- All current instructions, including incomplete fetches, are completed.
- All outstanding memory-controller transactions are completed.
- The system-bus clock stops for up to 150 µs. No new system-bus transactions are allowed during the change sequence. During this time,
  - — The LCD controller cannot transmit data to the LCD panel. If the panel cannot tolerate the 150-µs latency (for example, if it does not have a built-in frame buffer), disable the LCD controller before initiating the core-frequency change.
  - — The USB host controller cannot access its own FIFOs or configuration registers. This might require disabling the USB host controller before initiating the core-frequency change.
- SDRAMs are automatically placed in self-refresh mode.
- The CPU clock stops for up to 150 µs.

The suspension of DMA activity and CPU interrupt service can cause overrun or underrun in some peripheral modules. If a peripheral module cannot tolerate the 150-µs system-bus clock latency, disable the module **before** initiating the core-frequency change.

Note: If the PLLs are to be turned off using the xPDIS bits, then set the xPDIS bits **before** the frequency change, and clear the xPDIS bits **after** the frequency change.

Note: For best results, set both of the PMCR[xIDAE] bits in the Power Manager Control register while changing the core frequency (this allows software controlled entry into deep-sleep, if nBATT_FAULT or nVDD_FAULT asserts, rather than immediate entry to deep-sleep) (see Section 3.8.1.1 for register details). If the PLLs are disabled by setting the xPDIS bits, and then a standby mode entry is initiated, the CCCR[xPDIS] bits must not be altered until after exit from standby mode.

Table 3-9 summarizes the action required for each module when changing the core frequency.

Table 3-9. Required Actions Before and After Core-Frequency Change

Module	Before PLL Frequency Change (Core PLL = Clock Source)	After PLL Frequency Change (Core PLL = Clock Source)
Memory Controller	—	Reconfigure for new clock speed.
LCD Controller Quick Capture Interface	Disable unless LCD panel is tolerant of 150-µs pause in data transmittal.	Reconfigure for new clock speed and re-enable.
USB Host	Disable if intolerant of 150-µs system-bus clock stoppage and DMA/interrupt latency.	—
USB Client Infrared Port MSL I^2C Bus Interface Unit MMC UARTs AC'97 I^2S SSP Serial Ports	Disable if intolerant of 150-µs DMA/interrupt latency.	—
Keypad OS Timers RTC	Interrupts held until completion of frequency change.	—

3.5.7.3.2 Initiating Core-Frequency Change

To initiate a frequency-change operation:

1. Write the desired values for L, N, PPDIS, and CPDIS to CCCR and T and HT to CLKCFG.
2. Set CLKCFG[F]. When this write occurs, the new core-frequency change sequence begins and the values in CCCR are applied.

Note: Do not set CLKCFG[HT] while performing a frequency change.

3.5.7.3.3 Behavior During Core-Frequency Change

While the core-frequency change sequence is executing, the following occurs before the switch in CPU clock speed is made:

1. All processor activity is stopped and all interrupt requests to the processor are held.

2. All new DMA and LCD activity is suspended.
3. All CPU loads are completed and CPU stores are sent to the system bus.
4. All outstanding memory-controller transactions are completed.
5. The memory controller places the SDRAM in self-refresh mode and drives the nRAS/nSDCS<3:0> and nCAS/DQM<3:0> pins to their self-refresh state.
6. The system bus clocks are stopped for a period of at least four cycles of the system bus clock (at the new frequency).

The DMA controller and CPU experience a stoppage of the system bus clock for up to four system-bus cycles (during a clock-source change) or up to 150 µs (during a core PLL frequency change with the core PLL selected as the clock source), plus the time it takes the memory controller to finish its outstanding transactions. The total period of DMA and CPU inactivity depends on the source/destination of memory controller transactions and the type of frequency change. The lack of DMA service means that peripheral modules can experience overrun or underrun in their FIFOs, and the lack of CPU activity means increased interrupt latency.

If a power fault (nVDD_FAULT or nBATT_FAULT) is asserted during the period of CPU inactivity, the frequency-change sequence completes first. Then, the processor enters deep-sleep mode as described in Section 3.6.4.

3.5.7.3.4 Completion of Core-Frequency Change Sequence

After the clock speed has been changed, the CPU continues execution at the next instruction after the write to CLKCFG. interrupt requests are no longer held, and any interrupts that occurred during the change sequence are sent to the CPU.

CLKCFG remains in the state that was written to it. The values in it and the Clock Configuration Status register (CCSR—see Section 3.8.2.4) provide data about the operating frequencies of the core PLL, which determines the frequencies of the CPU, LCD controller, memory controller, and system bus.

The SDRAMs are automatically brought out of self-refresh mode. Before enabling external transactions to devices with frequency-dependent configurations, reconfigure the memory controller to account for the new clock frequency. For more information on memory configuration, see Chapter 6, “Memory Controller”.

If required, reconfigure the LCD controller’s pixel clock to account for the new frequency. If the LCD controller was disabled before the frequency change, re-enable it. For more information on LCD configuration, see Chapter 7, “LCD Controller”.

3.5.7.4 Turbo Mode

The processor CPU frequency depends on the setting of the T bit in the Clock Configuration register (CLKCFG—see Section 3.8.3.1):

- CLKCFG[T] set—CPU operates at the turbo frequency.
- CLKCFG[T] clear—CPU operates at the run frequency.

The turbo-mode and run-mode frequencies depend on the values in the Core Clock Configuration register (CCCR—see Section 3.8.2.1 for register details). Table 3-7 lists the turbo-mode frequencies.

The increased latency for entering or exiting turbo-mode is an interruption in execution while current instructions and loads are completed and stores are sent to the system bus. This latency varies depending on the number and destination and source of the stores and loads, respectively, as well as other bus activity and cacheability of the interrupt handler. Interrupt requests are held until the frequency is changed, resulting in longer and less predictable interrupt latency. The increased latency is greater when performing a frequency change (F bit) or fast-bus change (B bit) in the same write as the T bit in the CLKCFG register.

Also refer to Section 3.5.7.5 for half-turbo mode details.

3.5.7.4.1 Preparations for Turbo-Mode Entry and Exit

While entering or exiting turbo mode, the CPU clock halts, and interrupt requests to the CPU are held for up to eight core clock cycles.

The value of N is the ratio of the *turbo-mode-frequency* to the *run-mode frequency*. N is determined by CCCR[2N]. This value must have been loaded beforehand into the core PLL by means of a core-frequency change (see Section 3.5.7.3) and must be reflected in the Core Clock Status register (CCSR—see Section 3.8.2.4). The value reflected in CCSR[2N_S] is the value that is actually used when CLKCFG[T] is set.

Because entering or exiting turbo mode does not stop the system-bus clocks and does not alter any of the peripheral clocks (including the LCD and memory controller), no special steps are required with respect to the peripherals or memory controller.

3.5.7.4.2 Initiating Turbo-Mode Change

To enter or exit turbo mode, follow these steps:

1. Complete the preparations described in Section 3.5.7.4.1.
2. Set or clear CLKCFG[T]. When this write occurs, the CPU frequency switches to the indicated mode (turbo or normal run).

Note: If CCSR[2N_S] has the value of 0x2, the turbo-mode change sequence takes place, but the core frequency does not change.

3.5.7.4.3 Behavior During Turbo-Mode Change

While the processor is entering or exiting turbo mode, all processor activity is stopped and all interrupt requests are held. All loads initiated by the CPU are completed, and all stores are sent to the system bus before the switch in CPU clock speed is made. Peripheral and memory-controller activity continues without interruption or change in behavior.

A possible issue during the entry or exit from turbo mode is the additional interrupt latency caused by holding interrupt requests while all CPU loads are completed and CPU stores are sent to the system bus.

If a power fault (nVDD_FAULT or nBATT_FAULT) is asserted during the period of CPU inactivity, the turbo-mode change sequence completes first. Then, the processor enters deep-sleep mode as described in Section 3.6.4.

3.5.7.4.4 Completion of Turbo-Mode Change Sequence

After the clock speed has been changed, the CPU continues execution at the next instruction after the write to CLKCFG. interrupt requests are no longer held, and any interrupts that occurred during the change sequence are sent to the CPU.

CLKCFG remains in the state that was written to it. The values in it and the Clock Configuration Status register provide information about the current operating frequencies of the CPU, LCD controller, memory controller, and system bus.

3.5.7.5 Half-Turbo Mode

The processor CPU frequency depends on the setting of the HT bit in the Clock Configuration register (CLKCFG—see Section 3.8.3.1):

- CLKCFG[HT] set—the CPU operates at the turbo-mode frequency divided by two.
- CLKCFG[HT] clear—the CPU operates at the run frequency if the T bit is clear or at the turbo frequency if the T bit is set.

Note: Half-turbo mode can only be invoked when the CCSR reflects 2*N values of 6 or 8.

The half-turbo-mode and run-mode frequencies depend on the values in the Core Clock Configuration register (CCCR—see Section 3.8.2.1 for register details). Table 3-7 lists the turbo-mode frequencies.

The increased latency for entering or exiting half-turbo mode is an interruption in execution while current instructions and loads are completed and stores are sent to the system bus. This latency varies depending on the number and destination and source of the stores and loads, respectively, as well as other bus activity and cacheability of the interrupt handler. Interrupt requests are held until the frequency is changed, resulting in longer and less predictable interrupt latency. Do not set the HT bit in CLKCFG while performing a core PLL frequency change. After a frequency change has been performed, writing to the T bit and the HT bit at the same time results in the CPU frequency at the turbo-mode frequency divided by two.

3.5.7.5.1 Preparations for Half-Turbo Mode Entry and Exit

While entering or exiting half-turbo mode, the CPU clock halts, and interrupt requests to the CPU are held for up to eight core clock cycles.

The value of N is the ratio of the *turbo-mode-frequency* to the *run-mode frequency*. N is determined by CCCR[2N]. This value must have been loaded beforehand into the core PLL by means of a core-frequency change (see Section 3.5.7.3) and must be reflected in the Core Clock Status register (CCSR—see Section 3.8.2.4). The value reflected in CCSR[2N_S] is the value that is actually used when CLKCFG[HT] is set.

Because entering or exiting half-turbo mode does not stop the system-bus clocks and does not alter any of the peripheral clocks (including the LCD and memory controller), no special steps are required with respect to the peripherals or memory controller.

3.5.7.5.2 Initiating Half-Turbo Mode Change

To enter or exit half-turbo mode, follow these steps:

1. Complete the preparations described in Section 3.5.7.4.1.
2. Set or clear CLKCFG[HT] as follows:
   a. If CLKCFG[HT] is set, the CPU frequency switches to the half-turbo mode
   b. if CLKCFG[HT] is clear and T is clear, the CPU frequency is normal run-mode frequency
   c. if CLKCFG[HT] is clear and T is set, the CPU frequency is normal turbo frequency.

3.5.7.5.3 Behavior During a Half-Turbo Mode Change

While the processor is entering or exiting half-turbo mode, all processor activity is stopped and all interrupt requests are held. All loads initiated by the CPU are completed and all stores are sent to the system bus before the switch in CPU clock speed is made. Peripheral and memory-controller activity continues without interruption or change in behavior.

A possible issue during the entry or exit from half-turbo mode is the additional interrupt latency caused by holding interrupt requests while all CPU loads are completed and CPU stores are sent to the system bus.

If a power fault (nVDD_FAULT or nBATT_FAULT) gets asserted during the period of CPU inactivity, the turbo-mode change sequence completes first. Then, the processor enters deep-sleep mode as described in Section 3.6.4.

3.5.7.5.4 Completion of Half-Turbo Mode Change Sequence

After the clock speed has been changed, the CPU continues execution at the next instruction after the write to CLKCFG. interrupt requests are no longer held, and any interrupts that occurred during the change sequence are sent to the CPU.

CLKCFG remains in the state that was written to it. The values in it and the Clock Configuration Status register provide information about the current operating frequencies of the CPU, LCD controller, memory controller, and system bus.

3.5.7.6 Fast-Bus Mode

The processor system-bus frequency depends on the setting of the B bit in the Clock Configuration register (CLKCFG—see Section 3.8.3.1):

- CLKCFG[B] set—System bus operates at the full run-mode frequency.
- CLKCFG[B] clear—System bus operates at one-half the run-mode frequency.

The system-bus frequency is relative to the run-mode frequency indicated in the Core Clock Configuration register (CCCR—see Section 3.8.2.1 for register details and Table 3-7 for a summary of the frequencies).

Certain values of CCCR[2N] are illegal when CLKCFG[B] is set. For details, see Table 3-7.

The increased latency for a fast-bus mode change is a stoppage of LCD, memory controller, system bus and core clocks, lasting up to 150 µs. Additionally, an interruption in execution occurs while all outstanding processor or memory controller transactions are completed. Finally, some re-configuration of the LCD controller and memory controller might be required. Interrupt requests to the CPU are held until the new frequency is enacted, resulting in longer and less predictable interrupt latency.

3.5.7.6.1 Preparations for Fast-Bus-Mode Entry and Exit

Entering and exiting fast-bus mode incurs the same latencies and requires the same precautions as for a core frequency change. Thus, to prepare for a fast-bus mode change, follow the instructions in Section 3.5.7.3.1.

3.5.7.6.2 Initiating Fast-Bus Mode Change

To enter or exit fast-bus mode, follow these steps:

1. Complete the preparations described in Section 3.5.7.3.1.
2. Set or clear CLKCFG[B]. When this write occurs, the system-bus frequency switches to the indicated mode relative to the CPU run-mode frequency.

3.5.7.6.3 Behavior During Fast-Bus Mode Change

System behavior during the fast-bus-mode change sequence is identical to the behavior during a core frequency change, as described in Section 3.5.7.3.3.

3.5.7.6.4 Completion of Fast-Bus-Mode Change Sequence

After the system-bus speed has been changed, the peripheral continues execution at the next instruction after the write to CLKCFG. interrupt requests are no longer held, and any that occurred during the change sequence are sent to the peripheral.

