XS1-L8A-128-QF124 Datasheet

2013/02/26 Document Number: X5358,

Table of Contents

1 xCORE Multicore Microcontrollers

The XS1-L Series is a comprehensive range of 32-bit multicore microcontrollers that brings the low latency and timing determinism of the xCORE architecture to mainstream embedded applications. Unlike conventional microcontrollers, xCORE multicore microcontrollers execute multiple real-time tasks simultaneously. Devices consist of one or more xCORE tiles, each containing between four and eight independent xCORE logical processors. Each logical core can execute computational code, advanced DSP code, control software (including logic decisions and executing a state machine) or software that handles I/O.

Because xCORE multicore microcontrollers are completely deterministic, you can write software to implement functions that traditionally require dedicated hardware. You can simulate your program like hardware, and perform static timing analysis using the xTIMEcomposer development tools.

The devices include scheduling hardware that performs functions similar to those of an RTOS; and hardware that connects the cores directly to I/O ports, ensuring not only fast processing but extremely low latency. The use of interrupts is eliminated, ensuring deterministic operation.

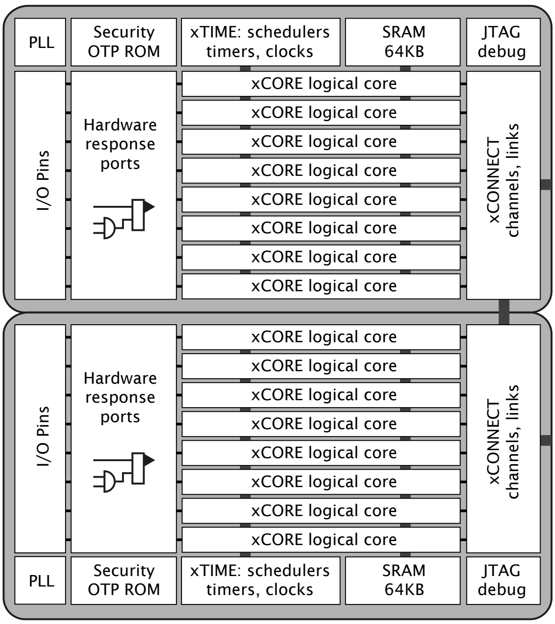


Figure 1: XS1-L Series: 4-16 core devices

XS1-L devices are available in a range of resource densities, package, performance and temperature grades depending on your needs. XS1-L devices range from 4-16 logical cores divided between one or two xCORE tiles, providing 400-1000 MIPS, up to 84 GPIO, and 64Kbytes or 128Kbytes of SRAM.

1.1 xSOFTip

xCORE devices are backed with tested and proven IP blocks from the xSOFTip library, which allow you to quickly add interface and processor functionality such as Ethernet, PWM, graphics driver, and audio EQ to your xCORE device.

xSOFTip blocks are written in high level languages and use xCORE resources to implement given function. This means xSOFTip is software and brings the associated benefits of easy maintenance and fast compilation time, while being accessible to anyone with embedded C skills.

The graphical xSOFTip Explorer tool lets you browse available xSOFTip blocks from our library, understand the resource usage, configure the blocks to your specification, and estimates the right device for your design. It is included in xTIMEcomposer Studio or available as a standalone tool from xmos.com/downloads.

1.2 xTIMEcomposer Studio

Designing with XS1-Ldevices is simple thanks to the xTIMEcomposer Studio development environment, which includes a highly efficient compiler, debugger and device programming tools. Because xCORE devices operate deterministically, they can be simulated like hardware within the development tools: uniquely in the embedded world, xTIMEcomposer Studio therefore includes a static timing analyzer, cycle-accurate simulator, and high-speed in-circuit instrumentation.

xTIMEcomposer can also be used to load the executable file onto the device and debug it over JTAG, programmed it into flash memory on the board, or write it to OTP memory on the device. The tools can also encrypt the flash image and write the decrpytion key securely to OTP memory.

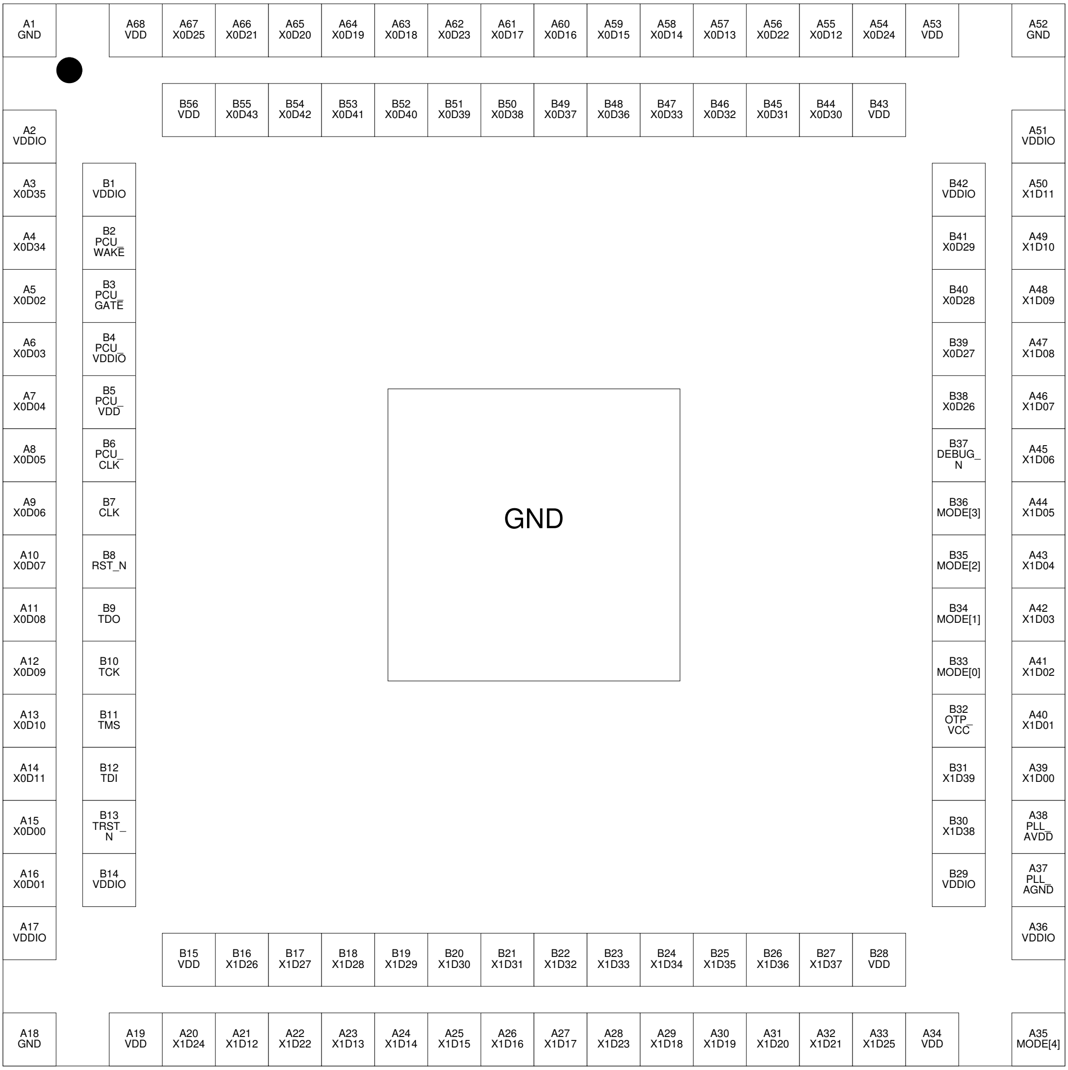
xTIMEcomposer can be driven from either a graphical development environment that will be familiar to any C programmer, or the command line. They are supported on Windows, Linux and MacOS X and available at no cost from xmos.com/downloads.

Information on using the tools is provided in a separate user guide, X1013.

2 XS1-L8A-128-QF124 Features

- **Eight-Core Multicore Microcontroller with Advanced Multi-Core RISC Architecture**
  - Up to 500 MIPS shared between up to 8 real-time logical cores across two tiles
  - Each logical core has:
    - Guaranteed throughput of 1/4 of tile MIPS
    - 16x32bit dedicated registers
  - 159 high-density 16/32-bit instructions
    - All have single clock-cycle execution (except for divide)
    - 32x32→64-bit MAC instructions for DSP, arithmetic and user-definable cryptographic functions
- **Programmable I/O**
  - 28 general-purpose I/O pins, configurable as input or output
  - Port sampling rates of up to 60 MHz with respect to an external clock
  - 64 channel ends for communication with other cores, on or off-chip
- **Memory**
  - 128KB internal single-cycle SRAM (max 64KB per tile) for code and data storage
  - 8KB internal OTP (max 8KB per tile) for application boot code
- **JTAG Module for On-Chip Debug**
- **Security Features**
  - Programming lock disables debug and prevents read-back of memory contents
  - AES bootloader ensures secrecy of IP held on external flash memory
- **Ambient Temperature Range**
  - Commercial qualification: 0 °C to 70 °C
  - Industrial qualification: -40 °C to 85 °C
- **Speed Grade**
  - 10: 1000 MIPS
  - 8: 800 MIPS
- **Power Consumption**
  - Active Mode
    - 400 mA at 500 MHz (typical)
    - 320 mA at 400 MHz (typical)
  - Standby Mode
    - 28 mA
  - Sleep Mode
    - Programmable PCU module puts device into sleep mode
    - Wakeup on external signal or timeout
- **124-pin QF124 package 0.5 mm pitch**

