Prospector P1100

User Guide

ARM

ARM DUI 0122A

Prospector P1100
User Guide

Release information

Change history

Date	Issue	Change
9 March 2000	A	New document

Proprietary notice

ARM, the ARM Powered logo, Thumb and StrongARM are registered trademarks of ARM Limited.

The ARM logo, AMBA, Angel, ARMulator, EmbeddedICE, ModelGen, Multi-ICE, PrimeCell ARM7TDMI, ARM7TDMI-S, ARM9TDMI, TDMI and STRONG are trademarks of ARM Limited.

All other products or services mentioned herein may be trademarks of their respective owners.

Neither the whole nor any part of the information contained in, or the product described in, this document may be adapted or reproduced in any material form except with the prior written permission of the copyright holder.

The product described in this document is subject to continuous developments and improvements. All particulars of the product and its use contained in this document are given by ARM Limited in good faith. However, all warranties implied or expressed, including but not limited to implied warranties or merchantability, or fitness for purpose, are excluded.

This document is intended only to assist the reader in the use of the product. ARM Limited shall not be liable for any loss or damage arising from the use of any information in this document, or any error or omission in such information, or any incorrect use of the product.

Document confidentiality status

This document is Open Access. This document has no restriction on distribution.

Product status

The information in this document is Final (information on a developed product).

ARM web address

`http://www.arm.com`

Electromagnetic conformity

This section contains *electromagnetic conformity* (EMC) notices.

Federal Communications Commission Notice

NOTE: This equipment has been tested and found to comply with the limits for a class A digital device, pursuant to part 15 of the FCC rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This equipment generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference in which case the user will be required to correct the interference at his own expense.

Caution

Changes or modifications not expressly approved by ARM Ltd. could void the user's authority to operate this equipment.

CE Declaration of Conformity

Warning

This is a class A product. In a domestic environment this product may cause radio interference in which case the user may be required to take adequate measures.

Contents
Prospector P1100 User Guide

Chapter 4 **Programmer's Reference**

Appendix A **Connector reference**

Appendix B **System Expansion Options**

List of Tables
Prospector P1100 User Guide

List of Figures
Prospector P1100 User Guide

Preface

This preface introduces the ARM Prospector P1100 and its reference documentation. It contains the following sections:

- *About this document* on page xii
- *Further reading* on page xv
- *Feedback* on page xvi.

About this document

This document is the user guide for the Prospector P1100 development system.

Intended audience

This document has been written for system hardware and software developers of ARM-based products.

Using this manual

This document is organized into the following chapters:

Chapter 1 *Introduction*
Read this chapter for a general description of the P1100, including the product options.

Chapter 2 *Getting started*
Read this chapter for a description of how to start using the P1100. Information about connecting power and peripherals are contained in this chapter.

Chapter 3 *Hardware Description*
Read this chapter for description of the major hardware devices. This chapter contains details about the hardware architecture of the various parts of the system.

Chapter 4 *Programmer’s Reference*
Read this chapter for a description of the system memory map and programming interfaces of the major devices. This chapter provides descriptions of the interface protocols for the major devices and provides timing information.

Appendix A *Connector reference*
Refer to this appendix for the connector pinouts.

Appendix B *System Expansion Options*
Refer to this appendix for a description of the system expansion options.

Typographical conventions

The following typographical conventions are used in this document:

bold Highlights ARM processor signal names, and interface elements such as menu names. Also used for terms in descriptive lists, where appropriate.

italic Highlights special terminology, cross-references and citations.

`typewriter` Denotes text that may be entered at the keyboard, such as commands, file names and program names, and source code.

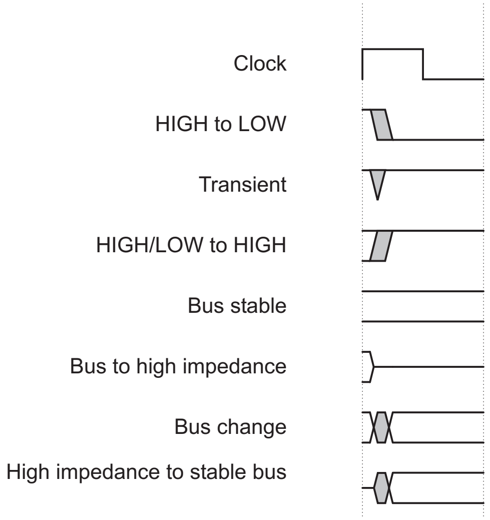
`typewriter` Denotes a permitted abbreviation for a command or option. The underlined text may be entered instead of the full command or option name.

`typewriter italic`
Denotes arguments to commands or functions where the argument is to be replaced by a specific value.

`typewriter bold`
Denotes language keywords when used outside example code.

Timing diagram conventions

This manual contains one or more timing diagrams. The following key explains the components used in these diagrams. Any variations are clearly labelled when they occur. Therefore, no additional meaning should be attached unless specifically stated.



Key to timing diagram conventions

Shaded bus and signal areas are undefined, so the bus or signal can assume any value within the shaded area at that time. The actual level is unimportant and does not affect normal operation.

Further reading

This section lists publications by ARM Limited, and by third parties.

ARM periodically provides updates and corrections to its documentation. See `http://www.arm.com` for current errata sheets and addenda.

See also the ARM Frequently Asked Questions list at:
`http://www.arm.com/DevSupp/Sales+Support/faq.html`

ARM publications

The following publications provide reference information about ARM architecture:

- *AMBA Specification Rev 2.0* (ARM IHI 0011)
- *ARM Architectural Reference Manual* (ARM DDI 0100).

The following publications provide information about ARM SDT 2.5:

- *ARM Software Development Toolkit User Guide* (ARM DUI 0040)
- *ARM Software Development Toolkit Reference Guide* (ARM DUI 0041).

The following publications provide information about the ARM Developer Suite:

- *Getting Started* (ARM DUI 0064)
- *ADS Tools Guide* (ARM DUI 0067)
- *ADS Debuggers Guide* (ARM DUI 0066)
- *ADS Debug Target Guide* (ARM DUI 0058)
- *ADS Developer Guide* (ARM DUI 0056)
- *ADS CodeWarrior IDE Guide* (ARM DUI 0065).

Further information can be obtained from the ARM website at:

`http://www.arm.com`

Other publications

This section lists relevant documents published by third parties.

- *Intel StrongARM SA-1100 Microprocessor Developer's Manual*
  April 1999, Intel Corporation.

Feedback

ARM Limited welcomes feedback both on the Prospector, and on the documentation.

Feedback on the Prospector/P1100

If you have any comments or suggestions about this product, please contact your supplier giving:

- the product name
- a concise explanation of your comments.

Feedback on this document

If you have any comments about this document, please send email to `errata@arm.com` giving:

- the document title
- the document number
- the page number(s) to which your comments refer
- a concise explanation of your comments.

General suggestions for additions and improvements are also welcome.

Chapter 1
Introduction

This chapter introduces the Prospector P1100 development system and contains the following sections:

- *About the Prospector P1100* on page 1-2
- *System overview* on page 1-4
- *Software development tools* on page 1-7.

1.1 About the Prospector P1100

The Prospector P1100 provides a flexible and portable StrongARM development and evaluation system. It enables you to work with a system that models your end product and allows you to realistically demonstrate your designs for mobile and hand-held applications. Application might include, for example, wireless-based systems, in-vehicle systems, and games consoles.

At the heart of every P1100 system is a the P1100 CPU board.

The P1100 is supplied at two levels:

- Compact system – a CPU board in small enclosure with power supply.
- Expanded system – a CPU board, keyboard, and LCD display in a larger enclosure. An expanded system is illustrated in Figure 1-1 on page 1-3.

Expanded systems are available with a choice of two displays:

- 8.4 inch VGA (640 x 480) 16bpp single panel with reflective screen
- 8.4 inch VGA (640 x 480) 16bpp single panel with transmissive screen, backlight, and touchscreen.

All P1100 systems are shipped with an external power supply and software development tools.

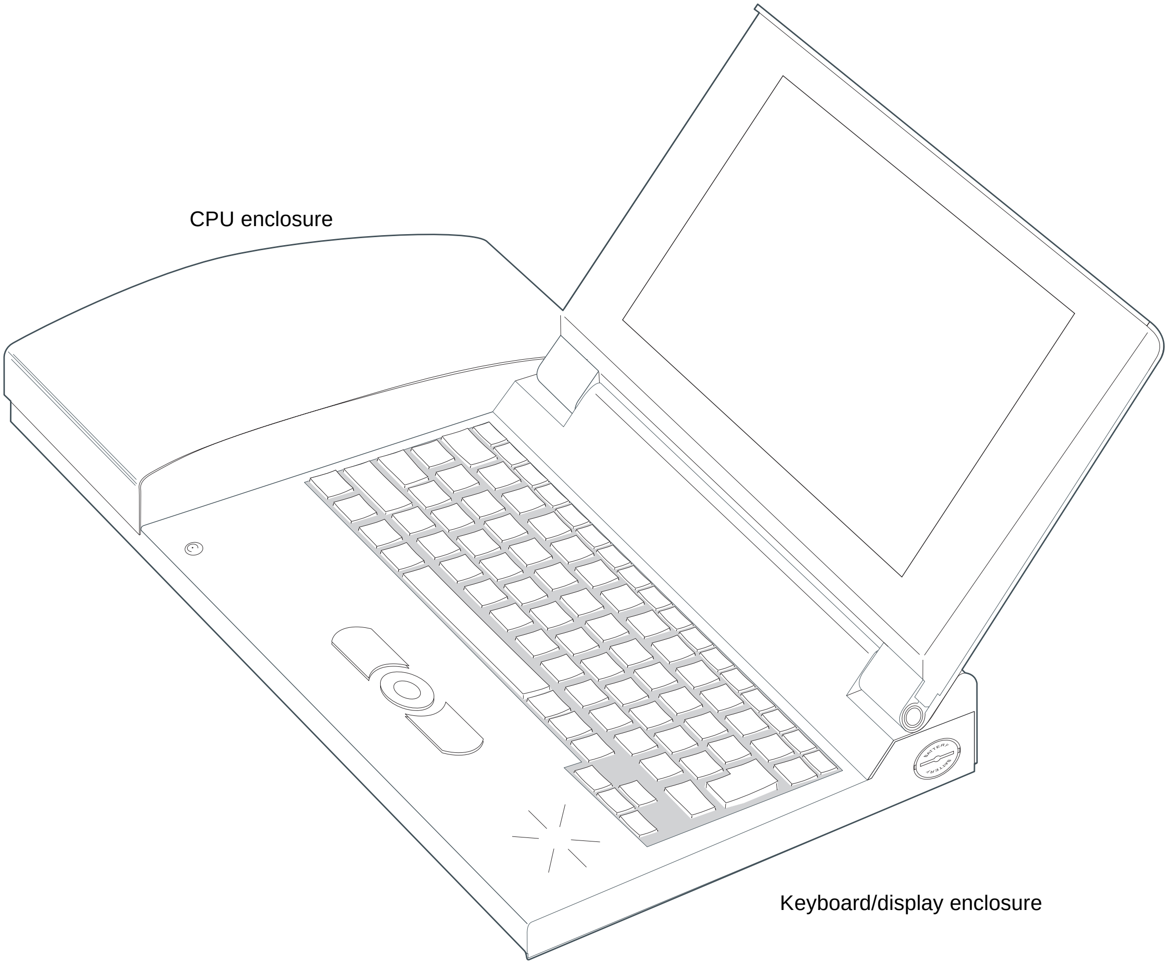


Figure 1-1 Prospector P1100 expanded system

1.2 System overview

Figure 1-1 shows the architecture of the Prospector P1100 system.

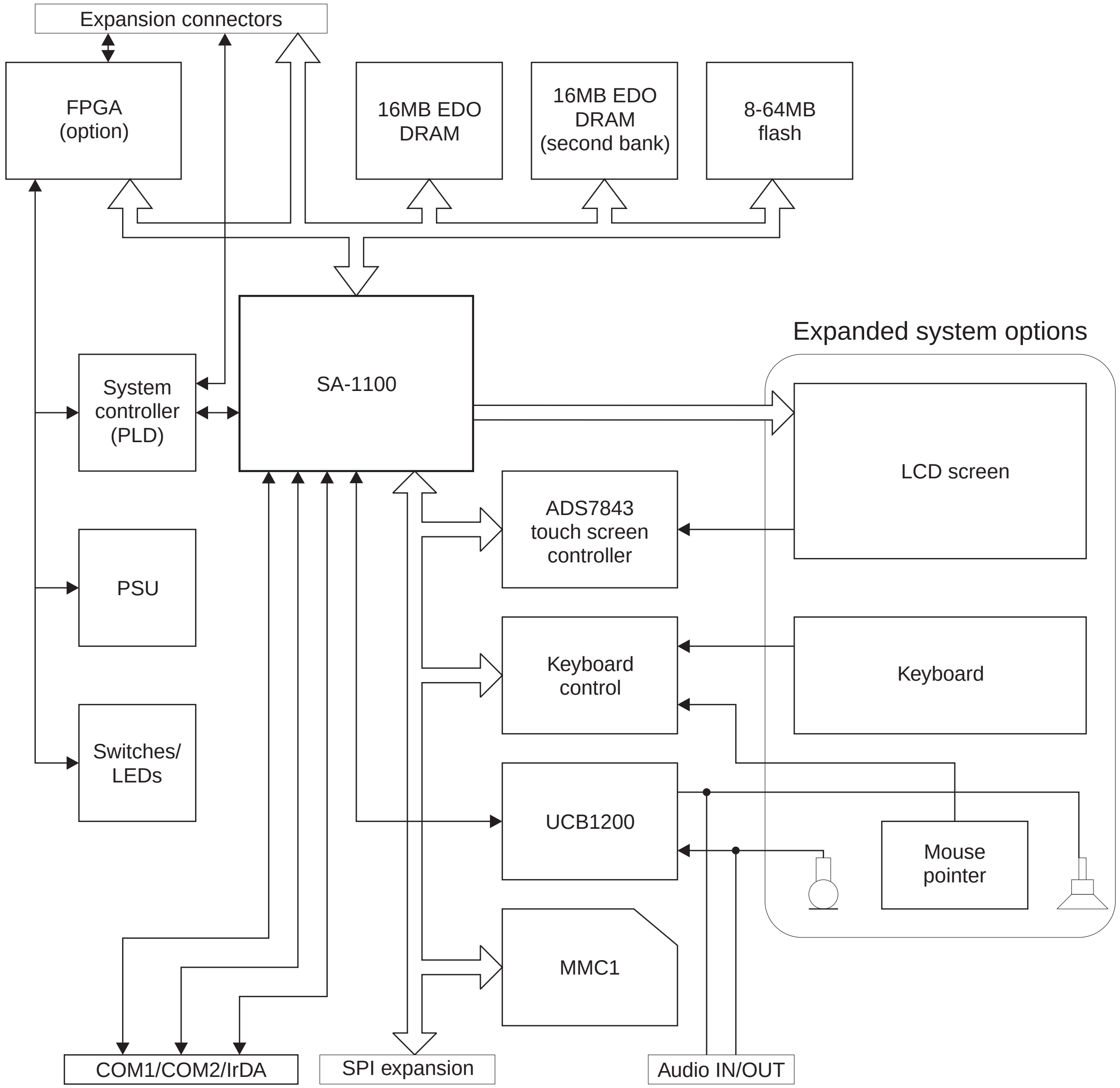


Figure 1-2 System architecture

1.2.1 CPU board

The CPU module houses the main P1100 CPU board. This features the following:

- StrongARM SA-1100 microprocessor
- 8-64MB Flash (32MB fitted as standard)
- 32MB EDO DRAM, (two banks of 16MB each)
- system controller *Programmable Logic Device* (PLD) providing:
  - power supply mode control
  - interrupt control
  - flash chip-select control
  - boot mode control.
- two RS232 COMs ports
- IrDA port
- keyboard controller
- touch screen controller
- SPI expansion (option)
- *Multi Media Card* (MMC) socket.

Figure 1-3 on page 1-6 shows the layout of the CPU board upper surface.

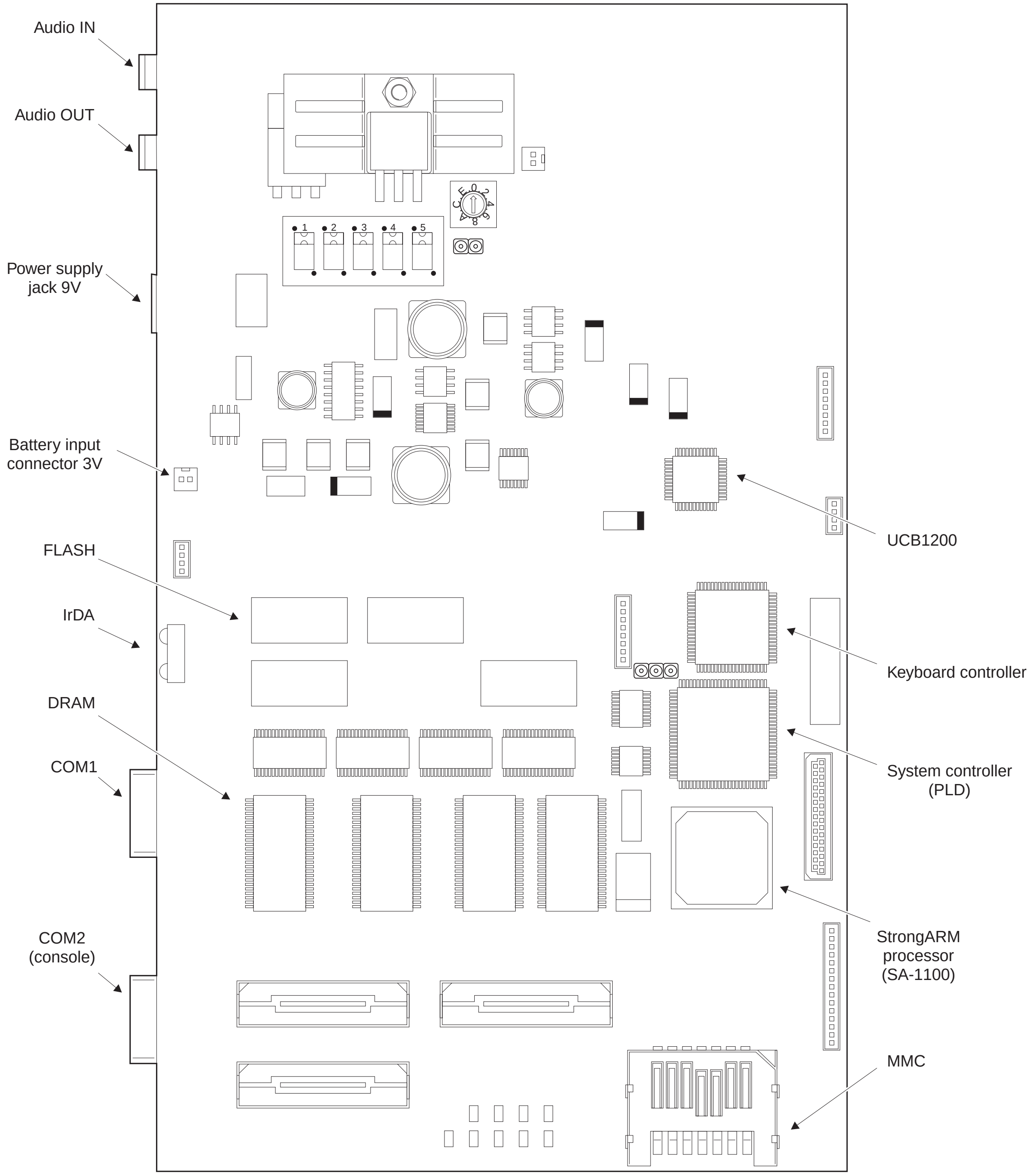


Figure 1-3 CPU board layout (top)

1.3 Software development tools

The P1100 is shipped with software development tools to aid your product development. This includes:

- *ARM Firmware Suite* (AFS)
- *ARM Developer Suite* (ADS) evaluation copy

1.3.1 AFS

The *ARM Firmware Suite* (AFS) is a collection of low-level software and utilities which aid you to:

- evaluate and benchmark ARM-based platforms
- port operating systems and applications
- develop applications and product architectures.

It provides you with a flexible environment for developing ARM-based products by providing:

- simplified applications programming interfaces
- operating system neutrality
- reusable code over a range or ARM and third-party development platforms
- a set of tried and tested functions that enable rapid development of applications and device drivers.

AFS provides a collection of low-level software which runs on a wide range of ARM-based platforms. The primary functions of the AFS are to:

- identify and initialize the system processor
- identify and initialize the system memory
- identify and initialize the system buses
- identify and initialize system devices, such as serial interfaces and timers
- initialize and handle interrupts.

The AFS provides supports for product development for users of SDT 2.5 or ADS 1.0 and can be used with Angel, Multi-ICE, and third-party debug monitors.

For more information about AFS, see the *ARM Firmware Suite Reference Guide*.

1.3.2 ADS

ADS is a software development suite used for creating applications for the ARM architecture. The suite comprises:

- Code generation tools – Embedded C++ and C compilers, assembler and linker for ARM and Thumb instruction sets
- An integrated development environment for Windows – CodeWarrior® IDE from Metrowerks
- Powerful GUI debuggers
- Instruction set simulators
- ROM-based debug tools (AFS)
- On-line documentation
- ARM Applications Library
- Real-time Trace support

For further information, refer to the ADS documentation.

Chapter 2
Getting started

This chapter describes how to start working with the P1100. It contains the following sections:

- *Interface connectors* on page 2-2
- *Supplying power* on page 2-4
- *Removing the CPU module cover* on page 2-6
- *Setting the switches* on page 2-3
- *System startup* on page 2-7.

2.1 Interface connectors

Figure 2-1 shows the external interface connectors on the side of the motherboard enclosure.

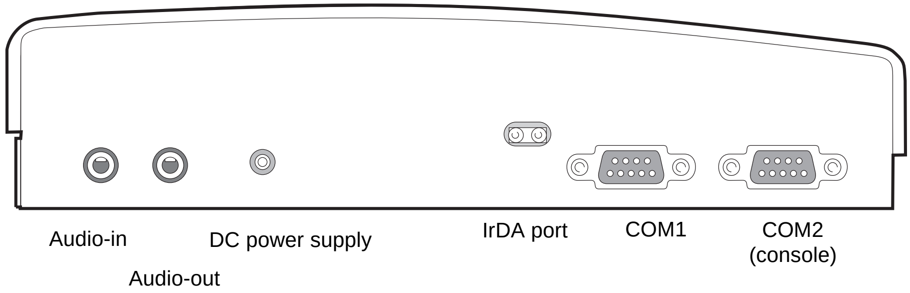


Figure 2-1 External interface connectors

2.2 Setting the switches

The P1100 CPU board provides two types of switches:

- a five element dual in-line (DIL) switch pack
- hexadecimal rotary switch.

The functions of the DIL switch elements are as follows:

SW1 – SW3 General purpose, see *Software readable switches* on page 3-21.

SW4 DEVMODE – controls the system controller PLD response to reset input signal (see *Boot and mode control* on page 3-14). The default setting is DEVMODE OFF.

SW5 BOOTSEL – controls the boot image selection at system start up (as described in *System startup* on page 2-7). The default setting for BOOTSEL selects the boot monitor.

Figure 2-2 shows the location and settings of the DEVMODE and BOOTSEL switches.

