
Chipsmall
深圳市津利帝科技有限公司
Geelead Technology Limited
Chipsmall
采购部

XS1-L12A-128-QF124 Datasheet

2015/04/14

Document Number: X7893,

Table of Contents

TO OUR VALUED CUSTOMERS

It is our intention to provide you with accurate and comprehensive documentation for the hardware and software components used in this product. To subscribe to receive updates, visit http://www.xmos.com/.

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 and communicate between tasks using a high speed network. Because xCORE multicore microcontrollers are completely deterministic, you can write software to implement functions that traditionally require dedicated hardware.

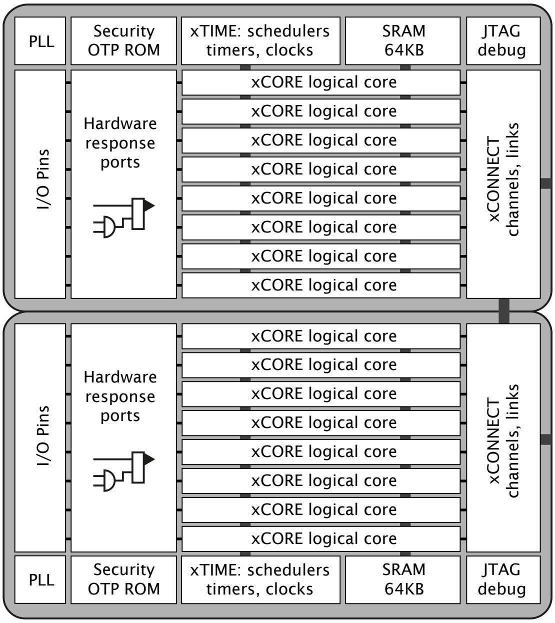


Figure 1: XS1-L Series: 4-16 core devices

Key features of the XS1-L12A-128-QF124 include:

- **Tiles**: Devices consist of one or more xCORE tiles. Each tile contains between four and eight 32-bit xCOREs with highly integrated I/O and on-chip memory.
- **Logical cores** Each logical core can execute tasks such as computational code, DSP code, control software (including logic decisions and executing a state machine) or software that handles I/O. Section 5.1
- **xTIME scheduler** The xTIME scheduler performs functions similar to an RTOS, in hardware. It services and synchronizes events in a core, so there is no requirement for interrupt handler routines. The xTIME scheduler triggers cores on events generated by hardware resources such as the I/O pins, communication channels and timers. Once triggered, a core runs independently and concurrently to other cores, until it pauses to wait for more events. Section 5.2

- **Channels and channel ends** Tasks running on logical cores communicate using channels formed between two channel ends. Data can be passed synchronously or asynchronously between the channel ends assigned to the communicating tasks. Section 5.5
- **xCONNECT Switch and Links** Between tiles, channel communications are implemented over a high performance network of xCONNECT Links and routed through a hardware xCONNECT Switch. Section 5.6
- **Ports** The I/O pins are connected to the processing cores by Hardware Response ports. The port logic can drive its pins high and low, or it can sample the value on its pins optionally waiting for a particular condition. Section 5.3
- **Clock blocks** xCORE devices include a set of programmable clock blocks that can be used to govern the rate at which ports execute. Section 5.4
- **Memory** Each xCORE Tile integrates a bank of SRAM for instructions and data, and a block of one-time programmable (OTP) memory that can be configured for system wide security features. Section 8
- **PLL** The PLL is used to create a high-speed processor clock given a low speed external oscillator. Section 6
- **JTAG** The JTAG module can be used for loading programs, boundary scan testing, in-circuit source-level debugging and programming the OTP memory. Section 9

1.1 Software

Devices are programmed using C, C++ or xC (C with multicore extensions). XMOS provides tested and proven software libraries, which allow you to quickly add interface and processor functionality such as USB, Ethernet, PWM, graphics driver, and audio EQ to your applications.