3 Pin Configuration



4 Signal Description

Module	Signal	Function	Type	Active	Properties
	PU=Pull Up, PD=Pull Down, ST=Schmitt Trigger Input, OT=Output Tristate, S=Switchable R_S=Required for SPI boot (§5.6), R_U=Required for USB-enabled devices (§E)				
Power	GND	Digital ground	GND	—	
	OTP_VCC	OTP power supply	PWR	—	
	PLL_AGND	Analog ground for PLL	GND	—	
	PLL_AVDD	Analog PLL power	PWR	—	
	RST_N	Global reset input	Input	Low	PU, ST
	VDD	Digital tile power	PWR	—	
	VDDIO	Digital I/O power	PWR	—	
Clocks	CLK	PLL reference clock	Input	—	PD, ST
	MODE[4:0]	Boot mode select	Input	—	PU, ST
JTAG	DEBUG_N	Multi-chip debug	I/O	Low	PU
	TCK	Test clock	Input	—	PU, ST
	TDI	Test data input	Input	—	PU, ST
	TDO	Test data output	Output	—	PD, OT
	TMS	Test mode select	Input	—	PU, ST
	TRST_N	Test reset input	Input	Low	PU, ST
I/O	X0D00	$P1A^0$	I/O	—	PD_S, R_S
	X0D01	XLA^{4o}_{5b} $P1B^0$	I/O	—	PD_S, R_S
	X0D02	XLA^{3o}_{5b} $P4A^0$ $P8A^0$ $P16A^0$ $P32A^{20}$	I/O	—	PD_S, R_U
	X0D03	XLA^{2o}_{5b} $P4A^1$ $P8A^1$ $P16A^1$ $P32A^{21}$	I/O	—	PD_S, R_U
	X0D04	$XLA^{1o}_{2b/5b}$ $P4B^0$ $P8A^2$ $P16A^2$ $P32A^{22}$	I/O	—	PD_S, R_U
	X0D05	$XLA^{0o}_{2b/5b}$ $P4B^1$ $P8A^3$ $P16A^3$ $P32A^{23}$	I/O	—	PD_S, R_U
	X0D06	$XLA^{0i}_{2b/5b}$ $P4B^2$ $P8A^4$ $P16A^4$ $P32A^{24}$	I/O	—	PD_S, R_U
	X0D07	$XLA^{1i}_{2b/5b}$ $P4B^3$ $P8A^5$ $P16A^5$ $P32A^{25}$	I/O	—	PD_S, R_U
	X0D08	XLA^{2i}_{5b} $P4A^2$ $P8A^6$ $P16A^6$ $P32A^{26}$	I/O	—	PD_S, R_U
	X0D09	XLA^{3i}_{5b} $P4A^3$ $P8A^7$ $P16A^7$ $P32A^{27}$	I/O	—	PD_S, R_U
	X0D10	XLA^{4i}_{5b} $P1C^0$	I/O	—	PD_S, R_S
	X0D11	$P1D^0$	I/O	—	PD_S, R_S
	X0D12	$P1E^0$	I/O	—	PD_S, R_U
	X0D13	XLB^{4o}_{5b} $P1F^0$	I/O	—	PD_S, R_U
	X0D14	XLB^{3o}_{5b} $P4C^0$ $P8B^0$ $P16A^8$ $P32A^{28}$	I/O	—	PD_S, R_U
	X0D15	XLB^{2o}_{5b} $P4C^1$ $P8B^1$ $P16A^9$ $P32A^{29}$	I/O	—	PD_S, R_U
	X0D16	$XLB^{1o}_{2b/5b}$ $P4D^0$ $P8B^2$ $P16A^{10}$	I/O	—	PD_S, R_U
	X0D17	$XLB^{0o}_{2b/5b}$ $P4D^1$ $P8B^3$ $P16A^{11}$	I/O	—	PD_S, R_U
	X0D18	$XLB^{0i}_{2b/5b}$ $P4D^2$ $P8B^4$ $P16A^{12}$	I/O	—	PD_S, R_U
	X0D19	$XLB^{1i}_{2b/5b}$ $P4D^3$ $P8B^5$ $P16A^{13}$	I/O	—	PD_S, R_U
	X0D20	XLB^{2i}_{5b} $P4C^2$ $P8B^6$ $P16A^{14}$ $P32A^{30}$	I/O	—	PD_S, R_U
	X0D21	XLB^{3i}_{5b} $P4C^3$ $P8B^7$ $P16A^{15}$ $P32A^{31}$	I/O	—	PD_S, R_U
	X0D22	XLB^{4i}_{5b} $P1G^0$	I/O	—	PD_S, R_U

(continued)

Module	Name	Function	Type	Active	Properties
	X0D23	$P1H^0$	I/O	—	PD_S, R_U
	X0D24	$P1I^0$	I/O	—	PD_S
	X0D25	$P1J^0$	I/O	—	PD_S
	X0D26	$P4E^0$ $P8C^0$ $P16B^0$	I/O	—	PD_S, R_U
	X0D27	$P4E^1$ $P8C^1$ $P16B^1$	I/O	—	PD_S, R_U
	X0D28	$P4F^0$ $P8C^2$ $P16B^2$	I/O	—	PD_S, R_U
	X0D29	$P4F^1$ $P8C^3$ $P16B^3$	I/O	—	PD_S, R_U
	X0D30	$P4F^2$ $P8C^4$ $P16B^4$	I/O	—	PD_S, R_U
	X0D31	$P4F^3$ $P8C^5$ $P16B^5$	I/O	—	PD_S, R_U
	X0D32	$P4E^2$ $P8C^6$ $P16B^6$	I/O	—	PD_S, R_U
	X0D33	$P4E^3$ $P8C^7$ $P16B^7$	I/O	—	PD_S, R_U
	X0D34	$P1K^0$	I/O	—	PD_S
	X0D35	$P1L^0$	I/O	—	PD_S
	X0D36	$P1M^0$ $P8D^0$ $P16B^8$	I/O	—	PD_S
	X0D37	$P1N^0$ $P8D^1$ $P16B^9$	I/O	—	PD_S, R_U
	X0D38	$P1O^0$ $P8D^2$ $P16B^{10}$	I/O	—	PD_S, R_U
	X0D39	$P1P^0$ $P8D^3$ $P16B^{11}$	I/O	—	PD_S, R_U
	X0D40	$P8D^4$ $P16B^{12}$	I/O	—	PD_S, R_U
	X0D41	$P8D^5$ $P16B^{13}$	I/O	—	PD_S, R_U
	X0D42	$P8D^6$ $P16B^{14}$	I/O	—	PD_S, R_U
	X0D43	$P8D^7$ $P16B^{15}$	I/O	—	PU_S, R_U
I/O	X1D00	$P1A^0$	I/O	—	PD_S, R_S
	X1D01	XLA^{4o}_{5b} $P1B^0$	I/O	—	PD_S, R_S
	X1D02	XLA^{3o}_{5b} $P4A^0$ $P8A^0$ $P16A^0$ $P32A^{20}$	I/O	—	PD_S, R_U
	X1D03	XLA^{2o}_{5b} $P4A^1$ $P8A^1$ $P16A^1$ $P32A^{21}$	I/O	—	PD_S, R_U
	X1D04	$XLA^{1o}_{2b/5b}$ $P4B^0$ $P8A^2$ $P16A^2$ $P32A^{22}$	I/O	—	PD_S, R_U
	X1D05	$XLA^{0o}_{2b/5b}$ $P4B^1$ $P8A^3$ $P16A^3$ $P32A^{23}$	I/O	—	PD_S, R_U
	X1D06	$XLA^{0i}_{2b/5b}$ $P4B^2$ $P8A^4$ $P16A^4$ $P32A^{24}$	I/O	—	PD_S, R_U
	X1D07	$XLA^{1i}_{2b/5b}$ $P4B^3$ $P8A^5$ $P16A^5$ $P32A^{25}$	I/O	—	PD_S, R_U
	X1D08	XLA^{2i}_{5b} $P4A^2$ $P8A^6$ $P16A^6$ $P32A^{26}$	I/O	—	PD_S, R_U
	X1D09	XLA^{3i}_{5b} $P4A^3$ $P8A^7$ $P16A^7$ $P32A^{27}$	I/O	—	PD_S, R_U
	X1D10	XLA^{4i}_{5b} $P1C^0$	I/O	—	PD_S, R_S
	X1D11	$P1D^0$	I/O	—	PD_S, R_S
	X1D12	$P1E^0$	I/O	—	PD_S, R_U
	X1D13	XLB^{4o}_{5b} $P1F^0$	I/O	—	PD_S, R_U
	X1D14	XLB^{3o}_{5b} $P4C^0$ $P8B^0$ $P16A^8$ $P32A^{28}$	I/O	—	PD_S, R_U
	X1D15	XLB^{2o}_{5b} $P4C^1$ $P8B^1$ $P16A^9$ $P32A^{29}$	I/O	—	PD_S, R_U
	X1D16	$XLB^{1o}_{2b/5b}$ $P4D^0$ $P8B^2$ $P16A^{10}$	I/O	—	PD_S, R_U
	X1D17	$XLB^{0o}_{2b/5b}$ $P4D^1$ $P8B^3$ $P16A^{11}$	I/O	—	PD_S, R_U
	X1D18	$XLB^{0i}_{2b/5b}$ $P4D^2$ $P8B^4$ $P16A^{12}$	I/O	—	PD_S, R_U
	X1D19	$XLB^{1i}_{2b/5b}$ $P4D^3$ $P8B^5$ $P16A^{13}$	I/O	—	PD_S, R_U
	X1D20	XLB^{2i}_{5b} $P4C^2$ $P8B^6$ $P16A^{14}$ $P32A^{30}$	I/O	—	PD_S, R_U
	X1D21	XLB^{3i}_{5b} $P4C^3$ $P8B^7$ $P16A^{15}$ $P32A^{31}$	I/O	—	PD_S, R_U

(continued)

Module	Name	Function	Type	Active	Properties
I/O	X1D22	XLB^{4i}_{5b} $P1G^0$	I/O	—	PD_S, R_U
	X1D23	$P1H^0$	I/O	—	PD_S, R_U
	X1D24	$P1I^0$	I/O	—	PD_S
	X1D25	$P1J^0$	I/O	—	PD_S
	X1D26	$P4E^0$ $P8C^0$ $P16B^0$	I/O	—	PD_S, R_U
	X1D27	$P4E^1$ $P8C^1$ $P16B^1$	I/O	—	PD_S, R_U
	X1D28	$P4F^0$ $P8C^2$ $P16B^2$	I/O	—	PD_S, R_U
	X1D29	$P4F^1$ $P8C^3$ $P16B^3$	I/O	—	PD_S, R_U
	X1D30	$P4F^2$ $P8C^4$ $P16B^4$	I/O	—	PD_S, R_U
	X1D31	$P4F^3$ $P8C^5$ $P16B^5$	I/O	—	PD_S, R_U
	X1D32	$P4E^2$ $P8C^6$ $P16B^6$	I/O	—	PD_S, R_U
	X1D33	$P4E^3$ $P8C^7$ $P16B^7$	I/O	—	PD_S, R_U
	X1D34	$P1K^0$	I/O	—	PD_S
	X1D35	$P1L^0$	I/O	—	PD_S
	X1D36	$P1M^0$ $P8D^0$ $P16B^8$	I/O	—	PD_S
	X1D37	$P1N^0$ $P8D^1$ $P16B^9$	I/O	—	PD_S, R_U
	X1D38	$P1O^0$ $P8D^2$ $P16B^{10}$	I/O	—	PD_S, R_U
	X1D39	$P1P^0$ $P8D^3$ $P16B^{11}$	I/O	—	PD_S, R_U
	PCU_CLK	Clock input		—	
	PCU_GATE	Power control gate control		—	
	PCU_VDD	PCU tile power		—	
	PCU_VDDIO	PCU I/O supply		—	
	PCU_WAKE	Wakeup reset		—	

5 Product Overview

The XMOS XS1-L8A-128-QF124 is a powerful device that provides a simple design process and highly-flexible solution to many applications. The device consists of two xCORE Tiles, each comprising a flexible multicore microcontroller with tightly integrated I/O and on-chip memory. The processors run mutiple tasks simultaneously using logical cores, each of which is guaranteed a slice of processing power and can execute computational code, control software and I/O interfaces. Logical cores use channels to exchange data within a tile or across tiles. The tiles are connected via an integrated switch network, which uses a proprietary physical layer protocol, and which can also be used to add additional resources to a design. The I/O pins are driven using intelligent ports that can serialize data, interpret strobe signals and wait for scheduled times or events, making the device ideal for real-time control applications.

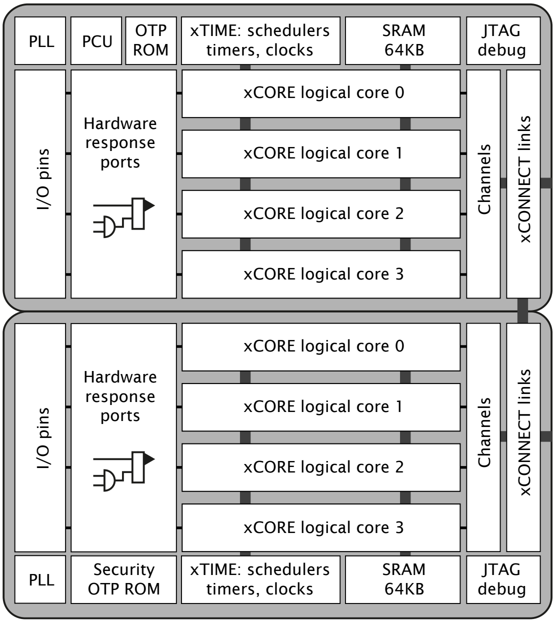


Figure 2: Block Diagram

The device can be configured using a set of software components that are rapidly customized and composed. XMOS provides source code libraries for many standard components. The device can be programmed using high-level languages such as C/C++ and XMOS-originated extensions to C, called XC, that simplify the control over concurrency, I/O and time.