CLKCFG remains in the state that was written to it. The values in it and the Clock Configuration Status register (CCSR—see Section 3.8.2.4) provide data about the operating frequencies of the peripheral, LCD controller, memory controller, and system bus.

3.5.7.7 13M Mode

When the CPDIS is set and the PPDIS is clear and a frequency change operation is performed by writing to the F bit in the CLKCFG register, the processor enters 13M mode. In this mode (CPDIS set and PPDIS clear), only the CPU core PLL is turned off, forcing all of the internal clocks to be derived from the 13-MHz oscillator. However, the peripheral PLL is still enabled and continues to provide the 312-MHz clock to all peripherals. Peripherals using external clocking are not altered; they continue to receive an external clock.

Allowing the peripheral PLL to continue running while the core PLL is disabled (forcing the CPU to run at 13 MHz) maintains peripheral functionality while enabling a lower CPU frequency to be used.

Software can optionally disable both CPU core PLL and peripheral PLL (CPDIS and PPDIS are set) however, the increased latency for this, is a stoppage of certain peripherals that cannot operate at 13 MHz.

Note: The LCD clock frequency (L_CLK) can be configured to 26 MHz while the processor is in 13M mode or deep-idle by setting CCCR[LCD_26].

To avoid stoppage of the LCD clock while exiting 13M mode (when CCCR[CPDIS] is set), follow these steps:

1. Remain in 13M mode, but early-enable the PLL (CCCR[CPDIS] = 1 and CCCR[PLL_EARLY_EN] = 1). This allows the PLL to be started early.
2. Read CCCR and compare to make sure that the data was correctly written.
3. Verify that CCSR[CPLOCK] and CCSR[PPLOCK] bits are both set, indicating that the PLLs are locked. Proceed to the next step only when this condition is true. At this point the processor is still in 13M mode, but the PLLs are running.

Note: Since the PLLs do not lock in less than 120 µs, software could utilize an OS timer to cause an interrupt after 120 µs. When the 120-µs timer interrupt occurs, the software could then begin polling CCSR[CPLOCK] and CCSR[PPLOCK]. This interrupt-driven delay would allow normal processing of other tasks at 13 MHz to continue during the time the PLLs are locking.

4. Exit 13M mode by writing 0x00 to CCCR[CPDIS,PPDIS] but maintaining CCCR[PLL_EARLY_EN] = 1. The CCCR[PLL_EARLY_EN] bit is automatically cleared after the frequency change.
5. Perform the frequency change only by setting the CLKCFG register's F bit.

Note: Entering idle mode from 13M mode puts the processor into deep-idle mode. Refer to Section 3.6.7 for details.

3.5.7.7.1 Prerequisites for Changing to 13M Mode

Because entering or exiting 13M mode is the same as a frequency change, see Section 3.5.7.3 for details. If software also disables the peripheral PLL, CCCR[PPDIS=1], certain peripherals cannot function in this mode—see Table 3-6 for details.

Note: Software can change into 13M mode from run mode only.

3.5.7.7.2 Initiating 13M Mode Change Sequence

To enter or exit 13M mode, perform a frequency change after setting the CCCR[CPDIS] bit. When this bit is written and a frequency change is performed by setting the F bit in CLKCFG, the processor enters or exits 13M mode.

Note: Other bits in CLKCFG cannot be changed while entering or exiting 13M mode. While in 13M mode, software must not write to CLKCFG [B, HT, T].

3.5.7.7.3 Behavior During 13M Mode Change Sequence

Entering and exiting 13M mode is the same as a frequency change. See Section 3.5.7.3 for details.

3.5.7.7.4 Completion of 13M Mode Change Sequence

After the clock speed has been changed, the peripheral continues execution at the next instruction after the write to CLKCFG. interrupt requests are no longer held, and any interrupts that occurred during the change sequence are sent to the peripheral.

CLKCFG remains in the state that was written to it. The values in it and the Clock Configuration Status register provide information about the current operating frequencies of the peripheral, LCD controller, memory controller, and system bus.

3.5.8 Summary of Clock Modes

The clock modes follow the sequences shown in Table 3-10. The total latency of each sequence is the sum of the latencies at each step.

Table 3-10. Summary of Clock Mode Sequences

Description of Action	Unit	Latency (cycles)	Half-Turbo/Turbo Chg.	Fast-Bus Chg.	Freq. Chg.
Write to CP14 CLKCFG register (software); interrupts gated.	CPU	1 CPU	x	x	x
All current instructions, including incomplete fetches, completed.	CPU	? CPU	x	x	x
All outstanding stores pending in CPU and memory controller are completed.	CPU	? SB	x	x	x
Wait for synchronization, Halt CPU clock.	Clks	<20 CPU	x	x	x
Deny all new bus requests (from LCD, USB-H, DMA, CPU).	PM	1 SB		x	x
Complete all memory controller transactions. Place SDRAM in self-refresh mode.	MEM	? SB			x
Synchronize clocks and power manager. Switch clock sources.	PM	<4 13M <8 SB			x
Frequency change sequence splits					
Disable PLLs if appropriate (xPDIS set).	PM	2 13M			x
Enable PLLs if appropriate (xPDIS clear) and wait for PLL to lock.	PM	2k* 13M			x
Synchronize clocks and power manager.	PM	<2 13M <4 SB			x
Frequency change completion sequence merges here					
Set PLL and/or SB divider.	PM	2 SB		x	x
Release bus for all transactions.	PM	1 SB		x	x
Wait for synchronization.	PM	<20 CPU	x	x	x
Enable clocks, interrupts to CPU and begin execution.	PM	2 SB	x	x	x

NOTES:

x	Must follow this step in the corresponding clock mode.
—	Not applicable
?	Variable—depends on system/software configuration.
13M	13-MHz processor oscillator
CPU	CPU
MEM	Memory controller
PM	Power manager
SB	System bus frequency

3.6 Power Manager Operation

The power manager controls all internal power domains, external power-supply functionality, and the entry and exit sequences for the processor power modes. The functional units within the power manager are:

- Power domains—Provide the connectivity and biasing to different regions for the various power modes. All units within a power domain receive the same power supply and must be powered on and off together. (The power domains are defined on page 3-5 and illustrated in Figure 3-2.)
- Sleep-mode power supply—Provides power during sleep mode to the 32.768-kHz timekeeping oscillator, real-time clock (RTC), and power manager.
- Power manager I^2C interface—Provides a hardware-controlled interface to the external regulator for voltage management.

The processor power modes are listed here in order of recovery time back to normal mode, with idle mode being the fastest and deep-sleep mode being the slowest:

- Normal mode—All internal power domains and external power supplies are fully powered and functional. The processor clocks are running.
- Idle mode—See Section 3.6.6 for more information. Clocks to the CPU are disabled; recovery is through interrupt assertion.
- Deep-idle mode—See Section 3.6.6 for more information. Clocks to the CPU are disabled; recovery is through interrupt assertion. When CCCR[CPDIS] is set, the mode is referred to as *deep-idle*.
- Standby mode—See Section 3.6.8 and Section 3.8.1.12 for more information. All internal power domains except VCC_RTC and VCC_OSC are placed in a low-power mode where state is retained but no activity is allowed. The clock sources can be disabled. Some of the internal power domains can be powered off, and both PLLs are disabled. Recovery is through external and selected internal wake-up events.
- Sleep mode—See Section 3.6.9 and Section 3.8.1.11 for more information. All internal power domains except VCC_RTC and VCC_OSC (both are internal supplies) can be powered off. All clock sources, except those used by the real-time clock (RTC) and the power manager, are disabled. The PXA27x processor's PWR_EN[1] output pin deasserts to optionally disable the external low-voltage power supplies to the processor's low-voltage domains. The remaining power domains are placed in a low-power state where state is retained but no activity is allowed. Recovery is through external and selected internal wake-up events. Because the program counter is invalid, recovery requires a system reboot (the program counter restarts from 0x0, so the core begins execution starting at the reset vector).
- Deep-sleep mode—See Section 3.6.10 for more information. All internal power domains except VCC_RTC and VCC_OSC can be powered off. All clock sources, except those used by the RTC and the power manager, are disabled. The PXA27x processor's PWR_EN[1] output pin deasserts to optionally disable the external low-voltage power supplies. The processor's SYS_EN[1] output pin also deasserts to optionally disable the external high-voltage power domains. All power domains are powered directly from the backup battery pin, VCC_BATT. The remaining power domains are placed in a low-power state where state is retained but no activity is allowed. Recovery is through external and selected internal wake-up events.

1. PWR_EN and SYS_EN are deasserted by hardware during the entry into sleep and deep-sleep modes. The system's power management IC must be configured to correctly control the system's power supplies.

Because the program counter is invalid, recovery requires a system reboot (the program counter restarts from 0x0, so the core begins execution starting at the reset vector).

Table 3-11 summarizes the action taken in each power mode. Figure 3-2 shows an overview of the power domains and units internal to the processor. Figure 3-3 summarizes the operational modes.

Note: VCC_BATT must be on at all times.

Table 3-11. Summary of Module Power and Clocks by Power Mode

Module	Clocks					Power				
	Normal	Idle	Standby	Sleep	Deep Sleep	Normal†	Idle†	Standby†	Sleep††	Deep Sleep††,†††
CPU (Processor Core)	On	Off	Off	Off	Off	On	On	St	Off	Off
Cache Contents	—	—	—	—	—	P	P	P	NP	NP
SRAMs (Internal SRAM Banks 0, 1, 2, and 3)	On	On	Off	Off	Off	On	On	St/Off	St[6]/Off	Off
Peripherals (All Units Not Otherwise Mentioned)	C[1]	C[1]	Off	Off	Off	On	On	St	Off	Off
OS Timer and Power Manager I^2C (PI) Power Domain	C[1]	C[1]	C[1]/Off	Off	Off	On	On	On/St/Off	On/St[6]/Off	St[7]/Off
Peripheral PLL	C[2]	C[2]	Off	Off	Off	On	On	Off	Off	Off
Core PLL	C[3]	C[3]	Off	Off	Off	On	On	Off	Off	Off
Real-Time Clock, Clocks/Power Manager	On	On	On	On	On	On	On	On	On[8]	On[9]
13-MHz Processor Oscillator	On	On	C[4]	C[4]	C[4]	On[8]	On[8]	On[8]	On[8]	On[9]
32.768-kHz Timekeeping Oscillator	C[5]	C[5]	C[5]	C[5]	C[5]	On[8]	On[8]	On[8]	On[8]	On[9]

NOTES:

On—Clock or power supply is active and fully functional.

Off—Clock is disabled or power supply is powered off.

St—Standby—power supply is in a low-power state-retaining mode; no activity is allowed.

C[1]—Configurable, using the clock-enable bits in CKEN.

C[2]—Configurable, using CCCR[PPDIS]; off if processor oscillator is off.

C[3]—Configurable, using CCCR[CPDIS]; off if processor oscillator is off.

C[4]—Configurable, using PCFR[OPDE]; on if OSCC[OOK] is clear.

C[5]—Configurable, using OSCC[OON].

St[6]—Standby state is powered by VCC_IO through the internal voltage regulator.

St[7]—Standby state is powered by VCC_BATT through the internal voltage regulator.

On[8]—Powered by VCC_IO through the internal voltage regulator.

On[9]—Powered by VCC_BATT through the internal voltage regulator.

P—Contents preserved.

NP—Contents not preserved.

† All power supplies that are being used must be enabled at all times.

†† When PWR_EN is deasserted, optionally remove external low-voltage power domains.

†††When SYS_EN is deasserted, optionally remove external high-voltage power domains.