1.2 xTIMEcomposer Studio

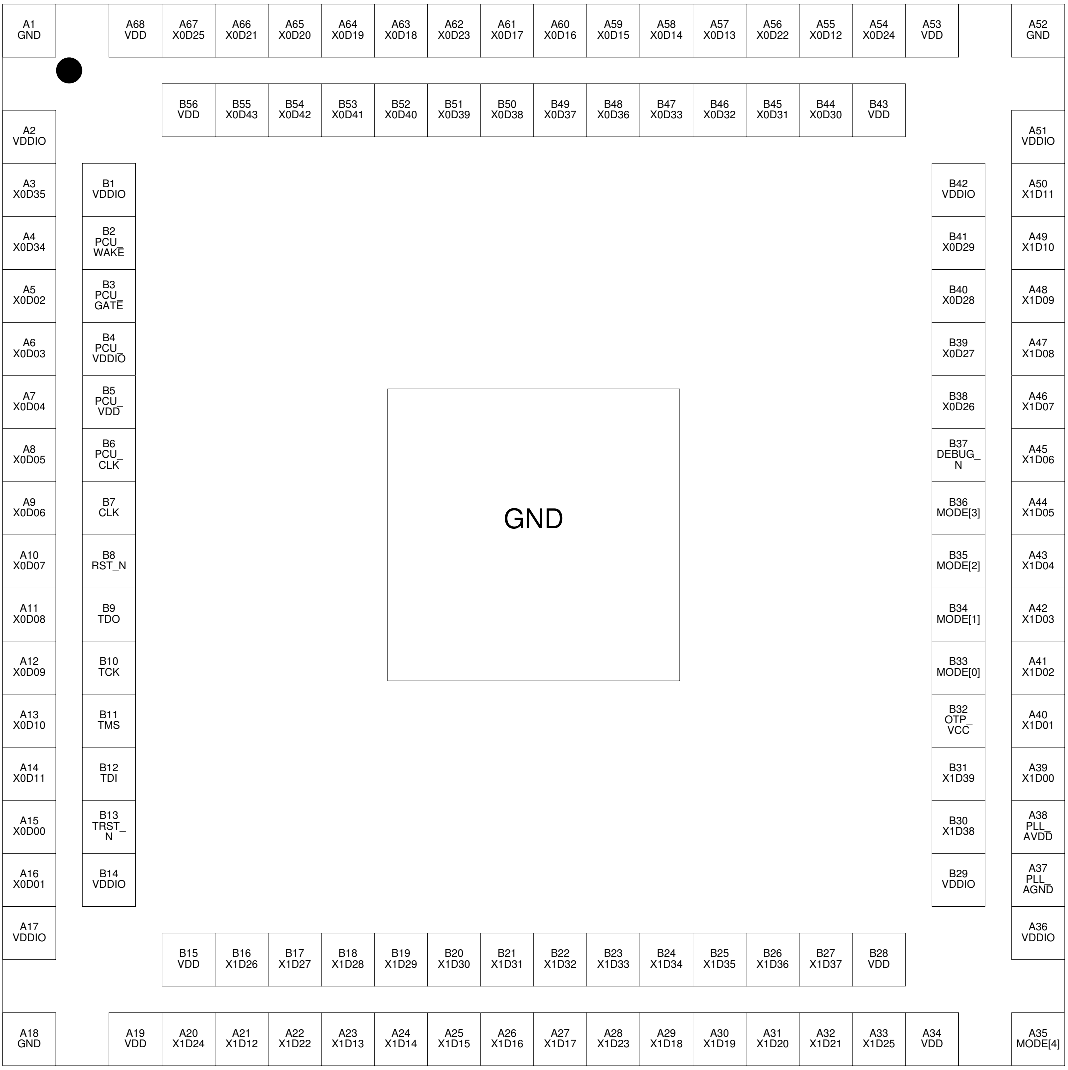
The xTIMEcomposer Studio development environment provides all the tools you need to write and debug your programs, profile your application, and write images into flash memory or OTP memory on the device. Because xCORE devices operate deterministically, they can be simulated like hardware within xTIMEcomposer: uniquely in the embedded world, xTIMEcomposer Studio therefore includes a static timing analyzer, cycle-accurate simulator, and high-speed in-circuit instrumentation.

xTIMEcomposer can be driven from either a graphical development environment, or the command line. The tools 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 the xTIMEcomposer User Guide, X3766.

2 XS1-L12A-128-QF124 Features

- **Multicore Microcontroller with Advanced Multi-Core RISC Architecture**
  - 12 real-time logical cores on 2 xCORE tiles
  - Cores share up to 500 MIPS
  - Each logical core has:
    - Guaranteed throughput of between 1/4 and 1/6 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
    - Up to 32 x 1bit port, 12 x 4bit port, 7 x 8bit port, 3 x 16bit port
    - 4 xCONNECT links
  - 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
- **Hardware resources**
  - 12 clock blocks (6 per tile)
  - 20 timers (10 per tile)
  - 8 locks (4 per tile)
- **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
- **124-pin QF124 package 0.5 mm pitch**

3 Pin Configuration



4 Signal Description

This section lists the signals and I/O pins available on the XS1-L12A-128-QF124. The device provides a combination of 1bit, 4bit, 8bit and 16bit ports, as well as wider ports that are fully or partially (gray) bonded out. All pins of a port provide either output or input, but signals in different directions cannot be mapped onto the same port.

Pins may have one or more of the following properties:

- PD/PU: The IO pin a weak pull-down or pull-up resistor. On GPIO pins this resistor can be enabled.
- ST: The IO pin has a Schmitt Trigger on its input.

Power pins (6)			
Signal	**Function**	**Type**	**Properties**
GND	Digital ground	GND	
OTP_VCC	OTP power supply	PWR	
PLL_AGND	Analog ground for PLL	GND	
PLL_AVDD	Analog PLL power	PWR	
VDD	Digital tile power	PWR	
VDDIO	Digital I/O power	PWR	

Clocks pins (2)			
Signal	**Function**	**Type**	**Properties**
CLK	PLL reference clock	Input	PD, ST
MODE[4:0]	Boot mode select	Input	PU, ST

JTAG pins (7)			
Signal	**Function**	**Type**	**Properties**
DEBUG_N	Multi-chip debug	I/O	PU
RST_N	Global reset input	Input	PU, ST
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	PU, ST

I/O pins (84)			
Signal	**Function**	**Type**	**Properties**
X0D00	$1A^0$	I/O	PD$_S$, R$_S$

(continued)