The XMOS toolchain includes compilers, a simulator, debugger and static timing analyzer. The combination of real-time software, a compiler and timing analyzer enables the programmer to close timings on components of the design without a detailed understanding of the hardware characteristics.

5.1 Logical cores, Synchronizers and Locks

Each tile has up to 4 active logical cores, which issue instructions down a shared four-stage pipeline. Instructions from the active cores are issued round-robin. Each core is allocated a quarter of the processing cycles.

Figure 3: Logical core performance

Speed Grade, MIPS, and frequency	MIPS per logical core
8: 800 MIPS, 400 MHz	100
10: 1000 MIPS, 500 MHz	125

Synchronizers are provided for fast synchronization in a group of logical cores. In a single instruction a logical core can block until all other logical cores in a group have reached the synchroniser. Locks are provided for fast mutual exclusion. A logical core can acquire or release a lock in a single instruction.

5.2 Channel Ends, Links and Switch

Logical cores communicate using point-to-point connections formed between two channel ends. Between tiles, channel communications are implemented over xConnect Links and routed through switches. The links operate in either 2 wires per direction or 5 wires per direction mode, depending on the amount of bandwidth required. Circuit switched, streaming and packet switched data can both be supported efficiently. Streams provide the fastest possible data rates between xCORE Tiles (up to 250 MBit/s), but each stream requires a single link to be reserved between switches on two tiles. All packet communications can be multiplexed onto a single link. A total of four 5bit links are available between both cores.

Information on the supported routing topologies that can be used to connect multiple devices together can be found in the XS1-L Link Performance and Design Guide, X2999.

5.3 Ports and Clock Blocks

Ports provide an interface between the logical cores and I/O pins. All pins of a port provide either output or input. Signals in different directions cannot be mapped onto the same port.

The operation of each port is synchronized to a clock block. A clock block can be connected to an external clock input, or it can be run from the divided reference clock. A clock block can also output its signal to a pin. On reset, each port is connected to clock block 0, which runs from the xCORE Tile reference clock.

The ports and links are multiplexed, allowing the pins to be configured for use by ports of different widths or links. If an xConnect Link is enabled, the pins of the underlying ports are disabled. If a port is enabled, it overrules ports with higher widths that share the same pins. The pins on the wider port that are not shared remain available for use when the narrower port is enabled. Ports always operate at their specified width, even if they share pins with another port.

5.4 Timers

Timers are 32-bit counters that are relative to the xCORE Tile reference clock. A timer is defined to tick every 10 ns. This value is derived from the reference clock, which is configured to tick at 100 MHz by default.

5.5 PLL

The PLL creates a high-speed clock that is used for the switch, tile, and reference clock. The PLL multiplication value is selected through the two MODE pins, and can be changed by software to speed up the tile or use less power. The MODE pins are set as shown in Figure 4:

Figure 4: PLL multiplier values and MODE pins

Oscillator Frequency	MODE 1	MODE 0	Tile Frequency	PLL Ratio	PLL settings OD	F	R
5-13 MHz	0	0	130-399.75 MHz	30.75	1	122	0
13-20 MHz	1	1	260-400.00 MHz	20	2	119	0
20-48 MHz	1	0	167-400.00 MHz	8.33	2	49	0
48-100 MHz	0	1	196-400.00 MHz	4	2	23	0

Figure 4 also lists the values of OD, F and R, which are the registers that define the ratio of the tile frequency to the oscillator frequency:

$$F_{core} = F_{osc} \times \frac{F+1}{2} \times \frac{1}{R+1} \times \frac{1}{OD+1}$$

OD, F and R must be chosen so that $0 \le R \le 63$, $0 \le F \le 4095$, $0 \le OD \le 7$, and $260MHz \le F_{osc} \times \frac{F+1}{2} \times \frac{1}{R+1} \le 1.3GHz$. The OD, F, and R values can be modified by writing to the digital node PLL configuration register.

The MODE pins must be held at a static value until the third rising edge of the system clock following the deassertion of the system reset.

Further details on configuring the clock can be found in the XS1-L Clock Frequency Control document, X1433.

5.6 Boot ROM

The device is kept in reset by driving RST_N low. When in reset, all GPIO pins are high impedance. When the device is taken out of reset by releasing RST_N the processor starts its internal reset process. After 15-150 μs (depending on the input clock), all GPIO pins have their internal pull-resistor enabled, and the processor boots at a clock speed that depends on MODE0 and MODE1.

The xCORE Tile boot procedure is illustrated in Figure 5. In normal usage, MODE[4:2] controls the boot source according to the table in Figure 6. If bit 5 of the security register (*see* §5.7.1) is set, the device boots from OTP.

If set to boot from SPI, the processor enables the four pins (X0D00, X0D01, X0D10, and X0D11) that connect to SPI, and drive the SPI clock at 2.5 MHz (assuming a 400 MHz core clock).

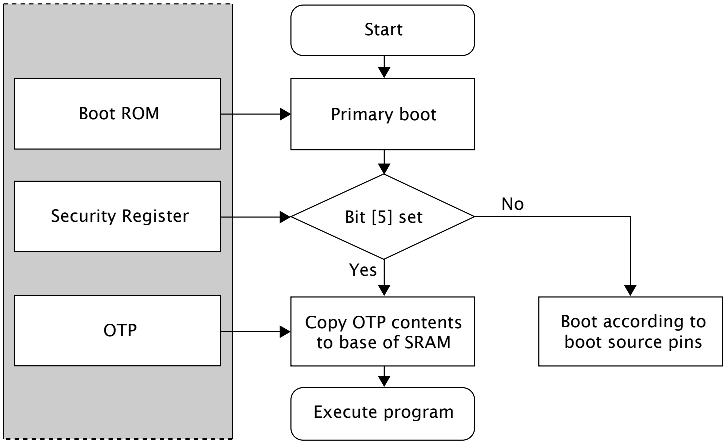


Figure 5: Boot procedure

Figure 6: Boot source pins

MODE[4]	MODE[3]	MODE[2]	Boot Source
X	0	0	None: Device waits to be booted via JTAG
X	0	1	Reserved
0	1	0	X0 boots from link B, X1 from channel end 0 via X0
0	1	1	X0 boots from SPI, X1 from channel end 0 via X0

Pin[A]	Signal	Description
X0D00	MISO	Master In Slave Out
X0D01	SS	Slave Select
X0D10	SCLK	Clock
X0D11	MOSI	Master Out Slave In

MODE[4]	MODE[3]	MODE[2]	Boot Source
1	1	0	X0 and X1 independently enable link B and internal links (E, F, G, H), and boot from channel end 0
1	1	1	Both tiles boot from SPI independently

A The pins used for SPI boot are hardcoded in the boot ROM and cannot be changed. An SPI boot program can be burned into OTP and used at any time.

If set to boot from a Link, the processor enables Link B around 200 ns after the boot process starts. Enabling the Link switches off the pull-down X8338, on resistors X0D16..X0D19, drives X0D16 and X0D17 low (the initial state for the Link), and monitors pins X0D19 and X0D20 for boot-traffic. X0D19 and X0D20 must be low at this stage. If the internal pull-down is too weak to drain any residual charge, external pull-downs of 10K may be required on those pins.

5.7 OTP

Each xCORE Tile integrates 8 KB one-time programmable (OTP) memory along with a security register that configures system wide security features. The OTP holds data in 2k rows x 32-bit configuration which can be used to implement secure bootloaders and store encryption keys. Data for the security register is loaded from the OTP on power up. All additional data in OTP is copied from the OTP to SRAM and executed first on the processor.

5.7.1 Security Register

The security register enables the following security features:

- **Secure Boot**: The xCORE Tile is forced to boot from address 0 of the OTP, allowing the xCORE Tile boot ROM to be bypassed (*see* §5.6). This feature can be used to implement a secure bootloader which loads an encrypted image from external flash, decrypts and CRC checks it with the processor, and discontinues the boot process if the decryption or CRC check fails. XMOS provides a default secure bootloader that can be written to the OTP along with secret decryption keys.
- **Disable JTAG**: The JTAG interface is disabled, making it impossible for the tile state or memory content to be accessed via the JTAG interface.
- **Disable Link access**: Other tiles are forbidden access to the processor state via the system switch.

  Disabling both JTAG and Link access transforms an xCORE Tile into a "secure island" with other tiles free for non-secure user application code.
- **Disable Global Debug access**: Disables access to the DEBUG_N pin.
- **OTP Master and Sector Lock**: Further access to the OTP is prevented by setting the master lock. Locks can also be applied to each of the four OTP sectors individually.

These security features provide a strong level of protection and are sufficient for providing strong IP security.

5.8 SRAM

Each xCORE Tile integrates a single 64 KB SRAM bank for both instructions and data. All internal memory is 32 bits wide, and instructions are either 16-bit or 32-bit. Byte (8-bit), half-word (16-bit) or word (32-bit) accesses are supported and are executed within one tile clock cycle. There is no dedicated external memory interface, although data memory can be expanded through appropriate use of the ports.

5.9 JTAG

The JTAG module can be used for loading programs, boundary scan testing, in-circuit source-level debugging and programming the OTP memory.

The JTAG chain structure is illustrated in Figure 7. Directly after reset, two TAP controllers are present in the JTAG chain for each xCORE Tile: the boundary scan TAP and the chip TAP. The boundary scan TAP is a standard 1149.1 compliant TAP

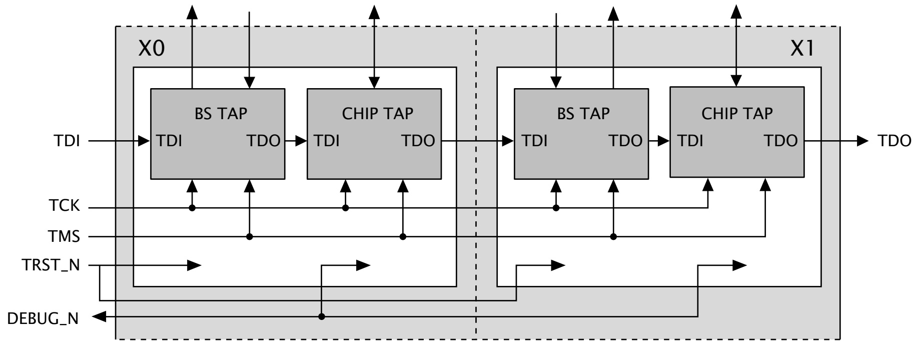


Figure 7: JTAG chain structure

that can be used for boundary scan of the I/O pins. The chip TAP provides access into the xCORE Tile, switch and OTP for loading code and debugging.

The TRST_N pin must be asserted low during and after power up for 100 ns. If JTAG is not required, the TRST_N pin can be tied to ground to hold the JTAG module in reset.

The DEBUG_N pin is used to synchronize the debugging of multiple xCORE Tiles. This pin can operate in both output and input mode. In output mode and when configured to do so, DEBUG_N is driven low by the device when the processor hits a debug break point. Prior to this point the pin will be tri-stated. In input mode and when configured to do so, driving this pin low will put the xCORE Tile into debug mode. Software can set the behavior of the xCORE Tile based on this pin. This pin should have an external pull up of 4K7-47K Ω or left not connected in single core applications.