Figure 3-2. Power Manager and Internal Power Domain Block Diagram

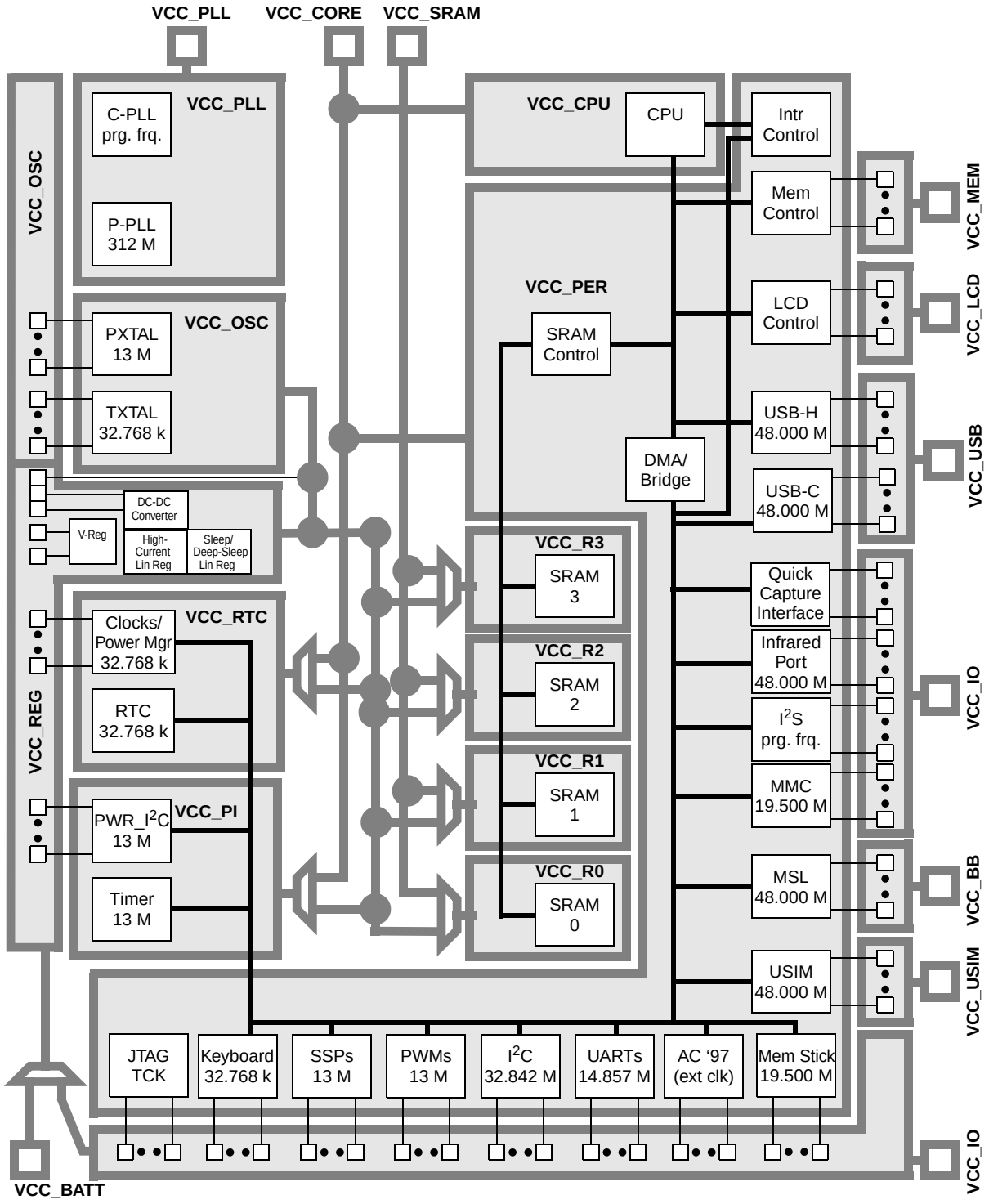
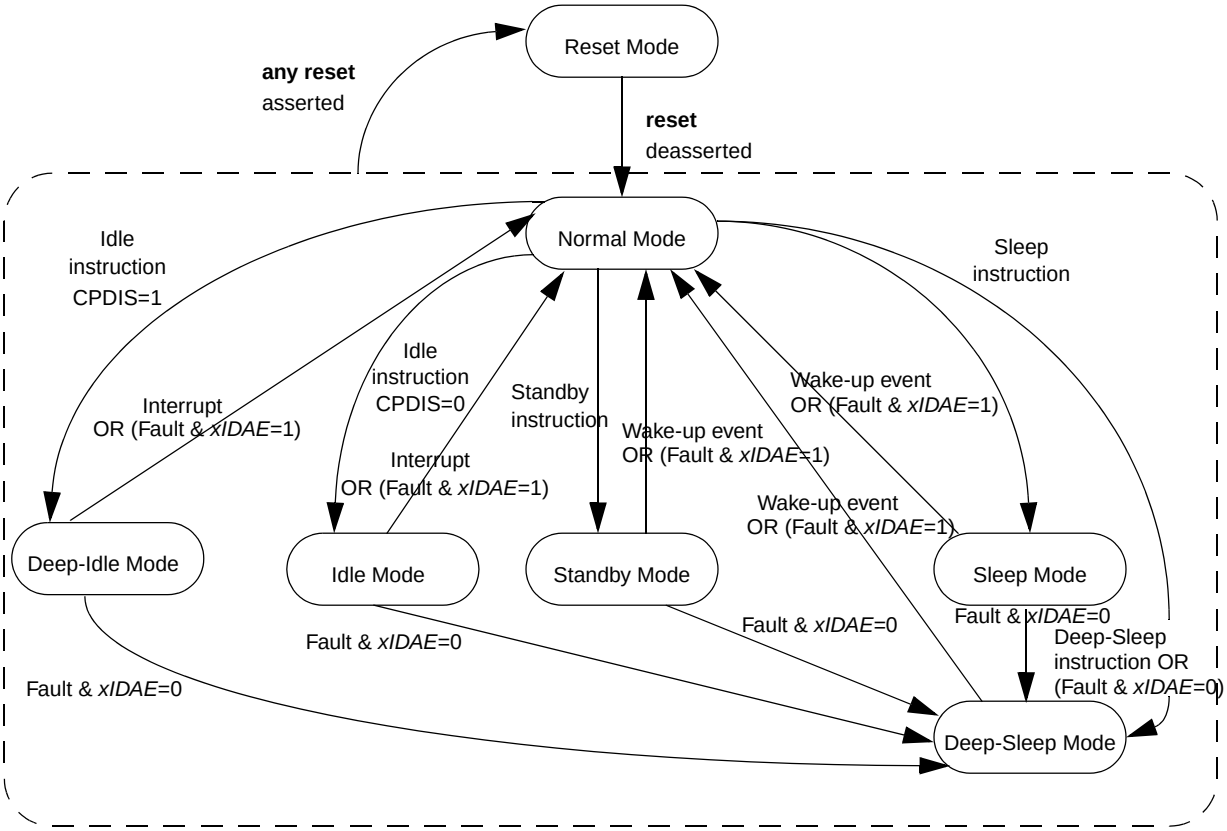


Figure 3-3. Overview of Power Manager Modes of Operation



3.6.1 Power Domains

The PXA27x processor has seven internal power domains (shown in Figure 3-2) and four I/O power supplies. All units within a power domain receive the same power and are powered on and off together. These domains are generated from one of seven externally applied power supplies, described in the *Intel® PXA27x Processor Family EMTS*.

The external power supplies create the internally-generated power domains. For more information, refer to the *Intel® PXA27x Processor Family EMTS*.

3.6.2 Internal Voltage Regulators

To achieve lowest system power in sleep and deep-sleep modes, the external power supplies can be disabled to prevent external regulator overhead power. Because the real-time clock (RTC), 32.768-kHz timekeeping oscillator, and power manager circuits must remain active, the PXA27x processor contains three internal voltage regulators, described in the sections that follow:

- High-Current Linear Regulator (Section 3.6.2.1)
- Sleep/Deep-Sleep Linear Regulator (Section 3.6.2.2)
- Sleep/Deep-Sleep DC-DC Converter (Section 3.6.2.3)

3.6.2.1 High-Current Linear Regulator

In sleep and deep-sleep modes, the external power supplies can be disabled. In these modes, however, there might be many current loads active:

- The real-time clock, 32.768-kHz timekeeping oscillator, and power manager are always active in these modes.
- The 13-MHz processor oscillator, OS timer, and power I^2C units are selectively active, based on software settings.
- In sleep mode only, the internal SRAM banks can be placed in a state that retains data at the expense of some current leakage.

The high-current linear regulator is always active in any of the following cases:

- The sleep/deep-sleep linear regulator and DC-DC converter are disabled by clearing PCFR[L1_EN] and PCFR[DC_EN]. For register details, see Section 3.8.1.8
- The 13-MHz processor oscillator is enabled.
- PCFR[OPDE] is set, attempting to disable the 13-MHz oscillator, but the 32.768-kHz oscillator has not yet stabilized (OSCC[OOK] is clear) before entering the low-power mode.
- Any of the internal SRAM banks or the power manager I^2C power domains are in a state-retaining mode during sleep mode.
- The power manager I^2C power domain is in a state-retaining mode during deep-sleep mode.

3.6.2.2 Sleep/Deep-Sleep Linear Regulator

The sleep/deep-sleep linear regulator cannot supply high current. This regulator is used when all of the following conditions apply (the sequence of setting the conditions is unimportant):

- The sleep/deep-sleep DC-DC converter is disabled (PCFR[DC_EN] is clear) and the sleep/deep-sleep linear regulator is enabled (PCFR[L1_EN] is set). For register details, see Section 3.8.1.8.
- The 13-MHz processor oscillator is disabled, and the 32-kHz oscillator has stabilized (OSCC[OOK] is set) before entering the low-power mode.
- The internal SRAM banks and power manager I^2C power domain do not retain state during sleep or deep-sleep modes.
- The power manager I^2C power domain and the OS timers are inactive.

3.6.2.3 Sleep/Deep-Sleep DC-DC Converter

During the lowest power sleep or deep-sleep modes, there are very few active loads. In these cases, the processor achieves best low-power efficiency when the internal DC-DC converter creates the internal supply. The DC-DC converter is used when all of the following conditions apply (the sequence of setting these conditions is unimportant):

- The sleep/deep-sleep DC-DC converter is enabled (PCFR[DC_EN] is set) and the sleep/deep-sleep linear regulator is disabled (PCFR[L1_EN] is clear). For register details, see Section 3.8.1.8.
- The 13-MHz processor oscillator is disabled and the 32-kHz oscillator has stabilized (OSCC[OOK] is set) before entering the low-power mode.

- The internal SRAM banks and power manager I^2C power domain are not in state-retaining modes during sleep or deep-sleep.
- The power manager I^2C power domain and the OS times are inactive.

The sleep/deep-sleep DC-DC converter requires the following external components:

- 0.1-µf capacitor connected between the PWR_OUT pin and ground
- 0.1-µf capacitor connected between the PWR_CAP0 and PWR_CAP1 pins
- 0.1-µf capacitor connected between the PWR_CAP2 and PWR_CAP3 pins

These capacitors must be ceramic, unpolarized capacitors with low equivalent series resistance (ESR).

Note: No other connections are allowed on the PWR_OUT and PWR_CAP<3:0> pins. Failure to adhere to this requirement can result in high currents, with associated potential for high temperature and permanent damage to the processor and the system.

3.6.3 Power Manager I^2C Interface

The PXA27x processor contains a dedicated I^2C module for communicating with an external regulator. See Chapter 9, "I2C Bus Interface Unit" for a full description. The only differences between the power manager I^2C (PWR_I2C) and the standard I^2C interfaces are the register addresses, which are summarized in Section 9.6.

3.6.4 Power Faults and Imprecise-Data Abort

Upon assertion of nBATT_FAULT or nVDD_FAULT, the processor enters the low-power deep-sleep mode, either automatically or under the control of a fault handler. For details of deep-sleep mode, see Section 3.6.10. For descriptions of the fault signals, see Section 3.3.13.

After a power fault, the ensuing action occurs in one of three ways, depending on the settings of the corresponding BIDAE and VIDAE bits and the IAS bit in the Power Manager Control register (PMCR—see Section 3.8.1.1 for register details):

- xIDAE clear—The processor immediately enters deep-sleep mode without issuing an imprecise-data abort or an interrupt to the core. All data and processor states are lost.
- xIDAE set, IAS clear—An imprecise-data abort is issued to the processor core, which immediately jumps to a pre-defined interrupt vector address. The imprecise-data-abort handler must be in place to manage deep-sleep entry. For more information, refer to the chapter "Handling Processor Exceptions" in the *ARM* Developer Suite Developer Guide*.
- xIDAE set, IAS set—An interrupt is issued to the processor core. The interrupt handler must be in place to manage deep-sleep entry. For more information about the processor interrupts, see Chapter 25, "Interrupt Controller".

In the last two cases (xIDAE set), the abort handler typically preserves critical cache and other data before initiating the deep-sleep entry with a write to the PWRMODE register.

The time available for the abort handler to take other actions, such as completion of current routines, depends upon how long the external regulator system can continue to supply power after the assertion of a power fault.

In some cases, such as during sleep mode while some units are retaining states, the wake-up latency and the abort handler's execution require more time than is available while the external supply ramps down. In these cases, clear xIDAE to force an immediate deep-sleep entry.

3.6.5 Modifying Power Modes

Coprocessor 14, register C7 (PWRMODE—see Section 3.8.3.2 for register details) initiates the following power-mode changes when the corresponding bit is set:

- PWRMODE[VC] initiates a voltage-change sequence. See Section 3.7.2 for details.
- PWRMODE[M] initiates the power mode corresponding to the value written to this bit field: normal, idle, deep-idle, standby, sleep, or deep-sleep. Section 3.6 summarizes these power modes, and the following subsections describe them in detail.

Legal values may be written to both VC and M in the same write operation. When a write to PWRMODE occurs, the processor implements the requested power-mode change and clears PWRMODE automatically.

Note: If CPSR[I] and CPSR[F] are set to mask the interrupt in the core before changing the power mode, the processor continues with the power-mode change despite any pending interrupts. In this case, if the power mode is sleep or deep-sleep and if there was a pending interrupt, the interrupt information is lost after waking from the mode. To avoid this situation, software must clear CPSR[I] and CPSR[F] before entering the power mode.

Note: Any two writes to CLKCFG or PWRMODE must be separated by six 13-MHz cycles. This requirement is achieved by reading CCCR and then comparing its value to the CLKCFG or PWRMODE register.

Note: If the memory controller is configured for synchronous external flash memory, it loses the synchronous configuration upon entry into sleep or deep-sleep mode. If PSLR[SL_ROD] is set, then nRESET_OUT does not get asserted. Thus, the external flash memory is still configured as synchronous, whereas the memory controller is configured for asynchronous operation. Software must manage this potential mismatch. For more information, see Section 6.5.2.1, "Synchronous Static Memory Configuration Register (SXCNFG)" on page 6-58.

3.6.6 Idle Mode

Idle mode allows stopping only the CPU clock during periods of processor inactivity, while continuing to monitor interrupt service requests. Idle mode is entered when PWRMODE[M] = 0b001. Generation of all other clocks remains unchanged so that, when an interrupt occurs, the CPU is quickly reactivated at the point where it entered idle mode. During idle mode, all other on-chip resources are active.

When CCCR[CPDIS] is set and PWRMODE[M] = 0b001, this mode is called deep-idle mode.

3.6.6.1 Preparation for Idle Mode

Before entering idle mode, enable any interrupts to be used as wake-ups from idle mode.

Idle mode does not stop the bus clocks and does not alter any of the peripheral clocks, including the memory and LCD controllers. Thus, no additional steps are required (with respect to the peripheral clocks) before entering idle mode.

3.6.6.2 Entering Idle Mode

To enter idle mode, first complete the preparations listed in Section 3.6.6.1. Then, write the appropriate value to the M bit field in the PWRMODE register (see Section 3.8.3.2). When the write to PWRMODE occurs, the following steps take place in order:

1. All processor activity is stopped. All interrupt requests to the CPU core are held.
2. All CPU loads are completed. CPU stores are sent to the system bus.
3. The CPU clock is halted.
4. Interrupts are no longer held and are recognized as wake-up sources from idle mode.