Signal	Function						Type	Properties
X0D01	$\mathrm{XLA}^{4}_{\mathrm{out}}$	1B^{0}					I/O	PD$_S$, R$_S$
X0D02	$\mathrm{XLA}^{3}_{\mathrm{out}}$		4A^{0}	8A^{0}	16A^{0}	32A^{20}	I/O	PD$_S$, R$_U$
X0D03	$\mathrm{XLA}^{2}_{\mathrm{out}}$		4A^{1}	8A^{1}	16A^{1}	32A^{21}	I/O	PD$_S$, R$_U$
X0D04	$\mathrm{XLA}^{1}_{\mathrm{out}}$		4B^{0}	8A^{2}	16A^{2}	32A^{22}	I/O	PD$_S$, R$_U$
X0D05	$\mathrm{XLA}^{0}_{\mathrm{out}}$		4B^{1}	8A^{3}	16A^{3}	32A^{23}	I/O	PD$_S$, R$_U$
X0D06	$\mathrm{XLA}^{0}_{\mathrm{in}}$		4B^{2}	8A^{4}	16A^{4}	32A^{24}	I/O	PD$_S$, R$_U$
X0D07	$\mathrm{XLA}^{1}_{\mathrm{in}}$		4B^{3}	8A^{5}	16A^{5}	32A^{25}	I/O	PD$_S$, R$_U$
X0D08	$\mathrm{XLA}^{2}_{\mathrm{in}}$		4A^{2}	8A^{6}	16A^{6}	32A^{26}	I/O	PD$_S$, R$_U$
X0D09	$\mathrm{XLA}^{3}_{\mathrm{in}}$		4A^{3}	8A^{7}	16A^{7}	32A^{27}	I/O	PD$_S$, R$_U$
X0D10	$\mathrm{XLA}^{4}_{\mathrm{in}}$	1C^{0}					I/O	PD$_S$, R$_S$
X0D11		1D^{0}					I/O	PD$_S$, R$_S$
X0D12		1E^{0}					I/O	PD$_S$, R$_U$
X0D13	$\mathrm{XLB}^{4}_{\mathrm{out}}$	1F^{0}					I/O	PD$_S$, R$_U$
X0D14	$\mathrm{XLB}^{3}_{\mathrm{out}}$		4C^{0}	8B^{0}	16A^{8}	32A^{28}	I/O	PD$_S$, R$_U$
X0D15	$\mathrm{XLB}^{2}_{\mathrm{out}}$		4C^{1}	8B^{1}	16A^{9}	32A^{29}	I/O	PD$_S$, R$_U$
X0D16	$\mathrm{XLB}^{1}_{\mathrm{out}}$		4D^{0}	8B^{2}	16A^{10}		I/O	PD$_S$, R$_U$
X0D17	$\mathrm{XLB}^{0}_{\mathrm{out}}$		4D^{1}	8B^{3}	16A^{11}		I/O	PD$_S$, R$_U$
X0D18	$\mathrm{XLB}^{0}_{\mathrm{in}}$		4D^{2}	8B^{4}	16A^{12}		I/O	PD$_S$, R$_U$
X0D19	$\mathrm{XLB}^{1}_{\mathrm{in}}$		4D^{3}	8B^{5}	16A^{13}		I/O	PD$_S$, R$_U$
X0D20	$\mathrm{XLB}^{2}_{\mathrm{in}}$		4C^{2}	8B^{6}	16A^{14}	32A^{30}	I/O	PD$_S$, R$_U$
X0D21	$\mathrm{XLB}^{3}_{\mathrm{in}}$		4C^{3}	8B^{7}	16A^{15}	32A^{31}	I/O	PD$_S$, R$_U$
X0D22	$\mathrm{XLB}^{4}_{\mathrm{in}}$	1G^{0}					I/O	PD$_S$, R$_U$
X0D23		1H^{0}					I/O	PD$_S$, R$_U$
X0D24		1I^{0}					I/O	PD$_S$
X0D25		1J^{0}					I/O	PD$_S$
X0D26			4E^{0}	8C^{0}	16B^{0}		I/O	PD$_S$, R$_U$
X0D27			4E^{1}	8C^{1}	16B^{1}		I/O	PD$_S$, R$_U$
X0D28			4F^{0}	8C^{2}	16B^{2}		I/O	PD$_S$, R$_U$
X0D29			4F^{1}	8C^{3}	16B^{3}		I/O	PD$_S$, R$_U$
X0D30			4F^{2}	8C^{4}	16B^{4}		I/O	PD$_S$, R$_U$
X0D31			4F^{3}	8C^{5}	16B^{5}		I/O	PD$_S$, R$_U$
X0D32			4E^{2}	8C^{6}	16B^{6}		I/O	PD$_S$, R$_U$
X0D33			4E^{3}	8C^{7}	16B^{7}		I/O	PD$_S$, R$_U$
X0D34		1K^{0}					I/O	PD$_S$
X0D35		1L^{0}					I/O	PD$_S$
X0D36		1M^{0}		8D^{0}	16B^{8}		I/O	PD$_S$
X0D37		1N^{0}		8D^{1}	16B^{9}		I/O	PD$_S$, R$_U$
X0D38		1O^{0}		8D^{2}	16B^{10}		I/O	PD$_S$, R$_U$
X0D39		1P^{0}		8D^{3}	16B^{11}		I/O	PD$_S$, R$_U$
X0D40				8D^{4}	16B^{12}		I/O	PD$_S$, R$_U$
X0D41				8D^{5}	16B^{13}		I/O	PD$_S$, R$_U$
X0D42				8D^{6}	16B^{14}		I/O	PD$_S$, R$_U$
X0D43				8D^{7}	16B^{15}		I/O	PU$_S$, R$_U$