The JTAG device identification register can be read by using the IDCODE instruction. Its contents are specified in Figure 8.

Figure 8: IDCODE return value

<table>
<tr><td colspan="4">Bit31</td><td colspan="24">Device Identification Register</td><td colspan="4">Bit0</td></tr>
<tr><td colspan="4">Version</td><td colspan="16">Part Number</td><td colspan="11">Manufacturer Identity</td><td>1</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>1</td><td>1</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>0</td><td>0</td><td>1</td><td>1</td></tr>
<tr><td colspan="4">0</td><td colspan="4">0</td><td colspan="4">0</td><td colspan="4">0</td><td colspan="4">2</td><td colspan="4">6</td><td colspan="4">3</td><td colspan="4">3</td></tr>
</table>

The JTAG usercode register can be read by using the USERCODE instruction. Its contents are specified in Figure 9. The OTP User ID field is read from bits [22:31] of the security register on xCORE Tile 0 (all zero on unprogrammed devices).

Figure 9: USERCODE return value

<table>
<tr><td colspan="4">Bit31</td><td colspan="24">Usercode Register</td><td colspan="4">Bit0</td></tr>
<tr><td colspan="10">OTP User ID</td><td colspan="4">Unused</td><td colspan="18">Silicon Revision</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td colspan="4">0</td><td colspan="4">0</td><td colspan="4">0</td><td colspan="4">2</td><td colspan="4">8</td><td colspan="4">0</td><td colspan="4">0</td><td colspan="4">0</td></tr>
</table>

5.10 PCU

The PCU can be used to isolate the core voltage of the device and reapply it under a controlled condition known as *sleep mode*. In sleep mode, all data in the SRAM is lost. The device recovers into functional mode under the control of an external PCU_WAKE signal or an internal timer.

If the PCU is not required, PCU_WAKE should be left unconnected, PCU_GATE should be left unconnected and PCU_CLK must be tied to CLK.

6 Board Integration

The device has the following power supply pins:

- VDD pins for the xCORE Tile
- VDDIO pins for the I/O lines
- PLL_AVDD pins for the PLL
- PCU_VDD and PCU_VDDIO pins for the PCU
- OTP_VCC pins for the OTP

Several pins of each type are provided to minimize the effect of inductance within the package, all of which must be connected. The power supplies must be brought up monotonically and input voltages must not exceed specification at any time.

The VDD supply must ramp from 0 V to its final value within 10 ms to ensure correct startup.

The VDDIO supply must ramp to its final value before VDD reaches 0.4 V.

The PLL_AVDD supply should be separated from the other noisier supplies on the board. The PLL requires a very clean power supply, and a low pass filter (for example, a 2.2 Ω resistor and 100 nF multi-layer ceramic capacitor) is recommended on this pin.

The PCU_VDD supply must be connected to the VDD supply.

The PCU_VDDIO supply must be connected to the VDDIO supply.

The OTP_VCC supply should be connected to the VDDIO supply.

The following ground pins are provided:

- PLL_AGND for PLL_AVDD
- GND for all other supplies

All ground pins must be connected directly to the board ground.

The VDD and VDDIO supplies should be decoupled close to the chip by several 100 nF low inductance multi-layer ceramic capacitors between the supplies and GND (for example, 4x100nF 0402 low inductance MLCCs per supply rail). The ground side of the decoupling capacitors should have as short a path back to the GND pins as possible. A bulk decoupling capacitor of at least 10 uF should be placed on each of these supplies.

RST_N is an active-low asynchronous-assertion global reset signal. Following a reset, the PLL re-establishes lock after which the device boots up according to the boot mode (*see* §5.6). RST_N and must be asserted low during and after power up for 100 ns.

6.1 Land patterns and solder stencils

The land pattern recommendations in this document are based on a RoHS compliant process and derived, where possible, from the nominal *Generic Requirements for Surface Mount Design and Land Pattern Standards* IPC-7351B specifications. This standard aims to achieve desired targets of heel, toe and side fillets for solder-joints.

Solder paste and ground via recommendations are based on our engineering and development kit board production. They have been found to work and optimised as appropriate to achieve a high yield. The size, type and number of vias used in the center pad affects how much solder wicks down the vias during reflow. This in turn, along with solder paster coverage, affects the final assembled package height. These factors should be taken into account during design and manufacturing of the PCB.

The following land patterns and solder paste contains recommendations. Final land pattern and solder paste decisions are the responsibility of the customer. These should be tuned during manufacture to suit the manufacturing process.

The package is a 124 pin dual row Quad Flat No lead package with exposed heat slug on a 0.5mm pitch.

An example land pattern is shown in Figure 10.

Pad widths and spacings are such that solder mask can still be applied between the pads using standard design rules. This is highly recommended to reduce solder shorts between pads. See the recommended PCB solder mask diagram in Figure 11.

6.2 Solder Stencil

The solder joints in the QFN package are formed exclusively from the solder paste deposited from the solder stencil. At the small aperture sizes required, the design of the stencil becomes important to ensure a reliable final solder joint volume and reliable solder joints.

The solder stencil recommendations here are based on those suggested in the IPC specification IPC-7525A "Stencil Design Guidelines".

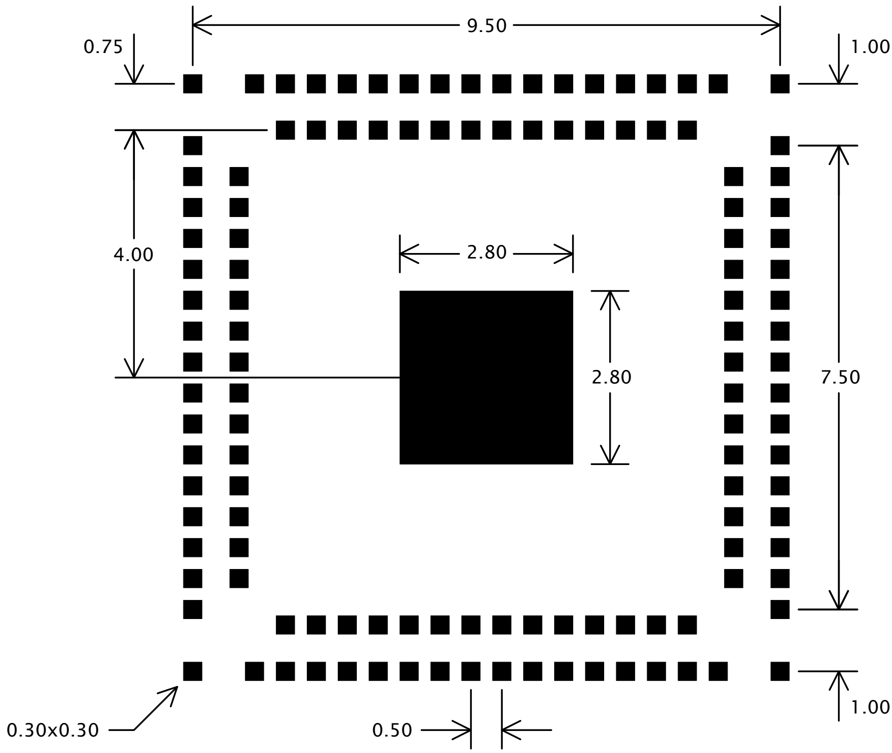


Figure 10: Example land pattern

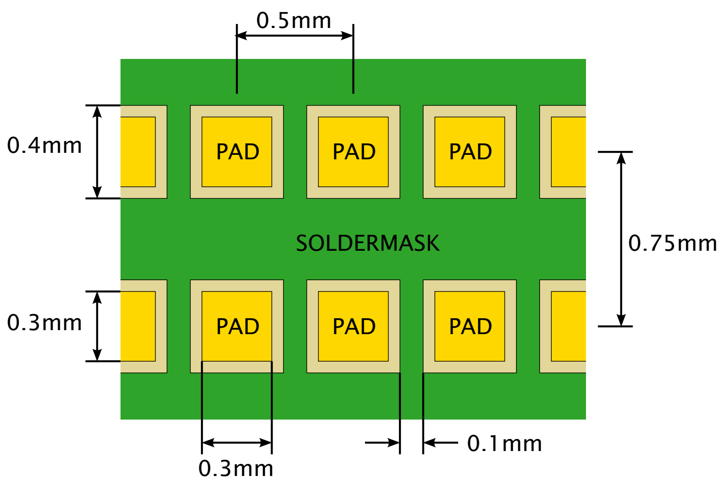


Figure 11: Detail of outer pads

As the aperture size in the stencil becomes very small, the amount of solder which remains on the PCB pad after printing is reduced. This occurs due to friction between the walls of the stencil and the solder paste dragging the paste from the pad when the stencil is removed. This effect is minimized as the thickness of the stencil is reduced.

For the 124 pin QFN package, our recommendations are to use a 4mil thick laser cut stencil. The solder stencil apertures for the pads should be 0.3mm square with 0.06mm radiused corners. This is the same size as the pads themselves apart from radiused corners to aid in paste transfer. This can be seen in the Figure 12.

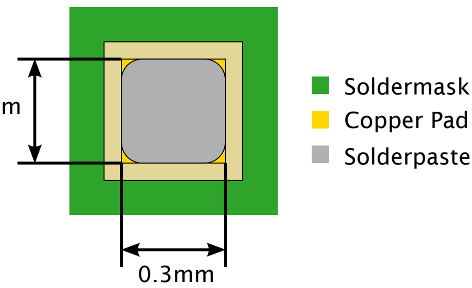


Figure 12: Solder stencil for outer pads

These dimensions should be the final aperture sizes used on the stencil, this should be agreed with the stencil makers or assembly house. It is common for assembly houses to subject the paste mask data to a global undersize before cutting the stencil. If this undersize is applied to these small apertures the paste transfer is likely to be poor and open solder joints may result.

For the center pad of this package, four squares of solder paste is recommended, 1mm on a side as shown in Figure 13. This gives a paste to pad area ratio of 51%.

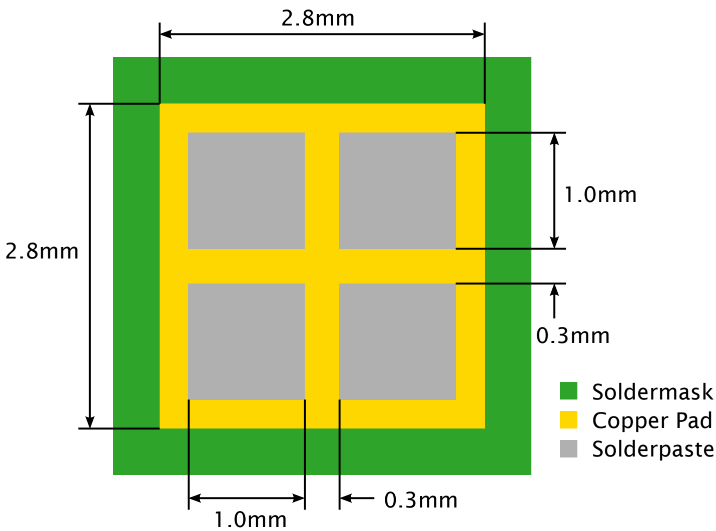


Figure 13: Solder stencil for centre pad

6.3 Ground and Thermal Vias

Vias under the heat slug into the ground plane of the PCB are recommended for a low inductance ground connection and good thermal performance. A 3 x 3 grid of vias, with a 0.6mm diameter annular ring and a 0.3mm drill, equally spaced across the heat slug, would be suitable.