3.6.6.3 Behavior During Idle Mode

During idle mode, all peripherals and system resources are fully operational, except that the CPU clock is stopped.

The only difference from normal peripheral operation is that any enabled interrupt can awaken the processor from idle mode, regardless of the state of ICMR (see Section 25.5.4, “Interrupt Controller Mask Registers (ICMR and ICMR2)” on page 25-19).

If normal interrupt masking is required, disable this feature by setting the disable idle mask bit, ICCR[DIM] (see Section 25.5.6, “Interrupt Controller Control Register (ICCR)” on page 25-27).

If ICCR[DIM] is clear and idle-mode wake-ups from a specific unit are not desired, the unit’s interrupt must be disabled at the unit level.

Note: There is additional interrupt latency caused by holding interrupt requests while the CPU loads are completed and CPU stores are sent to the system bus. This latency varies with the number, source, and destination of loads and stores, as well as with other bus activity.

Note: The watchdog timer, if enabled, is functional during idle mode and generates a watchdog reset if OSMR register 3 matches the OS timer counter.

3.6.6.4 Exiting Idle Mode

Idle mode ends with the assertion of either of the following idle-mode wake-up events:

- Any enabled interrupt, regardless of the state of ICMR.
- Assertion of nBATT_FAULT or nVDD_FAULT (see Section 3.6.4 for details of the ensuing action).

Following the assertion of an idle-mode wake-up event, the following occurs:

1. The CPU clock is restarted.
2. The processor continues execution at the next instruction after the write to PWRMODE or at the entry point to the imprecise-data abort or interrupt handler.

Note: Any two writes to the CLKCFG or PWRMODE registers must be separated by at least six 13-MHz cycles. To meet this requirement, read CCCR and then compare its value to the CLKCFG or PWRMODE register.

3.6.7 Deep-Idle Mode

Deep-idle mode is a combination of 13M mode and the processor idle feature. Deep-idle mode is a transition into idle mode from 13M mode. Refer to Section 3.6.6 for idle mode and Section 3.5.7.7 for 13M mode.

Refer to the *Intel® PXA27x Processor Family EMTS* for appropriate VCC_CORE voltage setting in deep-idle mode.

When CCCR[CPDIS] is set and PWRMODE[M] = 0b001, this mode is referred to as deep-idle.

3.6.8 Standby Mode

Standby mode is a low-power mode in which power consumption is reduced below the normal static power consumption while the processor retains state. All processor activity stops, except for the real-time clock (RTC) and the clocks and power manager. Since internal activity has stopped, recovery from standby mode must be through an external or RTC event. At recovery, execution resumes at the instruction following the write to the PWRMODE register (see Section 3.8.3.2).

Unit state retention during standby mode is as follows (see the block diagram in Figure 3-2):

- The following units always retain state:
  - The CPU, powered by VCC_CPU (internal domain)
  - Peripheral units powered by VCC_PER (internal domain)
- The following units may optionally retain state:
  - Internal SRAM banks, powered by VCC_SRAM or VCC_REG (internal domain) (depending on low-power mode)
  - The power manager I^2C unit and 13-MHz timer, powered by VCC_PI (internal domain)
- The PLLs are automatically disabled in standby mode.

Note: Any two writes to the CLKCFG or PWRMODE registers must be separated by six 13-MHz cycles. To meet this requirement, read of CCCR and then compare its value to the CLKCFG or POWERMODE register.

Refer to the *Intel® PXA27x Processor Family EMTS* for appropriate VCC_CORE voltage setting.

3.6.8.1 Preparation for Standby Mode

Before entering standby mode, complete the following steps:

1. Set the standby mode unit retention bits in the Standby Configuration register (PSTR) for any units that must retain state during standby mode (see Section 3.8.1.12 for register details).
2. Configure the memory controller to ensure that SDRAM contents are maintained during standby mode (all required boot sequences must be complete). See Chapter 6, “Memory Controller” for details.
3. Stop or disable all peripheral units except the RTC and, optionally, the OS timer. Also, if the keypad is configured to be the wake-up source, do not disable the keypad controller.

   The other peripherals do not function normally, since the clocks are stopped. This includes the LCD controller; thus, unless the external LCD can sustain operation without constant pixel information, disable the external LCD and the LCD controller.

4. Program the following registers to enable the standby-mode wake-up sources:
   Section 3.8.1.4 — Power Manager Wake-Up Enable Register (PWER)
   Section 3.8.1.15 — Power Manager Keyboard Wake-Up Enable Register (PKWR)
   Section 3.8.1.5 — Power Manager Rising-Edge Detect Enable Register (PRER)
   Section 3.8.1.6 — Power Manager Falling-Edge Detect Enable Register (PFER)
5. **If low-power operation is required**, set PCFR[OPDE] to disable the 13-MHz processor oscillator (see Section 3.8.1.8 for register details). In this case, wait until PCFR[OOK] is set before entering standby mode.

   If fast wake-up is required, clear PCFR[OPDE] to keep the oscillator on during standby.
6. After exiting standby mode, the code/data fetches by the core must not resume from the memory space that cannot be accessed due the PSSR[PH] bit being set by hardware.

Note: The GPIO block is not reset in standby mode. Hence, the GPIO alternate functions are restored automatically after standby mode to their states immediately preceding standby mode.

3.6.8.2 Entering Standby Mode

To enter standby mode, first complete the preparations listed in Section 3.6.8.1. Then, write the appropriate value to the M bit field in the PWRMODE register (see Section 3.8.3.2). When the write to PWRMODE occurs, the following steps take place:

1. All processor activity is stopped. All interrupt requests to the processor are held.
2. All CPU loads are completed. All CPU stores are sent to the system bus.
3. The CPU clock is halted.
4. All new transactions from the USB host, LCD controller, and DMA controller are ignored.
5. The memory controller completes all outstanding transactions in its buffers.
6. The memory controller places the SDRAM in self-refresh mode and drives the nRAS/nSDCS<3:0> and nCAS/DQM<3:0> pins to their self-refresh states.
7. All PLLs are disabled.
8. If PCFR[OPDE] and PCFR[OOK] are set, the 13-MHz processor oscillator is disabled.
9. The circuits that activate the low-current mode are energized for the following units:
   — units in the VCC_CPU and VCC_PER power domains (see the block diagram in Figure 3-2)
   — units selected by the PSTR[standby mode unit retention] bits.
10. Units not selected by the PSTR[standby mode unit retention] bits are powered off.

3.6.8.3 Behavior in Standby Mode

In standby mode, all clocks are disabled except those for the power manager and the RTC. No interrupts are recognized. No external pin transitions are recognized other than valid wake-up signals, reset signals, nBATT_FAULT, and nVDD_FAULT.

The power manager watches for wake-up events that were programmed prior to entering standby mode (for more information about wake-ups, see Section 3.6.8.1). Refer to the *Intel® PXA27x Processor Family EMTS* for GPIO timing specifications.

Deep-sleep entry due to a power fault (nVDD_FAULT or nBATT_FAULT asserted) normally occurs as described in Section 3.6.4. But in standby mode, with the corresponding PMCR[xIDAE] bit set, the imprecise-data abort or interrupt is not issued immediately. Instead, the power-fault event appears to the power manager first as a standby-mode wake-up. The power-fault abort is sent to the processor core only after exit from standby mode is complete. Thus, there is additional latency between the assertion of a power fault and its recognition.

If this additional latency is unacceptable, clear the corresponding PMCR[xIDAE] bit. In this case, entry into deep-sleep mode occurs immediately, but controlled entry using software is not possible.

3.6.8.4 Exiting Standby Mode

Following the assertion of a pre-programmed standby-mode wake-up event or the assertion of nBATT_FAULT or nVDD_FAULT with the corresponding PMCR[xIDAE] bit set, the following occurs:

1. If standby mode was entered with PCFR[OPDE] set, the 13-MHz processor oscillator gets re-enabled and allowed to stabilize.

   If PCFR[OPDE] is clear, the 13-MHz processor oscillator is already enabled and has stabilized.
2. If any of the standby mode unit retention bits are clear, power is restored to the selected unit.
3. The PLLs are restarted according to the corresponding values in the Core Clock Configuration register and allowed to stabilize.
4. The CPU clock is restarted. Interrupts are no longer held.
5. The processor resumes execution at the next instruction after the write to PWRMODE or at the entry point to the imprecise-data abort or interrupt handler.
6. The standby-mode configuration is automatically cleared in the PWRMODE register.
7. The SDRAM must be brought out of self-refresh mode, which requires that the SDRAM controller be switched to its idle state. See Chapter 6, “Memory Controller” for details on configuring the SDRAM interface.

If nBATT_FAULT or nVDD_FAULT is asserted:

- If the corresponding PMCR[xIDAE] bit is set, the regular standby-mode exit sequence takes place. The abort handler can then enter deep-sleep mode under controlled conditions, allowing software to save critical data. For more information about the abort handler, see Section 3.6.4.
- If the corresponding PMCR[xIDAE] bit is clear, the processor exits standby mode without performing steps 1-7 and immediately enters deep-sleep mode.

3.6.9 Sleep Mode

Sleep mode offers even lower power consumption by powering off most units. The increased latency for this low-power mode is that all states are lost. There is no activity inside the processor, except for the units programmed to retain their state in the PSLR register, the real-time clock, and the clocks and power manager. Because internal activity has stopped, recovery from sleep mode must occur through an external or a real-time clock event. All processor states are reset, and recovery begins with the required boot sequence (see Section 3.4 for information about boot sequences).

In sleep mode, the external low-voltage power domains can be disabled externally to further lower the system power consumption. For information on the power domains, see Figure 3-2.

3.6.9.1 Preparation for Sleep Mode

Before entering sleep mode, take the following steps:

1. For units that must retain their states during sleep, set the appropriate sleep mode unit retention bits in the Sleep Mode Configuration register (PSLR—see Section 3.8.1.11). The units that can retain state are the internal SRAM banks and the PI power domain (timer and power I^2C). The PLLs are automatically disabled.
2. Program PSLR[SYS_DEL] and PLSR[PWR_DEL] for the number of 32.768-kHz timekeeping oscillator cycles required for the external power supplies to stabilize (the default reset value is 125-ms delay for each).
3. For lowest power consumption, enable the sleep/deep-sleep DC-DC converter (see Section 3.6.2.3) by setting PCFR[DC_EN].

   If PCFR[DC_EN] is clear and PCFR[L1_EN] is set, the sleep/deep-sleep linear regulator is enabled. If both PCFR[DC_EN] and PCFR[L1_EN] are clear, the high-current linear regulator is enabled.

Note: Do **not** set both PCFR[DC_EN] and PCFR[L1_EN] at the same time.

4. Disable the LCD controller, unless the external LCD panel has a built-in frame buffer and can operate while the pixel clock is stopped. For more information, see Chapter 7, "LCD Controller".
5. Configure the appropriate power manager registers for the sleep-mode wake-up sources:

   Section 3.8.1.4 — Power Manager Wake-Up Enable Register (PWER)

   Section 3.8.1.15 — Power Manager Keyboard Wake-Up Enable Register (PKWR)

   Section 3.8.1.5 — Power Manager Rising-Edge Detect Enable Register (PRER)

   Section 3.8.1.6 — Power Manager Falling-Edge Detect Enable Register (PFER)

   Section 3.8.1.9 — Power Manager GPIO Sleep-State Registers (PGSRx)

Note: As the PGSRx registers get loaded onto the processor GPIO outputs, care must be taken to initialize these to the correct state. For example, pins related to the static and synchronous memory chip selects need to be deasserted, and PC Card control pins must be either deasserted or floated.

6. **For lowest-power operation**, set PCFR[OPDE] to disable the 13-MHz processor oscillator (see Section 3.8.1.8 for register details). In this case, wait until PCFR[OOK] is set before entering sleep mode.

   For fastest wake-up, clear PCFR[OPDE] to keep the oscillator running during sleep mode.
7. Prepare for possible power faults (assertion of nVDD_FAULT or nBATT_FAULT and subsequent deep-sleep entry) as described in Section 3.6.4.

3.6.9.2 Entering Sleep Mode

Note: The GPIO block is reset in sleep mode. Hence, the GPIO alternate functions must be re-programmed after sleep exit.

Entry into sleep mode occurs when the sleep configuration is written to the M bits in the PWRMODE register (see Section 3.8.3.2).

When the sleep configuration is written to PWRMODE, the following sequence occurs:

1. All processor activity stops, and all interrupt requests to the processor are ignored.
2. All CPU loads are completed and CPU stores are sent to the system bus.
3. The CPU clock is halted.
4. All new transactions from the USB host controller, LCD controller, and DMA controller are ignored.
5. The memory controller completes all outstanding transactions in its buffers.
6. The memory controller places the SDRAM in self-refresh mode and drives the nRAS/nSDCS<3:0> and nCAS/DQM<3:0> pins to their self-refresh state.
7. The PLL clock sources and their outputs are disabled.
8. The power manager switches the GPIO output pins to the sleep states programmed in the PGSR registers.
9. If PCFR[OPDE] and OSCC[OOK] are set, the 13-MHz processor oscillator is disabled.
10. An internal reset (reflected externally by the assertion of the nRESET_OUT pin if PSLR[SL_ROD] is clear) is generated to the CPU, to peripheral logic powered by VCC_PER (internal domain), and to all units not selected by the sleep mode unit retention bits.
11. The low-current state retention circuitry is enabled for the units that are selected by the sleep mode unit retention bits.
12. The units not selected by the sleep mode unit retention bits are powered off.
13. The power supply to the clocks and power manager, RTC, and any units selected by the sleep mode unit retention bits is switched from VCC_CORE to VCC_OSC (internal domain). The internal SRAM banks selected by the sleep mode unit retention bits are switched from VCC_SRAM to VCC_OSC (internal domain).
14. The PWR_EN pin is deasserted. Optionally, disable the external low-voltage power domains to minimize power consumption.