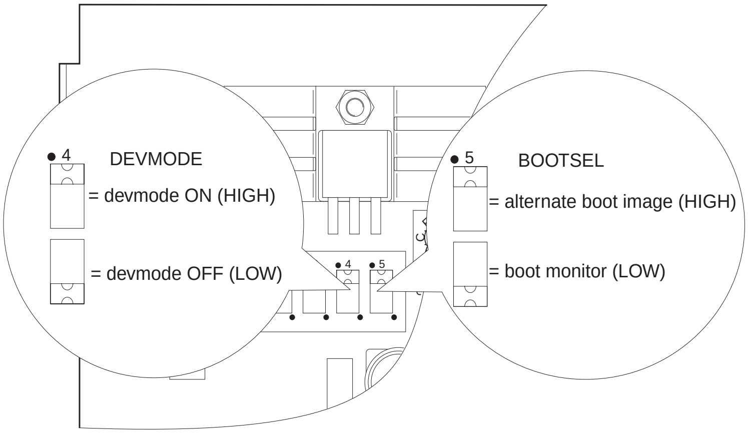


Figure 2-2 BOOTSEL and DEVMODE switches

The hexadecimal switch and remaining three elements within the DIL switch pack are intended for general purpose use (see *LED control and switch sensing* on page 3-20).

2.3 Supplying power

Depending on the type of system, you can power the P1100 as follows:

- compact systems can be powered using the external DC *Power Supply Unit* (PSU) supplied with the system.
- systems with a reflective display can be powered with the PSU or with batteries
- system with transmissive (backlit) display can only be powered with the PSU.

The system powers up immediately when either a PSU is connected or batteries are installed. During system startup, the system boot monitor displays self test information on a terminal connected to the console port (see Figure 2-1 on page 2-2). For systems with a keyboard and display fitted, the boot monitor uses the built-in display. Data entry can then be through either the console port or keyboard interface.

2.3.1 Connecting the external power supply

Connect power to the P1100 as follows:

1. Connect the DC out lead from the PSU to the DC power supply socket, see Figure 2-1 on page 2-2.
2. Plug the PSU into a suitable AC socket outlet.

2.3.2 Installing batteries

Caution

The P1100 CPU board is not polarity protected. To prevent damage, ensure the batteries are fitted the correct way round.

Systems units with a display provide a battery compartment for two AA-sized batteries. The battery compartment is located beneath the display hinges and can be accessed from the right side of the system unit, as shown in Figure 2-3 on page 2-5.

Note

Systems fitted with a backlit display cannot be powered from batteries.

The P1100 does not incorporate a battery charger.

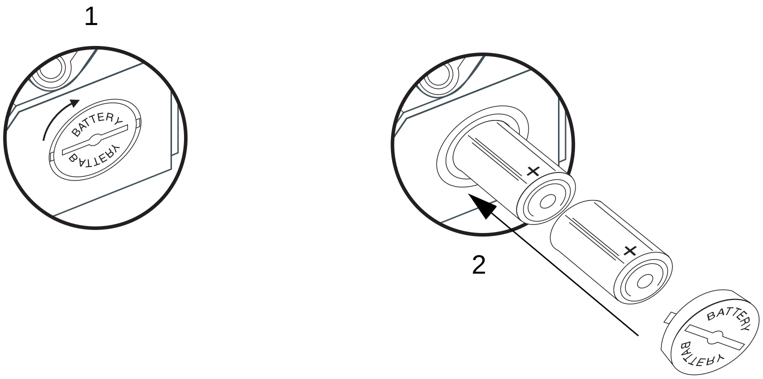


Figure 2-3 Battery installation

Referring to Figure 2-3, install batteries as follows:

1. Remove the battery cap by turning it one quarter turn counter-clockwise.
2. Insert two AA-size batteries
3. Replace the cap and re-secure by it turning it one quarter turn clockwise.

The system powers up immediately.

Note

Battery provision is intended for experimentation with battery operation and power control rather than for continuous operation.

2.4 Removing the CPU module cover

To gain access to the configuration links and switches it is necessary to remove the CPU module top cover, as shown in Figure 2-4.

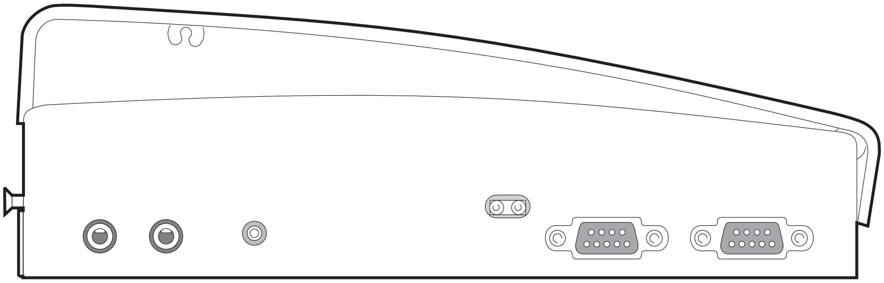


Figure 2-4 CPU module top cover removal

Remove the top cover as follows:

1. Loosen the screw at the rear of the CPU module.
2. Slide the back of the cover upwards until it clear of the CPU unit housing.
3. Slide the cover forwards.

On systems with a display, it may be necessary to open the display housing lid slightly to remove and refit the CPU module cover.

Note

The Prospector/P1100 has been tested with the cover in place and found to comply with FCC class A limits (see *Electromagnetic conformity* on page -iii). Operating the P1100 with covers removed may mean that it no longer complies and may interfere with other equipment. It is suggested that you remove the cover to make necessary adjustments and then replace the cover before operating the system.

2.5 System startup

When The P1100 powers up, the SA-1100 executes its first instruction from address `0x000000` (the reset vector). This is typically a jump instruction, because the first eight words are reserved for exception vectors (known as hard vectors in the *ARM Architecture Reference Manual*). The SA-1100 maps these addresses to static memory, but these are remapped by the *ARM Firmware Suite* (AFS). The AFS carries out this remapping as part of the initialization sequence, remapping the hard vectors to DRAM for application use. For example, to implement a system- or application-specific interrupt handler.

2.5.1 Boot switcher

The startup code for the P1100 includes a boot switcher routine. This reads SW5 in the DIL switch pack and then, depending on the setting either runs the boot monitor or an alternative image from flash. By using this mechanism, the boot switcher can relocate the boot monitor to DRAM before it is invoked (which is necessary if you want to reprogram the flash).

The boot switcher functions as follows:

- If the BOOTSEL switch (see Figure 2-2 on page 2-3) is LOW the boot monitor code is executed. The boot monitor is a reserved image in flash, with the image number 4280910 (`0x41524E`).
- If the BOOTSEL is (HIGH), the boot switcher:
  - Searches for a *System Information Block* (SIB) stored in the flash.
  - If a SIB is found, the image number is read and flash is examined for a matching image.
  - If an image match is detected, and the image has a valid CRC, the image is either executed from flash, or downloaded to DRAM and then executed.
  - If the boot switcher fails to find a valid image, or if the CRC fails, it executes the boot monitor.

2.5.2 Memory initialization

The boot monitor and boot switcher both use the standard µHAL memory initialization mechanism for The P1100. This tests the DRAM and remaps the memory spaces as follows:

- Checks for a second bank, then concatenates the two banks of DRAM to a single virtual address space starting at VA `0x00000000`.
- Remaps the flash to VA `0x04000000`.
- Maintains 1-to-1 mapped DRAM spaces for the fitted DRAM.

- Assigns 1-to-1 mapping to all other system input/output resources (SA-1100 internal and external devices).
- Sets protection for all unmapped areas so that they abort if accessed.

2.5.3 Initialization completion

When execution returns to the boot monitor, it:

- sets up a basic µHAL stack and memory environment
- sets up a basic µHAL timer environment
- sets up a basic µHAL interrupt handling environment
- displays a status report and a monitor prompt on the console.

Note

For systems with a keyboard and display fitted, the boot monitor configures the display for a blue screen text output, and replicates the console port output onto it. Data entry can then be through either the console port or keyboard.

An Angel debug monitor is provided in flash as image 911. Angel is also copied to DRAM before being executed. This enables the *ARM Flash Utilities* (`afu` and `bootfu`) to reprogram flash.

For details of the ARM Firmware Suite, refer to the *ARM Firmware Suite Reference Guide*.

Chapter 3
Hardware Description

This chapter describes the P100 hardware. It contains the following sections:

- *SA-1100 CPU* on page 3-2
- *System clocks* on page 3-6
- *Reset control* on page 3-8
- *System controller PLD* on page 3-11
- *LED control and switch sensing* on page 3-20
- *Memory subsystem* on page 3-23
- *Serial interfaces* on page 3-26
- *SPI bus* on page 3-29
- *Keyboard controller* on page 3-32
- *MMC interface* on page 3-37
- *Touch screen controller* on page 3-38
- *Philips UCB1200 codec and touch screen controller* on page 3-39
- *Display interface* on page 3-43
- *Distribution board* on page 3-44
- *Power supply* on page 3-45
- *JTAG interface* on page 3-51.

3.1 SA-1100 CPU

The Intel StrongARM SA-1100 is a general purpose 32-bit RISC processor which includes:

- 16Kbyte instruction cache
- 8Kbyte data cache
- 16-entry minicache
- memory management unit
- write buffer.

The SA-1100 is software compatible with the ARM v4 architecture.

In addition to the processor core, the SA-1100 incorporates system support modules:

- memory and PCMCIA controller
  - supporting EDO DRAM, and SRAM or flash
  - PCMCIA control signals (available for expansion).
- system control module
  - 28 *General Purpose Input/Output* (GPIO) pins
  - interrupt controller
  - real-time clock, watchdog, and interval timer
  - power management controller
  - reset controller
  - two on-chip oscillators.
- peripheral control module:
  - 6-channel DMA controller
  - LCD controller with programmable screen resolution,
  - five dedicated serial ports.

The core can be clocked at a range of clock frequencies (between 59 and 191.7MHz) with the upper limit determined by the speed grade of CPU fitted. The SA-1100 operates at a fixed core voltage of 1.5V and a fixed input/output voltage of 3.3V (10% tolerance).

Note

The IO is not 5V tolerant, buffering must be provided to isolate the part from any 5V logic domains. 74LCX devices are used as level converters. These parts have a +3V supply but are 5V tolerant. Their output switching range is within the TTL switching threshold. They are used to buffer 3V domain signals to the expansion connectors.

The SA-1100 variant used is packaged in a 256-pin *mini-Ball Grid Array* (mBGA).

Figure 3-1 shows the architecture of the SA-1100.

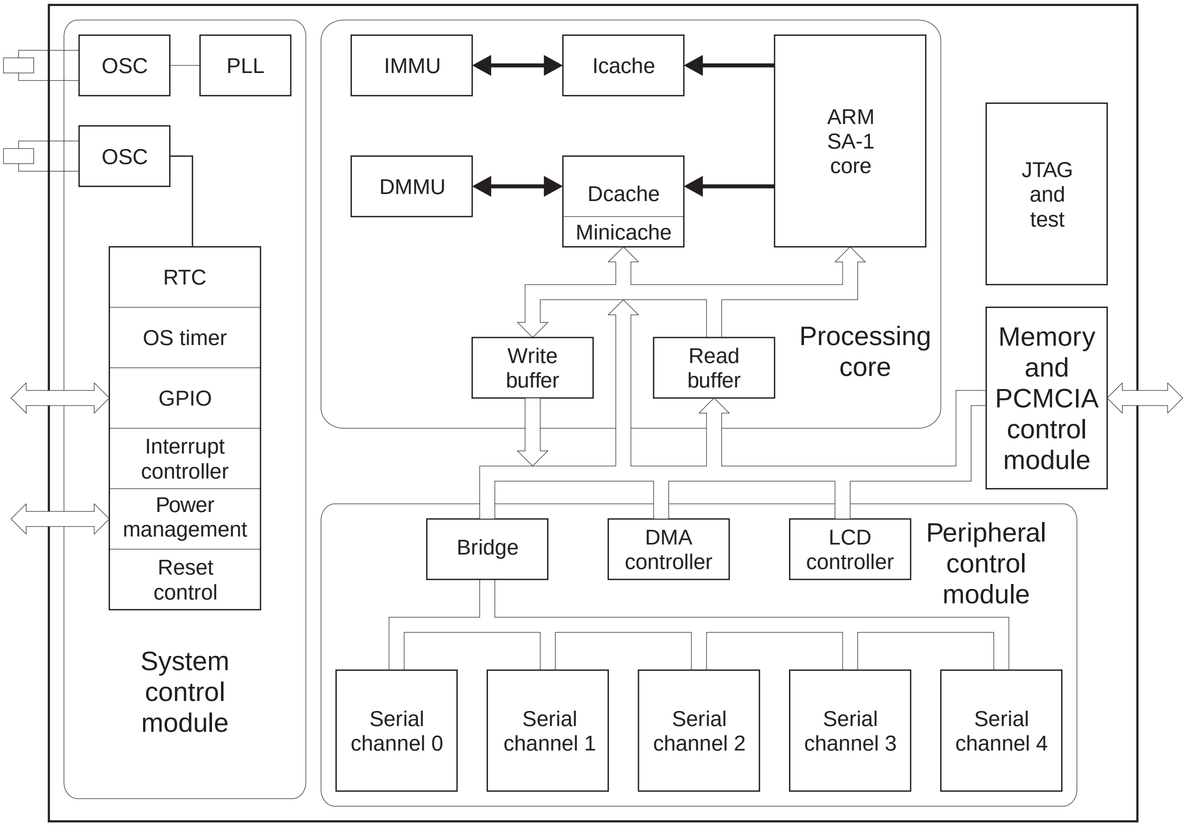


Figure 3-1 SA-1100 block diagram

3.1.1 Serial ports

The SA-1100 provides six serial ports. Five of these use dedicated serial interface ports and the sixth is implemented using GPIO pins. The serial ports are used to communicate with on-board devices and to provide external interfaces. The SA-1100 provides:

- two RS232 serial ports
- UDC device port (not used on P1100)
- IrDA port
- MCP port
- *Serial Peripheral Interface* (SPI) port.

For descriptions of these serial ports and their implementation in the P1100 design, see *Serial interfaces* on page 3-26.

3.1.2 GPIO pin assignment

The SA-1100 provides 28 GPIO pins. Each pin can be configured as an input or output, and to trigger an interrupt on a rising or falling edge. In the P1100 design these are used to provide:

- an expanded LCD data path
- SPI port with multiple chip selects
- serial port handshake control.

The SA-1100 GPIO pins are assigned as shown in Table 3-1. The table gives the name of the signal trace connected to each GPIO pin.

Table 3-1 SA-1100 GPIO pin assignment

Pin	Signal Name	Direction	Description
GPIO0	**CPU_GPIO0**	Input	PLD - Key_ATTN interrupt (wake-up interrupts)
GPIO1	**CPU_GPIO1**	Input	PLD - merged interrupts (all other external sources)
GPIO2	**LCD_DD_8**	Output	Bit8 of 16-bit LCD data path
GPIO3	**LCD_DD_9**	Output	Bit9 of 16-bit LCD data path
GPIO4	**LCD_DD_10**	Output	Bit10 of 16-bit LCD data path
GPIO5	**LCD_DD_11**	Output	Bit11 of 16-bit LCD data path
GPIO6	**LCD_DD_12**	Output	Bit12 of 16-bit LCD data path
GPIO7	**LCD_DD_13**	Output	Bit13 of 16-bit LCD data path
GPIO8	**LCD_DD_14**	Output	Bit14 of 16-bit LCD data path
GPIO9	**LCD_DD_15**	Output	Bit15 of 16-bit LCD data path
GPIO10	**CPU_SPI_TXD**	Output	SPI bus – transmit data
GPIO11	**CPU_SPI_RXD**	Input	SPI bus – receive data
GPIO12	**CPU_SPI_CLK**	Output	SPI bus – clock
GPIO13	**CPU_SPI_FRM**	Output	SPI bus – frame
GPIO14	**CPU_SPI_CS0**	Output	SPI encoded chip select 0
GPIO15	**CPU_SPI_CS1**	Output	SPI encoded chip select 1
GPIO16	**CPU_SPI_CS2**	Output	SPI encoded chip select 2

Table 3-1 SA-1100 GPIO pin assignment (continued)

Pin	Signal Name	Direction	Description
GPIO17	**CPU_UART1_RTS**	Output	UART1 - RTS
GPIO18	**UART1_CTS**	Input	UART1 - CTS
GPIO19	**CPU_UART2_RTS**	Output	UART2 - RTS
GPIO20	**UART2_CTS**	Input	UART2 - CTS
GPIO21	**CPU_MBGNT***	Output	Expansion: bus grant (SA-1100 memory request mode)
GPIO22	**CPU_MBREQ***	Input	Expansion: bus request (SA-1100 memory request mode)
GPIO23	**CPU_GPIO23**	-	Expansion: uncommitted
GPIO24	**CPU_GPIO24**	-	Expansion: uncommitted
GPIO25	**CPU_GPIO25**	-	LED current sink (can be configured as 1Hz from RTC)
GPIO26	**CPU_RCLK_OUT***		Buffered core frequency/2 clock
GPIO27	**CPU_32KHz***		Buffered untrimmed 32kHz clock

Note

Signals marked with * in Table 3-1 require the *Test Unit Control Register* (TUCR) set up in the SA-1100.

3.2 System clocks

The P1100 has three crystal oscillators. These supply clock references for the SA-1100 and the keyboard controller.

3.2.1 Processor clocks

Two crystal oscillators provide reference clock inputs for the SA-1100 as shown in Figure 3-2.

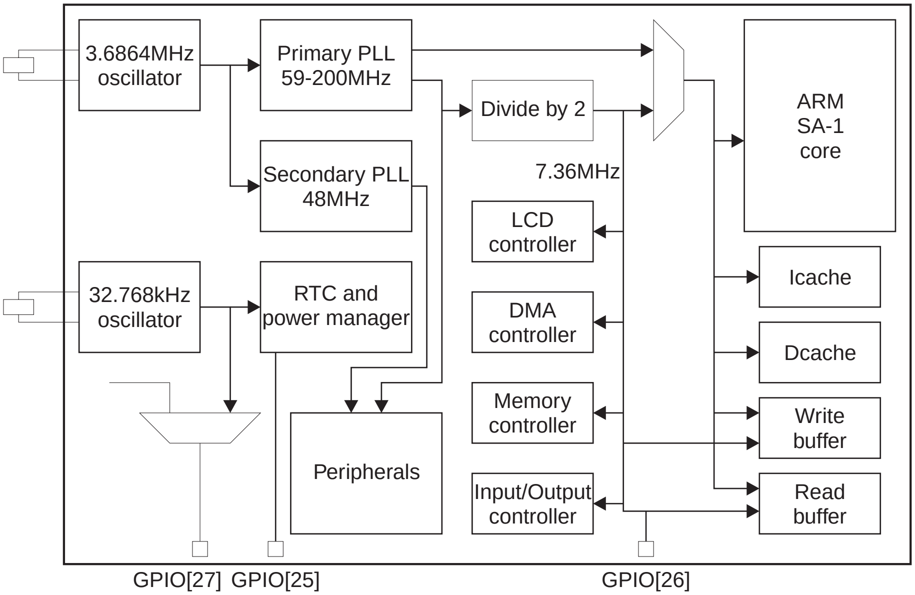


Figure 3-2 SA-1100 clocks

The reference input from the 3.6864 MHz crystal is used to drive two on-chip *Phase-Locked Loops* (PLLs), the primary and secondary PLLs.

The primary PLL supplies the core clock and a 7.36 MHz clock for some of the on-chip serial ports. A divide-by-2 and buffered version of the core clock is used to drive on-chip controllers and also appears on GPIO26 as **CPU_RCLK_OUT**.

The secondary PLL supplies a 48MHz clock for the GPIO-provided serial ports and pin controllers. The reference clock from the 32.768 kHz crystal is used to drive an on-chip *Real-Time Clock* (RTC) and power management controller. A buffered version of this clock also appears on GPIO27. An RTC derived 1Hz signal appears on GPIO25.

The clocks available from GPIO pins are under software control (see *GPIO pin assignment* on page 3-4).

3.2.2 Keyboard clock

In addition to the SA-1100 derived clocks, a 4 MHz ceramic resonator is used to drive the on-chip oscillator within the keyboard controller (see *Keyboard controller* on page 3-32).

3.3 Reset control

The CPU board provides five reset types:

- *Power-on reset* on page 3-9
- *Hardware reset* on page 3-10
- *Software reset* on page 3-10
- *Watchdog reset* on page 3-10
- *Sleep reset* on page 3-10.

The architecture of the reset control subsystem is illustrated in Figure 3-3.

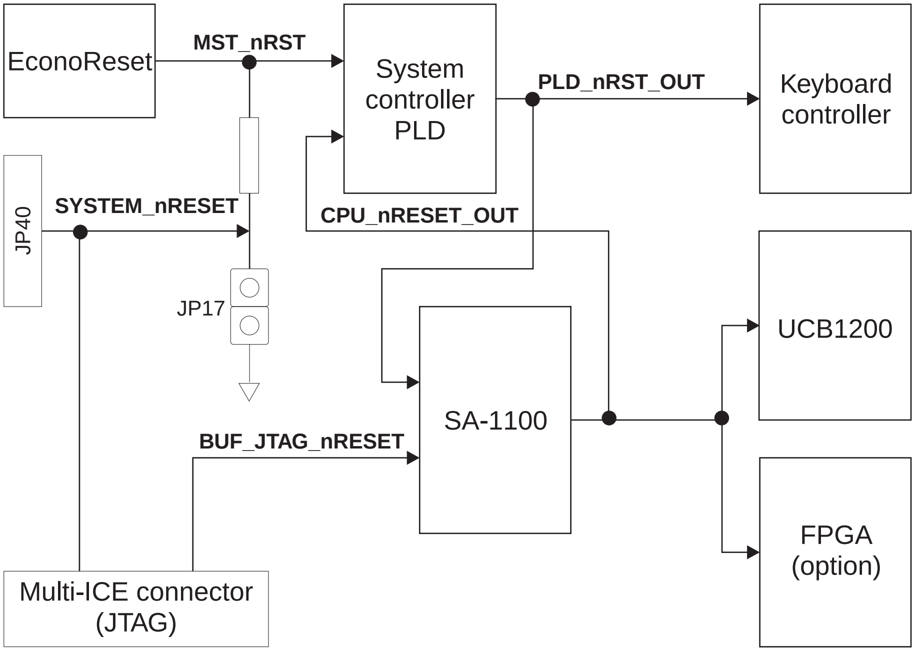


Figure 3-3 Reset system architecture

3.3.1 Reset signal summary

The reset signals are described in Table 3-2.

Table 3-2 Reset signal summary

Signal	Function
MST_nRST	**MST_nRST** is asserted at powerup (Vcc < 2.88V) or when the signal **SYSTEM_nRESET** is asserted. In response, the PLD asserts **PLD_nRST_OUT**. **MST_nRST** is ignored by the PLD when the SA-1100 is in sleep mode, unless the PLD is in development mode when (see *Boot and mode control* on page 3-14) the reset is always passed to the CPU.
SYSTEM_nRESET	This signal is a version **MST_nRST**. It is asserted by the button in the keyboard housing, when the pins on J17 are momentarily connected, or by a debug system connected to JP18. This signal can be used by the debug system to detect when the reset has been pressed.
CPU_nRESET_OUT	The SA-1100 asserts **CPU_nRESET_OUT** during reset and sleep mode. It is used to reset the UCB1200, the internal state machines and signals of the PLD, and the flash memory. It is also routed to a logic analyzer connector, the expansion connector and to the FPGA.
PLD_nRESET_OUT	The signal **PLD_nRST_OUT** is used to reset the keyboard controller.