(continued)

Signal	Function						Type	Properties
X1D00		$1A^0$					I/O	PD_S, R_S
X1D01	XLA^4_{out}	$1B^0$					I/O	PD_S, R_S
X1D02	XLA^3_{out}		$4A^0$	$8A^0$	$16A^0$	$32A^{20}$	I/O	PD_S, R_U
X1D03	XLA^2_{out}		$4A^1$	$8A^1$	$16A^1$	$32A^{21}$	I/O	PD_S, R_U
X1D04	XLA^1_{out}		$4B^0$	$8A^2$	$16A^2$	$32A^{22}$	I/O	PD_S, R_U
X1D05	XLA^0_{out}		$4B^1$	$8A^3$	$16A^3$	$32A^{23}$	I/O	PD_S, R_U
X1D06	XLA^0_{in}		$4B^2$	$8A^4$	$16A^4$	$32A^{24}$	I/O	PD_S, R_U
X1D07	XLA^1_{in}		$4B^3$	$8A^5$	$16A^5$	$32A^{25}$	I/O	PD_S, R_U
X1D08	XLA^2_{in}		$4A^2$	$8A^6$	$16A^6$	$32A^{26}$	I/O	PD_S, R_U
X1D09	XLA^3_{in}		$4A^3$	$8A^7$	$16A^7$	$32A^{27}$	I/O	PD_S, R_U
X1D10	XLA^4_{in}	$1C^0$					I/O	PD_S, R_S
X1D11		$1D^0$					I/O	PD_S, R_S
X1D12		$1E^0$					I/O	PD_S, R_U
X1D13	XLB^4_{out}	$1F^0$					I/O	PD_S, R_U
X1D14	XLB^3_{out}		$4C^0$	$8B^0$	$16A^8$	$32A^{28}$	I/O	PD_S, R_U
X1D15	XLB^2_{out}		$4C^1$	$8B^1$	$16A^9$	$32A^{29}$	I/O	PD_S, R_U
X1D16	XLB^1_{out}		$4D^0$	$8B^2$	$16A^{10}$		I/O	PD_S, R_U
X1D17	XLB^0_{out}		$4D^1$	$8B^3$	$16A^{11}$		I/O	PD_S, R_U
X1D18	XLB^0_{in}		$4D^2$	$8B^4$	$16A^{12}$		I/O	PD_S, R_U
X1D19	XLB^1_{in}		$4D^3$	$8B^5$	$16A^{13}$		I/O	PD_S, R_U
X1D20	XLB^2_{in}		$4C^2$	$8B^6$	$16A^{14}$	$32A^{30}$	I/O	PD_S, R_U
X1D21	XLB^3_{in}		$4C^3$	$8B^7$	$16A^{15}$	$32A^{31}$	I/O	PD_S, R_U
X1D22	XLB^4_{in}	$1G^0$					I/O	PD_S, R_U
X1D23		$1H^0$					I/O	PD_S, R_U
X1D24		$1I^0$					I/O	PD_S
X1D25		$1J^0$					I/O	PD_S
X1D26			$4E^0$	$8C^0$	$16B^0$		I/O	PD_S, R_U
X1D27			$4E^1$	$8C^1$	$16B^1$		I/O	PD_S, R_U
X1D28			$4F^0$	$8C^2$	$16B^2$		I/O	PD_S, R_U
X1D29			$4F^1$	$8C^3$	$16B^3$		I/O	PD_S, R_U
X1D30			$4F^2$	$8C^4$	$16B^4$		I/O	PD_S, R_U
X1D31			$4F^3$	$8C^5$	$16B^5$		I/O	PD_S, R_U
X1D32			$4E^2$	$8C^6$	$16B^6$		I/O	PD_S, R_U
X1D33			$4E^3$	$8C^7$	$16B^7$		I/O	PD_S, R_U
X1D34		$1K^0$					I/O	PD_S
X1D35		$1L^0$					I/O	PD_S
X1D36		$1M^0$		$8D^0$	$16B^8$		I/O	PD_S
X1D37		$1N^0$		$8D^1$	$16B^9$		I/O	PD_S, R_U
X1D38		$1O^0$		$8D^2$	$16B^{10}$		I/O	PD_S, R_U
X1D39		$1P^0$		$8D^3$	$16B^{11}$		I/O	PD_S, R_U

pins (5)

Signal	Function	Type	Properties
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 XS1-L12A-128-QF124 is a powerful device that consists of two xCORE Tiles, each comprising a flexible logical processing cores with tightly integrated I/O and on-chip memory.

5.1 Logical cores

Each tile has 6 active logical cores, which issue instructions down a shared four-stage pipeline. Instructions from the active cores are issued round-robin. If up to four logical cores are active, each core is allocated a quarter of the processing cycles. If more than four logical cores are active, each core is allocated at least $^{1}/_{n}$ cycles (for n cores). Figure 2 shows the guaranteed core performance depending on the number of cores used.

Figure 2: Logical core performance

Speed grade	MIPS	Frequency	Minimum MIPS per core (for *n* cores)							
			1	2	3	4	5	6		
8	800 MIPS	400 MHz	100	100	100	100	80	67		
10	1000 MIPS	500 MHz	125	125	125	125	100	83		

There is no way that the performance of a logical core can be reduced below these predicted levels. Because cores may be delayed on I/O, however, their unused processing cycles can be taken by other cores. This means that for more than four logical cores, the performance of each core is often higher than the predicted minimum but cannot be guaranteed.