3.6.9.3 Behavior in Sleep Mode

In sleep mode, all clocks are disabled to the processor and to all peripherals except the RTC. However, if the keypad is configured to be the wake-up source, do not disable the keypad controller. No interrupts are recognized, and no external pin transitions other than valid wake-up signals, reset signals, and the power fault (nVDD_FAULT and nBATT_FAULT) signals are recognized. The nVDD_FAULT pin is ignored if PSLR[IVF] is set.

The power manager watches for pre-programmed wake-up events that the CPU configures prior to entering sleep mode. Refer to the *Intel® PXA27x Processor Family EMTS* for GPIO timing specifications.

The processor does not recognize the imprecise-data abort or interrupt during sleep mode. Therefore, if the corresponding PMCR[xIDAE] bit is set, the assertion of nVDD_FAULT or nBATT_FAULT appears to the processor as a wake-up event from sleep mode. Once the processor has exited sleep mode, an imprecise-data abort or interrupt is reported to the core. For more information on these power faults and their management, see Section 3.6.4.

If nVDD_FAULT or nBATT_FAULT is asserted and if the corresponding PMCR[xIDAE] bit is set, the processor does not issue the imprecise-data abort or interrupt until sleep-mode exit completes. If this additional latency is unacceptable, the corresponding PMCR[xIDAE] bit must be cleared before entering sleep mode. In this case, however, assertion of the power fault results in the immediate loss of all processor states, with no software-controlled entry into deep-sleep mode.

3.6.9.4 Sleep Exit

Following the assertion of a pre-programmed sleep-mode wake-up event while the nVDD_FAULT and nBATT_FAULT pins are not asserted (nVDD_FAULT is ignored if PSLR[IVF] is set), the following occurs:

1. PWR_EN is asserted, enabling the external low-voltage power domains.
2. The processor waits the number of 32.768-kHz timekeeping oscillator cycles specified by the PSLR[PWR_DEL] bits.
3. The power supply to the clocks and power manager, RTC, and any units selected by the sleep mode unit retention bits in PSLR is switched from VCC_OSC to VCC_CORE or from VCC_OSC to VCC_SRAM.
4. If sleep mode was entered with PCFR[OPDE] set, the 13-MHz processor oscillator is enabled and allowed to stabilize.
5. If any of the sleep mode unit retention bits are clear, power to the selected units is restored.
6. The PLLs are restarted with the corresponding values in the Core Clock Configuration register and allowed to stabilize.
7. The sleep-mode configuration in the PWRMODE register is cleared.
8. The nRESET_OUT pin, if asserted, is deasserted, indicating that the processor is about to perform a fetch from the reset-vector location. The processor internal reset is deasserted.
9. The CPU begins the required boot sequence (see Section 3.4 for information about boot sequences), which includes the following items:
   a. Software must bring the SDRAM out of self-refresh mode, which requires that the SDRAM controller be switched to its idle state. See Chapter 6, “Memory Controller” for details on configuring the SDRAM interface.
   b. All units in the PXA27x processor, except those listed in Table 3-2, begin with their predefined reset conditions.
   c. Software must examine the Reset Controller Status register (RCSR[SMR]) to determine that the reset source was a sleep-exit reset and the Sleep Status register (PSSR[SSS]) to determine the reason for being in sleep mode.
   d. If the Scratch Pad register (PSPR) was used for saving any general processor state during sleep mode, the state can be recovered.

Note: If sleep mode was entered while the processor was in turbo, half-turbo or fast-bus mode, the sleep-mode exit returns the processor to normal run mode. If sleep mode was entered while in 13M mode, the sleep-mode exit returns the processor to 13M mode.

3.6.10 Deep-Sleep Mode

Deep-sleep mode offers the lowest power consumption by powering most units off. The increased latency for this low-power mode is that all state is lost and there is no activity inside the processor, except for the real-time clock (RTC) and the clocks and power manager. Because internal activity has stopped, recovery from deep-sleep mode must be through an external event or an RTC event. Because all state has been lost, the state of the processor is reset, and recovery begins with the required boot sequence (see Section 3.4 for information about boot sequences).

In deep-sleep mode, all the power supplies (VCC_CORE, VCC_SRAM, VCC_PLL, VCC_IO, VCC_LCD, VCC_USIM, VCC_USB, VCC_BB, and VCC_MEM) excluding VCC_BATT can be powered off for minimized power consumption.

If deep-sleep mode was entered with a software write to the PWRMODE register, deep-sleep can be exited by correctly programming the PWER, PFER and PRER registers for the RTC wake-up event and a wake-up event on GPIO<3>, GPIO<1>, or GPIO<0>. If deep-sleep mode was entered due to a power fault (assertion of nBATT_FAULT or nVDD_FAULT), then exit from deep-sleep is limited to a wake-up event on GPIO<1> or GPIO<0>. The PWER, PFER, and PRER registers are automatically forced to their reset values in this case.

3.6.10.1 Preparation for Deep-Sleep Mode

Before entering deep-sleep mode, complete the following steps:

1. For any units that must retain state during deep-sleep mode, set the sleep mode unit retention bits in the Sleep Mode Configuration register (see Section 3.8.1.11). Only the units in the PI power domain (RTC and power manager I^2C) can retain state. The PLLs are disabled automatically.
2. Program the SYS_DEL and PWR_DEL bits in PSLR for the number of 32.768-kHz timekeeping oscillator cycles required to stabilize the external power supplies (the reset value is 125ms delay each).
3. For lowest power consumption, enable the sleep/deep-sleep DC-DC converter (see Section 3.6.2.3) by setting PCFR[DC_EN].

   If PCFR[DC_EN] is clear and PCFR[L1_EN] is set, the sleep/deep-sleep linear regulator is enabled. Otherwise, the high current linear regulator is enabled.

Note: Do **not** set PCFR[DC_EN] and PCFR[L1_EN] at the same time.

4. The memory controller sends the self refresh command to the SDRAM banks. Ensure that the supply to the SDRAM is not removed if state retention is required during deep-sleep mode. See Chapter 6, “Memory Controller” for more details.
5. Disable the LCD controller. LCD operation during deep-sleep mode is possible only with an external LCD panel that has a built-in frame buffer.
6. Initialize the appropriate power manager registers to determine the deep-sleep wake-up sources:
   - Power Manager Wake-Up Enable register (PWER)
   - Power Manager Falling-Edge Detect Enable and Power Manager Rising-Edge Detect Enable registers (PFER and PRER)
   - Power Manager GPIO Sleep-State registers (PGSR0, PGSR1, PGSR2 and PGSR3).

Note: Because the PGSRx registers get loaded onto the GPIO outputs, be careful to initialize these to the correct states—for example, pins related to the static and synchronous memory chip selects, PC Card control, and so forth must be floated. When SYS_EN is deasserted and the power domains are powered-off, all GPIOs float. However, if software is configured to maintain power during deep-sleep mode, the GPIOs default to the pull-up, pull-down reset state as indicated in the *Intel® PXA27x Processor Family EMTS* Pin Usage section for reset states and the GPIO AC Timing Specification section for timing details.

7. **For lowest-power operation**, set PCFR[OPDE] to disable the 13-MHz processor oscillator (see Section 3.8.1.8 for register details). In this case, wait until PCFR[OOK] is set before entering sleep mode.

8. **For fastest wake-up**, clear PCFR[OPDE] to keep the oscillator running during sleep mode.

3.6.10.2 Entering Deep-Sleep Mode

Entry into deep-sleep mode occurs at any of the following deep-sleep entry events:

- The deep-sleep configuration is written to the mode bits in the PWRMODE register (see Section 3.8.3.2).
- The pin nBATT_FAULT or nVDD_FAULT is asserted while the corresponding xIDAE bit is clear.
- nBATT_FAULT or nVDD_FAULT is asserted while exiting from deep-sleep mode.

When the deep-sleep configuration is written, the following sequence occurs:

1. All processor activity is stopped and all interrupt requests to the processor are ignored.
2. All CPU loads are completed, and CPU stores are sent to the system bus.
3. The CPU clock is halted.

If nBATT_FAULT or nVDD_FAULT is asserted while the corresponding xIDAE bit is clear, the sequence begins here:

4. All new transactions from the LCD controller or DMA controller are ignored.
5. The memory controller completes all outstanding transactions in its buffers.
6. The memory controller places the SDRAM in self-refresh mode and drives the nRAS/nSDCS<3:0> and nCAS/DQM<3:0> pins to their self-refresh state.

If nVDD_FAULT or nBATT_FAULT is asserted during the sleep or deep-sleep exit sequence, deep-sleep mode is re-entered here:

7. The power manager switches the GPIO output pins to the sleep state programmed in registers PGSR0-3.
8. If the deep-sleep sequence was entered because of the assertion of nVDD_FAULT or nBATT_FAULT, regardless of the state of the corresponding xIDAE bit, the following actions occur:
   a. All wake-ups detected at this point are cleared (all GPIO edge-detects and the RTC alarm interrupt).
   b. The power manager wake-up source registers (PWER, PRER and PFER) are loaded with the value 0x0000_0003, the reset state after a hardware reset, which limits the potential wake-up sources to a rising or falling edge on GPIO<0> or GPIO<1>. This wake-up fault state prevents spurious events from causing an unwanted wake-up when a problem develops with the main battery or a power supply.
9. The PLL clock sources and their outputs are disabled.
10. If PCFR[OPDE] and OSCC[OOK] are set, the 13-MHz processor oscillator is disabled.
11. An internal reset (reflected externally by the assertion of the nRESET_OUT pin if PSLR[SL_ROD] is clear) is generated to the CPU, to peripheral logic powered by VCC_PER, and to all units not selected by the sleep mode unit retention bits.
12. The low-current state retention circuitry is enabled for the units that are selected by the sleep mode unit retention bits.
13. The units not selected by the sleep mode unit retention bits are powered off.

14. The power supply (VCC_REG) for the regulator that generates VCC_OSC is switched from VCC_IO to VCC_BATT.
15. The power supply to the clocks and power manager, RTC, and any units selected by the sleep mode unit retention bits is switched from VCC_CORE to VCC_OSC.
16. The PWR_EN pin is deasserted. Disable the external low-voltage power domains to minimize power consumption.
17. The SYS_EN pin is deasserted. Disable the external high-voltage power domains to minimize power consumption. If any of these power supplies is disabled, then all of the external low-voltage power domains must also be disabled.

3.6.10.3 Behavior in Deep-Sleep Mode

In deep-sleep mode, all clocks to the processor and to all peripherals except the RTC are disabled. Therefore, no interrupts are recognized, and no external pin transitions other than valid wake-up signals, reset signals, and the nBATT_FAULT signal are recognized. The nVDD_FAULT pin is ignored until the appropriate point in the deep-sleep wake-up sequence. The nVDD_FAULT pin is ignored if PSLR[IVF] is set.

The power manager watches for wake-up events that were programmed prior to entering deep-sleep mode. Refer to the *Intel® PXA27x Processor Family EMTS* for GPIO timing specifications.

In deep-sleep mode, the external GPIO wake-up sources are limited to GPIO<3:0>, unless the deep-sleep entry was caused by nBATT_FAULT or nVDD_FAULT assertion. In this case, the wake-ups are limited to GPIO<1:0>.

The imprecise-data abort or interrupt is not recognized in deep-sleep mode. If nBATT_FAULT is asserted during deep-sleep mode, the processor remains in deep-sleep mode.

3.6.10.4 Exiting Deep-Sleep Mode

Following the assertion of a pre-programmed deep-sleep mode wake-up event while the nBATT_FAULT pin is not asserted, the following occurs:

1. SYS_EN is asserted, enabling the external high-voltage power domains.
2. The processor waits the number of 32.768-kHz timekeeping oscillator cycles specified by PSLR[SYS_DEL].

   If PSLR[PSSD] is set, the processor shortens the wake-up sequence by asserting PWR_EN as soon as all of the high-voltage power supplies signal that they are powered on.
3. The PWR_EN signal is asserted, enabling the external low-voltage power domains.
4. The processor waits the number of 32.768-kHz timekeeping oscillator cycles specified by PSLR[PWR_DEL].

   If PSLR[PSSD] is set, the processor shortens the wake-up sequence by cutting short the PSLR[PWR_DEL] counting as soon as all of the low-voltage power supplies signal that they are powered on.

   Beyond this point, if nVDD_FAULT is asserted, the processor switches back into deep-sleep mode.
5. The power supply to the clocks and power manager, the RTC, and any units selected by the PSLR[sleep mode unit retention] bits is switched from VCC_OSC to VCC_CORE.
6. The power supply (VCC_REG) to the regulator that generates VCC_OSC is switched from VCC_BATT to VCC_IO.