3.3.2 Power-on reset

A Dallas Semiconductor DS1233AZ 3.3V EconoReset ensures the orderly power-up the P1100. This uses a precision reference with a comparator circuit which tests if the power supply is within tolerance. The typical threshold for the DS1233AZ is 2.88V.

When the power supply is below this threshold the EconoReset asserts **MST_nRST** to the PLD. When the voltage rises above 2.88V, the EconoReset holds **MST_nRST** asserted for an additional period of 350ms to provide a reliable debounce.

Sleep mode is controlled by the system controller module in the CPU. A pull-down resistor on the **PLD_nRST_OUT** signal ensures that the signal is pulled LOW until the PLD is initialized. The CPU asserts **CPU_nRESET_OUT** while **PLD_nRST_OUT** is asserted, and keeps it asserted until the PLD releases **PLD_nRST_OUT**.

3.3.3 Hardware reset

A hardware reset can be triggered from one of three sources:

- a push button switch attached to a 2-pole connector JP17
- push button switch in the keyboard housing panel
- a debug system, such as Multi-ICE, connected to the JTAG test connector

The EconoReset monitors its output for the assertion of any of these momentary-action switches. It provides switch debouncing by holding this output for a 350ms delay after the switch is released.

3.3.4 Software reset

A software reset is initiated by setting the software reset (SWR) bit in the SA-1100 *Reset controller Software Reset Register* (RSRR). Reset is applied to most of the internal circuitry of the SA-1100, and **CPU_nRESET_OUT** asserted. DRAM contents are retained during this reset because DRAM refresh and configuration settings are not cleared. The SA-1100 remains in reset for 256 processor clock cycles before it reboots.

3.3.5 Watchdog reset

The watchdog reset is triggered when the watchdog timer is enabled and the timer expires. In all other respects it is identical to the software reset described above, except that it sets the WDR bit in the RCSR.

3.3.6 Sleep reset

Sleep reset is invoked when the SA-1100 enters sleep mode, either by software setting the force sleep bit in the power manager control register or by a power supply fault. **CPU_RESET_OUT** is asserted, and the DRAMs placed in self refresh mode.

3.4 System controller PLD

A Xilinx XCR3128 CoolRunner PLD provides general system housekeeping and control functions. It features very low active power and near-zero power consumption when there are no signal transitions. The functions provided by the system controller in the P1100 are as follows:

- reset management
- *Interrupt control* on page 3-12
- *Boot and mode control* on page 3-14
- *LCD power control and battery sensing* on page 3-14
- *Interface enable* on page 3-17
- *Flash chip-select control* on page 3-18
- *Buffer control* on page 3-19
- *LED control and GPIO pins* on page 3-19.

Figure 3-4 shows the functional block diagram of the system controller PLD.

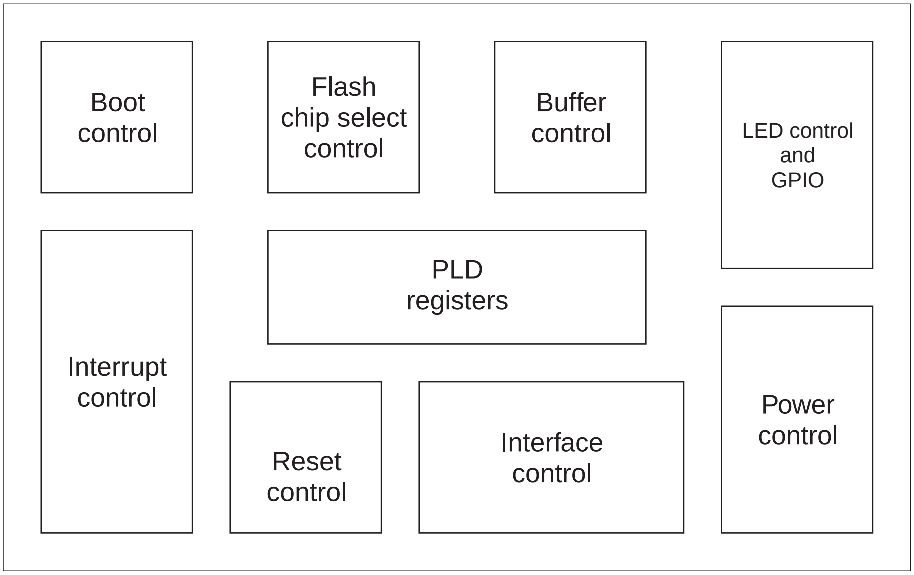


Figure 3-4 System controller PLD functional block diagram

A JTAG interface is provided for *in system programming* (ISP). Please see the section on JTAG for programming details.

3.4.1 Reset management

The reset management function ensures the orderly startup of the P1100 system. Resets are described in *Reset control* on page 3-8.

3.4.2 Interrupt control

There are a large number of interrupt sources in the P1100 system. Those from devices internal to the SA-1100 are controlled by the interrupt controller within the SA-1100. Interrupts from devices external to the SA-1100 are collated and controlled by the system controller PLD, as shown in Figure 3-5.

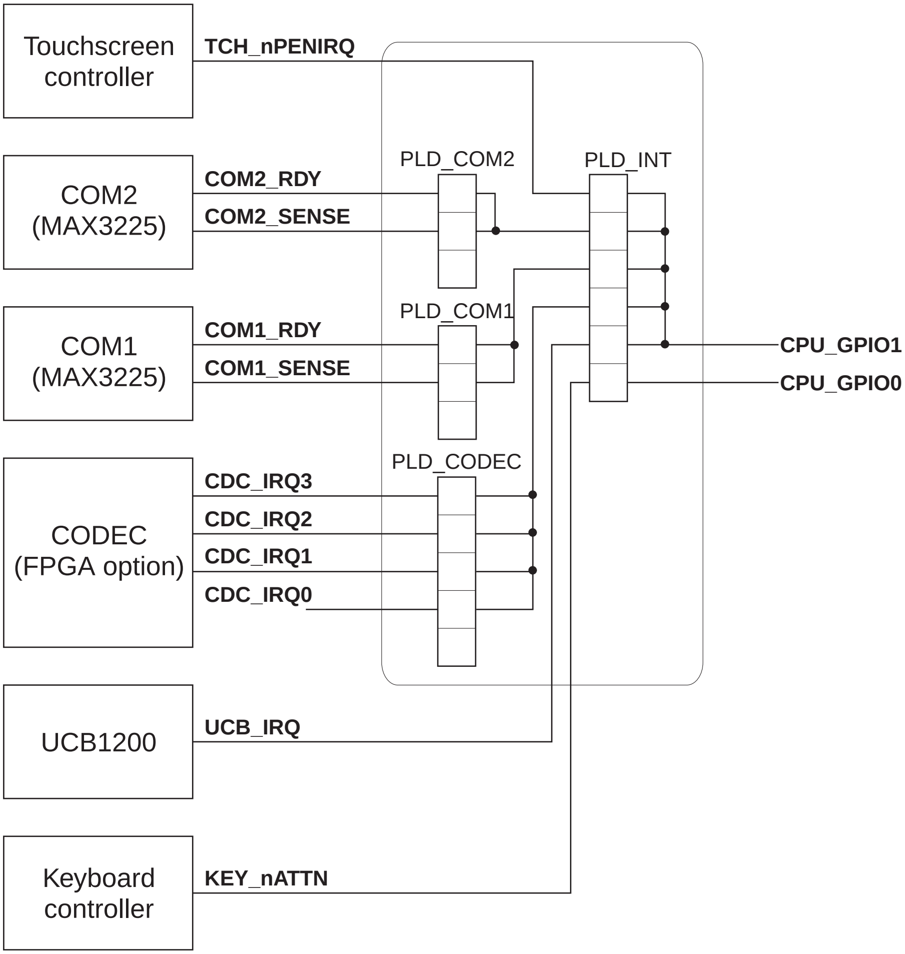


Figure 3-5 Interrupt controller

The PLD forwards interrupts to the processor using the signals **CPU_GPIO[1:0]**:

CPU_GPIO[0] is assigned to **KEY_nATTN** which can be used to bring the system out of sleep mode.

CPU_GPIO[1] is used to for all other interrupts

To determine the cause of an interrupt signalled on **CPU_GPIO[1]**, read the PLD interrupt register (see *Interrupt flag register* on page 4-13). In the case of the COMx interrupts you should also read the COMx_control registers (see *COM2 enable register* on page 4-17 and *COM1 enable register* on page 4-17).

The interrupts controlled by the PLD are listed in Table 3-3.

Note

The **COM[2:1]_RDY** and **COM[2:1]_SENSE** signals are not implemented as interrupts in the current release of the PLD code. However, the status of these signals can be read by polling the associated PLD registers (see *System controller PLD* on page 3-11).

Table 3-3 Interrupt summary

Interrupt signal	Source	SA-1100 pin	Function
CDC_IRQ[3:0]	FPGA (if fitted)	GPIO1	**CDC_IRQ[3:1]** are interrupt inputs that are allocated to the FPGA option. **CDC_IRQ0** is reserved for internal use by the PLD.
TCH_nPENIRQ	ADS7843	GPIO1	This interrupt is generated by the secondary touch screen controller when a touch screen event is detected.
COM[2:1]_RDY	COM[2:1] port	GPIO1	These are low priority interrupts generated when the serial port has valid signal levels and is ready for communication. This is not to be confused with the serial port handshake signals.

Table 3-3 Interrupt summary (continued)

Interrupt signal	Source	SA-1100 pin	Function
COM[2:1]_SENSE	COM[2:1] port	GPIO1	These are low priority interrupts generated when a change of state is detected on the COM ports, either when a cable is plugged in or unplugged.
UCB_IRQ	UCB1200 codec	GPIO1	The interrupt generated by a data transfer request, a touch screen event or an event on the GPIO pins. Each of these sources is programmable in the UCB1200.
KEY_nATTN	Keyboard controller	GPIO0	This interrupt generated by a keyboard event, a switch event, or as a handshake signal for the keyboard protocol.

3.4.3 Boot and mode control

Two switches housed within the 5-element DIL switch pack are connected to the system controller PLD (see *Software readable switches* on page 3-21). The system controller PLD senses S4 and S5 to support the following functions:

SW5 is the BOOTSEL switch. It is used by the boot monitor component of the *ARM Firmware Suite* (AFS) to select between the default or alternative boot image. If you are not using AFS, this switch can be assigned another meaning.

SW4 is the DEVMODE switch. It is used to select *development mode*. Development mode allows the system to be reset while the SA-1100 is in sleep mode. Development mode should only be used when DRAM contents can be considered as undefined when returning from sleep mode.

3.4.4 LCD power control and battery sensing

This function of the system controller provides:

- power supply mode control
- LCD power and bias control
- battery sensing.

Figure 3-6 on page 3-15 shows the architecture of the power output control function of the PLD.

Power supply switching mode control

The PLD provides a mode select signal **PLD_PMODE** to the MAX1703 DC-DC converter. This is controlled by the PMODE bit in the PLD_PWR register (see *Power control register* on page 4-14) and allows *Pulse-Width-Modulation* (PWM) or *Pulse-Frequency-Modulation* (PFM) to be selected. The use of modes is discussed in *Power supply* on page 3-45.

LCD power output and bias control

A number of signals from the system controller PLD, controlled by the PLD_PWR register, are used to enable and disable the display power supply outputs. The signals controlled by the PLD_LCDBC register are used to vary the monochrome bias output **LCD_OUT4**.

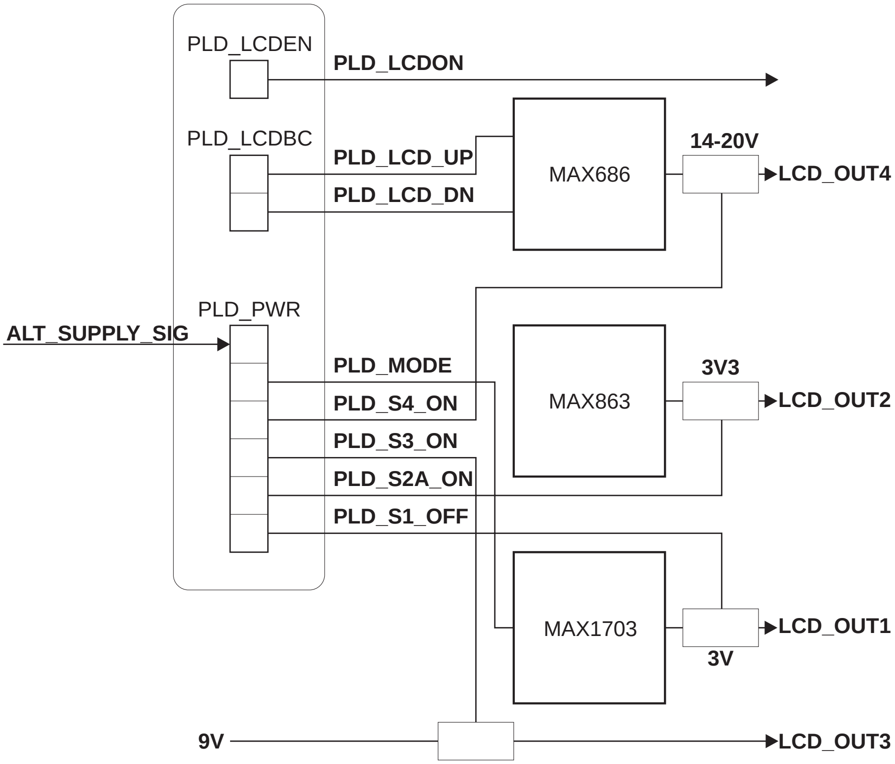


Figure 3-6 Power output control

The signals used to control the LCD power supply outputs are summarized in Table 3-4.

Table 3-4 LCD power configurations

Signal	Function
PLD_LCDON	Enables the monochrome LCD
PLD_LCD_S1_OFF	Enables 3V supply to the LCD
PLD_LCD_S2A_ON	Enables the 3.3V supply to LCD logic
PLD_LCD_S3_ON	Enables 9V
PLD_LCD_S4_ON	Enables the switched bias voltage for the monochrome LCD.
PLD_LCD_UP	LCD bias control – step up
PLD_LCD_DN	LCD bias control – step down

The signals are controlled by the power control register (see *Power control register* on page 4-14).

Battery sensing

The battery sense input (**ALT_SUPPLY_SIG**) controls the PWR_EXT bit in the PLD_PWR register. Software can read this bit to determine whether the system is running from battery power or an external power supply. The state of the sense line can be read from this bit in the power control register (see *Power control register* on page 4-14).

3.4.5 Interface enable

The system controller PLD provides enable controls for the serial channels, keyboard controller, and SPI chip-select decoder. This function of the PLD is illustrated in Figure 3-7.

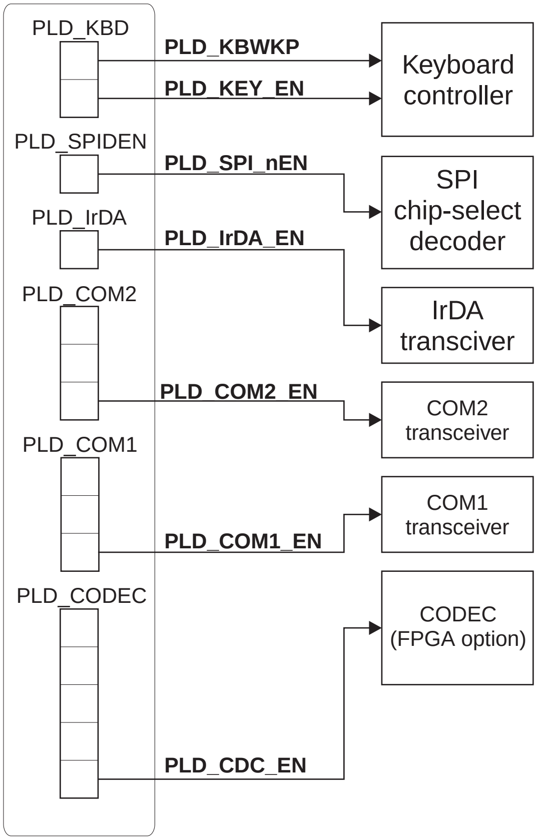


Figure 3-7 Interface control

These are controlled by PLD registers (see *System controller PLD registers* on page 4-13). The signals controlled by the PLD are summarized in Table 3-5 on page 3-18. Writing a 1 to the relevant bit enables the signal.

Table 3-5 Interface enable

Signal	Function
CDC_EN	FPGA codec enable
PLD_COM2_EN	Serial channel 2 transceiver enable
PLD_COM1_EN	Serial channel 1 transceiver enable
PLD_IrDA_EN	IrDA port enable
PLD_SPI_nEN	Enables the SPI bus decoder when LOW (see *SPI bus* on page 3-28).
PLD_KEY_EN	Keyboard enable

3.4.6 Flash chip-select control

The system controller PLD provides address decoding for the flash memory, expansion space, and its own internal registers. It decodes (**CPU_nCS[2:0]**) and **CPU_A[25:24]** from the SA-1100. All eight flash devices are accessed within a single 64MByte region. Each chip select defines a 16MByte region. Table 3-6 shows the decoding for the flash chip-select signals.

Table 3-6 Flash chip-select decoding

CPU_nCS[2:0]	CPU_A[25:24]	Chip select	Function	Base address
110	00	**FLASH_nCS0**	Flash region 0 - 16MByte	`0x00000000`
110	01	**FLASH_nCS1**	Flash region 16 - 32MByte	`0x01000000`
110	10	**FLASH_nCS2**	Flash region 32 - 48MByte	`0x02000000`
110	11	**FLASH_nCS3**	Flash region 48 - 64MByte	`0x03000000`
011	00	-	PLD Registers	`0x10000000`
011	10	**FLASH_nCS4**	Expansion connector	`0x12000000`
011	11	**FLASH_nCS5**	Expansion connector	`0x13000000`
101	xx	**FLASH_nCS6**	FPGA	`0x08000000`

Note

CPU_nCS1 and **CPU_nCS2** are routed to the expansion connector. **CPU_nCS3** is routed to the optional FPGA

3.4.7 Buffer control

The PLD provides enable and direction controls for the main address and data buffers. The standard PLD configuration supports one bus master (the SA-1100) in the system. However, the P1100 design is able to accommodate external bus masters, such as an optional FPGA, but the PLD must be reprogrammed to support this.

The significant signals are:

PLD_DBUF_DIR
a passed through version of **CPU_MEM_nWE**

PLD_MEMBUF_nOE
a passed through version of **CPU_MEM_nOE**.

3.4.8 LED control and GPIO pins

The PLD provides control for four LEDs on the CPU board (see *Software controlled LEDs* on page 3-21).

It also provides three GPIO pins (**PLD_GPIO[2:0]**). These are connected to the auxiliary connector for the optional FPGA and can be controlled by accessing the PLD input/output register (see *Software input/output control register* on page 4-16).

3.5 LED control and switch sensing

The switches and LEDs provided by the P1100 CPU card include:

- software controlled LEDs
- five software readable switches
- software readable hexadecimal switch.

Two additional LEDs provide status indication for the SA-1100 and FPGA, if fitted.

Figure 3-8 shows the LED control and switch reader.

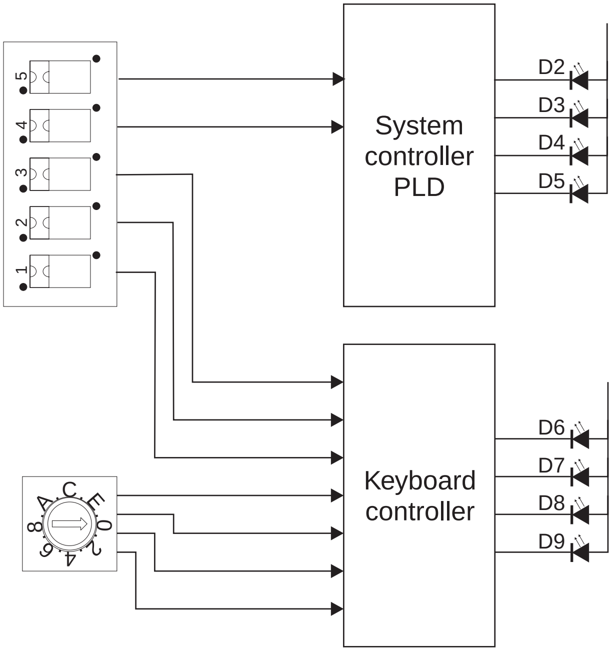


Figure 3-8 LEDs and switches

3.5.1 Software controlled LEDs

The software controllable LEDs provided on the CPU card are controlled by the system controller PLD or the keyboard controller. Table 3-7 lists these LEDs and indicates by which device they are controlled.

Table 3-7 LEDs connection and control

LED	Name	Color	Driving device	Driving signal
D1	CPU_LED*	Green	SA-1100	**CPU_GPIO25**
D2	PLD_LED0	Red	System controller PLD	not named
D3	PLD_LED1	Red	System controller PLD	not named
D4	PLD_LED2	Yellow	System controller PLD	not named
D5	PLD_LED3	Green	System controller PLD	not named
D6	KBD_LED0*	Red	Keyboard controller	**KBD_LED0**
D6	KBD_LED1*	Yellow	Keyboard controller	**KBD_LED1**
D8	KBD_LED2*	Green	Keyboard controller	**KBD_LED2**
D9	KBD_LED3*	Green	Keyboard controller	**KBD_LED3**
D10	FPGA_LED	Green	FPGA – signals when the FPGA has completed its programming on power up.	**-**

Note

* These LEDs are replicated on the distribution panel and visible through apertures in the keyboard housing (applies to system units only).

3.5.2 Software readable switches

The switches included on the P1100 CPU board are:

- a dual-in-line switch pack containing five individual switches
- hexadecimal rotary switch providing a 4-bit value

Three switches within the DIL pack and the rotary switch are connected to the GPIO pins on the keyboard controller. These switches can be assigned any meaning by the software developer.

The remaining two switches within the DIL pack are connected to the system controller PLD and control the boot and operating modes of the P1100 system (see *Boot and mode control* on page 3-14).

The switches are connected and assigned as shown in Table 3-8.

Table 3-8 Switch connections and assignment

Switch	Device	Pin name/signal	Function
SW1	Keyboard controller	GIO10/SW11	User definable
SW2		GIO10/SW12	
SW3		GIO10/SW13	
SW4	System controller (see *Boot and mode control* on page 3-14)	**PLD_DEVMODE**	Development mode control
SW5		**PLD_BOOTSEL**	AFS boot control
Rotary hex	Keyboard controller	GIO[17:14]	Supplies a 1’s compliment 4-bit value. Function of this switch is user definable.

3.6 Memory subsystem

The P1100 uses low power EDO DRAM for its main memory and screen buffer, and flash memory for storing boot images and other non-volatile data. The amount of memory fitted is a build-time option.

3.6.1 Flash memory

Non-volatile storage is provided by up to eight Intel 5V StrataFlash devices. The CPU card is fitted with 64Mbit devices organized as 4M x 16. These are fitted in pairs, with each pair providing a 16MB area of memory with a 32-bit data path. The P1100 provides support for up to four pairs (eight devices) providing 64MB of flash memory, although only the 32MB product variant is currently available. The flash memory is decoded within the region decoded by chip-select 0 from the SA-1100. This provides a single contiguous 64MB region for the flash memory (see *Flash chip-select control* on page 3-18).

The parts used are Intel G28F640J5-150 parts in a 56 ball microBGA package.

Note

The 16-bit ROM size SA-1100 option is not supported by the P1100.

Flash hardware interface

The hardware interface to the SA-1100 is as follows:

- buffered address and data buses
- a +5V level translated version of the system reset is connected to the flash **nRP** pins.
- data and address buffers controlled by the XCR3128 PLD (see *Buffer control* on page 3-19)
- +5V level translated versions of the chip selects decoded by the XCR3128 PLD (see *Flash chip-select control* on page 3-18).
- buffered versions of the **MEM_nOE** and **MEM_nWE** from the SA-1100

A 74HCT244 is used to translate the control signals to +5V signal levels. This decreases the power consumption by the flash devices.

Figure 3-9 on page 3-24 shows the architecture of the Flash memory. To aid clarity the diagram does not show the data and address buffers, or the output enables and write enable signals.

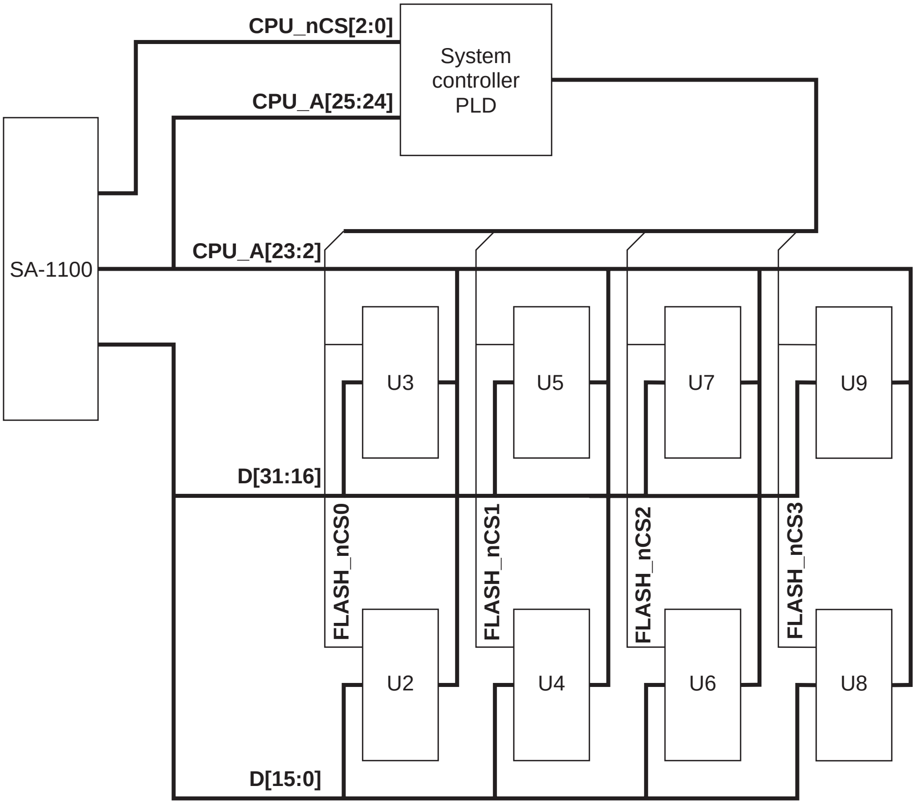


Figure 3-9 Flash memory block diagram

Programming and erasing flash

The flash devices are used in 16-bit mode. Flash address line A0 is ignored and is tied HIGH to avoid it floating and wasting unnecessary power.

Each flash chip contains sixty-four 128KB blocks. These can be programmed and erased independently using flash tools and utilities provided by the ARM Firmware Suite (see the *ARM Firmware Suite Reference Guide*).

Flash and power conservation

The flash devices enter standby mode automatically when their chip selects are not asserted. This state conserves power without time penalty because the devices power-up within a normal flash access cycle. Power-down mode is only used during reset because it provides no appreciable power advantage over using standby mode.

3.6.2 EDO DRAM

The main memory is provided by one or two banks of EDO DRAM. Two 64Mbit, 16-bit devices are used for each bank, providing 16MB of volatile memory per bank. The first bank is selected by **CPU_nRAS0** and the second by **CPU_nRAS1**. The devices are connected directly to the unbuffered SA-1100 address and data buses.

Hyundai HY51V65164A devices are used. The P1100 uses the SL low power 50ns part (60ns could also be used) in a 50-pin plastic TSOP type II package (400milx825mil). These operate from a 3V supply and provide a very low-power self-refresh mode with a refresh period of 128ms. The maximum current consumption parameters for the DRAM are summarized in Table 3-9.

Table 3-9 Maximum DRAM current

	50ns	60ns
EDO Mode current	105mA	85mA
Standby current	200µA	200µA
Self Refresh current	400µA	400µA
Refresh current	140mA	115mA

The DRAM interface connects directly to the SA-1100. The signals used to access and control the DRAM are summarized in Table 3-10.

Table 3-10 DRAM interface signal descriptions

Signal	Function
CPU_A[22:10]	These provide a multiplexed address bus, configured to use 12/10 address multiplexing scheme. **CPU_A22** is not used with the specified part (4k refresh).
CPU_D[31:0]	This is the 32-bit data bus to and from the SA-1100
CPU_RAS[1:0]	These are used to select the DRAM banks and to sample the row address.
CPU_CAS[3:0]	These are used to sample the column address data. One CAS line is provided for each byte lane.
CPU_MEM_nOE	This signal enables the data output buffers on read cycles when asserted (LOW).
CPU_MEM_nWE	This signal indicates a write cycle when asserted (LOW).

3.7 Serial interfaces

The SA-1100 provides five serial ports. These are assigned as shown in Table 3-11

Table 3-11 Serial port assignment

SA-1100 serial channel	Usage
0	Not used
1	COM1
2	IrDA
3	COM2
4	MCP/SSP

3.7.1 COM1 and COM2

COM1 and COM2 are asynchronous serial ports which use standard RS232 switching levels. Handshake signals are provided by using software controlled GPIO pins of the SA-1100. Port status and enable controls are provided by the system controller PLD. Baud rates in the range from 56.24bps to 230.4kbps are supported.

The host signals associated with the COMs channels are summarize in Table 3-12 on page 3-26.

Table 3-12 COMs channel host signals

Signal	Source	Function
CPU_UART1_RTS	SA-1100 GPIO_17	GPIO_17 is configured as an output for the **CPU_UART1_RTS** signal. Assert this signal when software is ready to transmit.
UART1_CTS	COM1 transceiver	GPIO_18 is configured as an input for the **UART1_CTS** line. This line is asserted by the connected device and should be configured to interrupt the SA-1100.
CPU_UART2_RTS	SA-1100 GPIO_19	GPIO_19 is configured as an output for the **CPU_UART1_RTS** signal. Assert this signal when software is ready to transmit.
UART2_CTS	COM2 transceiver	GPIO_20 is configured as an input for the **UART1_CTS** line. This line is asserted by the connected device and should be set to interrupt the SA-1100

Table 3-12 COMs channel host signals (continued)

Signal	Source	Function
COM1_RDY	COM1 transceiver	This signal is monitored by the PLD and controls the state of the COM1_READY bit in the Peripheral Control Register. Read this bit to determine if the serial port is ready to transmit.
COM1_SENSE	COM1 transceiver	This signal is monitored by the PLD and controls the state of the COM1_SENSE bit in the Peripheral Control Register. Read this bit to determine if a cable is plugged into the socket.
COM2_RDY	COM1 transceiver	This signal is monitored by the PLD and controls the state of the COM2_READY bit in the Peripheral Control Register. Read this bit to determine if the serial port is ready to transmit.
COM2_SENSE	COM1 transceiver	This signal is monitored by the PLD and controls the state of the COM2_SENSE bit in the Peripheral Control Register. Read this bit to determine if a cable is plugged into the socket.
PLD_COM1_EN	System controller PLD	This signal is used to enable/disable the COM1 transceiver. It is controlled by the COM1_EN bit in the peripheral control register (see *COM1 enable register* on page 4-17).
PLD_COM2_EN	System controller PLD	This signal is used to enable/disable the COM2 transceiver. It is controlled by the COM2_EN bit in the peripheral control register (see *COM2 enable register* on page 4-17).

Maxim MAX3225 transceivers provide signal level conversion for the RS232 interfaces. These feature:

- a guaranteed maximum data rate of 1Mbps (which exceeds the 230.4kbps upper limit of the UART)
- an automatic power-down mode. This is entered when a cable is not attached, or when transmitters are inactive for more than 30 seconds. The power-down mode can be overridden by asserting the **PLD_COM1_EN** signal.

The MAX3225 is powered from the non-switched +3V rail.

3.7.2 IrDA

The IrDA port conforms to SIR/MIR and FIR standards. It uses UART 2 of the SA-1100 and an IrDA transceiver. The XCR3128 PLD provides an enable signal to the transceiver. This is controlled by the IrDA_EN bit in a the IrDA enable register (see *IrDA enable register* on page 4-16).

3.7.3 MCP

The MCP channel is used as a dedicated communications link between the SA-1100 and Philips UCB1200 device. This is configured as a synchronous *Serial Interface Bus* (SIB) which operates at either 9.585 Mbs or 11.981 Mbs using frame-based data transfers. More details about this device are provided in *Philips UCB1200 codec and touch screen controller* on page 3-39.

The signals used to provide the interface between the SA-1100 and UCB1200 are summarized in Table 3-13.

Table 3-13 MCP signal summary

Signal	Source	Function
CPU_UCB_TXD	SA-1100 **TXD_C**	SIB transmit data (from the CPU)
CPU_UCB_RXD	UCB1200 sibdout pin	SIB receive data (to the CPU)
CPU_UCB_CLK	SA-1100 **CLK_C**	SIB clock
CPU_UCB_FRM	SA-1100 **SFRM_C**	SIB synchronization. Used to synchronize the start of each data frame

3.7.4 SPI bus

The SSP function of serial channel 4 is used to provide an SPI channel (see *SPI bus* on page 3-29). This uses the alternate functions of the **GPIO[13:10]** pins of the SA-1100 and is allows both types of serial interface to be supported at the same time. See the *Intel StrongARM SA-1100 Microprocessor Developer’s Manual* for further information.

3.8 SPI bus

The SPI bus is used to provide a serial interface between the SA-1100 and the:

- keyboard controller
- MMC sockets
- touch screen controller
- up to three expansion devices.

This section describes the SPI bus itself. For descriptions of the devices attached to the SPI bus, refer to the following sections:

- *Keyboard controller* on page 3-32
- *MMC interface* on page 3-37
- *Touch screen controller* on page 3-38
- *Philips UCB1200 codec and touch screen controller* on page 3-39.

3.8.1 SPI bus architecture

The architecture of the SPI bus on P1100 CPU board is shown in Figure 3-10.

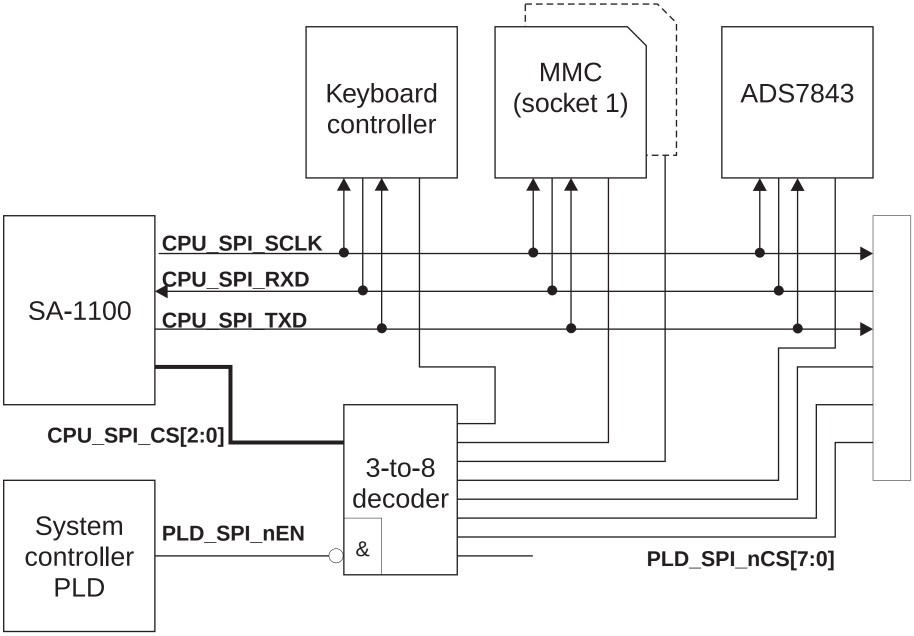


Figure 3-10 SPI bus architecture

The SPI transmit data, receive data and clock signals are buffered by a 74LCX16224. This is a +3V device with +5V tolerant inputs. The buffer is permanently enabled, and introduces a maximum propagation delay of 7ns into each signal. There is minimal skew because all signals pass through the same buffer.

The only SPI master in the P1100 system is the SA-1100. The SPI bus clock rate can be varied to suit the selected peripheral.

3.8.2 SPI bus signal summary

The SPI bus is implemented using the alternate output pins for serial port 4, and additional chip selects are provided by using the GPIO pins shown in Table 3-14.

Table 3-14 SPI signal summary

Signal	Source	Function
CPU_SPI_TXD	SA-1100 **GPIO10**	SPI transmit data (from the CPU)
CPU_SPI_RXD	SA-1100 **GPIO11**	SPI receive data (to the CPU)
CPU_SPI_CLK	SA-1100 **GPIO12**	SPI clock, should be varied according to the peripheral addressed.
CPU_SPI_FRM	SA-1100 **GPIO13**	SPI frame, used only as a logic analyzer monitor (devices used need extended timing provided by chip select decode lines)
CPU_SPI_CS[2:0]	SA-1100 **GPIO[16:14]**	These signals are used to select one of eight SPI devices, as shown in Table 3-15. These signals are ignored if **PLD_SPI_nEN** is set HIGH (by writing a 0 to the PLD_EN register).
PLD_SPI_nEN	PLD	This signal enables and disables a 74LVC138 3-to-8 decoder. This method ensures that only one SPI device can be selected at any one time.

3.8.3 Using the SPI chip selects

Peripheral selection is performed by decoding three chip address signals **CPU_SPI_CS[2:0]** from the CPU with a 3-to-8 decoder. This generates eight separate chip select signals, allowing only one chip select to be asserted at a time.

The decoder is enabled and disabled by a signal from the system controller PLD. This signal is controlled by the **SPI_EN** bit in the SPI enable register (see *SPI decoder enable register* on page 4-16.

The SPI chip select lines are assigned as shown in Table 3-15 on page 3-31. One chip select signal, **PLD_SPI_nCS7**, is reserved as a *null* select (do not use this signal for expansion). This is used to deselect devices on the SPI bus because they require their **nCS** line to be deasserted as part of the transfer protocol.

Table 3-15 SPI chip select assignment

CPU_SPI_CS[2:0]	Active chip select	Device
000	**PLD_SPI_nCS0**	Keyboard controller
001	**PLD_SPI_nCS1**	MMC Socket 1
010	**PLD_SPI_nCS2**	MMC Socket 2
011	**PLD_SPI_nCS3**	Touch screen controller
100	**PLD_SPI_nCS4**	System expansion
101	**PLD_SPI_nCS5**	System expansion
110	**PLD_SPI_nCS6**	System expansion
111	**PLD_SPI_nCS7**	Null chip select. This is asserted when no devices are required. Do not use this chip select for system expansion.

3.8.4 SPI bus timing

The different devices attached to the SPI bus have different timing requirements.

- The UR8HC007-001 SPI interface timing requirements are described in *Keyboard controller host interface timing* on page 4-23.
- The MMC interface timing requirements are described in *MMC transaction timing* on page 4-33.
- The touch screen controller timing requirements are described in *Programming the touch screen controller* on page 4-34.

3.9 Keyboard controller

The keyboard controller used is a multifunction device designed for low power hand-held applications. It is used to perform the following functions:

- *Key scanning and debounce* on page 3-33
- *LED control* on page 3-33
- *Switch sensing with the keyboard controller* on page 3-34
- *Power sensing and management* on page 3-36
- *Pulse Width Modulated* (PWM).

The architecture of the keyboard controller implementation is shown in Figure 3-11.

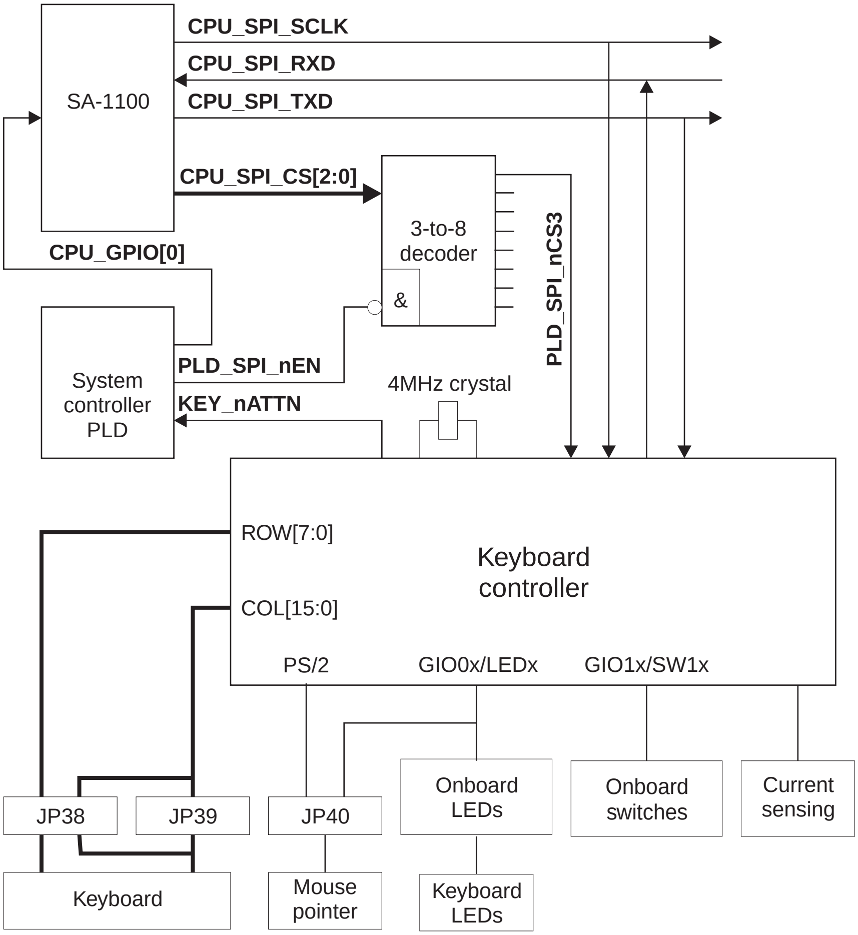


Figure 3-11 Keyboard controller

The keyboard controller has a typical power consumption of less than 1µA at 3V. Packaged in a 80-pin thin quad flat pack (TQFP), it has a maximum SPI clock frequency of 250KHz. The P1100 design supports the maximum clock rate.

During power-up the keyboard controller the SPI clock and transmit data inputs **CPU_SPI_SCLK** and **CPU_SPI_TXD** are pulled HIGH and the SPI portion of the buffer (U23) is permanently enabled. This ensures the SPI clock and transmit data inputs to the keyboard controller are held in their idle HIGH state until the SPI bus has been configured in the SA-1100, and ensures that the keyboard controller powers up in SPI mode.

For SPI bus timing requirements, see *Keyboard controller host interface timing* on page 4-23.

3.9.1 Key scanning and debounce

The keyboard controller scans and debounces a keyboard matrix of up to 8 rows x16 columns. The default matrix for the device is the Fujitsu FKB7654 miniature keyboard with which P1100 system units are equipped. The Fujitsu keyboard provides an IBM PC-AT style key layout within a minimal package size.

The matrix layout registers within the keyboard controller are automatically updated with data corresponding to this keyboard on power-up.

The controller provides rollover processing of any number of key presses and the minimizes *ghost* keys occurring when three or more keys are pressed. When a key is pressed, the keyboard controller asserts **KEY_nATTN** to signal an interrupt request to the PLD interrupt controller.

For information about keyboard controller interrupts, see *Keyboard controller interrupts* on page 4-25.

3.9.2 LED control

The keyboard controller features a 4-bit GPIO port which is used to control four LEDS. Each bit has a high current sink capability (20mA peak, 15mA average). On system units, these ports are also used to control the LEDs on the distribution board which appear through the keyboard housing. Each two LEDs are wired in parallel on when the distribution board is attached. The port assignments of the LEDs are shown in Table 3-16.

Table 3-16 Keyboard controller GPIO/LED assignment

Port Name	LED
GIO00/LED0	KBD_LED0 – red
GIO01/LED1	KBD_LED1 – yellow
GIO02/LED2	KBD_LED2 – green
GIO03/LED3	KBD_LED3 – green

The LEDs can be assigned to any function by the system developer. For example, they could be used to provide keyboard indicators, such as CapsLock and NumLock. The status of these indicators is under software control and is not automatically set by the keyboard controller.

The LEDs can be either on, off or blinking. To minimize power, it is recommended that the LEDs are set to blink on and off. In this mode, the on and off periods can be controlled in units of 1/16 of a second. Faster blink rates give the LED the appearance of being illuminated continuously. The LEDs can be made to flash by adding meta blink intervals within blink cycle. This give the appearance of the LED flashing.

For information about programming the LEDs, see *Programming the keyboard controller* on page 4-20.

Note

You cannot put the keyboard controller into SUSPEND mode if a LED is on or is blinking, so care should be taken in the use of the LEDs.

3.9.3 Switch sensing with the keyboard controller

The P1100 provides a number of switches that can be read from keyboard controller internal registers:

- DIL (three switches from a bank of five)
- hexadecimal rotary switch
- external switch (wired through JP11)
- lid switch (wired through JP9)

The switch inputs are connected to GPIO port 1 as described in Table 3-17 on page 3-35. JP9 provides an input for a lid-switch, if one is added to the system. This can be used by the power management subsystem (see *Power sensing and management* on page 3-36).

Table 3-17 Keyboard controller GPIO port assignments

Bit-port	Function	Description
GIO10/SW10	Expansion – JP11	This is an 8-bit port with individual pins configurable as inputs, outputs or as switch inputs. As inputs, these pins are at +3V levels but are +5V tolerant. As outputs, these pins can drive to both +3V and GND. GIO16 and GIO17 of this port are +5V tolerant but can only drive to GND because they use an *open n-channel drain* structure. GIO11-3 of this port are used for three of the DIL switch elements (+3V or GND). GIO15:7 are connected with pull-up resistors to the hex encoded rotary switch, JP10.
GIO11/SW11	DIL switch (SW1)	
GIO12/SW12	DIL switch (SW2)	
GIO13/SW13	DIL switch (SW3)	
GIO14/SW14	Rotary hex switch bit0	
GIO15/SW15	Rotary hex switch bit1	
GIO16/SW16	Rotary hex switch bit2	
GIO17/SW17	Rotary hex switch bit3	
GIO20/SW0/INT0	Expansion – JP11	This port is a 2-bit port with individual pins configurable as inputs, outputs or switch inputs. When configured as inputs they have +3V switching levels and can be used to generate interrupts on negative, positive or both edges. This port is connected to a DF13 style connector, JP11. A switch connected to this port is not disabled with the main switch matrix. In system units, this input could be used to signal when the lid is closed.
GIO21/SW1/INT1	Expansion – JP11	
GIO30/AD0	3V shunt – load x60 sample	This is a 4-bit port with individual pins configurable as inputs, outputs or 10-bit ADC inputs. GIO30 and GIO31 are ADCs input used for the current measuring circuitry. GIO32 and GIO33 are connected to the DF13 style connector, JP1.
GIO31/AD1	3V shunt – source sample	
GIO32/AD2	Expansion – JP1	
GIO33/AD3	Expansion – JP1	

3.9.4 Power sensing and management

Two of the *Analog-to-Digital Converters* (ADCs) within the keyboard controller are used to sense power usage by measuring the voltage across a 0.015Ω shunt resistor. A sample is taken from the source side of the shunt, and a second from the load side of the shunt. The current through the shunt can be calculated using Ohms Law:

$60 \text{ x } (V_{sample\_source} - V_{sample\_load})$

The keyboard controller has four states relating to power management, as listed in Table 3-18.

Table 3-18 Power management states

Host State	UR8HC007 Pin Name		
	PWROK	**_LID**	**_HSUS**
Critical Suspend (low battery)	0	x	x
Lid closed	1	1	x
SA-1100 suspended (sleep)	1	0	1
Active	1	0	0

CRITICAL SUSPEND

The **PSU_BATT_OK** signal from the power supply places the keyboard into the CRITICAL SUSPEND state on a transition from HIGH to LOW. The state disables all keys and turns off all LEDs. It remains in this state until the **PSU_BATT_OK** signal transitions to a HIGH. Normal state for this signal is HIGH.

LID CLOSED

This state can be entered from either the state of the physical pin (when enabled) or under software control. The controller can be activated as follows:

An external switch (GIO20 or GIO21 transition). If **_LID** is HIGH, only the external switch (XSW1 or 2) activates the keyboard controller. If the lid is open then any of the GIO switches activate the keyboard.

ACTIVE

This state is entered when a key switch is pressed. This method is valid only if the lid is open, the **PSU_BATT_OK** signal is asserted and the **nWKUP** pin is not being driven from the PLD (open drain buffer turned OFF). When the PLD shuts down the keyboard, only the software selected wake up keys can re-activate it.

3.10 MMC interface

The *MultiMedia Card* (MMCs) is a standard data storage device, that use either flash or ROM technology. It has a very small form factor of 32mm x 24mm and provides storage capacities of up to 64MByte at the present time.

The P1100 uses the SPI bus to communicate with the MMCs. The maximum clock rate for SPI accesses to the MMC is 1.8432MHz in the P1100 design.

The P1100 provides one or two sockets for MMCs:

- socket 1 is selected with **PLD_SPI_CS1**
- socket 2, an optional stacked socket, is selected with **PLD_SPI_CS2**.

Figure 3-12 shows the architecture of the MMC interface.

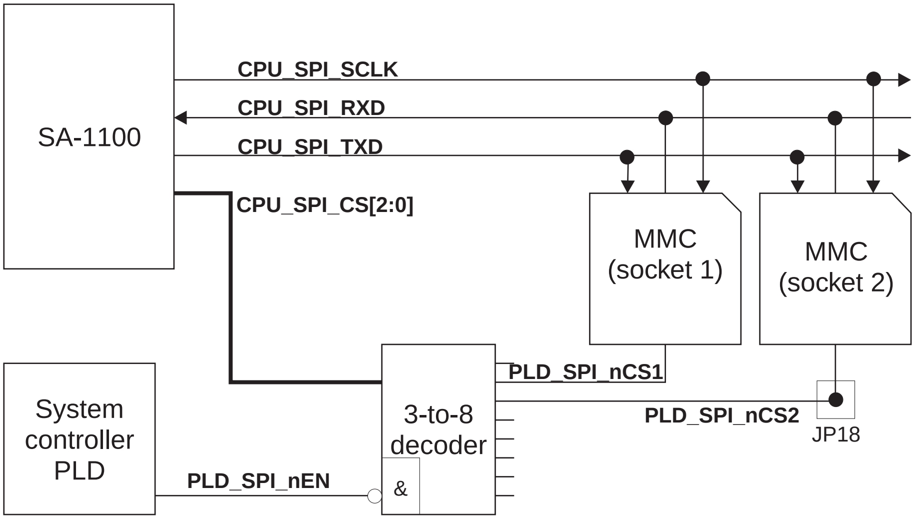


Figure 3-12 MMC interface

Note

Socket 2 is not fitted as standard. It is possible to obtain a stacking socket to fit onto socket 1. All pins except the chip select pin need to be soldered in parallel. **PLD_SPI_CS2** appears at a test point close to MMC socket, as shown in Figure 3-12.

The connector provides *hot swap* capability. The signal pins are recessed which allows the power to connect and stabilize before logic signals are applied to the card. The cards and connectors are keyed to prevent incorrect insertion. The +3V supply is permanently enabled.

3.11 Touch screen controller

The CPU board has two touch screen controllers:

- Burr-Brown ADS7843 touch screen controller, described in this section
- *Philips UCB1200 codec and touch screen controller* on page 3-39.

The default option in systems with a factory-fitted touch screen is the ADS7843. To use this controller, the touch screen is connected to the CPU board using the connector JP15.

The ADS7843 touch screen controller provides a direct interface to a 4-wire resistive touch screen. The ADS7843 operates from the 3V supply with a typical power consumption of 750µW.

Figure 3-13 shows the architecture of the touch screen interface.

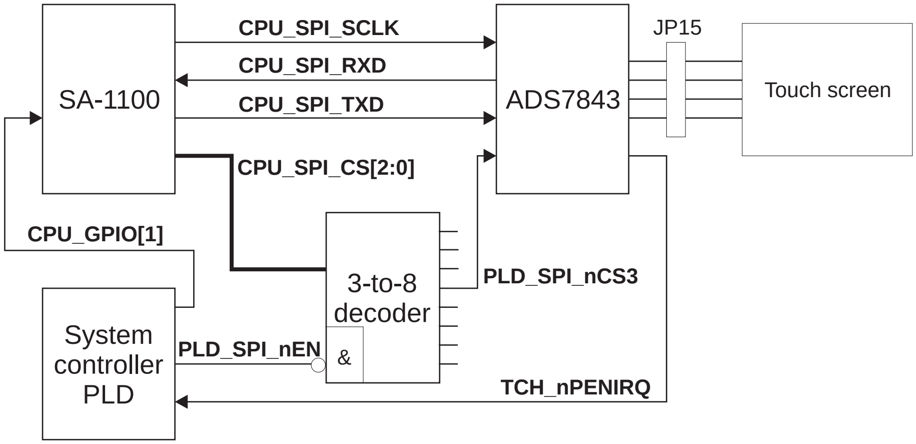


Figure 3-13 Touch screen interface (ADS7843)

The SA-1100 communicates with the ADS7843 using the SPI bus. The signal **PLD_SPI_CS3** is used to select the device. The maximum operating frequency for SPI bus accesses to the ADS7843 is 115 kHz. The SPI clock for this device idles HIGH and is sampled on the rising edge.

The ADS7843 issues interrupt requests by asserting **TCH_nPENIRQ** to the system interrupt controller PLD (see *Interrupt control* on page 3-12), which forwards it onto the CPU by asserting **CPU_GPIO[1]** and setting the PENIRQ bit in the interrupt flag register.

Information about programming the device and timing for the SPI bus are provided in *Programming the touch screen controller* on page 4-34.

3.12 Philips UCB1200 codec and touch screen controller

The Philips UCB1200 is a multifunction device. It incorporates the following functional blocks:

- *Touch screen controller* on page 3-40
- *Audio codec* on page 3-40
- *Telecom codec* on page 3-41
- *General purpose input/output pin block* on page 3-41
- *Multiplexed analog to digital converter* on page 3-42.

Figure 3-14 shows a simplified functional diagram of the UCB1200.

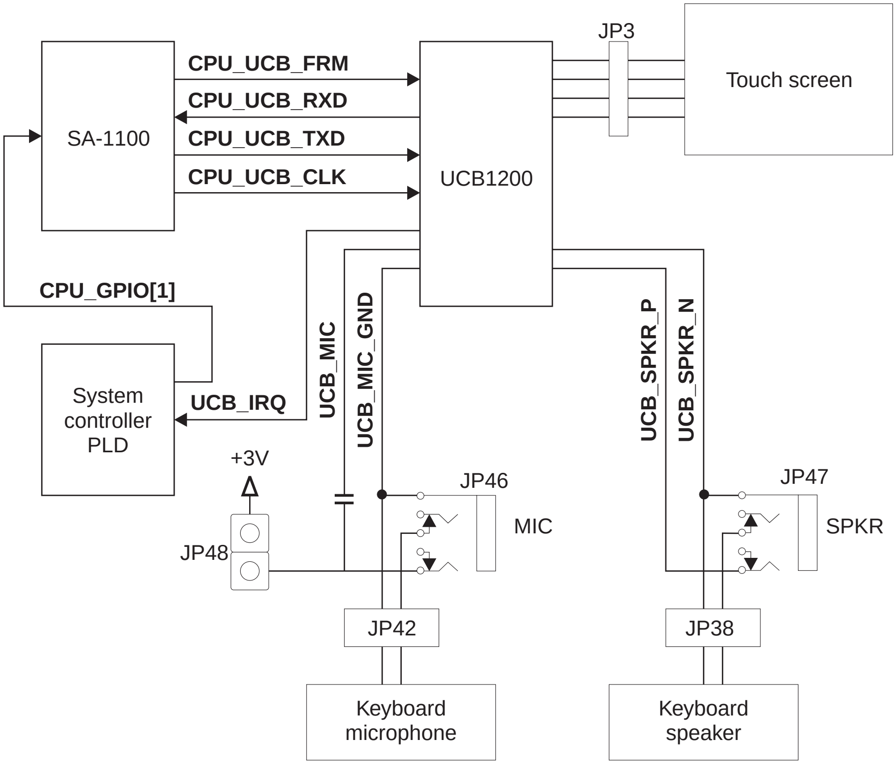


Figure 3-14 UCB1200 functional diagram

The UCB1200 is programmed using the *Serial Interface Bus* (SIB), which is provided by the MCP port of the SA-1100 (see *Programming the UCB1200* on page 4-27).

3.12.1 Touch screen controller

To use the touch screen controller within the UCB1200, you must connect the touch screen to the CPU board using the connector JP3.

This touch screen uses a four-wire interface consisting of the signals **UCB_TSPY**, **UCB_TSPX**, **UCB_TSMY**, and **UCB_TSM**. The controller carries out three types of measurement:

- X-Y position
- pressure
- plate resistance

The measurement type is defined in the touch screen control register. Measurements are typically taken by carrying out a sequence of:

1. Selecting the mode.
2. Selecting the appropriate input to the ADC.
3. Reading the value from the ADC data register.

The section *Programming the UCB1200* on page 4-27 provides more information about interface timing.

3.12.2 Audio codec

The audio codec is a 12-bit sigma-delta codec with programmable sample rates. It has outputs capable of directly driving a speaker and microphone, with programmable voltage levels.

The audio sample rate is derived from the SIB clock input to the UCB1200. The sample rate is programmable in the audio control register A. The sample rate is given by the following equation:

$$F_{sa} = \frac{(2 \cdot F_{sibclk})}{(64 \cdot audio\_divisor)}$$

The audio divisor must be greater than 5 and less than 128.

The UCB1200 samples both edges of the clock. The clock signal must have a 50% duty cycle if an odd divisor is used.

The audio codec drives the on-board microphone and speaker sockets, and the microphone and speaker on the distribution board, if connected. The distribution board devices are disconnected if external devices are connected into the jack sockets. The connector JP48 provides a +3V power source for active microphones.

3.12.3 Telecom codec

The telecom codec is a 14-bit sigma-delta codec with programmable sample rates. This codec has a programmable sample rate, and programmable input and output voltage levels. It is designed for direct connection to *Digital Access Arrangement* (DAA).

The telecom sample rate is derived from the SIB clock input to the UCB1200. The sample rate is programmable in the telecom control register A. The sample rate is given by the following equation:

$$F_{sa} = \frac{(2 \cdot F_{sibclk})}{(64 \cdot telecom\_divisor)}$$

The divisor must be greater than 5 and less than 128.

The UCB1200 samples both edges of the clock. The clock signal must have a 50% duty cycle if an odd divisor is used.

A number of the GPIO pins from the UCB1200 are designated for modem support, as shown in Table 3-19.

3.12.4 General purpose input/output pin block

The GPIO port provides 10 input/output pins. The pins can be individually configured as inputs or outputs in the IO port direction and register can be read from or written to by accessing the IO port data register. Each pin can be programmed to generate an interrupt on a rising or falling edge.

The pins are assigned as shown in Table 3-19.

Table 3-19 UCB1200 GPIO pin assignments

Port	Signal	Function
IO9	**UCB_ADSYNC**	AD converter synchronizing signal
IO8	**UCB_DAA_OH**	DAA off hook signal
IO7	**UCB_DAA_RI**	DAA ring indicator
IO6	**UCB_DAA_MUTE**	DAA mute signal
IO[5:0]	**UCB_LED[5:0]**	General purpose input/output. These are connected to the dual row 10-pin DF13 type connector, JP2.

3.12.5 Multiplexed analog to digital converter

The UCB1200 incorporates a 10-bit successive approximation ADC. The ADC has a 9-1 multiplexed input. The multiplexer is used to select between four switched resistive voltage divider inputs and the five touch screen inputs.

The switched resistive inputs are connected to the DF13-style connector JP3 and are available for user assigned functions. These inputs have a typical full-scale range of 7.5V (a maximum of 7.9V).

The touch screen inputs are described in *Touch screen controller* on page 3-40.

Conversions are synchronized by signal **UCB_ADCSYNC**. This is controlled by the IO9 pin from the general purpose input/output block.

3.13 Display interface

The P1100 provides support for a range of LCD options. The CPU card provides three connectors that support the following options:

- JP23: Active color TFT interface with 16bpp resolution
- JP24: Passive monochrome STN interface with 4bpp resolution
- JP22: to a power inverter for driving a cold cathode fluorescent lamp (CCFL) used with backlit displays

The SA-1100 incorporates a flexible LCD interface. It supports both single and dual panel displays. It must be configured appropriately for the type, screen size, refresh rate and resolution desired.

Systems equipped with a display are fitted with one of two Sharp TFT panels:

- 8.4inch VGA (640 x 480) 18bpp single panel reflective screen LQ084V2DS01 plus PE027-1 DF9 adaptor board

- 8.4inch VGA (640 x 480) 18bpp single panel transmissive screen LQ084V1DG21 with backlight and touchscreen

The relative merits of these two options are summarized in Table 3-20.

Table 3-20 Relative merits of the active TFT screen options

Type	Brightness	Contrast	Power Consumption	Touchscreen	Battery Operation
Reflective	LOW	LOW	LOW	NO	YES
Transmissive (backlit)	HIGH	HIGH	HIGH	YES	NO

Active color TFTs employ an active matrix of transistors to control the pixels. As a result they give fast switching times which prevents ghosting. Hyper-reflective screens illuminate the screen by using ambient light reflected back through the LCD matrix from a mirror finish behind the display elements. This makes them unsuitable for low light conditions and for use with a touch screen.

The SA-1100 supports 16-bit color. One color receives 6bits of color information and the two other receive 5bits. This requires1-bit of 2 of the colors to be tied either high or low. The best compromise is to tie the LSB of the red and blue low. This gives good colors and acceptable blacks to whites.

3.14 Distribution board

Systems fitted with an LCD display and keyboard contain a distribution board within the keyboard housing. It carries LEDs, a microphone and a speaker but is mainly used as a system interconnect. The distribution board connects to the P1100 module through four connectors on the CPU board. These are summarized in Table 3-21.

Table 3-21 LCD signal connectors

Connector	Function
JP38	Keyboard matrix row and some column. Speaker connections.
JP39	Provides the remaining keyboard column connections.
JP40	Provides signals for five LED drives (one driven by the SA-1100 and four by the keyboard controller), a PS/2 interface from the keyboard controller device, and system reset.
JP42	Provides microphone connections

Note

+5V, +3V and 0V are also provided by these connectors. Please see Appendix C for connector pinout information.

The LEDs are driven in parallel with surface mount LEDs on the CPU board.

Mechanical provision has been made for incorporating a touch pad as an alternate pointing device. However, this does require a redesign of the escutcheon plate used to mount the device.

The micro joystick used in the P1100 design is the MicroModule, part number VP5510. This is a compact button style joystick providing 2-button mouse functionality.

3.15 Power supply

This section describes power distribution and management on the P1100 CPU card and describes:

- Power supply architecture
- *Power supply inputs* on page 3-46
- *Internal power distribution* on page 3-47
- *Power management* on page 3-49.

3.15.1 Power supply architecture

The power supply system uses three MAXIM switching regulators and three IRF730x MOSFET switch packages. A number of the power supply outputs can be switched independently allowing you to reduce power when some devices are not being used.

Regulator mode control

The main +3V rail is supplied by a Maxim MAX1703 DC-DC converter. This can operate PWM mode or in PFM mode when powering light loads. The PWM mode is used for normal system operation and the more efficient PFM mode is when less power is required.

For example, when the system is in sleep mode, select the PFM mode by writing a 0 to the PWR_MODE bit in the power control register in the PLD. When the system is active, select the PWM mode by writing a 1 to the PWR_MODE bit.

All the switched outputs and the MAX1703 mode are controlled by the PLD_PWR register in the system controller PLD (see *LCD power control and battery sensing* on page 3-14). On power up the PLD places the MAX1703 in high power PWM mode.

Output switching

Switched outputs are controlled by applying a HIGH (3V) or LOW (0V) signal from the the PLD to the gate of a MOSFET switch. The switches used are devices manufactured by International Rectifier. These have the following operating functions:

- The IRF7306 U47 is used to switch the LCD bias supply **LCD_BIAS_SW**, and the LCD logic supply **LCD_OUT1**
- The IRF7304 U38 is used for power source selection (external PSU or battery) by a hardware controlled signal
- The IRF7304 U46 are to switch the +9V backlight inverter supply **LCD_OUT3**, and the switched + 3V3 LCD supply **LCD_OUT2**.

Power input to the P1100 is selected automatically. If an external supply is inserted and switched on, it causes a MOSFET switch arrangement (U38) to select the regulated +3V source in preference to the +3V battery supply.

3.15.2 Power supply inputs

The P1100 can be powered from two sources:

- external power supply
- battery

The choice of power source largely depends on the type of system you have.

External power supply

An external power supply can be connected to the 1.3mm jack JP32 (see *Supplying power* on page 2-4). The external supply should provide an output voltage of between +6V and +12V. If a backlit display is being used, then a 9V/1.5A supply is needed.

A suitable power supply is shipped with the P1100 system.

Two linear regulators provide +3V and +5V to the system:

3V regulator Linear Technology LT1085, positive adjustable linear regulator, able to supply an output current of 3A. Fitted with a 10.4 C/W heatsink.

5V regulator National Semiconductor LM2940CT-5.0 is a fixed 5V linear regulator, capable of providing 1A.

The +5V supply is used for system expansion only and the +3V supply is used directly for FPGA expansion in high current drain applications.

Battery power

A battery connector (JP33) allows the use of 2xAA batteries supplying +3V. An automatic switch is used to select one input supply or the other, with the external supply taking precedence. The common (selected) supply signal name is **3V_SUPPLY**.

Systems that are fitted with a display provide a battery compartment located beneath the display hinge (see *Installing batteries* on page 2-4). This can accommodate two AA sized batteries. You can also wire your own battery holder to JP33.

Caution

Take great care to connect the battery with the correct polarity. The positive pin (JP33 pin 2) is closest to the edge of the board.

Battery backup

The P1100 provides a backup capacitor that powers the system for a short while if neither the battery supply or external supply connected. This is a 1Farad SuperCAP, designed to provide power for a short duration while the batteries are being changed. It is designed to supply minimal power, with the system in sleep mode with DRAM in self refresh, for around 1 minute maximum. The backup is charged from the battery.

3.15.3 Internal power distribution

The power supply subsystem outputs are summarized in Table 3-22 on page 3-47.

Table 3-22 Power supply rail usage

Name	Switched	Source	Usage
3V_SUPPLY	No	Linear regulator U37 and	
PSU_+1V5	Yes	MAX1703 from **3V_SUPPLY** using the gain block in the MAX1703 and P-channel MOSFET Q3 as a linear regulator.	+1.5V to the SA-1100 core. Switched by the SA-1100 when the +3V rail is stable. **CPU_PWR_EN** is used to switch the supply on using a FMMT617 transistor.
+3V	No	MAX1703 from **3V_SUPPLY**.	+3V main system power, including SA-1100 input/output pad voltage.
LCD_OUT1	Yes	MAX1703 from **3V_SUPPLY**.	A switched version of **+3V** controlled by the PLD_S1_OFF bit in the PLD power control register.
LCD_OUT2	Yes	MAX863 from **3V_SUPPLY**.	Used to supply the LCDs, a **+3V3** controlled by the PLD_S2_ON bit in the PLD power control register.
+5VO	No	Maxim MAX863 from **3V_SUPPLY**.	Used to supply the LCDs, a **+5V** to the StrataFlash and associated 74AHCT244 (U18) control buffer.
LCD_OUT3	Yes	External power supply.	+9V used to power the backlight inverter in a transmissive color LCD configuration. Controlled by the PLD_S3_ON bit in the PLD power control register.

Table 3-22 Power supply rail usage (continued)

Name	Switched	Source	Usage
PSU_LCD_BIAS	Yes	Maxim MAX686.	Variable +14V to 20V Bias voltage for monochrome LCDs. Digitally controlled output ranging between +14V to +20V, controlled by setting either **PLD_LCD_UP** or **PLD_LCD_DN** in the LCD bias control register. Switched ON and OFF by the PLD_S4_ON bit in the power control register.
EXP_+3V	No	Regulator U37.	+3V alternate supply for the FPGA. The supply source is selected by using a link on header JP27.
EXP_+5V	No	Regulator U36.	Provided by a linear regulator from the external +9V source. This is the only source of power routed to the expansion connector JP20.

Note

To conserve power in the P1100 design the main power rail is +3V. This powers the 3.3V parts at the limit of their standard tolerance.

3.15.4 Power switching

Several of the power supply outputs can be switched ON and OFF in the PLD power control register, as indicated in Table 3-22 on page 3-47. There are a number of control signals from other devices. These are:

POK power OK signal. Provides the SA-1100 and keyboard controller as a power supply status flag.

PSU_BATT_OK supplied to the keyboard controller. When deasserted the keyboard controller enters its CRITICAL SUSPEND mode. It exits CRITICAL SUSPEND when **PSU_BATT_OK** is asserted.

PSU_VDD_FLT an inverted version of **PSU_BATT_OK** which is supplied to the SA-1100. When asserted the SA-1100 enters sleep mode.

3.15.5 Power management

The P1100 has three operating modes:

Normal mode The system is fully operational. For module-only variants, all the LCD supplies can be permanently powered off.

Configurations with keyboards and LCDs need to enable the **LCD_OUT2** LCD supply and, for transmissive displays with backlight, the **LCD_OUT3** inverter supply.

Idle mode Used when the case is open and the system is waiting for a keyboard event. The LCD must be refreshed, and all the peripherals capable of bringing the machine out of IDLE must be configured to generate an interrupt. This is primarily the keyboard controller and touchscreen, but can also include the serial ports and IrDA interfaces. Any power-saving is limited in this mode because the DRAM and SA-1100 core must remain active to allow screen-refresh to continue.

Sleep mode Used when the case is closed or no activity detected for a predetermined period of time. In this mode the screen and the SA-1100 core are powered down. The system consumes as little power as possible.

Entering sleep mode

To enter sleep mode the following steps should be taken:

1. Turn off LCD
2. Turn off LCD power supply
3. Disable the LCD controller
4. Turn off all other supplies
5. Enable wake-up keys on the keyboard controller.
6. Flush SA-1100 Dcaches and save the processor context into memory
7. Place DRAMs into self refresh mode
8. Place the SA-1100 into sleep mode

System wakeup

To wake from sleep mode:

1. Restore the SA-1100 configuration registers and any system context.
2. Take the DRAM out of self refresh mode.
3. Re-initialize and power up the LCD.

3.16 JTAG interface

The SA-1100 and XCR3128 PLD both contain a JTAG *Test Access Port* (TAP). The SA-1100 supports boundary scan, which can be used to program the flash, and the PLD supports in-circuit programming.

Figure 3-15 shows the JTAG data routing in The P1100.

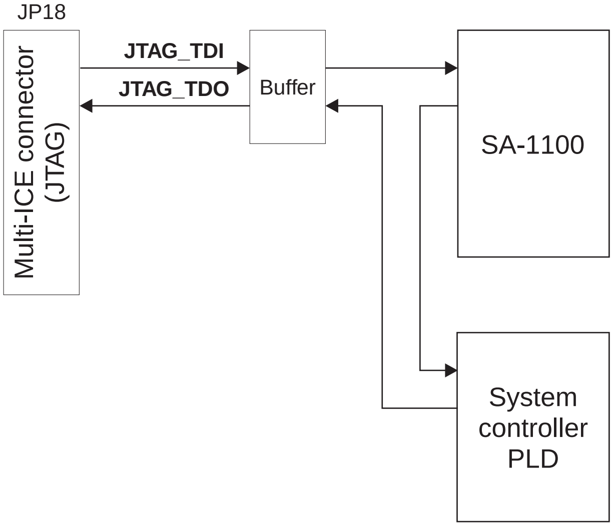


Figure 3-15 JTAG architecture

The P1100 connects the TAP controllers in the SA-1100 and PLD in a daisy chain. The JTAG connector is buffered through a 74LCX541 octal buffer. This is a 3-volt device with 5-volt tolerant inputs. The scan chain TDI signal is routed to the SA-1100 TAP as **BUF_JTAG_IN**. The TDO signal **CPU_TDO** from the SA-1100 provides the TDI input to the XCR3128. The TDO signal from the PLD is then returned to the JTAG interface through the buffer.

The JTAG reset signal has a pull down on the board to minimize power consumption in normal use, and a pullup on the cable to ensure correct functioning of Multi-ICE.

A small profile, single row 8-way vertical DF13 type connector (JP18) provides the JTAG external interface. An ARM supplied programming utility can be used to program the PLD using Multi-ICE. An SA-1100 boundary scan utility is supplied to enable you to program the flash.

Alternatively, Xilinx supply a software utility to download data to the PLD and a sample cable design to attach to the parallel port of an IBM PC or compatible.

3.16.1 Debug cables

The Xilinx cable specification is shown is Table 3-23.

Table 3-23 JTAG interface cable specification (Xilinx)

PC connector pin	JP18 pin (P1100)	Function
1	3	TDO
2	6	TCK
3	5	TDI
4	4	TMS
18,19,20,21	8	GND

The ARM Multi-ICE cable specification is shown in Table 3-24.

Table 3-24 Multi-ICE cable specification

Multi-ICE connector pin	JP18 pin (P1100)	Function
1, 2	1	**+3V**
13	3	**TDO**
9	6	**TCK**
5	5	**TDI**
7	4	**TMS**
12	8	**GND**
3	2	**nTRST**
15	7	**nSRST**

Note

Multi-ICE connector pin 3 (**nTRST**) is connected to pin 2 of the same connector with a 2kΩ resistor fitted inside the connector shell. This pulls the **nTRST** signal HIGH.

3.16.2 Debug ports

Three 38-way Mictor connectors are provided to connect a logic analyzer to the key system signals in the design. The Mictor connectors have been laid out to easily connect to a Hewlett Packard HP16xxx series logic analyzer.

The primary function of each connector is:

- JP34: SA-1100 address bus
- JP35: SA-1100 data bus
- JP36: SA-1100 and PLD generated system control signals.

Chapter 4
Programmer's Reference

This chapter describes the programming interfaces of the main components of the P1100. It contains the following sections:

4.1 About this chapter

This chapter describes the memory map and the system controller PLD. It also discusses how to access the major peripheral devices and describes the interface timing requirements.

The registers for devices, such as those in the SA-1100, are listed for convenience, but you should refer to the documentation supplied by the device manufacturers for detailed information. See *Other publications* on page -xv.

4.2 Memory map

The SA-1100 determines the memory map of the P1100. The SA-1100 memory map is shown in Table 4-1. The way in which the major regions are implemented is described below.

Table 4-1 SA-1100 memory map

Base address	Size	Description
0x00000000	128MB	Static bank select 0
0x08000000	128MB	Static bank select 1
0x10000000	128MB	Static bank select 2
0x18000000	128MB	Static bank select 3
0x20000000	256MB	Reserved for PCMCIA socket 0
0x30000000	256MB	Reserved for PCMCIA socket 1
0x40000000	1GB	Reserved
0x80000000	256MB	Peripheral module registers
0x90000000	256MB	System control module registers
0xA0000000	256MB	Memory and expansion registers
0xB0000000	256MB	LCD and DMA registers
0xC0000000	128MB	DRAM bank 0
0xC8000000	128MB	DRAM bank 1
0xD0000000	128MB	DRAM bank 2
0xD8000000	128MB	DRAM bank 3
0xE0000000	128MB	Zeroes bank
0xE8000000	128MB	Reserved

4.2.1 Static bank select 3:0

The static bank select regions are used to support accesses to the flash memory, system controller PLD internal registers, and expansion space. The system controller PLD encodes three of the bank select signals (**CPU_nCS[2:0]**) from the CPU and

CPU_A[25:24] to map eight 16MB areas. One of these areas contains the PLD internal registers and the other areas generate the chip select signals. The resulting physical memory map for the static bank select regions is shown in Table 4-2.

Table 4-2 Static memory region memory map

Base address	Device	Static bank select	PLD chip select
`0x00000000`	Flash region 0	0	Flash_CS0
`0x01000000`	Flash region 1	0	Flash_CS1
`0x02000000`	Flash region 2 (option)	0	Flash_CS2
`0x03000000`	Flash region 3 (option)	0	Flash_CS3
`0x08000000`	FPGA	1	Flash_CS6
`0x10000000`	PLD registers	2	-
`0x12000000`	Expansion	2	Flash_CS4
`0x13000000`	Expansion	2	Flash_CS5
`0x18000000`	FPGA decode (option)	3	-

For more information about how the flash chip selects are generated, see *Flash chip-select control* on page 3-18.

4.2.2 PCMCIA socket 1:0

This region is mapped conventionally. There are no PCMCIA devices on the P1100 CPU board.

4.2.3 Peripheral module registers

The peripheral module registers are summarized in Table 4-3. The notes column indicates required values and the default values set by the ARM Firmware Suite, where appropriate. Detailed descriptions of these registers can be found in the *Intel StrongARM SA-1100 Microprocessor Developer's Guide*.

Table 4-3 Peripheral module registers

Address	Register	Notes
COM1 (UART0 – serial port 0)		
`0x8001 0000`	UART1 Control Register 0	Bit 0 Parity Enable Bit 1 Parity Select Bit 2 Stop Bit Select Bit 3 Data Size Select Bit 4 Sample Clock Enable Bit 5 Receive Clock Edge Select Bit 6 Transmit Clock edge Select
`0x8001 0004`	UART1 Control Register 1	Bit 0-3 Baud Rate Divisor
`0x8001 0008`	UART1 Control Register 2	Bit 0-7 Baud Rate Divisor
`0x8001 000C`	UART1 Control Register 3	Bit 0 Receiver Enable Bit 1 Transmitter Enable Bit 2 Break Bit 3 Receive FIFO Interrupt Enable Bit 4 Transmit FIFO Interrupt Enable Bit 5 Loopback Mode
`0x8001 0010`	UART1 Control Register 4	
`0x8001 0014`	UART1 Data Register	
`0x8001 001C`	UART1 Status Register 0	Bit 0 Transmit FIFO Service Request Bit 1 Receive FIFO Service Request Bit 2 Receiver Idle Bit 3 Receiver Begin of Break Bit 4 Receiver End of Break Bit 5 Error in FIFO
`0x8001 0020`	UART1 Status Register 1	Bit 0 Transmitter Busy Flag Bit 1 Receive FIFO Not Empty Bit 2 Transmit FIFO Not Full Bit 3 Parity Error Bit 4 Framing Error Bit 5 Receive FIFO Overrun

Table 4-3 Peripheral module registers (continued)

Address	Register	Notes
IrDA (UART2 – Serial port 2)		
`0x8003 0000`	UART2 Control Register 0	
`0x8003 0004`	UART2 Control Register 1	
`0x8003 0008`	UART2 Control Register 2	
`0x8003 000C`	UART2 Control Register 3	
`0x8003 0010`	UART2 Control Register 4	
`0x8003 0014`	UART2 Data Register	
`0x8003 001C`	UART2 Status Register 0	
`0x8003 0020`	UART2 Status Register 1	
`0x8004 0060`	UART2 HSSP Control Register 0	
`0x8004 0064`	UART2 HSSP Control Register 1	
`0x8004 006C`	UART2 HSSP Data Register	
`0x8004 0074`	UART2 HSSP Status Register 0	
`0x8004 0078`	UART2 HSSP Status Register 1	
`0x9006 0028`	UART2 HSSP Control Register 2	This register is in the PPC space
COM 2 (UART 3 – serial port 3)		
`0x8005 0000`	UART3 Control Register 0	Bit 0 Parity Enable Bit 1 Parity Select Bit 2 Stop Bit Select Bit 3 Data Size Select Bit 4 Sample Clock Enable Bit 5 Receive Clock Edge Select Bit 6 Transmit Clock edge Select
`0x8005 0004`	UART3 Control Register 1	Bit 0-3 Baud Rate Divisor
`0x8005 0008`	UART3 Control Register 2	Bit 0-7 Baud Rate Divisor

Table 4-3 Peripheral module registers (continued)

Address	Register	Notes
`0x8005 000C`	UART3 Control Register 3	Bit 0 Receiver Enable Bit 1 Transmitter Enable Bit 2 Break Bit 3 Receive FIFO Interrupt Enable Bit 4 Transmit FIFO Interrupt Enable Bit 5 Loopback Mode
`0x8005 0014`	UART3 Data Register	
`0x8005 001C`	UART3 Status Register 0	
`0x8005 0020`	UART3 Status Register 1	Bit 0 Transmit FIFO Service Request Bit 1 Receive FIFO Service Request Bit 2 Receiver Idle Bit 3 Receiver Begin of Break Bit 4 Receiver End of Break Bit 5 Error in FIFO Bit 0 Transmitter Busy Flag Bit 1 Receive FIFO Not Empty Bit 2 Transmit FIFO Not Full Bit 3 Parity Error Bit 4 Framing Error Bit 5 Receive FIFO Overrun
UCB1200 (MCP – serial channel 4). These registers set up the port used to communicate with the UCB1200		
`0x8006 0000`	MCP Control Register 0	
`0x8006 0008`	MCP Data Register 0	
`0x8006 000C`	MCP Data Register 1	
`0x8006 0010`	MCP Data Register 2	
`0x8006 0018`	MCP Status Register	
SPI bus (SSP – serial channel 4)		
`0x8007 0060`	SSP Control Register 0	Bit 0-3 Data Size Select Bit 4-5 Frame Format Bit 7 Synchronous Serial Port Enable Bit 8-15 Serial Clock Rate

Table 4-3 Peripheral module registers (continued)

Address	Register	Notes
0x8007 0064	SSP Control Register 1	Bit 0 Receive FIFO Interrupt Enable Bit 1 Transmit FIFO Interrupt Enable Bit 2 Loopback Mode Bit 3 Serial Clock Polarity Bit 4 Serial Clock Phase Bit 5 External Clock Select
0x8007 006C	SSP Data Register	
0x8007 0074	SSP Status Register	Bit 1 Transmit FIFO Not Full Bit 2 Receive FIFO Not Empty Bit 3 SSP Busy Flag Bit 4 Transmit FIFO Service Request Bit 5 Receive FIFO Service Request Bit 6 Receive FIFO Overrun

4.2.4 System control module registers

The system control module registers provide status and control for the operating system timer, real-time clock, interrupt controller, and power management features of the SA-1100. This group of registers also provides access to and control of the GPIO pins. Table 4-4 lists the registers in this group. Detailed descriptions of these registers can be found in the *Intel StrongARM SA-1100 Microprocessor Developer's Guide*.

Table 4-4 System controller module registers

Address	Register	Notes
OS Timer		
0x9000 0000	OS Timer Match Register 0	
0x9000 0004	OS Timer Match Register 1	
0x9000 0008	OS Timer Match Register 2	
0x9000 000C	OS Timer Match Register 3	
0x9000 0010	OS Timer Counter Register	
0x9000 0014	OS Timer Status Register	
0x9000 0018	OS Timer Watchdog Enable Register	
0x9000 001C	OS Timer Interrupt Enable Register	

Table 4-4 System controller module registers

Address	Register	Notes
Real-time clock		
`0x9001 0000`	RTC Alarm Register	
`0x9001 0004`	RTC Count Register	
`0x9001 0008`	RTC Timer Trim Register	
`0x9001 0010`	RTC Status Register	
Power manager		
`0x9002 0000`	Power Manager Control Register	
`0x9002 0004`	Power Manager Sleep Status Register	
`0x9002 0008`	Power Manager Scratchpad Register	
`0x9002 000C`	Power Manager Wake up Register	
`0x9002 0010`	Power Manager Configuration Register	
`0x9002 0014`	Power Manager PLL Configuration Register	
`0x9002 0018`	Power Manager GPIO Sleep State Register	
`0x9002 001C`	Power Manager Oscillator Status Register	
Reset controller		
`0x9003 0000`	Reset Controller Software Reset Register	
`0x9003 0004`	Reset Controller Status Register	
GPIO		
`0x9004 0000`	GPIO Pin-Level Register	Bit 0 PLD Interrupt0 Status Bit 1 PLD Interrupt1 Status Bit 16:14 encoded SPI chip select Bit 17 UART1 RTS Status Bit 18 UART1 CTS Status Bit 19 UART2 RTS Status Bit 20 UART2 CTS Status Bit 25 CPU LED Status

Table 4-4 System controller module registers

Address	Register	Notes
0x9004 0004	GPIO Pin Direction Register	0 = Input 1 = Output Bit 0 Set To Input Bit 1 Set To Input Bit 14-16 Set To Output Bit 17 Set To Output Bit 18 Set To Input Bit 19 Set To Output Bit 20 Set To Input Bit 25 Set To Output
0x9004 0008	GPIO Pin Output Set Register	Set to 1 to Set Pin Bit 14-16 Encoded SPI Device Selection (Active High) Bit 17 UART1 RTS Bit 19 UART2 RTS Bit 25 CPU LED (Active Low)
0x9004 000C	GPIO Pin Output Clear Register	Set to 1 to Clear Pin Bit 14-16 Encoded SPI Device Selection Active High Bit 17 UART1 RTS Bit 19 UART2 RTS Bit 25 CPU LED Active Low
0x9004 0010	GPIO Rising Edge Detect Register	
0x9004 0014	GPIO Falling Edge Detect Register	
0x9004 0018	GPIO Edge Detect Status Register	
0x9004 001C	GPIO Alternate Function Register	Set The Following Bit 2-9 LCD High Order Pins Bit 10 SSP Transmit Bit 11 SSP Receive Bit 12 SSP Clock Bit 13 SSP Frame Bit 21 MBGNT Bit 22 MBREQ Bit 26 RCLK Out Bit 27 32KHz Out

Table 4-4 System controller module registers

Address	Register	Notes
Interrupt controller		
`0x9005 0000`	Interrupt Controller IRQ Pending Register	
`0x9005 0004`	Interrupt Controller Mask Register	
`0x9005 0008`	Interrupt Controller Level Register	
`0x9005 001C`	Interrupt Controller FIQ Pending Register	
`0x9005 0020`	Interrupt Controller Pending Register	
`0x9005 0010`	Interrupt Controller Control Register	

4.2.5 Memory and expansion registers

This group of registers is used to configure the memory and expansion interfaces of the SA-1100. These are set up by the ARM Firmware Suite. Detailed descriptions of these registers can be found in the *Intel StrongARM SA-1100 Microprocessor Developer's Guide*. Table 4-5

Table 4-5 Memory and expansion registers

Address	Register	Notes
`0xA000 0000`	DRAM Configuration Register	
`0xA000 0004`	DRAM CAS Waveform Shift Register 0	
`0xA000 0008`	DRAM CAS Waveform Shift Register 1	
`0xA000 000C`	DRAM CAS Waveform Shift Register 2	
`0xA000 0010`	Static memory Control Register 0	
`0xA000 0014`	Static memory Control Register 1	
`0xA000 0018`	Expansion Bus Configuration Register	

4.2.6 LCD controller registers

This set of registers is used to configure and control the display interface. These are set up by ARM Firmware Suite, where appropriate. Detailed descriptions of these registers can be found in the *Intel StrongARM SA-1100 Microprocessor Developer's Guide*

Table 4-6 LCD control registers

Address	Register	Notes
`0xB010 0000`	LCD controller control register 0	
`0xB010 0004`	LCD controller status register	
`0xB010 0010`	DMA channel 1 base address register	
`0xB010 0014`	DMA channel 1 current address register	
`0xB010 0018`	DMA channel 2 base address register	
`0xB010 001C`	DMA channel 2 current address register	
`0xB010 0020`	LCD controller control register 1	
`0xB010 0024`	LCD controller control register 2	
`0xB010 0028`	LCD controller control register 3	

4.2.7 DRAM banks

The P1100 provides two areas of DRAM which are located within the DRAM bank select 0 and 1 regions. These are used for general purpose data storage or for storing screen-image data. The DRAM configuration register must be set up correctly to support the size and memory device configuration for each bank, as shown in Table 4-7 (refer to the *Intel StrongARM SA-1100 Developer's Manual*).

Table 4-7 DRAM bank addresses

Base address	Bank	Configuration
`0xC0000000`	DRAM bank 0	Both banks are 16MB using 4Mword x 16bit devices in 12x10 configuration. Refresh period of 128ms.
`0xC8000000`	DRAM bank 1	

4.3 System controller PLD registers

The system controller PLD provides eleven registers:

- *Interrupt flag register*
- *Power control register* on page 4-14
- *Keyboard control register* on page 4-15
- *SPI decoder enable register* on page 4-16
- *Software input/output control register* on page 4-16
- *IrDA enable register* on page 4-16
- *COM2 enable register* on page 4-17
- *COM1 enable register* on page 4-17
- *CODEC enable and IRQ register* on page 4-18
- *LCD brightness register* on page 4-18
- *LCD enable register* on page 4-18
- *PLD version register* on page 4-19
- *GPIO register* on page 4-19
- *Touch screen register* on page 4-19.

4.3.1 Interrupt flag register

The interrupt flag register, PLD_INT (0x10000000), contains six bits that indicate the source of the interrupt request that caused an interrupt to be signalled on the GPIO0 or GPIO1 pins of the SA-1100. The bits function as follows:

- Bits are set to '1' to indicate an active interrupt from the associated source
- multiple interrupts result in multiple flags being set
- interrupt flags are cleared by clearing the interrupt request from the source device.

The bits in this register are summarized in Table 4-8.

Table 4-8 PLD interrupt flag register

Bit	Name	Function
5	PENIRQ	Indicates an interrupt generated by the ADS7843 touchscreen controller, if it is fitted.
4	COM2_IRQ	Indicates a change of state of COM2, either when a cable is connected or disconnected. Also indicates when COM2 has valid signal levels and is ready to communicate.
3	COM1_IRQ	Indicates a change of state of COM1, either when a cable is connected or disconnected. Also indicates when COM1 has valid signal levels and is ready to communicate.
2	CDC_IRQ	Indicates when an interrupt is generated FPGA internal codec. Only present if FPGA is fitted.
1	UCB_IRQ	Indicates when an interrupt is generated by the UCB1200, see *UCB1200 interrupts* on page 4-29.
0	KBD_ATN	Indicates the assertion of the signal **KBD_nATN** by the keyboard controller, see *Keyboard controller interrupts* on page 4-25.

4.3.2 Power control register

The power control register, PLD_PWR (0x10000004), is used to:

- enable LCD power supplies
- select the power supply operating mode
- detect an external power supply

The bits in this register are described in Table 4-9.

Table 4-9 PLD power control register

Bit	Name	Function
5	PWR_EXT (read-only)	Indicates when an external power supply is present: 0 = external supply not present 1 = external supply present
4	PWR_MODE	Selects the power supply operating mode: 0 = PFM 1 = PWM
3	PLD_S4_ON	Enable/disables the switched bias voltage for the monochrome LCD interface: 0 = disabled 1 = enabled
2	PLD_S3_ON	Enable/disables the 9V LCD backlight supply LCD_OUT3: 0 = disabled 1 = enabled
1	PLD_S2A_ON	Enable/disables the 3.3V supply output LCD_OUT2 used by the LCD: 0 = disabled 1 = enabled
0	PLD_S1_ON	Enable/disables the 3Vpower supply output LCD_OUT1: 0 = disabled 1 = enabled

4.3.3 Keyboard control register

The keyboard control register, PLD_KBD (0x10000008), contains two active bits, as shown in Table 4-10.

Table 4-10 PLD keyboard control register

Bit	Name	Function
1	KBD_WAKE	Keyboard wakeup
0	KBD_EN	Enables the keyboard controller

4.3.4 SPI decoder enable register

The SPI decoder enable register, PLD_EN (0x1000000C), contains one active bit (bit 0) which is used to enable/disable the SPI bus chip-select decoder (see *SPI bus* on page 3-28).

- 0 = SPI chip-select decoder disabled
- 1 = SPI chip-select decoder enabled.

4.3.5 Software input/output control register

The Software input/output control register, PLD_IO (0x10000010), register controls the LEDs connected to the PLD IO pins (see *LED control* on page 3-33) and monitors the DEVMODE and BOOTSEL switches.

Table 4-11 PLD input/output register

Bit	Name	Function
6	PLD_BOOTSEL (read-only)	This bit indicates the setting of the BOOTSEL switch (see *Setting the switches* on page 2-3).
5	PLD_DEVMODE (read-only)	This bit indicates the setting of the DEMODE switch (see *Setting the switches* on page 2-3).
4	PLD_LED3	These bits are used to enable/disable individual LEDs that are connected to the PLD: 0 = LED OFF 1 = LED ON
3	PLD_LED2	
2	PLD_LED1	
1	PLD_LED0	
0	PLD_LEDEN	This bit is used to enable/disable all of the LEDs connected to the PLD, overriding the individual control bits. 0 = LEDs disabled 1 = LEDs enabled

4.3.6 IrDA enable register

The IrDA enable register, PLD_IrDA (0x10000014), contains one active bit (bit 0) which is used to enable/disable the IrDA port transceiver.