6.4 Moisture Sensitivity

XMOS devices are, like all semiconductor devices, susceptible to moisture absorption. When removed from the sealed packaging, the devices slowly absorb moisture from the surrounding environment. If the level of moisture present in the device is too high during reflow, damage can occur due to the increased internal vapour pressure of moisture. Example damage can include bond wire damage, die lifting, internal or external package cracks and/or delamination.

All XMOS devices are Moisture Sensitivity Level (MSL) 3 - devices have a shelf life of 168 hours between removal from the packaging and reflow, provided they are stored below 30C and 60% RH. If devices have exceeded these values or an included moisture indicator card shows excessive levels of moisture, then the parts should be baked as appropriate before use. This is based on information from *Joint IPC/JEDEC Standard For Moisture/Reflow Sensitivity Classification For Nonhermetic Solid State Surface-Mount Devices* J-STD-020 Revision D.

7 DC and Switching Characteristics

7.1 Operating Conditions

Symbol	Parameter	MIN	TYP	MAX	UNITS	Notes
VDD	Tile DC supply voltage	0.95	1.00	1.05	V	
VDDIO	I/O supply voltage	3.00	3.30	3.60	V	
PLL_AVDD	PLL analog supply	0.95	1.00	1.05	V	
PCU_VDD	PCU tile DC supply voltage	0.95	1.00	1.05	V	
PCU_VDDIO	PCU I/O DC supply voltage	3.00	3.30	3.60	V	
OTP_VCC	OTP supply voltage	3.00	3.30	3.60	V	
Cl	xCORE Tile I/O load capacitance			25	pF	
Ta	Ambient operating temperature (Commercial)	0		70	°C	
	Ambient operating temperature (Industrial)	-40		85	°C	
Tj	Junction temperature			125	°C	
Tstg	Storage temperature	-65		150	°C	

Figure 14: Operating conditions

7.2 DC Characteristics

Symbol	Parameter	MIN	TYP	MAX	UNITS	Notes
V(IH)	Input high voltage	2.00		3.60	V	A
V(IL)	Input low voltage	-0.30		0.70	V	A
V(OH)	Output high voltage	2.70			V	B, C
V(OL)	Output low voltage			0.60	V	B, C
R(PU)	Pull-up resistance		35K		Ω	D
R(PD)	Pull-down resistance		35K		Ω	D

Figure 15: DC characteristics

A All pins except power supply pins.

B Ports 1A, 1D, 1E, 1H, 1I, 1J, 1K and 1L are nominal 8 mA drivers, the remainder of the general-purpose I/Os are 4 mA.

C Measured with 4 mA drivers sourcing 4 mA, 8 mA drivers sourcing 8 mA.

D Used to guarantee logic state for an I/O when high impedance. The internal pull-ups/pull-downs should not be used to pull external circuitry.

7.3 ESD Stress Voltage

Symbol	Parameter	MIN	TYP	MAX	UNITS	Notes
HBM	Human body model	-2.00		2.00	KV	
MM	Machine model	-200		200	V	

Figure 16: ESD stress voltage

7.4 Reset Timing

Figure 17: Reset timing

Symbol	Parameters	MIN	TYP	MAX	UNITS	Notes
T(RST)	Reset pulse width	5			us	
T(INIT)	Initialization time			150	µs	A

A Shows the time taken to start booting after RST_N has gone high.

7.5 Power Consumption

Figure 18: xCORE Tile currents

Symbol	Parameter	MIN	TYP	MAX	UNITS	Notes
I(DDCQ)	Quiescent VDD current		28		mA	A, B, C
PD	Tile power dissipation		450		µW/MIPS	A, D, E, F
IDD	Active VDD current (Speed Grade 8)		320	600	mA	A, G
	Active VDD current (Speed Grade 10)		400	750	mA	A, H
I(ADDPLL)	PLL_AVDD current			14	mA	I

A Use for budgetary purposes only.
B Assumes typical tile and I/O voltages with no switching activity.
C Includes PLL current.
D Assumes typical tile and I/O voltages with nominal switching activity.
E Assumes 1 MHz = 1 MIPS.
F PD(TYP) value is the usage power consumption under typical operating conditions.
G Measurement conditions: VDD = 1.0 V, VDDIO = 3.3 V, 25 °C, 400 MHz, average device resource usage.
H Measurement conditions: VDD = 1.0 V, VDDIO = 3.3 V, 25 °C, 500 MHz, average device resource usage.
I PLL_AVDD = 1.0 V

The tile power consumption of the device is highly application dependent and should be used for budgetary purposes only.

More detailed power analysis can be found in the XS1-L Power Consumption document, X2999.

7.6 Clock

Symbol	Parameter	MIN	TYP	MAX	UNITS	Notes
f	Frequency	4.22	20	100	MHz	
SR	Slew rate	0.10			V/ns	
TJ(LT)	Long term jitter (pk-pk)			2	%	A
f(MAX)	Processor clock frequency (Speed Grade 8)			400	MHz	B
	Processor clock frequency (Speed Grade 10)			500	MHz	B

A Percentage of CLK period.

B Assumes typical tile and I/O voltages with nominal activity.

Figure 19: Clock

Further details can be found in the XS1-L Clock Frequency Control document, X1433.

7.7 xCORE Tile I/O AC Characteristics

Symbol	Parameter	MIN	TYP	MAX	UNITS	Notes
T(XOVALID)	Input data valid window	8			ns	
T(XOINVALID)	Output data invalid window	9			ns	
T(XIFMAX)	Rate at which data can be sampled with respect to an external clock			60	MHz	

Figure 20: I/O AC characteristics

The input valid window parameter relates to the capability of the device to capture data input to the chip with respect to an external clock source. It is calculated as the sum of the input setup time and input hold time with respect to the external clock as measured at the pins. The output invalid window specifies the time for which an output is invalid with respect to the external clock. Note that these parameters are specified as a window rather than absolute numbers since the device provides functionality to delay the incoming clock with respect to the incoming data.

Information on interfacing to high-speed synchronous interfaces can be found in the XS1 Port I/O Timing document, X5821.

7.8 xConnect Link Performance

Symbol	Parameter	MIN	TYP	MAX	UNITS	Notes
B(2blinkP)	2b link bandwidth (packetized)			87	MBit/s	A, B
B(5blinkP)	5b link bandwidth (packetized)			217	MBit/s	A, B
B(2blinkS)	2b link bandwidth (streaming)			100	MBit/s	B
B(5blinkS)	5b link bandwidth (streaming)			250	MBit/s	B

A Assumes 32-byte packet in 3-byte header mode. Actual performance depends on size of the header and payload.

B 7.5 ns symbol time.

Figure 21: Link performance

The asynchronous nature of links means that the relative phasing of CLK clocks is not important in a multi-clock system, providing each meets the required stability criteria.

7.9 JTAG Timing

Figure 22: JTAG timing

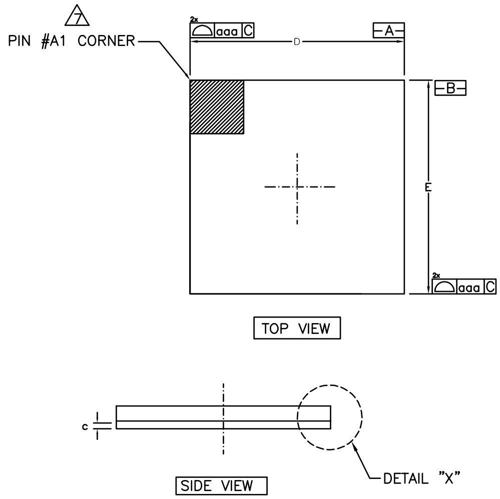
Symbol	Parameter	MIN	TYP	MAX	UNITS	Notes
f(TCK_D)	TCK frequency (debug)			18	MHz	
f(TCK_B)	TCK frequency (boundary scan)			10	MHz	
T(SETUP)	TDO to TCK setup time	5			ns	A
T(HOLD)	TDO to TCK hold time	5			ns	A
T(DELAY)	TCK to output delay			15	ns	B

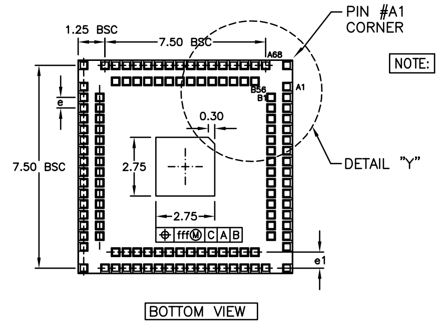
A Timing applies to TMS and TDI inputs.

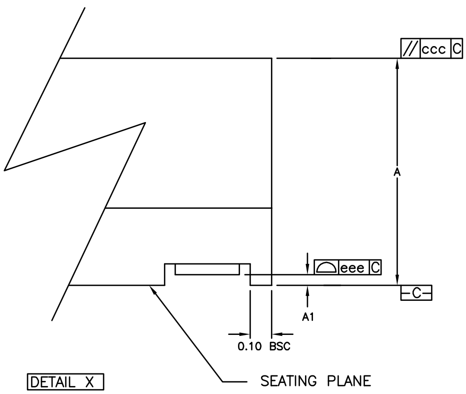
B Timing applies to TDO output from negative edge of TCK.

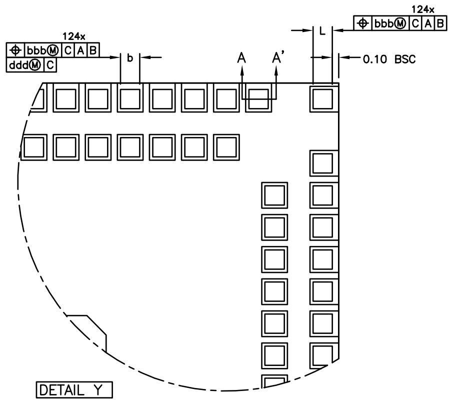
All JTAG operations are synchronous to TCK apart from the global asynchronous reset TRST_N.

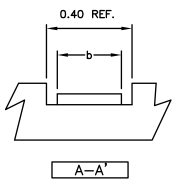
8 Package Information











SYMBOL	MIN.	NOM.	MAX.
A	1.01	1.06	1.11
A1	0.00	–	0.05
D	9.90	10.00	10.10
E	9.90	10.00	10.10
aaa			0.10
bbb			0.10
ccc			0.10
ddd			0.08
eee			0.08
fff			0.10
b	0.25	0.30	0.35
L	0.25	0.30	0.35
e	0.50 BSC.		
e1	0.75 BSC.		
c	0.36 REF.		

1. ALL DIMENSIONS ARE IN MILLIMETERS.
2. DIMENSIONING AND TOLERANCING PER ASME Y14.5M–1994
3. PRIMARY DATUM -C- AND SEATING PLANE ARE DESIGNED BY THE SPHERICAL CROWNS OF THE SOLDER BALLS.
4. A1 CORNER MUST BE IDENTIFIED BY INK OR LASER MARK.

8.1 Part Marking

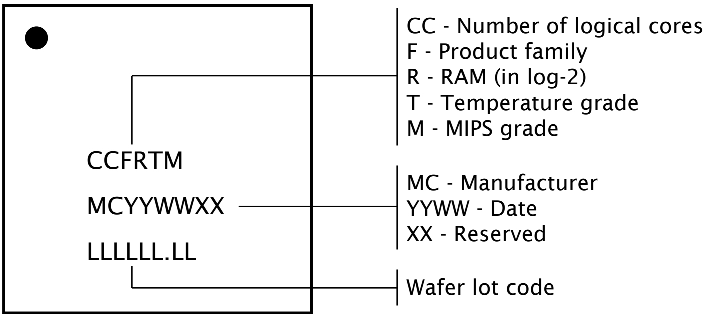


Figure 23: Part marking scheme

9 Ordering Information

Figure 24: Orderable part numbers

Product Code	Marking	Qualification	Speed Grade
XS1-L8A-128-QF124-C8	8L7C8	Commercial	800 MIPS
XS1-L8A-128-QF124-C10	8L7C10	Commercial	1000 MIPS
XS1-L8A-128-QF124-I8	8L7I8	Industrial	800 MIPS
XS1-L8A-128-QF124-I10	8L7I10	Industrial	1000 MIPS

Appendices

A Configuration of the XS1

The device is configured through three banks of registers, as shown in Figure 25.

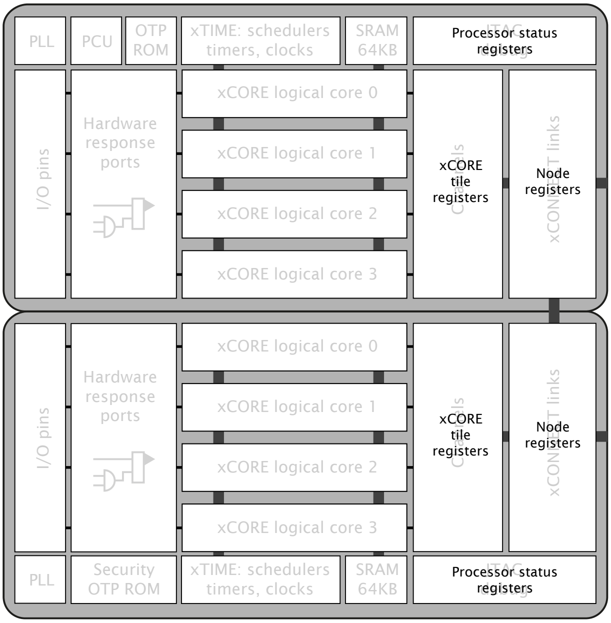


Figure 25: Registers

The following communication sequences specify how to access those registers. Any messages transmitted contain the most significant 24 bits of the channel-end to which a response is to be sent. This comprises the node-identifier and the channel number within the node. if no response is required on a write operation, supply 24-bits with the last 8-bits set, which suppresses the reply message. Any multi-byte data is sent most significant byte first.

A.1 Accessing a processor status register

The processor status registers are accessed directly from the processor instruction set. The instructions GETPS and SETPS read and write a word. The register number should be translated into a processor-status resource identifier by shifting the register number left 8 places, and ORing it with 0x0C. Alternatively, the functions `getps(reg)` and `setps(reg,value)` can be used from XC.

A.2 Accessing an xCORE Tile configuration register

xCORE Tile configuration registers can be accessed through the interconnect using the functions `write_tile_config_reg(tileref, ...)` and `read_tile_config_reg(tile`

↪ `ref, ...)`, where `tileref` is the name of the xCORE Tile, e.g. `tile[1]`. These functions implement the protocols described below.

Instead of using the functions above, a channel-end can be allocated to communicate with the xCORE tile configuration registers. The destination of the channel-end should be set to `0xnnnnC20C` where `nnnnnn` is the tile-identifier.

A write message comprises the following:

control-token 192	24-bit response channel-end identifier	16-bit register number	32-bit data	control-token 1

The response to a write message comprises either control tokens 3 and 1 (for success), or control tokens 4 and 1 (for failure).

A read message comprises the following:

control-token 193	24-bit response channel-end identifier	16-bit register number	control-token 1

The response to the read message comprises either control token 3, 32-bit of data, and control-token 1 (for success), or control tokens 4 and 1 (for failure).

A.3 Accessing node configuration

Node configuration registers can be accessed through the interconnect using the functions `write_node_config_reg(device, ...)` and `read_node_config_reg(device,` ↪ `...)`, where `device` is the name of the node. These functions implement the protocols described below.

Instead of using the functions above, a channel-end can be allocated to communicate with the node configuration registers. The destination of the channel-end should be set to `0xnnnnC30C` where `nnnn` is the node-identifier.

A write message comprises the following:

control-token 192	24-bit response channel-end identifier	16-bit register number	32-bit data	control-token 1

The response to a write message comprises either control tokens 3 and 1 (for success), or control tokens 4 and 1 (for failure).

A read message comprises the following:

control-token 193	24-bit response channel-end identifier	16-bit register number	control-token 1

The response to a read message comprises either control token 3, 32-bit of data, and control-token 1 (for success), or control tokens 4 and 1 (for failure).

B Processor Status Configuration

The processor status control registers can be accessed directly by the processor using processor status reads and writes (use `getps(reg)` and `setps(reg,value)` for reads and writes).

Number	Perm	Description
0x00	RW	RAM base address
0x01	RW	Vector base address
0x02	RW	xCORE Tile control
0x03	RO	xCORE Tile boot status
0x05	RO	Security configuration
0x06	RW	Ring Oscillator Control
0x07	RO	Ring Oscillator Value
0x08	RO	Ring Oscillator Value
0x09	RO	Ring Oscillator Value
0x0A	RO	Ring Oscillator Value
0x10	DRW	Debug SSR
0x11	DRW	Debug SPC
0x12	DRW	Debug SSP
0x13	DRW	DGETREG operand 1
0x14	DRW	DGETREG operand 2
0x15	DRW	Debug interrupt type
0x16	DRW	Debug interrupt data
0x18	DRW	Debug core control
0x20 .. 0x27	DRW	Debug scratch
0x30 .. 0x33	DRW	Instruction breakpoint address
0x40 .. 0x43	DRW	Instruction breakpoint control
0x50 .. 0x53	DRW	Data watchpoint address 1
0x60 .. 0x63	DRW	Data watchpoint address 2
0x70 .. 0x73	DRW	Data breakpoint control register
0x80 .. 0x83	DRW	Resources breakpoint mask
0x90 .. 0x93	DRW	Resources breakpoint value
0x9C .. 0x9F	DRW	Resources breakpoint control register

Figure 26: Summary

B.1 RAM base address: 0x00

This register contains the base address of the RAM. It is initialized to 0x00010000.

0x00: RAM base address

Bits	Perm	Init	Description
31:2	RW		Most significant 16 bits of all addresses.
1:0	RO	-	Reserved

B.2 Vector base address: 0x01

Base address of event vectors in each resource. On an interrupt or event, the 16 most significant bits of the destination address are provided by this register; the least significant 16 bits come from the event vector.

0x01: Vector base address

Bits	Perm	Init	Description
31:16	RW		The most significant bits for all event and interrupt vectors.
15:0	RO	-	Reserved

B.3 xCORE Tile control: 0x02

Register to control features in the xCORE tile

0x02: xCORE Tile control

Bits	Perm	Init	Description
31:6	RO	-	Reserved
5	RW	0	Set to 1 to select the dynamic mode for the clock divider when the clock divider is enabled. In dynamic mode the clock divider is only activated when all active logical cores are paused. In static mode the clock divider is always enabled.
4	RW	0	Set to 1 to enable the clock divider. This slows down the xCORE tile clock in order to use less power.
3:0	RO	-	Reserved

B.4 xCORE Tile boot status: 0x03

This read-only register describes the boot status of the xCORE tile.

0x03: xCORE Tile boot status

Bits	Perm	Init	Description
31:24	RO	-	Reserved
23:16	RO		xCORE tile number on the switch.
15:9	RO	-	Reserved
8	RO		Set to 1 if boot from OTP is enabled.
7:0	RO		The boot mode pins MODE0, MODE1, ..., specifying the boot frequency, boot source, etc.

B.5 Security configuration: 0x05

Copy of the security register as read from OTP.

0x05: Security configuration

Bits	Perm	Init	Description
31:0	RO		Value.

B.6 Ring Oscillator Control: 0x06

There are four free-running oscillators that clock four counters. The oscillators can be started and stopped using this register. The counters should only be read when the ring oscillator is stopped. The counter values can be read using four subsequent registers. The ring oscillators are asynchronous to the xCORE tile clock and can be used as a source of random bits.

0x06: Ring Oscillator Control

Bits	Perm	Init	Description
31:2	RO	-	Reserved
1	RW	0	Set to 1 to enable the xCORE tile ring oscillators
0	RW	0	Set to 1 to enable the peripheral ring oscillators

B.7 Ring Oscillator Value: 0x07

This register contains the current count of the xCORE Tile Cell ring oscillator. This value is not reset on a system reset.

0x07: Ring Oscillator Value

Bits	Perm	Init	Description
31:16	RO	-	Reserved
15:0	RO	-	Ring oscillator counter data.

B.8 Ring Oscillator Value: 0x08

This register contains the current count of the xCORE Tile Wire ring oscillator. This value is not reset on a system reset.

0x08: Ring Oscillator Value

Bits	Perm	Init	Description
31:16	RO	-	Reserved
15:0	RO	-	Ring oscillator counter data.

B.9 Ring Oscillator Value: 0x09

This register contains the current count of the Peripheral Cell ring oscillator. This value is not reset on a system reset.

0x09: Ring Oscillator Value

Bits	Perm	Init	Description
31:16	RO	-	Reserved
15:0	RO	-	Ring oscillator counter data.

B.10 Ring Oscillator Value: 0x0A

This register contains the current count of the Peripheral Wire ring oscillator. This value is not reset on a system reset.

0x0A: Ring Oscillator Value

Bits	Perm	Init	Description
31:16	RO	-	Reserved
15:0	RO	-	Ring oscillator counter data.

B.11 Debug SSR: 0x10

This register contains the value of the SSR register when the debugger was called.

0x10: Debug SSR

Bits	Perm	Init	Description
31:0	RO	-	Reserved

B.12 Debug SPC: 0x11

This register contains the value of the SPC register when the debugger was called.

0x11: Debug SPC

Bits	Perm	Init	Description
31:0	DRW		Value.

B.13 Debug SSP: 0x12

This register contains the value of the SSP register when the debugger was called.

0x12: Debug SSP

Bits	Perm	Init	Description
31:0	DRW		Value.

B.14 DGETREG operand 1: 0x13

The resource ID of the logical core whose state is to be read.

0x13: DGETREG operand 1

Bits	Perm	Init	Description
31:8	RO	-	Reserved
7:0	DRW		Thread number to be read

B.15 DGETREG operand 2: 0x14

Register number to be read by DGETREG

0x14: DGETREG operand 2

Bits	Perm	Init	Description
31:5	RO	-	Reserved
4:0	DRW		Register number to be read

B.16 Debug interrupt type: 0x15

Register that specifies what activated the debug interrupt.

0x15: Debug interrupt type

Bits	Perm	Init	Description
31:18	RO	-	Reserved
17:16	DRW		If the debug interrupt was caused by a hardware breakpoint or hardware watchpoint, this field contains the number of the breakpoint or watchpoint. If multiple breakpoints or watchpoints trigger at once, the lowest number is taken.
15:8	DRW		If the debug interrupt was caused by a logical core, this field contains the number of that core. Otherwise this field is 0.
7:3	RO	-	Reserved
2:0	DRW	0	Indicates the cause of the debug interrupt 1: Host initiated a debug interrupt through JTAG 2: Program executed a DCALL instruction 3: Instruction breakpoint 4: Data watch point 5: Resource watch point

B.17 Debug interrupt data: 0x16

On a data watchpoint, this register contains the effective address of the memory operation that triggered the debugger. On a resource watchpoint, it countains the resource identifier.

0x16: Debug interrupt data

Bits	Perm	Init	Description
31:0	DRW		Value.

B.18 Debug core control: 0x18

This register enables the debugger to temporarily disable logical cores. When returning from the debug interrupts, the cores set in this register will not execute. This enables single stepping to be implemented.