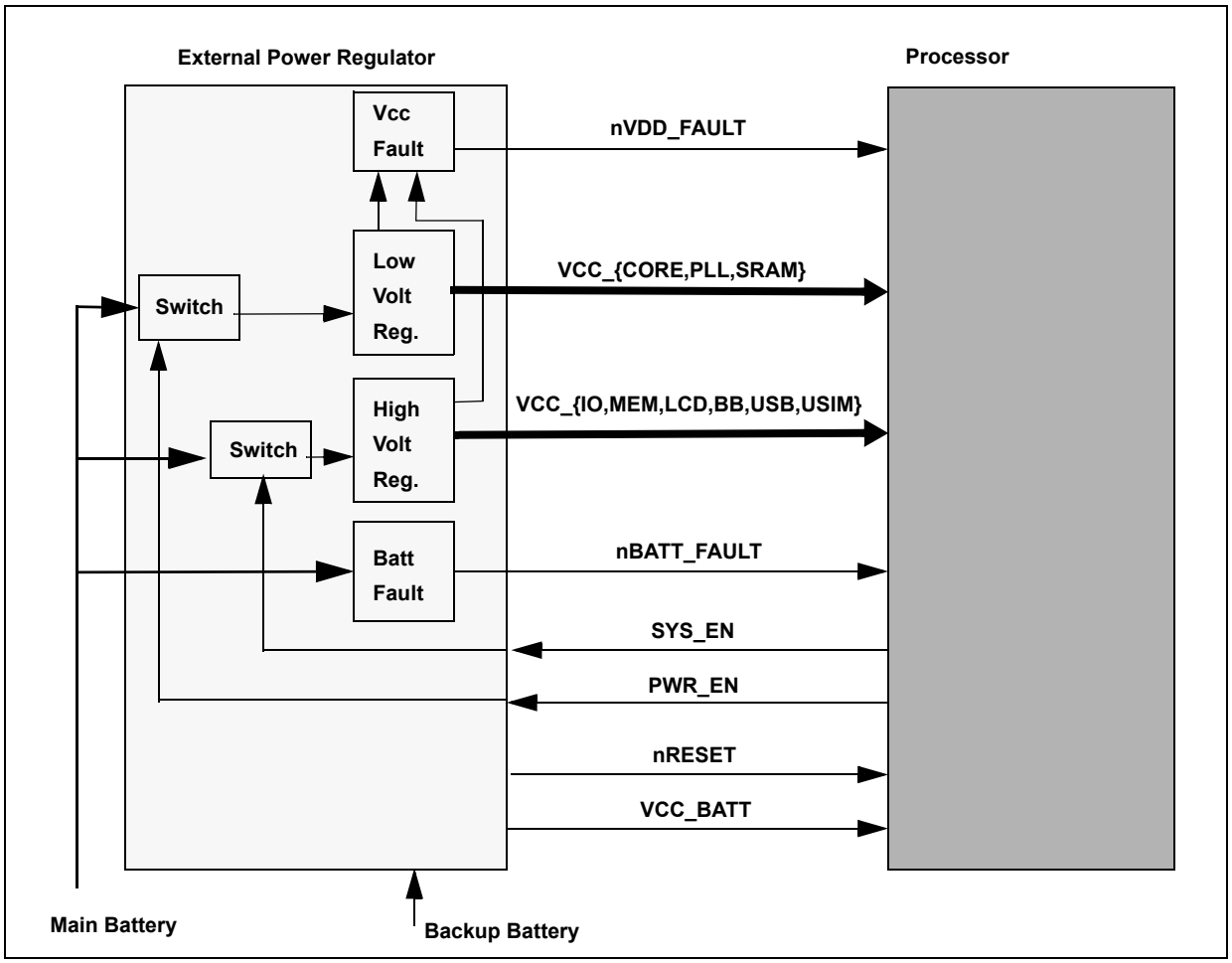
7. If deep-sleep mode was entered with PCFR[OPDE] set, the 13-MHz processor oscillator is enabled and allowed to stabilize.
8. If any of the PSLR[sleep mode unit retention] bits are clear, power to the selected unit is restored. If any of the PSLR[sleep mode unit retention] bits are set, the low-current state-retention circuitry is disabled, and power to the selected unit is restored.
9. The PLLs are reprogrammed with the corresponding values in the Core Clock Configuration register and allowed to stabilize.
10. The deep-sleep configuration in the PWRMODE register is cleared.
11. If it is asserted, the nRESET_OUT pin is deasserted, indicating that the processor is about to perform a fetch from the reset-vector location. The processor's internal reset is deasserted.
12. The CPU begins the required boot sequence (see Section 3.4 for information about boot sequences), which includes the following items:
    a. Software must bring the SDRAM out of self-refresh mode, which requires that the SDRAM controller be switched to its idle state. See Chapter 6, "Memory Controller" for details on configuring the SDRAM interface.
    b. All processor units, except those listed in Table 3-2, begin with their predefined reset states.
    c. Software must examine the Reset Controller Status register (RCSR[SMR]) to determine that the reset source was a sleep-exit reset from deep-sleep mode and the Sleep Status register (PSSR) to determine the reason for being in deep-sleep mode.
    d. If the Scratch Pad register (PSPR) was used for saving any general processor states during deep-sleep mode, the states can be recovered.

Note: If deep-sleep mode was entered while the processor was in turbo, half-turbo or fast-bus mode, the deep-sleep exit returns the processor to normal run mode. If deep-sleep mode was entered while the processor was in 13M mode, the deep-sleep exit returns the processor to13M mode.

3.6.11 Initial Power-On and Deep-Sleep Exit Sequence

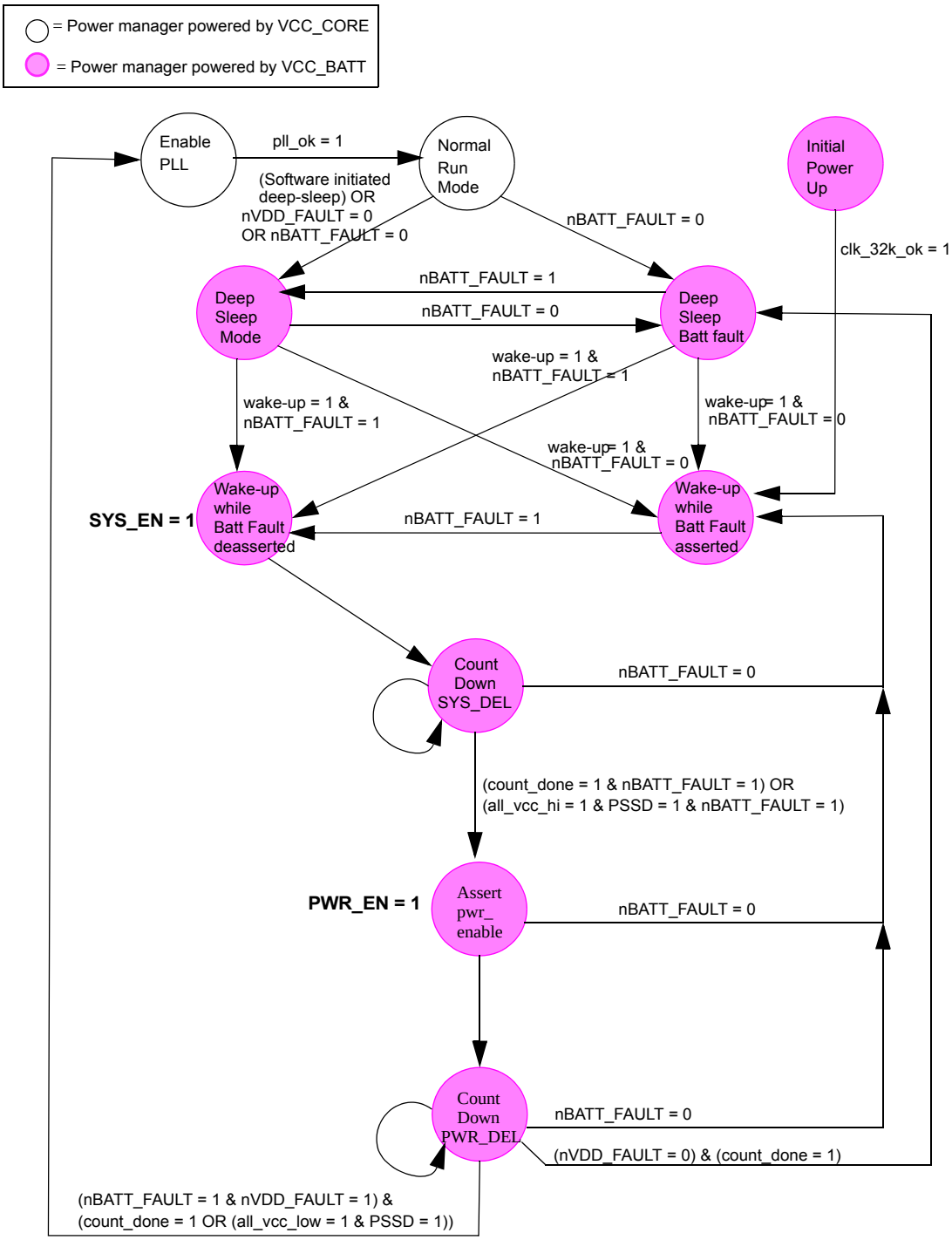
As shown in Figure 3-4, the external voltage regulator supplies the high-voltage and low-voltage power supplies to the processor. The external voltage regulator also sources the nBATT_FAULT and nVDD_FAULT signals to the processor. The processor's SYS_EN and PWR_EN signals control the high (VCC_IO, VCC_MEM, VCC_LCD, VCC_BB, VCC_USB, and VCC_USIM) and low (VCC_CORE, VCC_SRAM, VCC_PLL) voltages, respectively.

Figure 3-4. Typical System Diagram



The state diagram in Figure 3-5 shows typical steps taken by the power manager while initially powering up and while exiting from deep-sleep mode. Based on the wake-up events and states of the nBATT_FAULT and nVDD_FAULT signals, the processor exits from deep-sleep mode and enters the normal power mode. Refer to the *Intel® PXA27x Processor Family EMTS* for timing information on the initial power-on sequence and the deep-sleep exit sequence.

Figure 3-5. Initial Power-On and Deep-Sleep Exit States



3.6.12 Summary of Power Modes

The power modes follow the entry and exit sequences shown in Table 3-12. The total latency of the entry into each sequence is the sum of the latencies at each step. The remaining latency visible to software depends on latency in the boot up or interrupt service routine.

Table 3-12. Summary of Power and Clock Mode Sequences (Sheet 1 of 2)

Description of Action	Unit	Latency (cycles)	Idle	Standby	Sleep	Deep Sleep
Write to CP14 PWRMODE register (software); interrupts gated off.	CPU	1 CPU	x	x	x	x
All current instructions, including incomplete fetches, completed.	CPU	? CPU	x	x	x	x
All outstanding stores completed.	CPU	? SB	x	x	x	x
Wait for synchronization. Halt CPU clock.	CPM	< 20 CPU	x	x	x	x
Entry point for sleep/deep-sleep entry, xIDAE bits clear						
Deny all bus requests (from LCD, USB-H, DMA, CPU, or memory controller).	CPM	1 SB		x	x	x
Complete all memory controller transactions (allow memory controller requests). Place SDRAM in self-refresh mode.	MEM	? SB		x	x	x
Switch GPIO output pins to sleep state in PGSR registers.	CPM	1 SB			x	x
Re-Entry point for sleep/deep-sleep entry, Fault pin asserted during exit sequence						
Clear all wake-up sources; set PWER, PRER, PFER to 0x0000 0003 if entered from FAULT pin.	CPM	1 SB			x	x
Synchronize clocks and power manager. Switch clock sources if needed.	CPM	<4 13M <8 SB		x	x	x
Disable PLLs if appropriate (OPDE set, xPDIS set).	CPM	2 13M		x	x	x
Switch clocks and power manager from 13M to 32k clock if OPDE set and disable 13M oscillator.	CPM	3 32k		x	x	x
Assert reset to internal units if appropriate. Assert nRESET_OUT.	CPM	1 32k		x	x x	x x
Power off/standby selected units according to the mode's retention units if any.	CPM	2 32k		x	x	x
Deassert PWR_EN.	CPM	2 32k			x	x
Deassert SYS_EN.	CPM	2 32k				x
Power/clock mode entry sequence completed; waiting for external wake-up event						
Enable wake-up events/interrupts to CPU.	CPM	1 32k	x	x	x	x
Wake-up/interrupt received in clocks and power manager.	CPM	—	x	x	x	x
External wake-up synchronized to clocks and power manager clock.	CPM	2 32k		x	x	x
Assert SYS_EN.	CPM	1 32k				x
Wait for ramp time set by SYS_DEL.	CPM	? 32k				x
Assert PWR_EN.	CPM	1 32k			x	x
Wait for ramp time set by PWR_DEL.	CPM	? 32k			x	x
The 13-MHz processor oscillator stabilizes if OPDE is set.	CPM	64k* 13M		x	x	x
Switch CPM to 13-MHz clock if OPDE set.	CPM	2* 13M		x	x	x
Power on internal units.	CPM	? 13M		x	x	x
Enable PLLs if appropriate (xPDIS clear) and wait for PLL to lock.	CPM	1290 * 13M		x	x	x
Synchronize clocks and power manager.	CPM	<2 13M <4 SB		x	x	x

Table 3-12. Summary of Power and Clock Mode Sequences (Sheet 2 of 2)

Description of Action	Unit	Latency (cycles)	Idle	Standby	Sleep	Deep Sleep
Enable I/O, clocks to all units, set PLL divider.	CPM	3 SB		x	x	x
Release bus for all transactions.	CPM	1 SB		x	x	x
Wait for synchronization.	CPM	<20 CPU	x	x	x	x
Enable clocks, interrupts to CPU and begin execution.	CPM	2 SB	x	x	x	x
Deassert internal reset, nRESET_OUT.	CPM	1 SB			x	x

NOTES:

x	Step is followed in the corresponding power mode
?	Variable
13M	13-MHz processor oscillator
32k	32.768-kHz timekeeping oscillator
CPM	Clocks and power manager
CPU	CPU
MEM	Memory controller
SB	System bus

3.7 Voltage Manager Operation

The voltage manager provides dynamic and static voltage management to the processor through the use of an I^2C module (PWR_I^2C) dedicated to communication with the external regulator. The voltage manager provides the following features:

- Static (halted) or dynamic (operational) voltage change
- Up to 32 I^2C commands automatically sent to external PWR_I^2C module
- Programmable delay between commands

The voltage manager consists of two primary components:

- Dedicated power manager I^2C module (PWR_I^2C)
- Command sequencer

3.7.1 Power Manager I^2C and Restrictions

The dedicated I^2C module used by the voltage manager is nearly identical to the I^2C described in Chapter 9, “I2C Bus Interface Unit”. The power manager I^2C (PWR_I^2C) is optimized for connection to the external voltage regulator only. The power manager I^2C is a full-featured I^2C module that can connect to other units, although operation during a voltage-change operation may be limited.