- 0 = IrDA port disabled
- 1 = IrDA port enabled.

4.3.7 COM2 enable register

The COM2 enable register, PLD_COM2 (0x1000018), provides one control bit and two status bits for COM2, as shown in Table 4-12.

Table 4-12 PLD COM2 enable register

Bit	Name	Function
2	COM2_RDY (read only)	When set indicates that the COM2 ready signal caused the COM2 interrupt indicated in the PLD_INT register.
1	COM2_SENSE (read only)	When set indicates a cable connection or disconnection from COM2 caused the COM2 interrupt indicated by the PLD_INT register.
0	COM2_EN	This bit is used to enable/disable the COM2 port transceiver: 0 = disabled 1 = enabled

4.3.8 COM1 enable register

The COM1 enable register, PLD_COM1 (0x1000001C), provides one control bit and two status bits for COM1, as shown in Table 4-13.

Table 4-13 PLD COM1 enable register

Bit	Name	Function
2	COM1_RDY (read only)	When set indicates that the COM1 ready signal caused the COM1 interrupt indicated by the PLD_INT register.
1	COM1_SENSE (read only)	When set indicates a cable connection or disconnection from COM1 caused the COM1 interrupt indicated by the PLD_INT register.
0	COM1_EN	This bit is used to enable/disable the COM2 port transceiver: 0 = disabled 1 = enabled

4.3.9 CODEC enable and IRQ register

The CODEC enable and IRQ register, PLD_CODEC (0x10400000), provides an enable/disable control and read-only indicators from the FPGA codec, if it is present. Table 4-14 describes the bits in this register. The bit assignments assume the use of the optional FPGA.

Table 4-14 FPGA codec register

Bit	Name	Function
4	CODEC_IRQ3	This bit is only used if the FPGA is fitted
3	CODEC_IRQ2	This bit is only used if the FPGA is fitted
2	CODEC_IRQ1	This bit is only used if the FPGA is fitted
1	Reserved	This bit reserved, read as 0
0	CODEC_EN	This bit is only used if the FPGA is fitted to enable/disable the codec: 0 = disabled 1 = enabled

4.3.10 LCD brightness register

The LCD brightness control register, PLD_BRITE (0x10400004), is used to vary the brightness of a monochrome LCD display.

Table 4-15 PLD LCD brightness register

Bit	Name	Function
1	LCD_UP	Write 1 to this bit to increase the display brightness.
0	LCD_DN	Write 1 to this bit to decrease the display brightness.

4.3.11 LCD enable register

The LCD enable register, PLD_LCDEN (0x10400008), contains one active bit (bit 0) used to switch a monochrome display ON or OFF, when fitted. Write a 1 to enable the LCD display, write a 0 to disable the LCD display.

4.3.12 PLD version register

The PLD version register, PLD_ID (0x1040000C), is a read-only register which returns the version number of the system controller PLD.

4.3.13 GPIO register

The PLD GPIO register, PLD_GPIO (0x10400014), is used to control or read the three GPIO pins on the PLD. The bit assignment is shown in Table 4-16.

Table 4-16 GPIO register

Bit	Name	Function
2	PLD_GPIO2	The value set for a bit in this register appears on the associated PLD GPIO pin.
1	PLD_GPIO1	
0	PLD_GPIO0	

4.3.14 Touch screen register

The PLD touch screen register, PLD_TCH (0x10400010), is used to control communications between the touchscreen controller and SA-1100 The bit assignment is shown in Table 4-17.

Table 4-17 Touchscreen register

Bit	Name	Function
1	TCH_nPENIRQ	Inicates the level on the touchscreen controller IRQ pin.
0	TCH_EN	Is used to enable or disable touchscreen interface and interrupts 0 = interrupts enabled and interface disabled 1 = interrupts disabled and interface enabled.

The PLD_TCH register is used as follows:

1. Write 0 to TCH_EN to disable the interface, and enable the PENIRQ interrupt.
2. Wait for an interrupt, is indicated by a 1 in TCH_nPENIRQ bit.
3. When an interrupt is received, write a 1 to TCH_EN to enable the interface and disable the PENIRQ interrupt, and then commence transactions with the ADS7843.
4. When the transaction is complete, write a 0 to TCH_EN to disable the interface and re-enable the PENIRQ interrupt.

4.4 Programming the keyboard controller

The SA-1100 and the keyboard controller communicate using the SPI bus, which utilizes a number of GPIO pins provided by the SA-1100. The operation of the keyboard controller is controlled by accessing internal registers, of which it contains two banks of 255 registers. These are accessed using a message exchange protocol. The timing for the SPI bus is described in *Keyboard controller host interface timing* on page 4-23.

4.4.1 Message structure

The SA-1100 uses messages to read from and write to individual bits within the internal registers. Messages are also used by the keyboard controller to report events and to respond to reads and writes by the host. Each message comprises:

- a header
- a command or a report identifier
- a message body, if necessary
- a *Longitudinal Redundancy Check* (LRC)

The header identifies the message type and is followed by a byte containing a command code or register offset. The message body contains any data to be written or read from a register. The LRC byte is used to verify the packet data. The read/write/report message blocks contains a message length descriptor and between 1 and 32 bytes of data.

Figure 4-1 shows the general structure of messages.

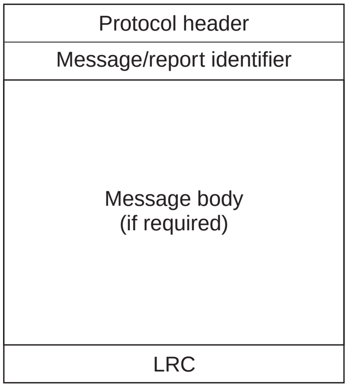


Figure 4-1 Keyboard controller message structure

4.4.2 Message types

The keyboard controller supports four message types:

- commands from the host
- commands from the keyboard controller
- HID reports to the host
- event reports to the host.

Table 4-18 provides a summary of the message types and their corresponding headers.

Table 4-18 Message type summary

Message type	Message header	Message length
Commands from the host		
Simple commands	0x80	3 bytes
Write register bit	0x81	4 bytes
Read register bit	0x82	4 bytes
Write register	0x83	4 bytes
Read register	0x84	4 bytes
Write block	0x85	5 to 36 bytes
Read block	0x86	5 to 36 bytes
Responses from the controller		
Simple commands	0x80	3 bytes
Report register bit or Event alert	0x81	4 bytes
Report register	0x83	4 bytes
Report block	0x85	5 to 36 bytes
Pointing device report	0x87	6 bytes
Keyboard report	0x88	3 bytes

Note

The first byte of message body for read block and report block message types indicates a the number data bytes contained in the message body. The body length can vary between 2 and 32 bytes.

4.4.3 Selecting register banks

The keyboard controller provides two register banks. Register bank 0 contains all of the control and status registers, and bank 1 contains scanned matrix and alternate layout register. If you use the standard keyboard layout, it is only bank 0 that is accessed. Figure 4-2 shows the organization of the registers.

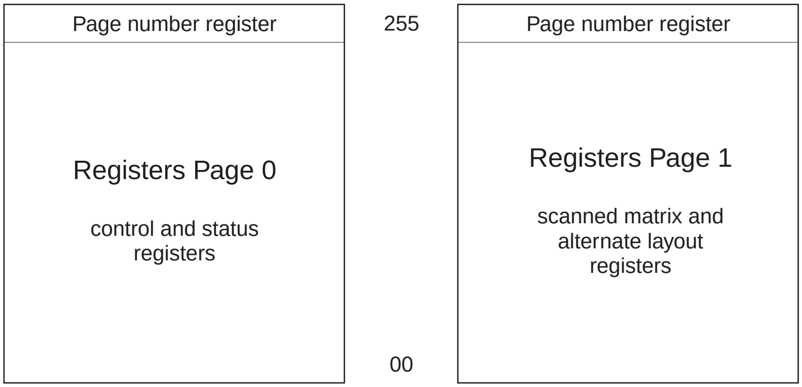


Figure 4-2 UR8HC007 register organization

Both register banks provide access to the page number register (at offset 255 in both banks). This is used to select the bank to be accessed. To access bank 1 registers, you must first write 1 to the page number register. To revert back to page 0, you must write 0 to the page number register. The default is bank 0. All other values written to this register are ignored.

Individual registers in the bank are selected by the register offset value contained in the message, as shown Figure 4-1 on page 4-20.

Note

For a full description of the keyboard controller registers, see the *Technical Reference Manual USAR Keyboard controller*.

4.4.4 Accessing the keyboard controller

To pass messages between the keyboard controller and SA-1100, you must:

1. Set the correct SPI chip-select value on the SA-1100 GPIO pins.
2. Enable the SPI chip-select decoder which is controlled by a GPIO pin on the system controller PLD.
3. Manipulate the SPI clock signal, as described below.

4.4.5 Keyboard controller host interface timing

A number of SA-1100 GPIO pins are used for data transfers and interface timing between the SA-1100 and keyboard controller.

Keyboard controller host writes

Figure 4-3 shows the timing for host writes to the keyboard controller.

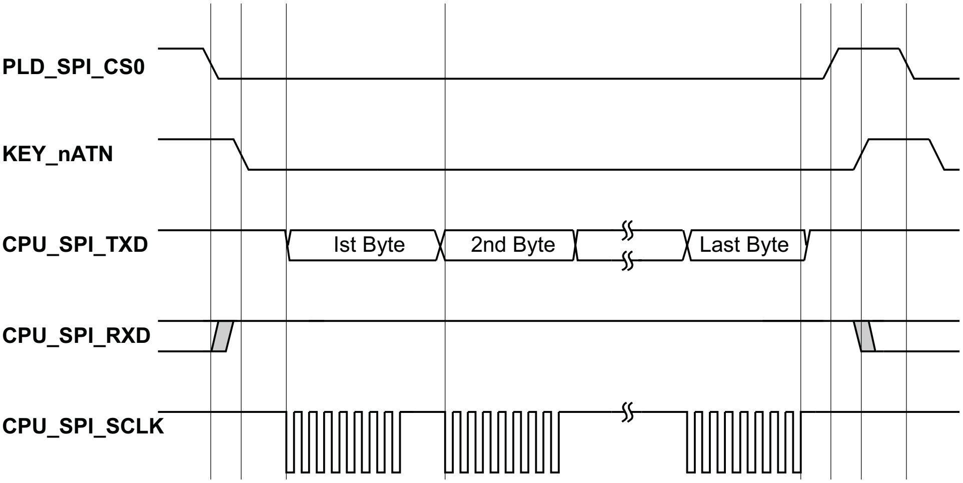


Figure 4-3 SPI master timing: host writes to the keyboard controller

Host writes to the keyboard controller are as follows:

- Host transfers to the keyboard controller start with the assertion of **PLD_SPI_nCS0**. The keyboard controller responds by asserting **KEY_nATN** and driving **CPU_SPI_RXD** HIGH.
- The host must wait a minimum of 0.1ms and a maximum of 5ms before beginning to transfer the first byte.

- The host places a sequence of eight data bits onto **CPU_SPI_TXD** and supplies a sequence of eight clock pulses on **CPU_SPI_SCLK**, one pulse for each bit. The minimum clock period is 4µs, with a minimum clock LOW period of 2µs. Each bit is read by the keyboard controller on the rising edge of **CPU_SPI_SCLK**.
- The next byte is transferred by holding **PLD_SPI_nCS0** LOW, placing another sequence of eight bits of the next byte on **CPU_SPI_TXD** and supplying another burst of clock pulses. At least 150µs must be allowed between the start of one byte transfer and the start of the next.
- After the last byte is transferred, the host de-asserts **PLD_SPI_nCS0** to indicate that the transfer is complete. In response, the keyboard controller de-asserts **KEY_nATN**. A new transfer can begin 120µs after the de-assertion of **KEY_nATN**.

Note

The signals are described in *SPI bus signal summary* on page 3-30.

Keyboard controller host reads and report transfers

Figure 4-4 shows the timing for data transfers from the keyboard controller to the host.

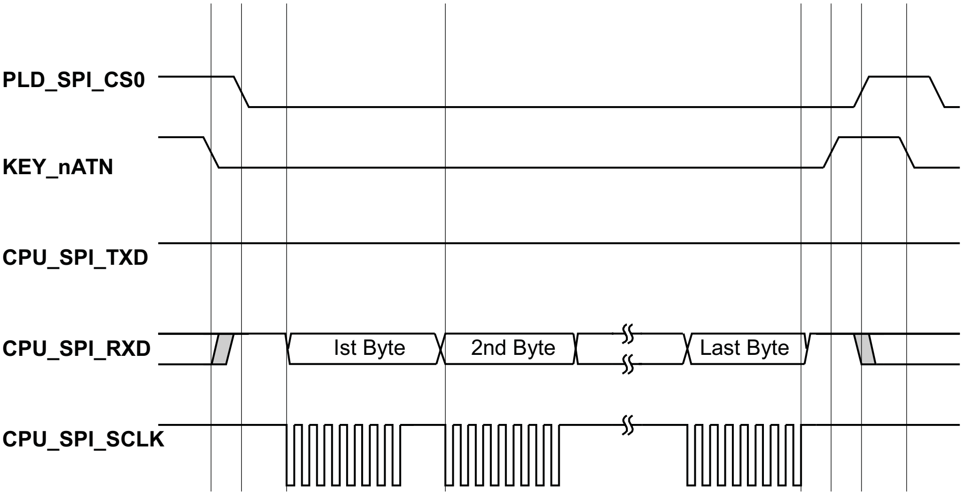


Figure 4-4 SPI slave timing: keyboard controller to host

Keyboard controller transfers to the host are as follows:

- The keyboard controller alerts the host by asserting **KEY_nATN** and driving **CPU_SPI_RXD** high.

- The host responds by asserting **PLD_SPI_nCS0** between 0.1ms and 5ms and supplying the first burst of clock pulses.
- The slave places a sequence of eight data bits onto **CPU_SPI_RXD** on the falling edge of **CPU_SPI_SCLK**.
- The next byte is transferred if **KEY_nATN** is held LOW, signaling that more data is available. At least 150µs must be allowed between the start of one byte transfer and the start of the next.
- After the last byte is transferred, the keyboard controller de-asserts **KEY_nATN** to indicate that the transfer is complete. In response, the host de-asserts **PLD_SPI_CS0** to release the SPI bus. A new transfer can begin 120µs after the de-assertion of **KEY_nATN**.

4.4.6 Keyboard controller interrupts

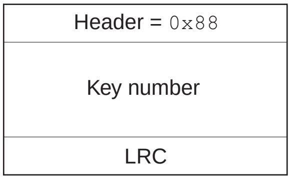
The keyboard controller asserts the signal **KEY_nATN** when any of the *Human Input Device* (HID) manager blocks are enabled and detect certain events. The **KEY_nATN** signal is used by the system controller PLD to assert the keyboard interrupt which is connected to the GIO0 pin of the SA-1100.

The interrupt service routine should read a report from the keyboard controller, as described in *Keyboard controller host reads and report transfers* on page 4-24.

The HID managers issue the following report types:

Keyboard data reports

These return changes on the internal or external keyboard matrix. Input may also be from an external keyboard device connected to J19. Figure 4-5 shows the format of a keyboard data report.



Header = `0x88`

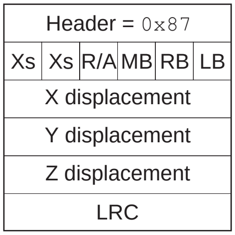
Key number

LRC

Figure 4-5 Keyboard data report

Pointing device report

These return absolute or relative positioning data from the mouse pointer. Figure 4-6 shows the format of a pointing device report.



Header = `0x87`

Xs | Xs | R/A | MB | RB | LB

X displacement

Y displacement

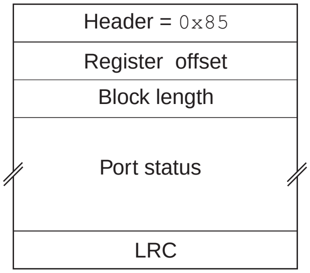
Z displacement

LRC

Figure 4-6 Pointing device report

Report block

These are generated by the direct report manager using a report block message. This can be used to relay data generated by an external PS/2 device. Figure 4-7 shows the format of a report block.



Header = `0x85`

Register offset

Block length

Port status

LRC

Figure 4-7 Report block

JP19 can used to connect an external mouse and keyboard.

4.5 Programming the UCB1200

The MCP port of SA-1100 is used to provide a communications link with the UCB1200. This provides a *Serial Interface Bus* (SIB) compatible with the UCB. Data is transferred using SIB data frames, which are typically 128 bits long. These are divided into two 64-bit subframes. The UCB reads and writes data using the lower subframe (bits 0 to 63) but ignores the upper subframe (bits 64 to 127).

4.5.1 SIB frame format

The format of subframes used to read and write telecom data, audio data and register contents are shown in Table 4-19.

Table 4-19 UCB SIB frame format summary

Bit	SIBDin definition	SIBDout definition
63:48	Control register write data [15:0] bit 0 = MSB	Control register read data [15:0] bit 0 = MSB
47:32	Telecom input path data [15:0] bit 0 = MSB 14 MSB bits are read	Telecom output path data [15:0] bit 0 = MSB bits [15:14] are 0
31	Telecom valid sample flag	Telecom valid flag
30	Audio valid sample flag	Audio valid flag
29:22	Reserved	Always 0s
21	Write bit (write=1)	Always 0
20:17	Control register address [3:0] bit 0 = MSB	Control register address [3:0] bit 0 = MSB = copy of the address present in SIBDin
16	Write bit (write=1)	Always 0
15:0	Audio input path data [15:0] bit 0 = MSB 12 MSB bits are read	Audio output path data [15:0] bit 0 = MSB bits [15:12] are 0

4.5.2 SIB frame timing

Figure 4-8 show the general timing characteristics for the SIB frame transfers between the SA-1100 and the UCB1200 (see the *UCB1200 Data Sheet* for more information).

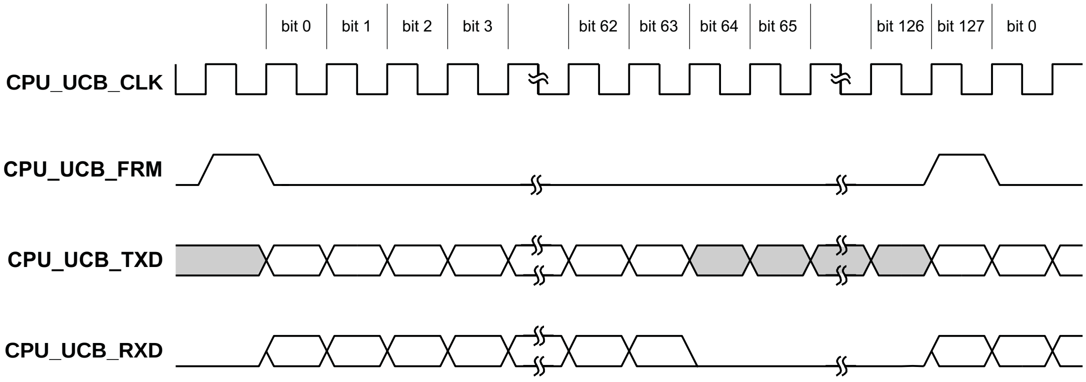


Figure 4-8 SIB frame timing

4.5.3 UCB1200 registers

The UCB1200 codec contains 16 register locations which are selected using bits 20:17 in the SIB subframe. The data for the addressed register is carried by bits 63:48. Table 4-20 provides a summary of the UCB1200 register. For details of these register, please refer to the *UCB1200 Data Sheet.*

Table 4-20 UCB1200 register summary

Address	Name
15	Null register
14	Reserved
13	Mode register
12	ID register
11	ADC data register
10	ADC control register
9	Touch screen control register
8	Audio control register B

Table 4-20 UCB1200 register summary

Address	Name
7	Audio control register A
6	Telecom control register B
5	Telecom control register A
4	Interrupt status/clear register
3	Falling edge interrupt enable register
2	Rising edge interrupt enable register
1	IO port direction register
0	IO port data register

4.5.4 Using the GPIO pins

The direction of the GPIO pins are controlled by the state of bits 9:0 in the IO port direction register. Setting a bit to 1 defines the corresponding pin as an output. Setting a bit to 0 defines the corresponding pin as an input.

Writing to the IO port data register controls the state of the pins that are defined as outputs. Reading the IO port data register returns the current states of the IO port pins.

The GPIO pins can be configured to generate interrupts on rising or falling edges.

A number of the GPIO pins have predefined functionality for the P1100. For example, GPIO8 is used as a DAA off hook signal (see *General purpose input/output pin block* on page 3-41).

4.5.5 UCB1200 interrupts

The UCB1200 interrupt request signal **UCB_IRQ** is routed to the interrupt controller block of system controller PLD and then to the interrupt on the GPIO1 pin of the SA-1100. The interrupt handler must read the PLD interrupt register to determine the source of any interrupt on this pin. Bit 1 in the PLD interrupt register is a 1 if the UCB1200 is the source of the pending interrupt.

The UCB1200 provides the following interrupt sources:

- 10 input/output pins

- audio and telecom chip detect
- ADC ready signal
- tspx_low
- tsmx_low

These sources are enabled, acknowledged and cleared by accessing the interrupt rising and falling edge registers and the interrupt status/clear register within the UCB1200.

4.6 MMC programming

The SPI bus is used to provide the interface between the SA-1100 and MMC interface (see *SPI bus* on page 3-29).

This section provides an overview of how the MMC SPI interface operates. However, because different card types have different timing requirements, you are advised to consult the documentation supplied by the MultiMediaCard manufacturer for more specific information.

4.6.1 MMC/SPI bus protocol

As well as flash memory or ROM (depending upon card type), MMCs incorporate an intelligent interface controller. Data transfers are handled using a message-based protocol. The SPI implementation uses a subset of the MMC protocol.

Three types of messages are used:

command these are always 6 bytes in length, as shown in Figure 4-9, and include a 7-bit CRC checksum. The checksum is ignored in SPI mode. Commands are transferred serially on the **CPU_SPI_TXD** signal.

response responses are replies by the card to a previously received command. These are transferred on the **CPU_SPI_RXD** signal.

data data is transferred on the **CPU_SPI_RXD** when read from the card or **CPU_SPI_TXD** when written to the card. Data transfers always use multiples of 8 bits on the SPI bus which are byte aligned to the chip select signal

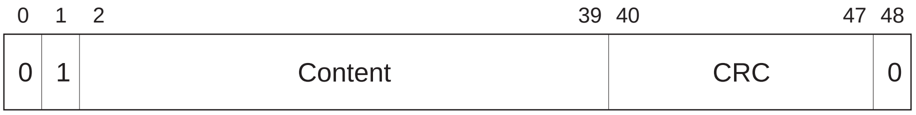


Figure 4-9 MMC command token

4.6.2 MMC registers

The SPI mode provides access to three registers, as shown in Table 4-21. Register are accessed using command messages.

Table 4-21 MMC register in SPI mode

Name	Size (bytes)	Description
CID	16	Card identification data
CSD	16	Card specific data
OCR		Operation condition register (optional)

4.6.3 MMC command summary

Table 4-22 provides a summary of the MMC commands available in SPI mode. The main difference is that commands related to multiblock transfers and software addressing are not available. Individual cards are addressed by asserting the appropriate chip-select signals.