0x18: Debug core control

Bits	Perm	Init	Description
31:8	RO	-	Reserved
7:0	DRW		1-hot vector defining which logical cores are stopped when not in debug mode. Every bit which is set prevents the respective logical core from running.

B.19 Debug scratch: 0x20 .. 0x27

A set of registers used by the debug ROM to communicate with an external debugger, for example over JTAG. This is the same set of registers as the Debug Scratch registers in the xCORE tile configuration.

0x20 .. 0x27: Debug scratch

Bits	Perm	Init	Description
31:0	DRW		Value.

B.20 Instruction breakpoint address: 0x30 .. 0x33

This register contains the address of the instruction breakpoint. If the PC matches this address, then a debug interrupt will be taken. There are four instruction breakpoints that are controlled individually.

0x30 .. 0x33: Instruction breakpoint address

Bits	Perm	Init	Description
31:0	DRW		Value.

B.21 Instruction breakpoint control: 0x40 .. 0x43

This register controls which logical cores may take an instruction breakpoint, and under which condition.

0x40 .. 0x43: Instruction breakpoint control

Bits	Perm	Init	Description
31:24	RO	-	Reserved
23:16	DRW	0	A bit for each logical core in the tile allowing the breakpoint to be enabled individually for each logical core.
15:2	RO	-	Reserved
1	DRW	0	Set to 1 to cause an instruction breakpoint if the PC is not equal to the breakpoint address. By default, the breakpoint is triggered when the PC is equal to the breakpoint address.
0	DRW	0	When 1 the instruction breakpoint is enabled.

B.22 Data watchpoint address 1: 0x50 .. 0x53

This set of registers contains the first address for the four data watchpoints.

0x50 .. 0x53: Data watchpoint address 1

Bits	Perm	Init	Description
31:0	DRW		Value.

B.23 Data watchpoint address 2: 0x60 .. 0x63

This set of registers contains the second address for the four data watchpoints.

0x60 .. 0x63: Data watchpoint address 2

Bits	Perm	Init	Description
31:0	DRW		Value.

B.24 Data breakpoint control register: 0x70 .. 0x73

This set of registers controls each of the four data watchpoints.

0x70 .. 0x73: Data breakpoint control register

Bits	Perm	Init	Description
31:24	RO	-	Reserved
23:16	DRW	0	A bit for each logical core in the tile allowing the breakpoint to be enabled individually for each logical core.
15:3	RO	-	Reserved
2	DRW	0	Set to 1 to enable breakpoints to be triggered on loads. Breakpoints always trigger on stores.
1	DRW	0	By default, data watchpoints trigger if memory in the range [Address1..Address2] is accessed (the range is inclusive of Address1 and Address2). If set to 1, data watchpoints trigger if memory outside the range (Address2..Address1) is accessed (the range is exclusive of Address2 and Address1).
0	DRW	0	When 1 the instruction breakpoint is enabled.

B.25 Resources breakpoint mask: 0x80 .. 0x83

This set of registers contains the mask for the four resource watchpoints.

0x80 .. 0x83: Resources breakpoint mask

Bits	Perm	Init	Description
31:0	DRW		Value.

B.26 Resources breakpoint value: 0x90 .. 0x93

This set of registers contains the value for the four resource watchpoints.

0x90 .. 0x93: Resources breakpoint value

Bits	Perm	Init	Description
31:0	DRW		Value.

B.27 Resources breakpoint control register: 0x9C .. 0x9F

This set of registers controls each of the four resource watchpoints.

0x9C .. 0x9F: Resources breakpoint control register

Bits	Perm	Init	Description
31:24	RO	-	Reserved
23:16	DRW	0	A bit for each logical core in the tile allowing the breakpoint to be enabled individually for each logical core.
15:2	RO	-	Reserved
1	DRW	0	By default, resource watchpoints trigger when the resource id masked with the set Mask equals the Value. If set to 1, resource watchpoints trigger when the resource id masked with the set Mask is not equal to the Value.
0	DRW	0	When 1 the instruction breakpoint is enabled.

C Tile Configuration

The xCORE Tile control registers can be accessed using configuration reads and writes (use `write_tile_config_reg(tileref, ...)` and `read_tile_config_reg(tileref, ↪ ...)` for reads and writes).

Number	Perm	Description
0x00	RO	Device identification
0x01	RO	xCORE Tile description 1
0x02	RO	xCORE Tile description 2
0x04	CRW	Control PSwitch permissions to debug registers
0x05	CRW	Cause debug interrupts
0x06	RW	xCORE Tile clock divider
0x07	RO	Security configuration
0x10 .. 0x13	RO	PLink status
0x20 .. 0x27	CRW	Debug scratch
0x40	RO	PC of logical core 0
0x41	RO	PC of logical core 1
0x42	RO	PC of logical core 2
0x43	RO	PC of logical core 3
0x60	RO	SR of logical core 0
0x61	RO	SR of logical core 1
0x62	RO	SR of logical core 2
0x63	RO	SR of logical core 3
0x80 .. 0x9F	RO	Chanend status

Figure 27: Summary

C.1 Device identification: 0x00

Bits	Perm	Init	Description
31:24	RO		Processor ID of this xCORE tile.
23:16	RO		Number of the node in which this xCORE tile is located.
15:8	RO		xCORE tile revision.
7:0	RO		xCORE tile version.

0x00: Device identification

C.2 xCORE Tile description 1: 0x01

This register describes the number of logical cores, synchronisers, locks and channel ends available on this xCORE tile.

0x01: xCORE Tile description 1

Bits	Perm	Init	Description
31:24	RO		Number of channel ends.
23:16	RO		Number of locks.
15:8	RO		Number of synchronisers.
7:0	RO	-	Reserved

C.3 xCORE Tile description 2: 0x02

This register describes the number of timers and clock blocks available on this xCORE tile.

0x02: xCORE Tile description 2

Bits	Perm	Init	Description
31:16	RO	-	Reserved
15:8	RO		Number of clock blocks.
7:0	RO		Number of timers.

C.4 Control PSwitch permissions to debug registers: 0x04

This register can be used to control whether the debug registers (marked with permission CRW) are accessible through the tile configuration registers. When this bit is set, write -access to those registers is disabled, preventing debugging of the xCORE tile over the interconnect.

0x04: Control PSwitch permissions to debug registers

Bits	Perm	Init	Description
31:1	RO	-	Reserved
0	CRW		Set to 1 to restrict PSwitch access to all CRW marked registers to become read-only rather than read-write.

C.5 Cause debug interrupts: 0x05

This register can be used to raise a debug interrupt in this xCORE tile.

0x05: Cause debug interrupts

Bits	Perm	Init	Description
31:2	RO	-	Reserved
1	RO	0	Set to 1 when the processor is in debug mode.
0	CRW	0	Set to 1 to request a debug interrupt on the processor.

C.6 xCORE Tile clock divider: 0x06

This register contains the value used to divide the PLL clock to create the xCORE tile clock. The divider is enabled under control of the tile control register

0x06: xCORE Tile clock divider

Bits	Perm	Init	Description
31:8	RO	-	Reserved
7:0	RW		Value of the clock divider minus one.

C.7 Security configuration: 0x07

Copy of the security register as read from OTP.

0x07: Security configuration

Bits	Perm	Init	Description
31:0	RO		Value.

C.8 PLink status: 0x10 .. 0x13

Status of each of the four processor links; connecting the xCORE tile to the switch.

0x10 .. 0x13: PLink status

Bits	Perm	Init	Description
31:26	RO	-	Reserved
25:24	RO		00 - ChannelEnd, 01 - ERROR, 10 - PSCTL, 11 - Idle.
23:16	RO		Based on SRC_TARGET_TYPE value, it represents channelEnd ID or Idle status.
15:6	RO	-	Reserved
5:4	RO		Two-bit network identifier
3	RO	-	Reserved
2	RO		1 when the current packet is considered junk and will be thrown away.
1	RO	0	Set to 1 if the switch is routing data into the link, and if a route exists from another link.
0	RO	0	Set to 1 if the link is routing data into the switch, and if a route is created to another link on the switch.

C.9 Debug scratch: 0x20 .. 0x27

A set of registers used by the debug ROM to communicate with an external debugger, for example over the switch. This is the same set of registers as the Debug Scratch registers in the processor status.

0x20 .. 0x27: Debug scratch

Bits	Perm	Init	Description
31:0	CRW		Value.

C.10 PC of logical core 0: 0x40

Value of the PC of logical core 0.

0x40: PC of logical core 0

Bits	Perm	Init	Description
31:0	RO		Value.

C.11 PC of logical core 1: 0x41

0x41: PC of logical core 1

Bits	Perm	Init	Description
31:0	RO		Value.

C.12 PC of logical core 2: 0x42

0x42: PC of logical core 2

Bits	Perm	Init	Description
31:0	RO		Value.

C.13 PC of logical core 3: 0x43

0x43: PC of logical core 3

Bits	Perm	Init	Description
31:0	RO		Value.

C.14 SR of logical core 0: 0x60

Value of the SR of logical core 0

0x60: SR of logical core 0

Bits	Perm	Init	Description
31:0	RO		Value.

C.15 SR of logical core 1: 0x61

0x61: SR of logical core 1

Bits	Perm	Init	Description
31:0	RO		Value.

C.16 SR of logical core 2: 0x62

0x62: SR of logical core 2

Bits	Perm	Init	Description
31:0	RO		Value.

C.17 SR of logical core 3: 0x63

0x63: SR of logical core 3

Bits	Perm	Init	Description
31:0	RO		Value.

C.18 Chanend status: 0x80 .. 0x9F

These registers record the status of each channel-end on the tile.

0x80 .. 0x9F: Chanend status

Bits	Perm	Init	Description
31:26	RO	-	Reserved
25:24	RO		00 - ChannelEnd, 01 - ERROR, 10 - PSCTL, 11 - Idle.
23:16	RO		Based on SRC_TARGET_TYPE value, it represents channelEnd ID or Idle status.
15:6	RO	-	Reserved
5:4	RO		Two-bit network identifier
3	RO	-	Reserved
2	RO		1 when the current packet is considered junk and will be thrown away.
1	RO	0	Set to 1 if the switch is routing data into the link, and if a route exists from another link.
0	RO	0	Set to 1 if the link is routing data into the switch, and if a route is created to another link on the switch.

D Node Configuration

The digital node control registers can be accessed using configuration reads and writes (use `write_node_config_reg(device, ...)` and `read_node_config_reg(device, ↪ ...)` for reads and writes).

Number	Perm	Description
0x00	RO	Device identification
0x01	RO	System switch description
0x04	RW	Switch configuration
0x05	RW	Switch node identifier
0x06	RW	PLL settings
0x07	RW	System switch clock divider
0x08	RW	Reference clock
0x0C	RW	Directions 0-7
0x0D	RW	Directions 8-15
0x10	RW	DEBUG_N configuration
0x1F	RO	Debug source
0x20 .. 0x27	RW	Link status, direction, and network
0x40 .. 0x43	RW	PLink status and network
0x80 .. 0x87	RW	Link configuration and initialization
0xA0 .. 0xA7	RW	Static link configuration

Figure 28: Summary

D.1 Device identification: 0x00

This register contains version and revision identifiers and the mode-pins as sampled at boot-time.

Bits	Perm	Init	Description
31:24	RO	0x00	Chip identifier.
23:16	RO		Sampled values of pins MODE0, MODE1, ... on reset.
15:8	RO		SSwitch revision.
7:0	RO		SSwitch version.

0x00: Device identification

D.2 System switch description: 0x01

This register specifies the number of processors and links that are connected to this switch.

0x01: System switch description

Bits	Perm	Init	Description
31:24	RO	-	Reserved
23:16	RO		Number of links on the switch.
15:8	RO		Number of cores that are connected to this switch.
7:0	RO		Number of links per processor.

D.3 Switch configuration: 0x04

This register enables the setting of two security modes (that disable updates to the PLL or any other registers) and the header-mode.

0x04: Switch configuration

Bits	Perm	Init	Description
31	RO	0	Set to 1 to disable any write access to the configuration registers in this switch.
30:9	RO	-	Reserved
8	RO	0	Set to 1 to disable updates to the PLL configuration register.
7:1	RO	-	Reserved
0	RO	0	Header mode. Set to 1 to enable 1-byte headers. This must be performed on all nodes in the system.

D.4 Switch node identifier: 0x05

This register contains the node identifier.

0x05: Switch node identifier

Bits	Perm	Init	Description
31:16	RO	-	Reserved
15:0	RW	0	The unique 16-bit ID of this node. This ID is matched most-significant-bit first with incoming messages for routing purposes.

D.5 PLL settings: 0x06

An on-chip PLL multiplies the input clock up to a higher frequency clock, used to clock the I/O, processor, and switch, see Oscillator. Note: a write to this register will cause the tile to be reset.

0x06: PLL settings

Bits	Perm	Init	Description
31:26	RO	-	Reserved
25:23	RW		OD: Output divider value The initial value depends on pins MODE0 and MODE1.
22:21	RO	-	Reserved
20:8	RW		F: Feedback multiplication ratio The initial value depends on pins MODE0 and MODE1.
7	RO	-	Reserved
6:0	RW		R: Oscilator input divider value The initial value depends on pins MODE0 and MODE1.

D.6 System switch clock divider: 0x07

Sets the ratio of the PLL clock and the switch clock.

0x07: System switch clock divider

Bits	Perm	Init	Description
31:16	RO	-	Reserved
15:0	RW	0	Switch clock divider. The PLL clock will be divided by this value plus one to derive the switch clock.

D.7 Reference clock: 0x08

Sets the ratio of the PLL clock and the reference clock used by the node.

0x08: Reference clock

Bits	Perm	Init	Description
31:16	RO	-	Reserved
15:0	RW	3	Architecture reference clock divider. The PLL clock will be divided by this value plus one to derive the 100 MHz reference clock.

D.8 Directions 0-7: 0x0C

This register contains eight directions, for packets with a mismatch in bits 7..0 of the node-identifier. The direction in which a packet will be routed is goverened by the most significant mismatching bit.

0x0C: Directions 0-7

Bits	Perm	Init	Description
31:28	RW	0	The direction for packets whose first mismatching bit is 7.
27:24	RW	0	The direction for packets whose first mismatching bit is 6.
23:20	RW	0	The direction for packets whose first mismatching bit is 5.
19:16	RW	0	The direction for packets whose first mismatching bit is 4.
15:12	RW	0	The direction for packets whose first mismatching bit is 3.
11:8	RW	0	The direction for packets whose first mismatching bit is 2.
7:4	RW	0	The direction for packets whose first mismatching bit is 1.
3:0	RW	0	The direction for packets whose first mismatching bit is 0.

D.9 Directions 8-15: 0x0D

This register contains eight directions, for packets with a mismatch in bits 15..8 of the node-identifier. The direction in which a packet will be routed is goverened by the most significant mismatching bit.

0x0D: Directions 8-15

Bits	Perm	Init	Description
31:28	RW	0	The direction for packets whose first mismatching bit is 15.
27:24	RW	0	The direction for packets whose first mismatching bit is 14.
23:20	RW	0	The direction for packets whose first mismatching bit is 13.
19:16	RW	0	The direction for packets whose first mismatching bit is 12.
15:12	RW	0	The direction for packets whose first mismatching bit is 11.
11:8	RW	0	The direction for packets whose first mismatching bit is 10.
7:4	RW	0	The direction for packets whose first mismatching bit is 9.
3:0	RW	0	The direction for packets whose first mismatching bit is 8.

D.10 DEBUG_N configuration: 0x10

Configures the behavior of the DEBUG_N pin.

0x10: DEBUG_N configuration

Bits	Perm	Init	Description
31:2	RO	-	Reserved
1	RW	0	Set to 1 to enable signals on DEBUG_N to generate DCALL on the core.
0	RW	0	When set to 1, the DEBUG_N wire will be pulled down when the node enters debug mode.

D.11 Debug source: 0x1F

Contains the source of the most recent debug event.

Bits	Perm	Init	Description
31:5	RO	-	Reserved
4	RW		If set, the external DEBUG_N pin is the source of the most recent debug interrupt.
3:1	RO	-	Reserved
0	RW		If set, the xCORE Tile is the source of the most recent debug interrupt.

0x1F: Debug source

D.12 Link status, direction, and network: 0x20 .. 0x27

These registers contain status information for low level debugging (read-only), the network number that each link belongs to, and the direction that each link is part of.

Bits	Perm	Init	Description
31:26	RO	-	Reserved
25:24	RO		If this link is currently routing data into the switch, this field specifies the type of link that the data is routed to: 0: external link 1: plink 2: internal control link
23:16	RO	0	If the link is routing data into the switch, this field specifies the destination link number to which all tokens are sent.
15:12	RO	-	Reserved
11:8	RW	0	The direction that this this link is associated with; set for routing.
7:6	RO	-	Reserved
5:4	RW	0	Determines the network to which this link belongs, set for quality of service.
3	RO	-	Reserved
2	RO	0	Set to 1 if the current packet is junk and being thrown away. A packet is considered junk if, for example, it is not routable.
1	RO	0	Set to 1 if the switch is routing data into the link, and if a route exists from another link.
0	RO	0	Set to 1 if the link is routing data into the switch, and if a route is created to another link on the switch.

0x20 .. 0x27: Link status, direction, and network

D.13 PLink status and network: 0x40 .. 0x43

These registers contain status information and the network number that each processor-link belongs to.

0x40 .. 0x43: PLink status and network

Bits	Perm	Init	Description
31:26	RO	-	Reserved
25:24	RO		If this link is currently routing data into the switch, this field specifies the type of link that the data is routed to: 0: external link 1: plink 2: internal control link
23:16	RO	0	If the link is routing data into the switch, this field specifies the destination link number to which all tokens are sent.
15:6	RO	-	Reserved
5:4	RW	0	Determines the network to which this link belongs, set for quality of service.
3	RO	-	Reserved
2	RO	0	Set to 1 if the current packet is junk and being thrown away. A packet is considered junk if, for example, it is not routable.
1	RO	0	Set to 1 if the switch is routing data into the link, and if a route exists from another link.
0	RO	0	Set to 1 if the link is routing data into the switch, and if a route is created to another link on the switch.

D.14 Link configuration and initialization: 0x80 .. 0x87

These registers contain configuration and debugging information specific to external links. The link speed and width can be set, the link can be initialized, and the link status can be monitored.

0x80 .. 0x87: Link configuration and initialization

Bits	Perm	Init	Description
31	RW	0	Write '1' to this bit to enable the link, write '0' to disable it. This bit controls the muxing of ports with overlapping links.
30	RW	0	Set to 0 to operate in 2 wire mode or 1 to operate in 5 wire mode
29:28	RO	-	Reserved
27	RO	0	Set to 1 on error: an RX buffer overflow or illegal token encoding has been received. This bit clears on reading.
26	RO	0	1 if this end of the link has issued credit to allow the remote end to transmit.
25	RO	0	1 if this end of the link has credits to allow it to transmit.
24	WO	0	Set to 1 to initialize a half-duplex link. This clears this end of the link's credit and issues a HELLO token; the other side of the link will reply with credits. This bit is self-clearing.
23	WO	0	Set to 1 to reset the receiver. The next symbol that is detected will be assumed to be the first symbol in a token. This bit is self-clearing.
22	RO	-	Reserved
21:11	RW	0	The number of system clocks between two subsequent transitions within a token
10:0	RW	0	The number of system clocks between two subsequent transmit tokens.

D.15 Static link configuration: 0xA0 .. 0xA7

These registers are used for static (ie, non-routed) links. When a link is made static, all traffic is forwarded to the designated channel end and no routing is attempted.

0xA0 .. 0xA7: Static link configuration

Bits	Perm	Init	Description
31	RW	0	Enable static forwarding.
30:5	RO	-	Reserved
4:0	RW	0	The destination channel end on this node that packets received in static mode are forwarded to.

E XMOS USB Interface

XMOS provides a low-level USB interface for connecting the device to a USB transceiver using the UTMI+ Low Pin Interface (ULPI). The ULPI signals must be connected to the pins named in Figure 29. Note also that some ports on the same tile are used internally and are not available for use when the USB driver is active (they are available otherwise).

Pin	Signal
X*n*D02	Unavailable when USB active
X*n*D03	
X*n*D04	
X*n*D05	
X*n*D06	
X*n*D07	
X*n*D08	
X*n*D09	

Pin	Signal
X*n*D12	ULPI_STP
X*n*D13	ULPI_NXT
X*n*D14	ULPI_DATA[0]
X*n*D15	ULPI_DATA[1]
X*n*D16	ULPI_DATA[2]
X*n*D17	ULPI_DATA[3]
X*n*D18	ULPI_DATA[4]
X*n*D19	ULPI_DATA[5]
X*n*D20	ULPI_DATA[6]
X*n*D21	ULPI_DATA[7]
X*n*D22	ULPI_DIR
X*n*D23	ULPI_CLK

Pin	Signal
X*n*D26	Unavailable when USB active
X*n*D27	
X*n*D28	
X*n*D29	
X*n*D30	
X*n*D31	
X*n*D32	
X*n*D33	

X*n*D37	Unavailable when USB active
X*n*D38	
X*n*D39	
X*n*D40	
X*n*D41	
X*n*D42	
X*n*D43	

Figure 29: ULPI signals provided by the XMOS USB driver

F Device Errata

This section describes minor operational differences from the data sheet and recommended workarounds. As device and documentation issues become known, this section will be updated the document revised.

To guarantee a logic low is seen on the pins RST_N, DEBUG_N, MODE[4:0], TRST_N, TMS, TCK and TDI, the driving circuit should present an impedance of less than 100 Ω to ground. Usually this is not a problem for CMOS drivers driving single inputs. If one or more of these inputs are placed in parallel, however, additional logic buffers may be required to guarantee correct operation.

For static inputs tied high or low, the relevant input pin should be tied directly to GND or VDDIO.

G Associated Design Documentation

Document Title	Information	Document Number
XS1-L Hardware Design Checklist	Board design checklist	X6277
Estimating Power Consumption For XS1-L Devices	Power consumption	X4271
Programming XC on XMOS Devices	Timers, ports, clocks, cores and channels	X9577
xTIMEcomposer User Guide	Compilers, assembler and linker/mapper Timing analyzer, xScope, debugger Flash and OTP programming utilities	X3766

H Related Documentation

Document Title	Information	Document Number
The XMOS XS1 Architecture	ISA manual	X7879
XS1 Port I/O Timing	Port timings	X5821
XS1-L System Specification	Link, switch and system information	X1151
XS1-L Link Performance and Design Guidelines	Link timings	X2999
XS1-L Clock Frequency Control	Advanced clock control	X1433
XS1-L Active Power Conservation	Low-power mode during idle	X7411

I Revision History

Date	Description
2013-01-30	New datasheet - revised part numbering
2013-02-26	New multicore microcontroller introduction Moved configuration sections to appendices