The logical cores are triggered by events instead of interrupts and run to completion. A logical core can be paused to wait for an event.

5.2 xTIME scheduler

The xTIME scheduler handles the events generated by xCORE Tile resources, such as channel ends, timers and I/O pins. It ensures that all events are serviced and synchronized, without the need for an RTOS. Events that occur at the I/O pins are handled by the Hardware-Response ports and fed directly to the appropriate xCORE Tile. An xCORE Tile can also choose to wait for a specified time to elapse, or for data to become available on a channel.

Tasks do not need to be prioritised as each of them runs on their own logical xCORE. It is possible to share a set of low priority tasks on a single core using cooperative multitasking.

5.3 Hardware Response Ports

Hardware Response ports connect an xCORE tile to one or more physical pins and as such define the interface between hardware attached to the XS1-L12A-128-QF124, and the software running on it. A combination of 1bit, 4bit, 8bit, 16bit and 32bit ports are available. All pins of a port provide either output or input. Signals in different directions cannot be mapped onto the same port.

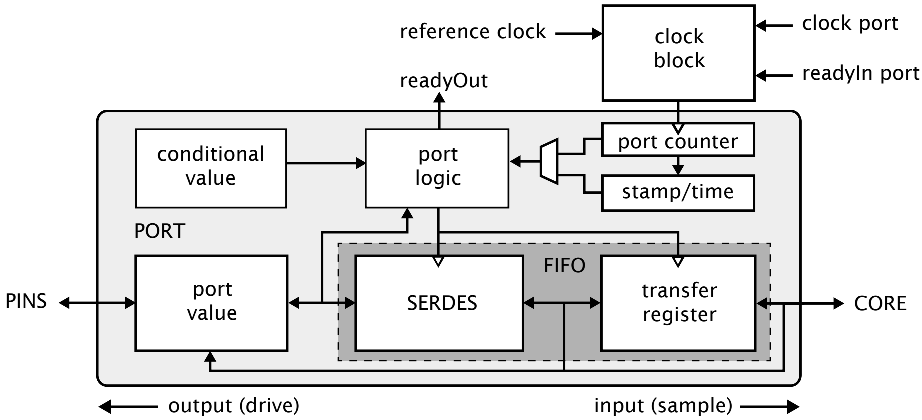


Figure 3: Port block diagram

The port logic can drive its pins high or low, or it can sample the value on its pins, optionally waiting for a particular condition. Ports are accessed using dedicated instructions that are executed in a single processor cycle.

Data is transferred between the pins and core using a FIFO that comprises a SERDES and transfer register, providing options for serialization and buffered data.

Each port has a 16-bit counter that can be used to control the time at which data is transferred between the port value and transfer register. The counter values can be obtained at any time to find out when data was obtained, or used to delay I/O until some time in the future. The port counter value is automatically saved as a timestamp, that can be used to provide precise control of response times.

The ports and xCONNECT links are multiplexed onto the physical pins. 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 Clock blocks

xCORE devices include a set of programmable clocks called clock blocks that can be used to govern the rate at which ports execute. Each xCORE tile has six clock blocks: the first clock block provides the tile reference clock and runs at a default frequency of 100MHz; the remaining clock blocks can be set to run at different frequencies.

A clock block can use a 1-bit port as its clock source allowing external application clocks to be used to drive the input and output interfaces.

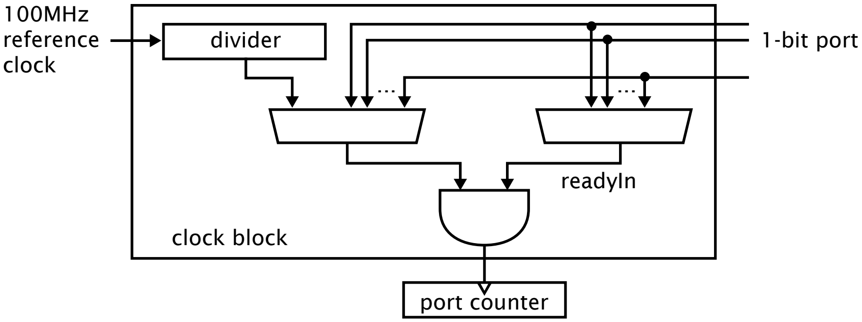


Figure 4: Clock block diagram

In many cases I/O signals are accompanied by strobing signals. The xCORE ports can input and interpret strobe (known as readyIn and readyOut) signals generated by external sources, and ports can generate strobe signals to accompany output data.

On reset, each port is connected to clock block 0, which runs from the xCORE Tile reference clock.

5.5 Channels and Channel Ends

Logical cores communicate using point-to-point connections, formed between two channel ends. A channel-end is a resource on an xCORE tile, that is allocated by the program. Each channel-end has a unique system-wide identifier that comprises a unique number and their tile identifier. Data is transmitted to a channel-end by an output-instruction; and the other side executes an input-instruction. Data can be passed synchronously or asynchronously between the channel ends.

5.6 xCONNECT Switch and Links

XMOS devices provide a scalable architecture, where multiple xCORE devices can be connected together to form one system. Each xCORE device has an xCONNECT interconnect that provides a communication infrastructure for all tasks that run on the various xCORE tiles on the system.