Refer to the *Intel® PXA27x Processor Family EMTS* for voltage-change timing specifications.

3.7.1.1 Programming Restrictions

Except for the ICCR fixed selections, power manager I^2C is a full-function I^2C capable of all normal operations, including master and slave, receive and transmit operation. The power manager I^2C supports standard-speed operation of 40 kbits/sec and fast-speed operation of 160 kbits/sec. When used with the voltage-change sequencer, the power manager I^2C operates at standard speed. When used apart from the voltage-change sequencer, the power manager I^2C supports all I^2C specifications. The PI^2C_EN bit in register PCFR (see Section 3.8.1.8) must be set to use the power manager I^2C, either with or without the voltage-change sequencer.

When the voltage-change sequencer is operating, indicated by hardware setting the PVCR[VCSA] (see Section 3.8.1.13), the following condition applies:

- The power manager I^2C registers (PCMDx and PVCR) are not writable and reads return unknown values.

Thus, software must check PVCR[VCSA] before reading or writing to the power manager I^2C registers and must read the registers following a write to ensure that the write took place.

The following restrictions apply to the voltage-change sequencer:

- The sequencer allows a maximum of 32 commands in the transmission of data to the external regulator.
- The sequencer allows only master-transmitter operations to a single, predefined slave.
- The sequencer does not send interrupts to the CPU.
- Only the standard-speed operation at 40 kbits/sec is used.

3.7.2 Voltage-Change Sequencer

The voltage manager contains a voltage-change sequencer, which automatically sends commands to the external regulator when triggered by the voltage-change mode. The sequencer can send up to 32 commands, which can be categorized as dynamic commands and static commands.

Dynamic commands are executed when the core is running. The power manager requests the voltage manager for command execution. The sequencer starts sending out the commands as soon as the request is received. The voltage manager gives an acknowledgement to the power manager after completing all of the commands.

Static commands are executed after clocks to the processor are disabled. Static commands are transmitted by the voltage manager when voltage change is coupled with a frequency change (see Section 3.7.6.3) or a power-mode change (see Section 3.7.6.4).

3.7.2.1 Voltage-Change Sequencer Controls

The following control bits in PCFR (see Section 3.8.1.8), PVCR (see Section 3.8.1.13), and PCMD (see Section 3.8.1.17) affect execution of a voltage-change sequence:

- Frequency/voltage change bit (PCFR[FVC])—When set, a frequency-change sequence also triggers a voltage-change sequence.
- Read Pointer in PVCR—These bits point to the PCMD register location that contains the command to be sent out. The command sequence can start from any PCMD register by programming these bits accordingly. After a command is sent out, the read pointer increments

to point to the next PCMD register location. The read pointer is not incremented if the current command is the last command, as indicated by PCMD[LC] set.

- Delay command execution bit (PCMD[DCE])—If DCE is set in the current PCMD, a counter (set by the command delay bits in PVCR) waits for a programmable number of 13-MHz processor-oscillator cycles before continuing execution of the command. This is useful if a longer period between commands is required (for example, to allow for stabilization).
- Multi-byte command bit (PCMD[MBC])—If set, the voltage-change sequencer continues sending bytes to the slave with no delay or handshaking with the power manager until a command with PCMD[MBC] clear is executed. PCMD[SQC] and PCMD[LC] must be the same for all bytes of a multi-byte command except for the last byte, which has PCMD[MBC] clear and PCMD[LC] set.
- Last command bit (PCMD[LC])—When clear, the voltage-change sequencer expects the PCMD register at the next higher address to contain an additional command. If PCMD[LC] is clear in PCMD31, the PVCR read pointer rolls over to PCMD0 after executing the command in PCMD31. When PCMD[LC] is set, the voltage-change sequencer considers the current command the last one and finishes after execution completes. Each voltage-change command sequence must be terminated by setting PCMD[LC] for the last command in the sequence. The PVCR read pointer is not incremented if PCMD[LC] is set.

3.7.2.2 Static Voltage-Change Sequence Configurations

Execution of the voltage-change sequence is controlled by the following configurations set by the SQC bits in PCMDx:

- Continue configuration (PCMD[SQC] set to Continue)—Execution of this command is automatic when the command is read from PCMDx.
- Pause configuration (PCMD[SQC] set to Pause)—When this command is read, the voltage manager sends a trigger to the power manager after executing the command. The PVCR read pointer is incremented, but execution of the next command pauses until the power manager issues a new request to the voltage manager.

3.7.2.3 Prerequisites for Voltage-Change Sequence

Before initiating an automatic voltage-change sequence, configure the following items:

1. Set PCFR[PI^{2}C_EN].
2. If the voltage-change sequence is to be used in conjunction with a clock frequency change, set PCFR[FVC].
3. Program the required delay between commands into PVCR[Command Delay].
4. Program the address for the external voltage regulator into PVCR[Slave Address].
5. Configure the external voltage regulator as a slave with the same address that is specified in PVCR[Slave Address].
6. Set all units on the power manager I^{2}C bus to slave-receive mode. They must be prevented from transmitting on the I^{2}C bus during the voltage-change sequence.
7. Load PCMDx with the commands to be sent.
8. Write the starting command location to PVCR[Read Pointer].

3.7.2.4 Sequence Initiation

The first command in the Power Manager I^2C Command register file (PCMDx) is initiated immediately when the power manager sends a request to the voltage manager, triggered by one of the following events:

- 0b1 is written to PWRMODE[VC])
- 0b1 is written to CLKCFG[F] while PCFR[FVC] is set.

Once the voltage-change sequence is initiated, reads from and writes to the power manager I^2C module are ignored until the voltage-change sequence is complete. All registers return unknown values if read, writes are ignored, and interrupts from power manager I^2C are directed to the automatic voltage-change sequencer instead of to the interrupt controller. This state is indicated when PVCR[VCSA]. is set.

Each command is executed from PCMDx in order of increasing addresses, starting with the address in PVCR[Read Pointer]. The command is executed if all of the following events have occurred:

1. PWRMODE[VC] is set, or 0b1 is written to CLKCFG[F], while PCFR[FVC] is set.
2. All previous (lower-address) commands have completed execution but did not set their LC configuration.
3. If the Pause configuration was set for the previous command, then a request has been received from the power manager.

Once initiated, PVCR[VCSA] is set and remains set until the voltage-change sequence is complete. While VCSA is set, the power manager I^2C registers ignore writes and return undefined values if read.

3.7.2.5 Command Execution

Each command is set up as a master-mode transmission to the slave device (the slave's address is stored in (PVCR[Slave Address]). Since all commands use the same transmission type and slave address, automatic communication is limited to a single device. The voltage-change sequence takes place automatically, as follows.

1. If Pause is set in the previously-transmitted PCMDx command, wait for a request from the power manager.
2. If PCMDx[DCE] is set in the current command, count the specified number of 13-MHz processor oscillator cycles.
3. Write the slave address to the Power I^2C Data Buffer register, PIDBR[7:1]. Clear PIDBR[0]. For PIDBR register details, see Section 9.5.4, "I2C Data Buffer Register (IDBR, PIDBR)" on page 9-29.
4. In the I^2C Control register, set the transmit empty interrupt enable (ITEIE), I2C unit enable, (IUE), SCL enable (SCLEA), transfer byte (TB), and start (START) bits. This sends one byte in master-transmit mode to the power manager I^2C bus. For PICR register details, see Section 9.5.1, "I2C Control Registers (ICR, PICR)" on page 9-23.
5. Wait for the IDBR transmit empty interrupt, which automatically triggers the next step.
6. Read the Power I^2C Status register (PISR), looking for the IDBR transmit empty (ITE) and unit busy (UB) bits to be set. For PISR register details, see Section 9.5.2, "I2C Status Registers (ISR, PISR)" on page 9-26.
7. Read PISR to clear PISR[ITEIE].

8. Write the currently-pointed-to PCMDx[Command Data] to PIDBR[7:0].
9. Set PICR[ITEIE], PICR[IUE], PICR[SCLEA], and PICR[TB]. If PCMDx[SQC] is set to Pause for this command (indicating a pause after the command transmission), **or** if PCMDx[MBC] is clear and PCMDx[LC] is set (indicating the last command of the sequence), set PICR[STOP]. This sends the command byte in master-transmit mode to the power I^2C bus with a stop bit to terminate the communication.
10. Wait for the IDBR transmit empty interrupt, which automatically triggers the next step.
11. Read PISR, looking for PISR[ITE] and PISR[UB] to be set.
12. If the current PCMDx[MBC] bit is set, repeat steps 7-11 until a command is executed in which PCMDx[MBC] is clear.
13. If the current PCMDx[LC] or PCMDx[Pause] is set:
    a. Trigger the power manager to begin modification or power-off of the power supplies.
    b. Trigger the power manager to re-enable the clocks.
    c. Exit the voltage-change sequence and clear PVCR[VCSA].
14. If the current PCMDx[LC] is clear (PCMDx[LC] is set to Continue), increment PVCR[Read Pointer] and execute the next command. (repeat steps 1-14).

3.7.3 External Voltage Regulator Requirements

The external voltage regulator must meet the requirements described in the *Intel® PXA27x Processor Family EMTS*.

3.7.4 Sending Commands Using Voltage-Change Sequencer

The voltage-change sequencer sends single-byte, multi-byte, or sets of single- and multi-byte commands to the external regulator. These sequences can be used individually or as part of complex power-mode voltage changes.

3.7.4.1 Single-Byte Command Voltage Change

Power manager I^2C commands can be sent to the external regulator at any time. Sending a single command to the external regulator's I^2C module is efficient, using direct software control of the power manager I^2C bus. This sequence is the basic building block for more complex change sequences. To send a single command to the external regulator, use the following sequence, where:

n = number of the PCMD register containing the single-byte command (see Section 3.8.1.17)

1. Clear PCFR[FVC] (see Section 3.8.1.8) and PVCR[Command Delay] (see Section 3.8.1.13). These functions are not required when sending a single command to the external regulator.
2. Program PVCR[Slave Address] with the external regulator's I^2C address. Since PCMD register *n* contains the single-byte command, write *n* to PVCR[Read Pointer].
3. In PCMD*n*, clear the SQC, MBC and DCE bits. These functions are not required when sending a single command to the external regulator.
4. Set PCMD*n*[LC], which indicates that this is the last command.
5. Program PCMD*n*[Command Data] with the data to be sent to the external regulator.

6. Because PCMD*n*[LC] is set, the voltage-change sequencer ignores the remaining PCMD registers.
7. Enable power manager I^2C by setting PCFR[PI^2C_EN] **before** writing 0b1 to PWRMODE[VC]. PCFR[PI^2C_EN] must remain set as long as the power manager I^2C is being used on the I^2C bus.
8. Execute the voltage-change sequence by setting PWRMODE[VC]. See Section 3.8.3.2 for register details.

The voltage-change sequence begins as soon as PWRMODE[VC] is written with 0b1. One byte (PCMD*n*[Command Data]) is sent to the external regulator. When the sequence is complete, PVCR[VCSA] is cleared.

3.7.4.2 Single, Multiple-Byte Command Voltage-Change

The voltage-change sequencer can send a multi-byte I^2C command to the external regulator at any time. A multi-byte command is a single command that contains more than one byte of PCMDx[Command Data]. These bytes are sent without delay or re-arbitration for the I^2C bus between bytes, even if multiple masters are present on the bus. This sequence is a basic building block for more complex sequences. To send a single, multi-byte command to the external regulator, use the following sequence, where:

n = number of the PCMD register containing the multi-byte command (see Section 3.8.1.17)

m = number of bytes to be sent

1. Clear PCFR[FVC] (see Section 3.8.1.8) and PVCR[Command Delay] (see Section 3.8.1.13). These functions are not required when sending a single command to the external regulator.
2. Program PVCR[Slave Address] with the external regulator's I^2C address. Since PCMD register *n* contains the multi-byte command, write *n* to PVCR[Read Pointer].
3. In registers PCMD*n* through PCMD*n+m–1,* clear the SQC and DCE bits. These functions are not required when sending a single command to the external regulator.
4. In registers PCMD*n* through PCMD*n+m–2*, set the MBC bit. Setting MBC informs the sequencer logic that the next highest PCMD register contains an additional byte of Command Data to be sent as part of the command.
5. Clear PCMD*n+m–1*[MBC]. Clearing MBC informs the sequencer logic that the last byte of Command Data for the command is in the current register.
6. In registers PCMD*n* through PCMD*n+m–1,* set LC. This indicates that the first command in the PCMD is also the last (although the command contains several bytes).
7. In registers PCMD*n* through PCMD*n+m–1*, program Command Data with the data to be sent to the external regulator.
8. Because PCMD*n+m–1*[LC] is set and PCMD*n+m–1*[MBC] is cleared, the voltage-change sequencer ignores the remaining PCMD registers.
9. Enable power manager I^2C by setting PCFR[PI^2C_EN] **before** writing 0b1 to PWRMODE[VC]. PCFR[PI^2C_EN] must remain set as long as the power manager I^2C is being used on the I^2C bus.
10. Execute the voltage-change sequence by setting PWRMODE[VC]. See Section 3.8.3.2 for register details.

The voltage-change sequence begins as soon as PWRMODE[VC] is written with 0b1. *m* bytes (PCMD*n* through PCMD*n+m–1 Command Data*) are sent to the external regulator. When the sequence is complete, PVCR[VCSA] is cleared.

3.7.4.3 Multiple Single-Byte Command Voltage Change

The voltage-change sequencer can send up to 32 I^2C commands to the external regulator, with programmable delays between commands. Such commands control the ramp rate of the external regulator.

Regulators designed specifically for dynamic voltage control of the processor have built-in ramp control that can be specified with a minimum number of I^2C commands. However, to control the ramp beyond the regulator's built-in capability, or for regulators without built-in ramp control, multiple I^2C commands might be required.