Table 4-22 MMC command summary

SPI Index	Name	Function
CMD0	GO_IDLE_STATE	Reset card to idle state
CMD1	SEND_OP_COND	Request and confirm operating conditions
CMD9	SEND_CSD	Request card to send CSD data
CMD10	SEND_CID	Request card to send ID number
CMD13	SEND_STATUS	Request card to send status
CMD16	SET_BLOCKLEN	Set block length for block commands
CMD17	READ_SINGLE_BLOCK	Read a single block
CMD27	PROGRAM_CSD	Protect writable bits in CSD register
CMD28	SET_WRITE_PROT	Protection ON for addressed group
CMD29	CLR_WRITE_PROT	Protection OFF for dressed group

Table 4-22 MMC command summary (continued)

SPI Index	Name	Function
CMD30	SEND_WRITE_PROT	Request card write protection status
CMD32	TAG_SECTOR_START	First sector in erase list
CMD33	TAG_SECTOR_END	Last sector in erase list
CMD34	UNTAG_SECTOR	Remove sector from erase list
CMD35	TAG_ERASE_GROUP_START	First group in erase list
CMD36	TAG_ERASE_GROUP_END	Last group in erase list
CMD37	UNTAG_ERASE_GROUP	Remove group from erase list
CMD38	ERASE	Erase all previously selected sectors
CMD39	FAST_IO	Access application-specific registers
CMD59	CRC_ON_OFF	Enable/disable CRC

4.6.4 MMC transaction timing

The SPI bus timing used for transfers to and from MMC cards is similar to that used for the keyboard controller (see *Keyboard controller host interface timing* on page 4-23), although the way that data is assembled into valid messages is entirely different.

For more detailed timing requirements, refer to the documentation supplied by your card vendor.

4.7 Programming the touch screen controller

The SA-1100 and ADS7843 communicate using the SPI bus. The host sends control bytes to set operating parameters and to request X and Y co-ordinate data. The ADS7843 responds by returning coordinate data.

4.7.1 Touch screen controller control byte

The ADS7843 carries out acquisition on the requested channel as soon as it has received enough information contained in the control byte, reducing the delay before it returns the acquired data. Table 4-23 defines the bit assignment in a control byte from the SA-1100.

Table 4-23 Control byte

Bit	Name	Function
7	S	Start bit, always 1.
6:4	A2:0	Channel select bits: 001 = X 101 =Y 010 = No function on The P1100 110 = No function on The P1100
3	MODE	Selects 12-bit or 8-bit conversion: 0 = 12 bits 1 = 8 bits.
2	SER/DFR	Selects single-ended or differential mode: 0 = Differential 1 = Single-ended
1:0	PD1:0	Power down mode: 00 = Power down between transaction. No delays are required when the device returns to full operation and the first conversion is valid. PENIRQ is enabled 01 = Power down between conversions. PENIRQ is disabled. 10 = Reserved. 11 = Device is always powered.

4.7.2 Data exchange protocol

A transaction between the SA-1100 and ADS7843 consist of:

- writing a control byte to the ADS7843
- reading an 8 or 12-bit conversion word from the ADS7843.

Each complete transaction between the CPU and ADS7843 takes 16 clock cycles (**CPU_SPI_SCLK**). Figure 4-10 shows the timing for an 8-bit conversion.

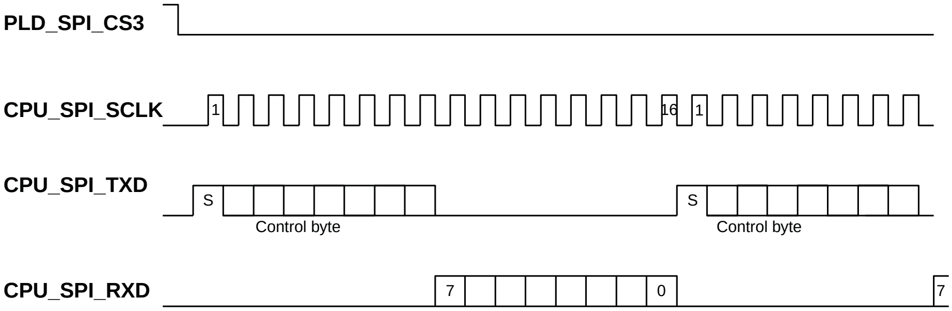


Figure 4-10 ADS7843 SPI bus timing

Note

- Before a transaction can begin, you must enable the interface to the ADS7843 and disable the PENIRQ interrupt by setting the TCH_EN bit in the PLD_TCH register HIGH (see *Touch screen register* on page 4-19).
- When the transaction is completed, you must disable the interface to the ADS7843 and enable the PENIRQ interrupt by setting the TCH_EN bit LOW.

The transaction between the CPU and ADS7843 is as follows:

- The CPU enables the interface by setting the TCH_EN bit in the PLD_TCH register HIGH.
- During the first eight clock cycles, the control byte is written to the ADS7843 on **CPU_SPI_TXD**, with the ADS7843 sampling on the rising edge of the clock signal
- As soon as it has enough data (after the first five bits), the ADS7843 begins to acquire a sample.
- The CPU begins to read the conversion data on the 9th clock, starting with the MSB.

- The CPU can begin writing the next control byte on the 15th clock cycle for 12-bit conversion, or on the 11th clock for 8-bit conversions.
- The CPU disbables the interface by setting the TCH_EN bit LOW.

Note

The SPI bus timing for the ADS7843 differs from that used for the keyboard controller in that for the ADS7843 **CPU_SPI_SCLK** idles LOW and is sampled on the rising edge.

Appendix A
Connector reference

This appendix contains connector pinouts. It contains the following sections:

Note

The abbreviation NC used in the tables in this appendix means *not connected*.

A.1 Summary of P1100 Connectors

Table A-1 shows a summary of the types and function of the connectors in the P1100. Connectors marked with * are not fitted as standard.

Table A-1 P1100 connector summary

ID	Type	Function	Schematic
JP1	DF13-10DP-125V	UCB1200 ADCs Keyboard controller ADCs	8*
JP2	DF13-10DP-125V	UCB1200 controlled LED Keyboad controller PWM	8*
JP3	DF13-4P-125H	UCB1200 touch screen interface	8
JP9	0.1" HDRx3	Keyboard controller lid switch sense	13
JP10	Hex rotary SW	Keyboard controller GPIO	13
JP11	DF13-5P-125V	Keyboard controller GPIO/interrupt	13*
JP14	MMC_conn	MMC	16
JP15	DF13-4P-125V	ADS7843 touch screen	16
JP16	Plated through hole	SPI_CS2 (for 2nd MMC)	16
JP17	0.1" HDRx2	RESET	16
JP18	DF13-8P-125V	JTAG	16
JP19	DF13-10DP-125V	Keyboard controller PS2 interfaces	16*
JP20	PCR-128FDNG1	Expansion port	20*
JP21	DF13-10DP-125V	Auxiliary SPI	17*
JP22	DF13-8P-125V	LCD backlight	18
JP23	DF9-31P-1V	LCD TFT 16bpp color	18
JP24	DF13-15P-125H	LCD STN monochrome	18
JP26	DF13-40DP-125V	FPGA IO support	19*
JP27	0.1" HDR1x3	FPGA/AC97 3V select	19*
JP28	0.1" HDR1x2	AC97 mono select	19*

Table A-1 P1100 connector summary (continued)

ID	Type	Function	Schematic
JP29	0.1" HDR2x15	AC97	19*
JP30	K31-E9P-N	Serial port COM1	20
JP31	K31-E9P-N	Serial port COM2	20
JP32	1.3mm	Switched DC-IN	25
JP33	0.1" HDRx2	3V battery	25
JP34	Mictor-38way	LA-1 address	31
JP35	Mictor-38way	LA-2 data	31
JP36	Mictor-38way	LA-3 control	31
JP37	DF13-10DP-125V	UCB1200 DAA	8*
JP38	DF13-15P-125H	Distribution board keyboard and loudspeaker	15
JP39	DF13-15P-125H	Distribution board keyboard	15
JP40	DF13-15P-125H	Distribution board LEDs, PS2 port and reset	15
JP42	DF13-4P-125H	Distribution board: microphone	15
JP46	StereoJack	External microphone	8
JP47	StereoJack	External headphones	8
JP48	0.1" HDRx2	Microphone power	8
JP49	StereoJack	AC97 audio	15*
JP50	0.1" HDR1x11	AC97 header	15*
JP51	StereoJack	AC97 audio	15*
JP53	StereoJack	AC97 audio	15*
JP56	RJ11	AC97 modem	15*

A.2 UCB1200 Touchscreen Interface – JP3

Table A-2 shows the pinout of JP3.

Table A-2 JP3 pinout

Pin	Function
1	**UCB_TSPY** (Positive Y)
2	**UCB_TSPX** (Positive X)
3	**UCB_TSMY** (Negative Y)
4	**UCB_TSMX** (Negative X)

A.3 MMC Connector – JP14

Table A-3 shows the pinout of JP14.

Table A-3 JP14 pinout

Pin	Function
1	**PLD_SPI_nCS1**
2	**BUF_SPI_TXD**
3	**GND**
4	**+3V**
5	**BUF_SPI_CLK**
6	**GND**
7	**BUF_SPI_RXD**

A.4 ADS7843 Touchscreen Interface – JP15

Table A-4 shows the pinout of JP15.

Table A-4 JP15 pinout

Pin	Function
1	Positive Y
2	Positive X
3	Negative Y
4	Negative X

A.5 JTAG Interface connector – JP18

Table A-5 shows the pinout of JP18.

Table A-5 JP18 pinout

Pin	Function
1	**+3V**
2	**JTAG_nRESET**
3	**BUF_PLD_TDO**
4	**JTAG_TMS**
5	**JTAG_TDI**
6	**JTACK_TCK**
7	**SYSTEM_nRESET**
8	**GND**

A.6 Auxiliary PS2 Interface – JP19

Table A-6 shows the pinout of JP19.

Table A-6 JP19 pinout

Pin	Function
1	**PS2_0DAT**
2	**PS2_EN**
3	**PS2_0CLK**
4	**PS2_IDAT** (Internal PS2 device)
5	**PS2_1DAT**
6	**PS2_ICLK** (Internal PS2 device)
7	**PS2_1CLK**
8	**+3V**
9	**GND**
10	**GND**

A.7 Auxiliary SPI Interface Connector – JP21

Table A-7 shows the pinout of JP21.

Table A-7 JP21 pinout

Pin	Function	Pin	Function
1	**BUF_SPI_RXD**	6	**+3V**
2	**PLD_SPI_nCS7**	7	**PLD_SPI_nCS5**
3	**PLD_SPI_TXD**	8	**GND**
4	**GND**	9	**PLD_SPI_nCS6**
5	**BUF_SPI_CLK**	10	**PLD_SPI_nCS4**

A.8 TFT color connector – JP23

Table A-8 shows the pinout of JP23.

Table A-8 JP23 pinout

Pin	Function	Pin	Function
1	**GND**	17	**LCD_DD_9** (Green 4)
2	**LCD_PCLK** (Pixel Clock)	18	**LCD_DD_10** (Green 5)
3	**LCD_LCLK** (Line Clock Hsync)	19	**GND**
4	**LCD_FCLK** (Frame Clock Vsync)	20	Tied LOW (Blue LSB)
5	**GND**	21	**LCD_DD_0** (Blue 1)
6	Tied LOW (Red LSB)	22	**LCD_DD_1** (Blue 2)
7	**LCD_DD_11** (Red 1)	23	**LCD_DD_2** (Blue 3)
8	**LCD_DD_12** (Red 2)	24	**LCD_DD_3** (Blue 4)
9	**LCD_DD_13** (Red 3)	25	**LCD_DD_4** (Blue 5)
10	**LCD_DD_14** (Red 4)	26	**GND**
11	**LCD_DD_15** (Red 5)	27	**LCD_BIAS** (ENAB)
12	**GND**	28	**+3V3**
13	**LCD_DD_5** (Green 0)	29	**+3V3**
14	**LCD_DD_6** (Green 1)	30	Tied HIGH (Horizontal Display Mode)
15	**LCD_DD_7** (Green 2)	31	Tied HIGH (Vertical Display Mode)
16	**LCD_DD_8** (Green 3)		

A.9 Serial port connectors

This section provides pinouts for the serial port connectors.

A.9.1 COM1 – JP30

Table A-9 shows the pinout of JP30.

Table A-9 JP30 pinout

Pin	Function
1	NC
2	**UART1_RXD**
3	**CPU_UART1_TXD**
4	NC
5	**GND**
6	NC
7	**CPU_UART1_RTS**
8	**UART1_CTS**
9	NC

A.9.2 COM2 – JP31

Table A-10 shows the pinout of JP31.

Table A-10 JP31 pinout

Pin	Function
1	NC
2	**UART2_RXD**
3	**CPU_UART2_TXD**
4	NC
5	**GND**
6	NC

Table A-10 JP31 pinout (continued)

Pin	Function
7	**CPU_UART2_RTS**
8	**UART2_CTS**
9	NC

A.10 UCB1200 DAA (modem) interface – JP37

Table A-11 shows the pinout of JP37.

Table A-11 JP37 pinout

Pin	Function
1	**AGND**
2	**UCB_TOUTP** (Telephone Out Positive)
3	**+5V**
4	**UCB_TINP** (Telephone In Positive)
5	**UCB_DAA_OH** (Off Hook)
6	**UCB_DAA_MUTE** (Mute)
7	NC
8	**UCB_DAA_RI** (Ring Indicate)
9	**GND**
10	**+3V**

A.11 Distribution panel connectors

This section shows the pinouts of the connectors associated with the keybaord controller.

A.11.1 Keyboard controller connector 1 – JP38

Table A-12 shows the pinout of JP38. The connector provides the keyboard row signals and speaker connections.

Table A-12 JP38 pinout

Pin	Function
8	**KBD_ROW6**
1	**UCB_SPKR_P**
2	**UCB_SPKR_N**
3	**GND**
4	**KBD_COL15**
5	**KBD_COL14**
6	**KBD_COL13**
7	**KBD_ROW7**
9	**KBD_ROW5**
10	**KBD_ROW4**
11	**KBD_ROW3**
12	**KBD_ROW2**
13	**KBD_ROW1**
14	**KBD_ROW0**
15	**GND**

A.11.2 Keyboard controller connector 2 – JP39

Table A-13 shows the pinout of JP39. The connector provides the keyboard column signals.

Table A-13 JP39 pinout

Pin	Function
1	**GND**
2	**KBD_COL12**
3	**KBD_COL11**
4	**KBD_COL10**
5	**KBD_COL9**
6	**KBD_COL8**
7	**KBD_COL7**
8	**KBD_COL6**
9	**KBD_COL5**
10	**KBD_COL4**
11	**KBD_COL3**
12	**KBD_COL2**
13	**KBD_COL1**
14	**KBD_COL0**
15	**+3V**

A.11.3 Keyboard controller connector 3 – JP40

Table A-14 shows the pinout of JP40. The connector provides signals for the LEDs, mouse pointer, and reset button in the keyboard housing.

Table A-14 JP40 pinout

Pin	Function
1	**+3V**
2	**SYSTEM_nRESET**
3	**PLD_nRST_OUT**
4	**PS2_EN**
5	**+3V**
6	**PS2_ICLK**
7	**PS2_IDAT**
8	**GND**
9	**CPU_GPIO25**
10	**GND**
11	**KBD_LED3**
12	**KBD_LED2**
13	**KBD_LED1**
14	**KBD_LED0**
15	**GND**

A.11.4 Microphone connector – JP42

Table A-15 shows the pinout of JP42.

Table A-15 JP42 pinout

Pin	Function
1	**UCB_MIC_GND**
2	**UCB_MIC**
3	**GND**
4	**+5V0**

Appendix B
System Expansion Options

This appendix describes the P1100 expansion options. It contains the following sections:

B.1 Power considerations

The power supplies on the P1100 CPU board are able to supply a limited additional loading, as shown in Table B-1.

Table B-1 Power supply support for expansion

Rail	Total	Used	Available
3V	1A	500mA Max	500mA
3V3	350mA	100mA	250mA
5V	300mA	100mA	200mA
9V	3600mW	-	-
1V5	200mA	100mA	100mA

These figures can vary with the number of items within the P1100 system that are active and should be used for guidance only.

B.2 System expansion

The system expansion connector JP20 can be fitted to allow a second board to be attached to the underside of the P1100 CPU board. It provides an interface that is compatible with the Intel SA-1101 Evaluation Board, with access to:

- address and data buses
- PCMCIA control signals
- GPIO
- buffered address and data buses
- system control signals and DRAM strobes.

Note

This connector is not fitted as standard.

B.3 SPI expansion

The SPI bus is expandable by connecting devices to JP21. In addition to data transmit, receive, and clock signals, the P1100 CPU board provides three spare chip-select signals.

Table B-2 lists the SPI chip select signals.

Table B-2 SPI chip select assignment

CPU_SPI_CS[2:0]	Chip select	Device
000	**PLD_SPI_nCS0**	Keyboard controller
001	**PLD_SPI_nCS1**	MMC Socket 1
010	**PLD_SPI_nCS2**	MMC Socket 2
011	**PLD_SPI_nCS3**	Touch screen controller
100	**PLD_SPI_nCS4**	System expansion
101	**PLD_SPI_nCS5**	System expansion
110	**PLD_SPI_nCS6**	System expansion
111	**PLD_SPI_nCS7**	Null chip select. Do not use for system expansion.

Note

The chip select signal **PLD_SPI_nCS7** is reserved as a *null* select and should not be used for expansion (see *Using the SPI chip selects* on page 3-30).

B.4 Flash memory expansion

The buffered address and data busses along with PLD control outputs are routed to the expansion connector JP20. This allows connection of additional ROM/Flash or provision for an alternate boot path.

B.5 UCB1200 expansion

The signals from the functional blocks are routed to various connectors to provide:

- a *Data Access Arrangement* (DAA) interface
- four ADC inputs
- a 10-bit GPIO port.

The DAA interface enables you to add a DAA to the P1100 to allow connection to telephone networks. The interface uses GPIO pins to provide Mute, Ring Indicate (RI) and Off Hook (OH) signals. Connection is made at a single dual row 10-pin DF13 type connector, JP37.

Note

The connector JP37 is not fitted as standard.

B.6 FPGA and AC97 CODEC Interface

Note

The FPGA and AC97 interface connector (JP29) are not used or fitted in the standard product variants.

Circuitry is provided to support a Xilinx XC4036XLA FPGA. The circuitry includes unbuffered upper address lines and DRAM control lines, **MBREQ**, **MBGNT**, a full 32-bit buffered data bus, and lower order buffered address lines.

This allows the FPGA to act as a memory mapped peripheral and to implement a DRAM memory controller for a multi-master environment.

Provision has also been made to allow prototyping of an AC '97 codec. The codec interface is a serial bus that allows connection of modems, or a high-quality audio codec of up to 20-bits resolution and with up to 6 audio channels for surround sound.

The configuration data for the FPGA is held in an 8-bit FLASH ROM (AT49LV040). The FPGA automatically loads the data at power up, using its master parallel configuration mode. An LED can be driven from the FPGA to show that it has finished configuration. The configuration PROM is placed in a socket allowing its removal for programming.

Most of the unused IO pins of the FPGA are connected to a 40-way expansion connector, JP26, for use by alternate functions.

The AC '97 codec interface logic essentially consists of an interface to the SA-1100 memory bus, 4 transmit FIFOs, 4 receive FIFOs, serializing, plus control and muxing logic. The interface to the codec is a serial bus running at a bit clock rate of 12.288 MHz and a fixed frame rate of 48 KHz. Each frame is divided into 13 time slots that carry data for different codec channels. For full details of the interface please refer to the AC'97 interface specification. An oscillator module is needed on the CPU board to supply the required 12.288 MHz clock.

The AC '97 interface connects to the MDC header connector, JP29, to allow standard MDC daughter cards to be plugged onto the CPU board. The two smallest sizes of MDC card will fit onto the CPU board, however, there is not enough space to fit the largest variant. Three stereo 3.5mm jack sockets and an RJ11 telephone socket are positioned on the CPU board and connected to a header to allow a hard wired connection to the miniature input/output connector found on MDC cards.

B.7 Connector support for expansion

Figure B-1 shows the modem and audio expansion connectors on the rear of the compact module enclosure (with the blanking plate removed). The signal traces to these connectors appear on JP50 and enable you to connect, for example, an audio codec and modem that you have added to the system.

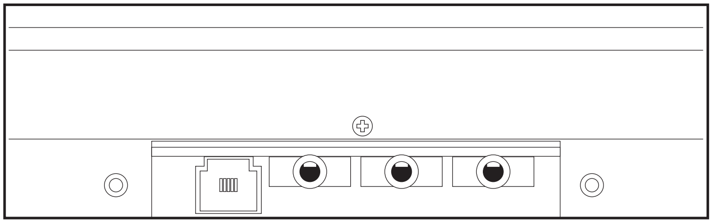


Figure B-1 Modem and audio expansion connectors

Note

These connectors are not fitted on the standard product variant.

B.8 PLD enhanced clocking

The system controller PLD code currently used in the P1100 design assumes that the PLD is not an enhanced clocking device. This means that the code must steal pins to use as clocks for the internal registers. If an anhanced clocking device is used then these pins can be used. The pins affected by this are:

- pin 92 – **CDC_IRQ0**
- pin 83 – **CPU_RCLK_OUT**
- pin 37 – **CPU_32KHz**.

B.9 GPIO support for expansion

A number of the GPIO pins on the SA-1100 can be used to support additional functions. These are:

Table B-3 GPIO pins used to support system expansion

Port	Signal	Function
GPIO[24:23]	**CPU_GPIO[24:23]**	These two pins can be individually configured as inputs or outputs. They are connected to JP26. CPU_GPIO23 is also connected to the system expansion connector JP20.
GPIO25	**CPU_GPIO25**	This pin is connected to a LED, JP40 and JP26. However, when a distribution panel is connected (to JP40), this signal is a current sink for two LEDs. The SA-1100 can provide a 1Hz clock on this pin as an alternate function using a buffered, trimmed 1 Hz signal from the RTC in the SA-1100.
GPIO26	**CPU_RCLK_OUT**	This signal is routed to the XCR3128 PLD and JP20 expansion connector. The SA-1100 can provide a buffered divide-by-two version of the core clock on this pin as an alternate function. The clock is currently unused on the SA-1100 and, to save power, need not be configured by the software. It could be used by the PLD for high speed state machines or for latching transient data, but is not used in the first release of the PLD.
GPIO27	**CPU_32KHz**	This signal is routed to the XCR3128 PLD and JP20 expansion connector. The SA-1100 can provide a buffered untrimmed version of the 32.768kHz crystal on this pin as an alternate function. The clock is currently unused on the SA-1100 and, to save power, need not be configured by the software. It can be used by the PLD for slow speed state machines, but is not used in the first release of the PLD.
GPIO22	**CPU_MEMREQ**	These signals are routed to the FPGA and to its associated expansion connector, J26.
GPIO21	**CPU_MEMGNT**	
GPIO1	**CPU_GPIO1**	This signal is routed to the expansion connector, J26. It can be used to generate interrupts to the SA-1100 (see *Interrupt control* on page 3-12).

Index

The items in this index are listed in alphabetic order, with symbols and numerics appearing at the end. The references given are to page numbers.