The interconnect relies on a collection of switches and XMOS links. Each xCORE device has an on-chip switch that can set up circuits or route data. The switches are connected by xConnect Links. An XMOS link provides a physical connection between two switches. The switch has a routing algorithm that supports many different topologies, including lines, meshes, trees, and hypercubes.

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.

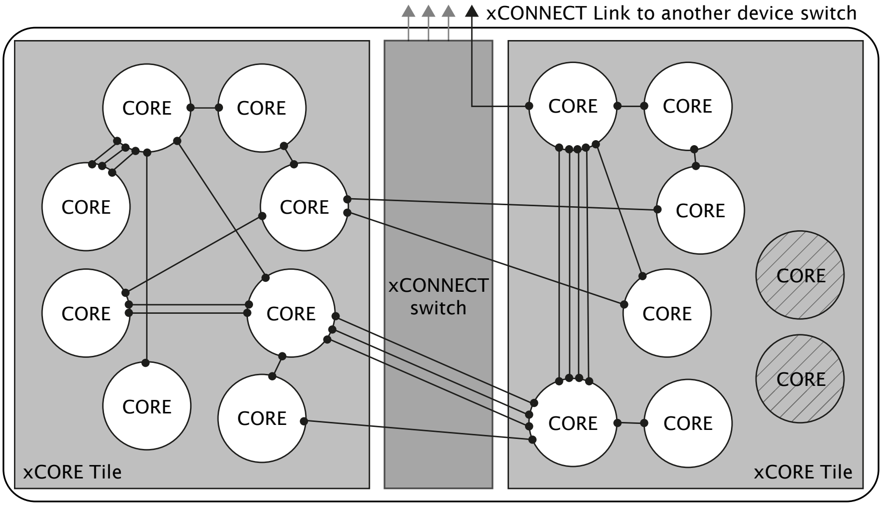


Figure 5: Switch, links and channel ends

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.

6 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 6:

Oscillator Frequency	**MODE**		**Tile Frequency**	**PLL Ratio**	**PLL settings**		
	1	**0**			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 6: PLL multiplier values and MODE pins

Figure 6 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 during and after deassertion of the system reset.

If a different tile frequency is required (eg, 500 MHz), then the PLL must be reprogrammed after boot to provide the required tile frequency. The XMOS tools perform this operation by default. Further details on configuring the clock can be found in the XS1-L Clock Frequency Control document, X1433.

7 Boot Procedure

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 7. In normal usage, MODE[4:2] controls the boot source according to the table in Figure 8. If bit 5 of the security register (*see* §8.1) is set, the device boots from OTP.

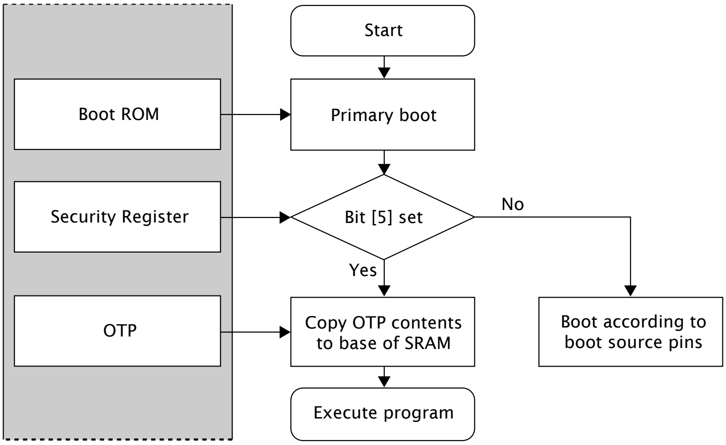


Figure 7: Boot procedure

The boot image has the following format:

- A 32-bit program size s in words.
- Program consisting of $s \times 4$ bytes.
- A 32-bit CRC, or the value 0x0D15AB1E to indicate that no CRC check should be performed.

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	Tile0 boots from link B, Tile1 from channel end 0 via Tile0
0	1	1	Tile0 boots from SPI, Tile1 from channel end 0 via Tile0
1	1	0	Tile0 and Tile1 independently enable link B and internal links (E, F, G, H), and boot from channel end 0
1	1	1	Tile0 and Tile 1 boot from SPI independently

Figure 8: Boot source pins

The program size and CRC are stored least significant byte first. The program is loaded into the lowest memory address of RAM, and the program is started from that address. The CRC is calculated over the byte stream represented by the program size and the program itself. The polynomial used is 0xEDB88320 (IEEE 802.3); the CRC register is initialized with 0xFFFFFFFF and the residue is inverted to produce the CRC.

7.1 Boot from SPI master

If set to boot from SPI master, the processor enables the four pins specified in Figure 9, and drives the SPI clock at 2.5 MHz (assuming a 400 MHz core clock). A READ command is issued with a 24-bit address 0x000000. The clock polarity and phase are 0 / 0.

Pin	Signal	Description
X0D00	MISO	Master In Slave Out (Data)
X0D01	SS	Slave Select
X0D10	SCLK	Clock
X0D11	MOSI	Master Out Slave In (Data)

Figure 9: SPI master pins

The xCORE Tile expects each byte to be transferred with the *least-significant bit first*. Programmers who write bytes into an SPI interface using the most significant bit first may have to reverse the bits in each byte of the image stored in the SPI device.

If a large boot image is to be read in, it is faster to first load a small boot-loader that reads the large image using a faster SPI clock, for example 50 MHz or as fast as the flash device supports.

The pins used for SPI boot are hardcoded in the boot ROM and cannot be changed. If required, an SPI boot program can be burned into OTP that uses different pins.

7.2 Boot from xConnect Link

If set to boot from an xConnect Link, the processor enables Link B around 200 ns after the boot process starts. Enabling the Link switches off the pull-down on resistors X0D16..X0D19, drives X0D16 and X0D17 low (the initial state for the Link), and monitors pins X0D18 and X0D19 for boot-traffic. X0D18 and X0D19 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.

The boot-rom on the core will then:

1. Allocate channel-end 0.
2. Input a word on channel-end 0. It will use this word as a channel to acknowledge the boot. Provide the null-channel-end 0x0000FF02 if no acknowledgment is required.
3. Input the boot image specified above, including the CRC.
4. Input an END control token.
5. Output an END control token to the channel-end received in step 2.
6. Free channel-end 0.
7. Jump to the loaded code.

7.3 Boot from OTP

If an xCORE tile is set to use secure boot (see Figure 7), the boot image is read from address 0 of the OTP memory in the tile's security module.

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.

Each tile has its own individual OTP memory, and hence some tiles can be booted from OTP while others are booted from SPI or the channel interface. This enables systems to be partially programmed, dedicating one or more tiles to perform a particular function, leaving the other tiles user-programmable.

7.4 Security register

The security register enables security features on the xCORE tile. The features shown in Figure 10 provide a strong level of protection and are sufficient for providing strong IP security.

8 Memory

8.1 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

Feature	Bit	Description
Disable JTAG	0	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	1	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.
Secure Boot	5	The xCORE Tile is forced to boot from address 0 of the OTP, allowing the xCORE Tile boot ROM to be bypassed (*see* §7).
Redundant rows	7	Enables redundant rows in OTP.
Sector Lock 0	8	Disable programming of OTP sector 0.
Sector Lock 1	9	Disable programming of OTP sector 1.
Sector Lock 2	10	Disable programming of OTP sector 2.
Sector Lock 3	11	Disable programming of OTP sector 3.
OTP Master Lock	12	Disable OTP programming completely: disables updates to all sectors and security register.
Disable JTAG-OTP	13	Disable all (read & write) access from the JTAG interface to this OTP.
Disable Global Debug	14	Disables access to the DEBUG_N pin.
	21..15	General purpose software accessable security register available to end-users.
	31..22	General purpose user programmable JTAG UserID code extension.

Figure 10: Security register features

data in four sectors each containing 512 rows of 32 bits 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.

The OTP memory is programmed using three special I/O ports: the OTP address port is a 16-bit port with resource ID 0x100200, the OTP data is written via a 32-bit port with resource ID 0x200100, and the OTP control is on a 16-bit port with ID 0x100300. Programming is performed through `libotp` and `xburn`.

8.2 SRAM

Each xCORE Tile integrates a single 64KB 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.

9 JTAG

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

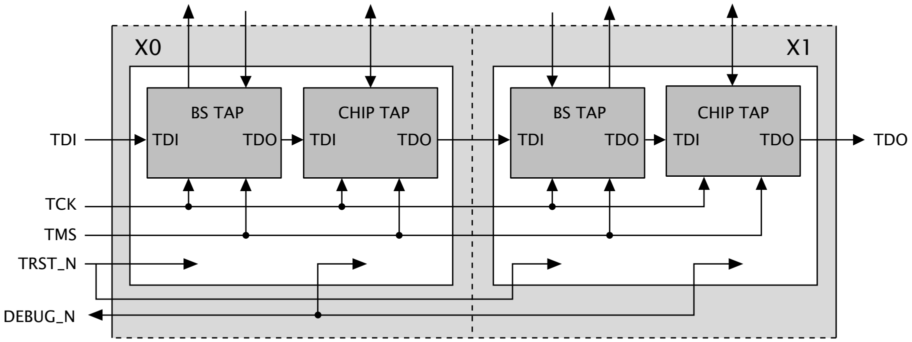


Figure 11: JTAG chain structure

The JTAG chain structure is illustrated in Figure 11. 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 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 12.

<table>
<tr><td colspan="32">Bit31 Device Identification Register 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>

Figure 12: IDCODE return value

The JTAG usercode register can be read by using the USERCODE instruction. Its contents are specified in Figure 13. The OTP User ID field is read from bits [22:31] of the security register on xCORE Tile 0, *see* §8.1 (all zero on unprogrammed devices).

Figure 13: USERCODE return value

<table>
<tr><td colspan="10">Bit31</td><td colspan="12">Usercode Register</td><td colspan="10">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>

9.1 PCU

PCU_WAKE should be left unconnected, PCU_GATE should be left unconnected and PCU_CLK must be tied to CLK.

10 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 and OTP_VCC 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* §7). RST_N and must be asserted low during and after power up for 100 ns.

10.1 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 optimized 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 14.

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 15.

10.2 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.

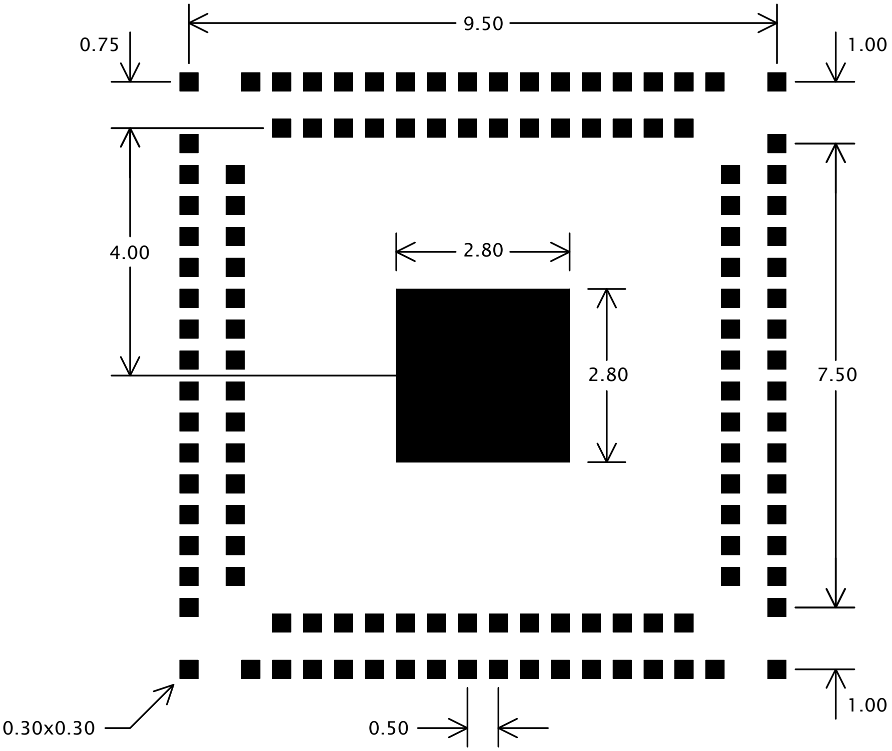


Figure 14: Example land pattern

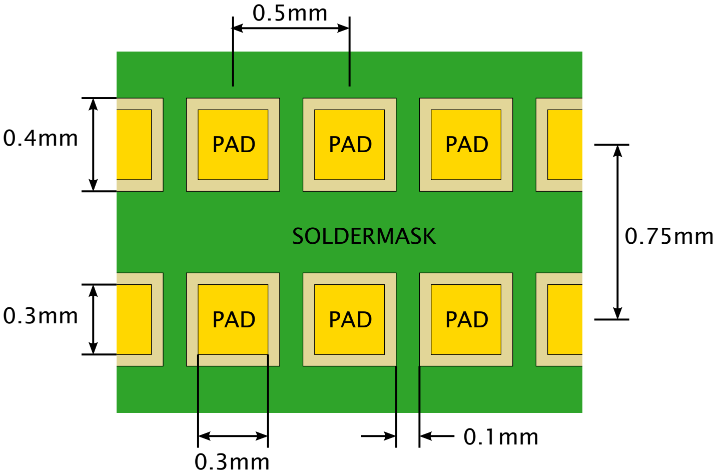


Figure 15: Detail of outer pads

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

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 16.

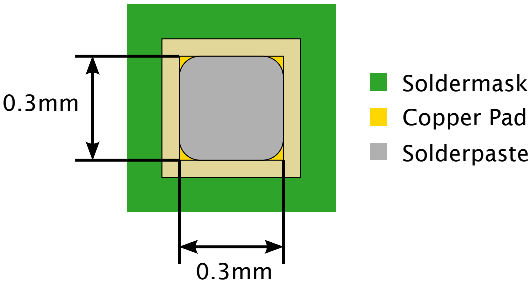


Figure 16: Solder stencil for outer pads

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 17. This gives a paste to pad area ratio of 51%.

10.3 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.

10.4 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.

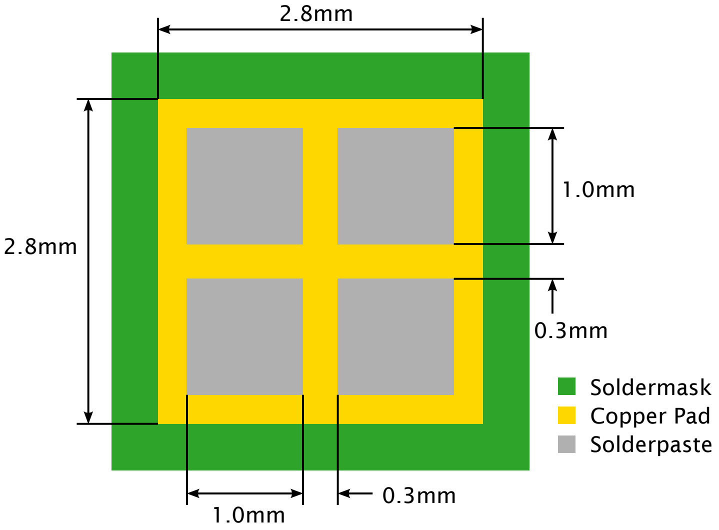


Figure 17: Solder stencil for centre pad

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.

11 DC and Switching Characteristics

11.1 Operating Conditions

Figure 18: 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	

11.2 DC Characteristics

Figure 19: 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	2.00			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

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.

11.3 ESD Stress Voltage

Figure 20: 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	