To send multiple single-byte commands to the external regulator, use the following sequence, where:

n = number of the PCMD register containing the first command (see Section 3.8.1.17)

m = number of commands to be sent

1. Clear PCFR[FVC] (see Section 3.8.1.8). This function is not required when sending multiple commands to the external regulator.
2. Program PVCR[Command Delay] with the required delay between single-byte commands (this delay can control the ramp rate). For register details, see Section 3.8.1.13.
3. Program PVCR[Slave Address] with the external regulator's I^2C address. Since PCMD register *n* contains the first single-byte command, write *n* to PVCR[Read Pointer].
4. In registers PCMD*n* through PCMD*n+m–1,* clear the SQC and MBC bits. These functions are not required when sending multiple single-byte commands to the external regulator.
5. Set PCMD*n+m–1*[LC]. This indicates that PCMD*n+m–1* contains the last command.
6. In registers PCMD*n* through PCMD*n+m–1*, program Command Data with the data to be sent to the external regulator.

   For each command that is to be delayed by PVCR[Command Delay], set PCMDx[DCE].
7. Because PCMD*n+m–1*[LC] is set, the sequencer logic ignores the remaining PCMD registers.
8. Enable power manager I^2C by setting PCFR[PI^2C_EN] **before** writing 0b1 to PWRMODE[VC]. PCFR[PI^2C_EN] must remain set as long as the power manager I^2C is being used on the I^2C bus.
9. Execute the voltage-change sequence by setting PWRMODE[VC]. See Section 3.8.3.2 for register details.

The voltage-change sequence begins as soon as PWRMODE[VC] is written with 0b1. One byte is sent to the external regulator for each PCMD register, for a total of *m* bytes. Commands with PCMD[DCE] set are delayed by PVCR[Command Delay] before being sent. When the sequence is complete, PVCR[VCSA] is cleared.

3.7.4.4 Multiple Single- and Multi-Byte Command Voltage Change

Depending on the type of external regulator used, some commands may need to be single-byte commands and others multiple-byte commands. Additionally, some commands may require a delay between commands, while other commands may need to be sent immediately following the previous command. To send multiple, single, and multi-byte commands to the external regulator, use the following sequence, where:

n = number of the PCMD register containing the first command (see Section 3.8.1.17)

m = number of commands to be sent

t = total number of command bytes to be sent

1. Clear PCFR[FVC] (see Section 3.8.1.8). This function is not required when sending multiple commands to the external regulator.
2. Program PVCR[Command Delay] with the required delay between commands (this delay can control the ramp rate). For register details, see Section 3.8.1.13.
3. Program PVCR[Slave Address] with the external regulator's I^2C address.
4. Since PCMD register *n* contains the first command of the sequence, write *n* to PVCR[Read Pointer].
5. For registers PCMD*n* through PCMD*n+t–1*, clear the SQC bits, which are not required when sending multiple commands to the external regulator.
6. For each multi-byte command of length *m* starting at location *x*:
   a. Set MBC in registers PCMD*x* through PCMD*x+m–2*. MBC set indicates that the following PCMD register contains an additional byte of Command Data.
   b. Clear MBC in PCMD*x+m*. MBC clear indicates that the current PCMD register contains the last byte of Command Data for the multi-byte command.
   c. If a delay is required before executing the multi-byte command, set PCMD*x*[DCE].
7. For each single-byte command starting at PCMD*x*, clear PCMD*x*[MBC].
8. If the last command of the sequence is a multi-byte command of *m* bytes,
   a. For registers PCMD*n+t–m-2* through PCMD*n+t–1*, set LC.
   b. For registers PCMD*n+t–m-2* through PCMD*n+t–2*, set MBC.
   c. Clear PCMD*n+t–1*[MBC]. MBC clear indicates that the last command is contained in registers PCMD*n+t–m-2* through PCMD*n+t–1*.
9. For registers PCMD*n* through PCMD*n+t–1*, program Command Data with the data to be sent to the external regulator for each command.
10. Because PCMD*n+t–1*[LC] is set, the sequencer logic ignores the remaining PCMD registers.
11. Enable power manager I^2C by setting PCFR[PI^2C_EN] **before** writing 0b1 to PWRMODE[VC]. PCFR[PI^2C_EN] must remain set as long as the power manager I^2C is being used on the I^2C bus.
12. Execute the voltage-change sequence by setting PWRMODE[VC]. See Section 3.8.3.2 for register details.

The voltage-change sequence begins as soon as PWRMODE[VC] is written with 0b1.

For each single-byte command, one byte of Command Data is sent to the external regulator. For each multi-byte command, multiple bytes are sent, as specified by the MBC bits. A total of *t* bytes is sent. Commands with PCMD[DCE] set are delayed by PVCR[Command Delay] before being sent.

When the sequence is complete, PVCR[VCSA] is cleared.

3.7.5 Behavior During Power-Fault Assertion

If nBATT_FAULT or nVDD_FAULT is asserted while the voltage manager is transmitting a command and the corresponding xIDAE bits (BIDAE, VIDAE) are clear in the Power Manager Control register (PMCR), the command sequence is aborted with a STOP condition on the I^2C bus after completing the next command in the sequence. In this case, programmed delay between commands, if any, is ignored. Even if the command being transmitted is part of a multi-byte command, the behavior is the same as above.

If the nBATT_FAULT or nVDD_FAULT occurs during the programmed delay between commands while the corresponding PMCR[xIDAE] bits are clear, the sequence terminates after executing the command following the delay.

If nBATT_FAULT or nVDD_FAULT is asserted while the corresponding PMCR[xIDAE] bits are set, the command sequence is not terminated.

3.7.6 Using the Voltage Manager

The power manager I^2C bus and voltage-change sequencer are normally used to control the voltage applied to the internal logic (supplied by VCC_CORE) based on frequency. The voltage can be adjusted at any time relative to software execution, provided that the applied voltage meets the requirements for the frequency in use at the time. However, the most efficient use of the voltage manager is achieved when used in conjunction with the power-mode and frequency controls.

3.7.6.1 Voltage Change at Initialization

Voltage regulators for the PXA27x processor must power on with the correct default voltages. For details, see the *Intel® PXA27x Processor Family EMTS*. During the boot sequence, the power manager I^2C can be used to adjust the regulator output voltages to meet the system requirements. These adjustments can be made with direct software access to the power manager I^2C, so that use of the voltage-change sequencer is not necessary.

3.7.6.2 Coupling Voltage Change with Turbo Modes

Adjusting the processor frequency and voltage according to application's requirements achieves optimal system power consumption. The fastest way of adjusting the processor frequency is through the use of turbo mode (see Section 3.5.7.4) or half-turbo mode (see Section 3.5.7.5), which adjusts the frequencies of the CPU and the system bus without affecting operation of the peripheral modules. To adjust the voltage accordingly, follow these steps.

To raise the voltage and frequency:

1. At boot-up, set the turbo mode to run mode ratio (CCCR[2N]) to the required values. If required, execute a frequency change.

2. Program the PCMD and the PVCR registers to send the types of commands necessary to raise the voltage. For example, for a controlled ramp, follow the procedure given in Section 3.7.4.3.
3. Initiate a voltage-change sequence by setting PWRMODE[VC].
4. Wait for PVCR[VCSA] to clear by periodically polling the register.
5. Enable turbo mode (or half-turbo mode) by writing to the appropriate bits in the CLKCFG register.

To lower the voltage and frequency:

1. At boot-up, set the turbo mode to run mode ratio (CCCR[2N]) to the required values. If required, execute a core frequency change, turbo-mode change, or fast-bus mode change.
2. Program the PCMD and the PVCR registers to send the types of commands necessary to lower the voltage. For example, for a controlled ramp, follow the procedure given in Section 3.7.4.3.
3. Exit turbo mode (or half-turbo mode) by writing to the appropriate bits in the CLKCFG register.
4. Initiate a voltage-change sequence by setting PWRMODE[VC].
5. The voltage change is complete when PVCR[VCSA] is clear.

3.7.6.3 Coupling Voltage Change with Frequency Change

A frequency change (clock source change or core PLL frequency change) can be used to change the frequency of the CPU, system bus, memory controller, and LCD controller to a value not available with turbo or fast-bus modes. This change can be coupled with a voltage change in a similar way as turbo mode. The only additional requirement is that PCFR[FVC] be set. Similarly, voltage change can be coupled with fast-bus mode. For details of these frequency changes, see sections Section 3.5.7.3 and Section 3.5.7.6.

In the case of a core PLL frequency change where the core PLL is also the core clock source (CCCR[CPDIS] = 0), there will be a delay while the PLL re-locks. To reduce the overall delay caused by the frequency change and the voltage change as well as the software overhead required to do both, use the following sequence to change the voltage and frequency:

1. Set the turbo-mode-to-run-mode ratio (CCCR[2N]) to the required values.
2. Program the PCMD registers with the required commands. For each command, set PCMDx[SQC] to Continue. For the last command, set PCMD[LC].

   The first command is executed as soon as the power manager asserts the request. After the last command is executed, the voltage manager triggers the clocks manager to perform a frequency change.
3. Set PCFR[FVC].
4. Initiate a frequency-change sequence by writing to CLKCFG[F] (or CLKCFG[B] if fast-bus mode is to be used).
5. The frequency-change sequence exits at the new voltage and frequency.

Note: Observe the appropriate frequency/voltage specification (refer to the *Intel® PXA27x Processor Family EMTS* for details) for the VCC_CORE power domain when initiating any frequency or frequency-coupled voltage changes.

3.7.6.4 Coupling Voltage Change with Power-Mode Changes

Low-power modes (deep-idle, idle, standby, sleep, and deep-sleep) can significantly reduce power consumption by gating clocks, reducing leakage, or powering off large sections of the processor. Before entering these modes, the voltage can be adjusted for optimum power consumption.

The voltage can be changed in anticipation of entry into one of these power modes in the same way that frequency and voltage can be changed, requiring a frequency change and a voltage change before the power mode is changed and again after the power-mode change is finished, potentially increasing the latency of the power-mode change. Power-mode changes can be more efficiently coupled with voltage change by using the following sequences.

To raise the voltage during a power-mode change:

1. Set the turbo mode to run mode ratio (CCCR[2N]) to the required value.
2. Program the PCMD registers in the following order:
   a. The first set of commands consists of those that may be required by the external voltage regulator but do not actually change the voltage. These commands all have the SQC bits set to Continue. The first command is executed as soon as the power manager asserts a request to the voltage manager.
   b. The second set of commands consists of those that actually raise the voltage; it can be a ramp or a single command. Because the voltage must be raised before the power mode is entered, these commands all have the SQC bits set to Continue, except the last command of this set, which has the SQC bits set to Pause. After execution of this command, the voltage manager sends a trigger to the power manager to enter the power mode.
   c. The third set of commands consists of those that again lower the voltage after the power-mode change is complete (these commands are executed after the power-mode wake-up source is asserted and the PWR_EN and SYS_EN pins and associated timers have been asserted). The last command in this set must have the PCMDx[LC] set. After executing the last command, the voltage manager again sends a trigger to the power manager, signaling the end of the voltage-change sequence.
3. Initiate a voltage-change sequence and power-mode change concurrently by writing to PWRMODE[VC] and PWRMODE[M] at the same time.

This sequence raises the voltage, enters the power mode, exits the power mode according to its normal exit mechanism, and then lowers the voltage.

To lower the voltage during a power-mode change:

1. Set the turbo-mode-to-run-mode ratio (CCCR[2N] to the required value.
2. Program the PCMD registers in the following order:
   a. The first set of commands consists of those that may be required by the external voltage regulator but do not actually change the voltage. These commands all have the SQC bits set to Continue. The first command is executed as soon as power manager asserts a request to the voltage manager.
   b. The second set of commands consists of those that actually lower the voltage; it can be a ramp or a single command. Since clocks were already stopped before the power manager asserted the request, these commands have the SQC bits set to Continue, except the last command of this set, which has the SQC bits set to Pause. After the execution of this command, the voltage manager sends a trigger to the power manager to enter the power mode.

c. The third set of commands consists of those that again raise the voltage after the power mode is complete (these commands are executed after the power-mode wake-up source is asserted and the PWR_EN and SYS_EN pins and associated timers have been asserted). The last command in this set must have the LC bit set. After executing the last command, the voltage manager sends a trigger to the power manager signaling the end of voltage-change sequence.

3. Initiate a voltage-change sequence and power-mode change concurrently by writing to PWRMODE[VC] and PWRMODE[M] at the same time.

This sequence lowers the voltage, enters the power mode, exits the power mode according to its normal exit mechanism, and then raises the voltage.

3.7.6.5 Alternate Method: Voltage, Frequency, and Power-Mode Changes

An alternative means of performing a voltage change coupled with a frequency change does not require the use of the PWRMODE[VC] or PVCR[VCSA] functionality. The following sequences illustrate this alternative method, which is as efficient as using the automatic power manager I^2C coupling described in Section 3.7.6.3.

To raise the voltage and frequency (for example, enter turbo mode or increase the value of L):

1. Perform the voltage change by instructing the external power-management device to raise the voltage to the final level.
2. Optionally, the external regulator can generate an interrupt after the transfer is completed.
3. If necessary, add a delay to allow for the time required by the external device to change the voltage.
4. Perform the frequency change.

To lower the voltage and frequency (for example, exit turbo mode or decrease the value of L):

1. Perform the frequency change.
2. Perform the voltage change by instructing the external power-management device to lower the voltage to the final level.
3. Optionally, the external regulator can generate an interrupt after the transfer is completed.

To couple a voltage change with a power-mode change, use a sequence that is similar to the ones shown above.

3.8 Register Descriptions

The following sections describe the registers used by the clocks and power manager:

Section 3.8.1 — Power Manager Registers

Section 3.8.2 — Clocks Manager Registers

Section 3.8.3 — Coprocessor 14: Clock and Power Management

3.8.1 Power Manager Registers

The power manager uses the following 32-bit registers: