XU210-256-TQ128 Datasheet

2015/08/19 Document Number: X008621,

Table of Contents

1 xCORE Multicore Microcontrollers

The xCORE-200 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.

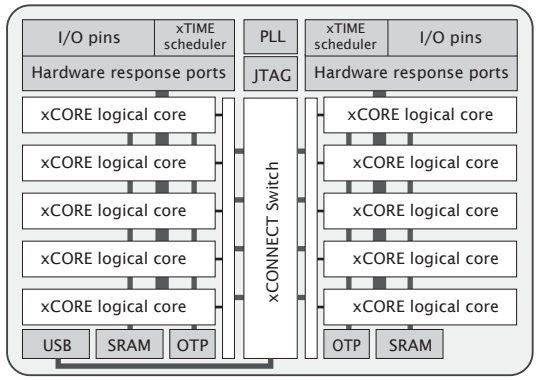


Figure 1: XU210-256-TQ128 block diagram

Key features of the XU210-256-TQ128 include:

- **Tiles**: Devices consist of one or more xCORE tiles. Each tile contains between five 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 6.1
- **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 6.2
- **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 6.5
- **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 6.6

- **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 6.3
- **Clock blocks** xCORE devices include a set of programmable clock blocks that can be used to govern the rate at which ports execute. Section 6.4
- **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 9
- **PLL** The PLL is used to create a high-speed processor clock given a low speed external oscillator. Section 7
- **USB** The USB PHY provides High-Speed and Full-Speed, device, host, and on-the-go functionality. Data is communicated through ports on the digital node. A library is provided to implement USB device functionality. Section 10
- **JTAG** The JTAG module can be used for loading programs, boundary scan testing, in-circuit source-level debugging and programming the OTP memory. Section 11

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

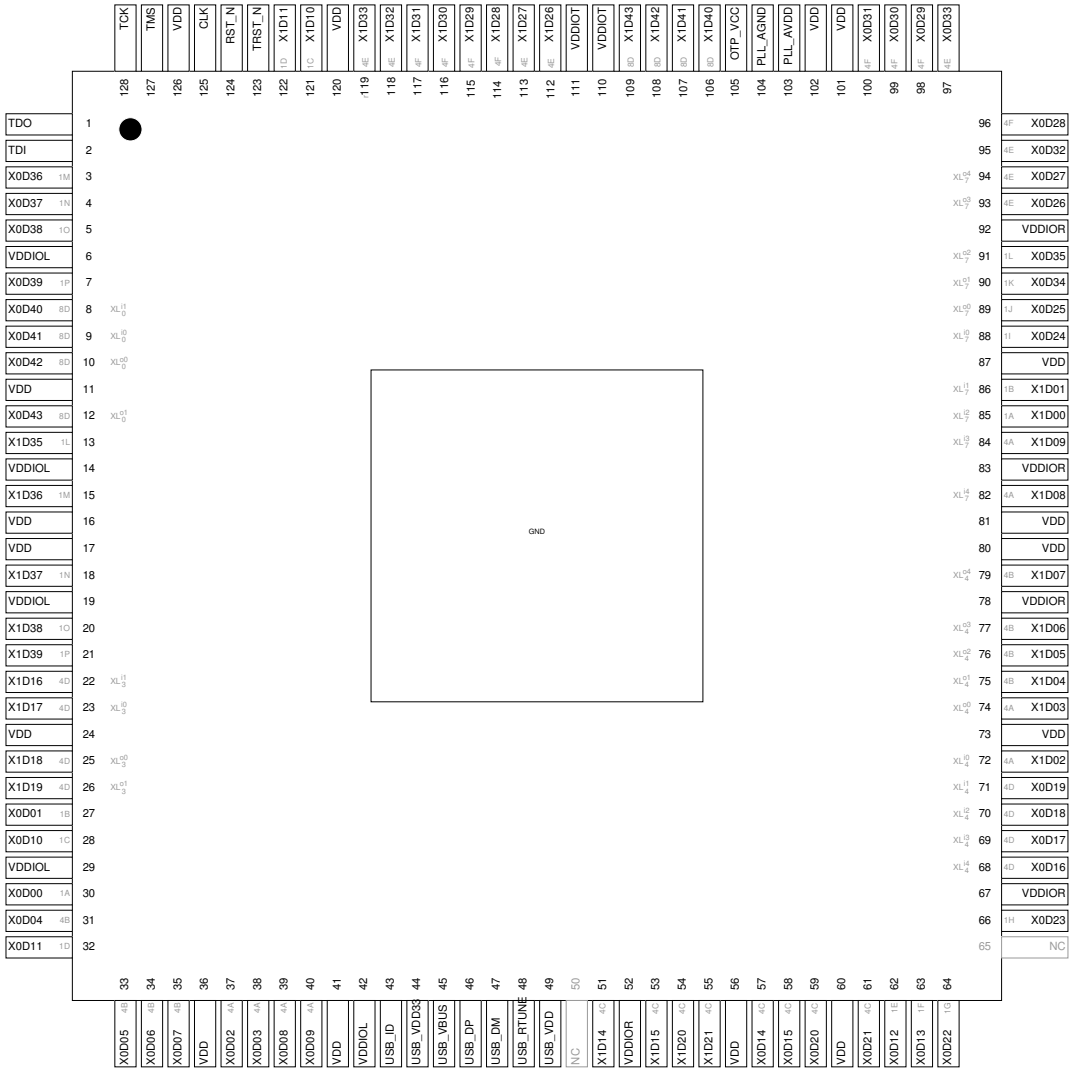
The xTIMEcomposer Studio development environment provides all the tools you need to write and debug your programs, profile your application, and write images into flash memory or OTP memory on the device. Because xCORE devices operate deterministically, they can be simulated like hardware within xTIMEcomposer: 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 XU210-256-TQ128 Features

- **Multicore Microcontroller with Advanced Multi-Core RISC Architecture**
  - 10 real-time logical cores on 2 xCORE tiles
  - Cores share up to 1000 MIPS
    - Up to 2000 MIPS in dual issue mode
  - Each logical core has:
    - Guaranteed throughput of between 1/5 and 1/5 of tile MIPS
    - 16x32bit dedicated registers
  - 167 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
- **USB PHY, fully compliant with USB 2.0 specification**
- **Programmable I/O**
  - 81 general-purpose I/O pins, configurable as input or output
    - Up to 25 x 1bit port, 12 x 4bit port, 8 x 8bit port, 4 x 16bit port
    - 4 xCONNECT links
  - Port sampling rates of up to 60 MHz with respect to an external clock
  - 32 channel ends for communication with other cores, on or off-chip
- **Memory**
  - 256KB internal single-cycle SRAM (max 128KB per tile) for code and data storage
  - 16KB 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**
  - 20: 1000 MIPS
- **Power Consumption**
  - 570 mA (typical)
- **128-pin TQFP package 0.4 mm pitch**

3 Pin Configuration



4 Signal Description

This section lists the signals and I/O pins available on the XU210-256-TQ128. 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.
- IOL/IOT/IOR: The IO pin is powered from VDDIOL, VDDIOT, and VDDIOR respectively

Power pins (10)			
Signal	**Function**	**Type**	**Properties**
GND	Digital ground	GND	
OTP_VCC	OTP power supply	PWR	
PLL_AGND	Analog ground for PLL	PWR	
PLL_AVDD	Analog PLL power	PWR	
USB_VDD	Digital tile power	PWR	
USB_VDD33	USB Analog power	PWR	
VDD	Digital tile power	PWR	
VDDIOL	Digital I/O power (left)	PWR	
VDDIOR	Digital I/O power (right)	PWR	
VDDIOT	Digital I/O power (top)	PWR	

JTAG pins (6)			
Signal	**Function**	**Type**	**Properties**
RST_N	Global reset input	Input	IOL, PU, ST
TCK	Test clock	Input	IOL, PD, ST
TDI	Test data input	Input	IOL, PU
TDO	Test data output	Output	IOL, PD
TMS	Test mode select	Input	IOL, PU
TRST_N	Test reset input	Input	IOL, PU, ST

I/O pins (81)			
Signal	**Function**	**Type**	**Properties**
X0D00	$1A^0$	I/O	IOL, PD
X0D01	$1B^0$	I/O	IOL, PD

(continued)

Signal	Function					Type	Properties
X0D02		4A[0]	8A[0]	16A[0]	32A[20]	I/O	IOL, PD
X0D03		4A[1]	8A[1]	16A[1]	32A[21]	I/O	IOL, PD
X0D04		4B[0]	8A[2]	16A[2]	32A[22]	I/O	IOL, PD
X0D05		4B[1]	8A[3]	16A[3]	32A[23]	I/O	IOL, PD
X0D06		4B[2]	8A[4]	16A[4]	32A[24]	I/O	IOL, PD
X0D07		4B[3]	8A[5]	16A[5]	32A[25]	I/O	IOL, PD
X0D08		4A[2]	8A[6]	16A[6]	32A[26]	I/O	IOL, PD
X0D09		4A[3]	8A[7]	16A[7]	32A[27]	I/O	IOL, PD
X0D10	1C[0]					I/O	IOL, PD
X0D11	1D[0]					I/O	IOL, PD
X0D12	1E[0]					I/O	IOR, PD
X0D13	1F[0]					I/O	IOR, PD
X0D14		4C[0]	8B[0]	16A[8]	32A[28]	I/O	IOR, PD
X0D15		4C[1]	8B[1]	16A[9]	32A[29]	I/O	IOR, PD
X0D16	$XL4^{4}_{in}$	4D[0]	8B[2]	16A[10]		I/O	IOR, PD
X0D17	$XL4^{3}_{in}$	4D[1]	8B[3]	16A[11]		I/O	IOR, PD
X0D18	$XL4^{2}_{in}$	4D[2]	8B[4]	16A[12]		I/O	IOR, PD
X0D19	$XL4^{1}_{in}$	4D[3]	8B[5]	16A[13]		I/O	IOR, PD
X0D20		4C[2]	8B[6]	16A[14]	32A[30]	I/O	IOR, PD
X0D21		4C[3]	8B[7]	16A[15]	32A[31]	I/O	IOR, PD
X0D22	1G[0]					I/O	IOR, PD
X0D23	1H[0]					I/O	IOR, PD
X0D24	$XL7^{0}_{in}$	1I[0]				I/O	IOR, PD
X0D25	$XL7^{0}_{out}$	1J[0]				I/O	IOR, PD
X0D26	$XL7^{3}_{out}$	4E[0]	8C[0]	16B[0]		I/O	IOR, PD
X0D27	$XL7^{4}_{out}$	4E[1]	8C[1]	16B[1]		I/O	IOR, PD
X0D28		4F[0]	8C[2]	16B[2]		I/O	IOR, PD
X0D29		4F[1]	8C[3]	16B[3]		I/O	IOR, PD
X0D30		4F[2]	8C[4]	16B[4]		I/O	IOR, PD
X0D31		4F[3]	8C[5]	16B[5]		I/O	IOR, PD
X0D32		4E[2]	8C[6]	16B[6]		I/O	IOR, PD
X0D33		4E[3]	8C[7]	16B[7]		I/O	IOR, PD
X0D34	$XL7^{1}_{out}$	1K[0]				I/O	IOR, PD
X0D35	$XL7^{2}_{out}$	1L[0]				I/O	IOR, PD
X0D36		1M[0]		8D[0]	16B[8]	I/O	IOL, PD
X0D37		1N[0]		8D[1]	16B[9]	I/O	IOL, PD
X0D38		1O[0]		8D[2]	16B[10]	I/O	IOL, PD
X0D39		1P[0]		8D[3]	16B[11]	I/O	IOL, PD
X0D40	$XL0^{1}_{in}$			8D[4]	16B[12]	I/O	IOL, PD
X0D41	$XL0^{0}_{in}$			8D[5]	16B[13]	I/O	IOL, PD
X0D42	$XL0^{0}_{out}$			8D[6]	16B[14]	I/O	IOL, PD
X0D43	$XL0^{1}_{out}$			8D[7]	16B[15]	I/O	IOL, PD
X1D00	$XL7^{2}_{in}$	1A[0]				I/O	IOR, PD

(continued)

Signal	Function	Type	Properties
X1D01	$XL7^{1}_{in}$ $1B^{0}$	I/O	IOR, PD
X1D02	$XL4^{0}_{in}$ $4A^{0}$ $8A^{0}$ $16A^{0}$ $32A^{20}$	I/O	IOR, PD
X1D03	$XL4^{0}_{out}$ $4A^{1}$ $8A^{1}$ $16A^{1}$ $32A^{21}$	I/O	IOR, PD
X1D04	$XL4^{1}_{out}$ $4B^{0}$ $8A^{2}$ $16A^{2}$ $32A^{22}$	I/O	IOR, PD
X1D05	$XL4^{2}_{out}$ $4B^{1}$ $8A^{3}$ $16A^{3}$ $32A^{23}$	I/O	IOR, PD
X1D06	$XL4^{3}_{out}$ $4B^{2}$ $8A^{4}$ $16A^{4}$ $32A^{24}$	I/O	IOR, PD
X1D07	$XL4^{4}_{out}$ $4B^{3}$ $8A^{5}$ $16A^{5}$ $32A^{25}$	I/O	IOR, PD
X1D08	$XL7^{4}_{in}$ $4A^{2}$ $8A^{6}$ $16A^{6}$ $32A^{26}$	I/O	IOR, PD
X1D09	$XL7^{3}_{in}$ $4A^{3}$ $8A^{7}$ $16A^{7}$ $32A^{27}$	I/O	IOR, PD
X1D10	$1C^{0}$	I/O	IOT, PD
X1D11	$1D^{0}$	I/O	IOT, PD
X1D14	$4C^{0}$ $8B^{0}$ $16A^{8}$ $32A^{28}$	I/O	IOR, PD
X1D15	$4C^{1}$ $8B^{1}$ $16A^{9}$ $32A^{29}$	I/O	IOR, PD
X1D16	$XL3^{1}_{in}$ $4D^{0}$ $8B^{2}$ $16A^{10}$	I/O	IOL, PD
X1D17	$XL3^{0}_{in}$ $4D^{1}$ $8B^{3}$ $16A^{11}$	I/O	IOL, PD
X1D18	$XL3^{0}_{out}$ $4D^{2}$ $8B^{4}$ $16A^{12}$	I/O	IOL, PD
X1D19	$XL3^{1}_{out}$ $4D^{3}$ $8B^{5}$ $16A^{13}$	I/O	IOL, PD
X1D20	$4C^{2}$ $8B^{6}$ $16A^{14}$ $32A^{30}$	I/O	IOR, PD
X1D21	$4C^{3}$ $8B^{7}$ $16A^{15}$ $32A^{31}$	I/O	IOR, PD
X1D26	$4E^{0}$ $8C^{0}$ $16B^{0}$	I/O	IOT, PD
X1D27	$4E^{1}$ $8C^{1}$ $16B^{1}$	I/O	IOT, PD
X1D28	$4F^{0}$ $8C^{2}$ $16B^{2}$	I/O	IOT, PD
X1D29	$4F^{1}$ $8C^{3}$ $16B^{3}$	I/O	IOT, PD
X1D30	$4F^{2}$ $8C^{4}$ $16B^{4}$	I/O	IOT, PD
X1D31	$4F^{3}$ $8C^{5}$ $16B^{5}$	I/O	IOT, PD
X1D32	$4E^{2}$ $8C^{6}$ $16B^{6}$	I/O	IOT, PD
X1D33	$4E^{3}$ $8C^{7}$ $16B^{7}$	I/O	IOT, PD
X1D35	$1L^{0}$	I/O	IOL, PD
X1D36	$1M^{0}$ $8D^{0}$ $16B^{8}$	I/O	IOL, PD
X1D37	$1N^{0}$ $8D^{1}$ $16B^{9}$	I/O	IOL, PD
X1D38	$1O^{0}$ $8D^{2}$ $16B^{10}$	I/O	IOL, PD
X1D39	$1P^{0}$ $8D^{3}$ $16B^{11}$	I/O	IOL, PD
X1D40	$8D^{4}$ $16B^{12}$	I/O	IOT, PD
X1D41	$8D^{5}$ $16B^{13}$	I/O	IOT, PD
X1D42	$8D^{6}$ $16B^{14}$	I/O	IOT, PD
X1D43	$8D^{7}$ $16B^{15}$	I/O	IOT, PD

usb pins (5)			
Signal	**Function**	**Type**	**Properties**
USB_DM	USB Serial Data Inverted	I/O	
USB_DP	USB Serial Data	I/O	
USB_ID	USB Device ID (OTG) - Reserved	I/O	

(continued)

Signal	Function	Type	Properties
USB_RTUNE	USB resistor	I/O	
USB_VBUS	USB Power Detect Pin	I/O	

System pins (1)			
Signal	**Function**	**Type**	**Properties**
CLK	PLL reference clock	Input	IOL, PD, ST

5 Example Application Diagram

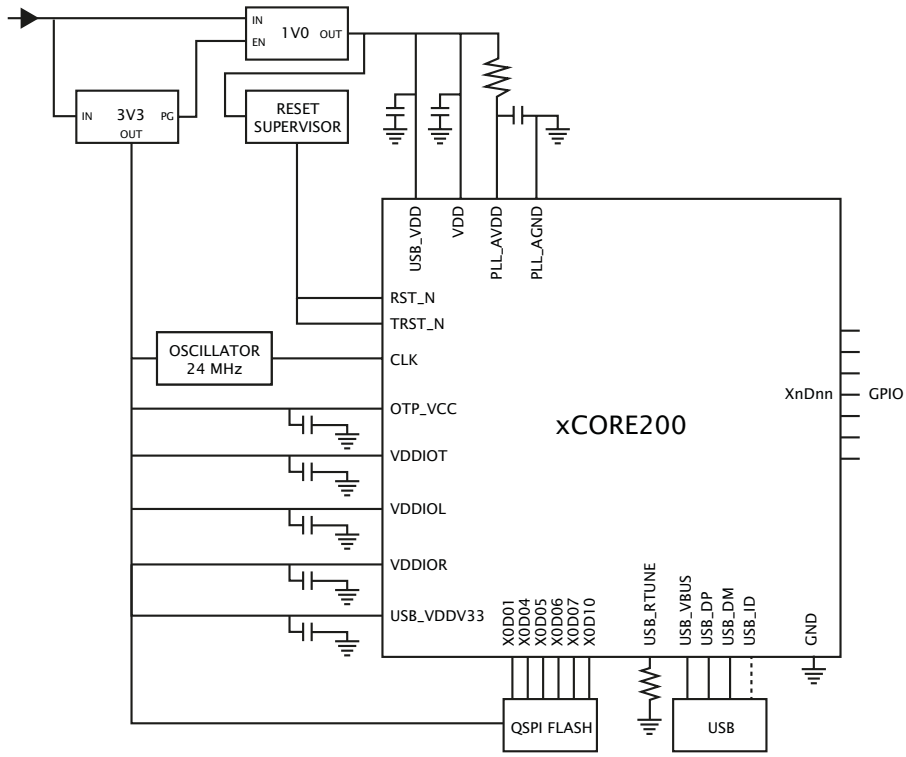


Figure 2: Simplified Reference Schematic

6 Product Overview

The XU210-256-TQ128 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.

6.1 Logical cores

Each tile has up to 5 active logical cores, which issue instructions down a shared five-stage pipeline. Instructions from the active cores are issued round-robin. Each core is allocated a fifth of the processing cycles. Figure 3 shows the guaranteed core performance.

Figure 3: Logical core performance

Speed grade	MIPS	Frequency	MIPS per logical core
10	1000 MIPS	500 MHz	100

There is no way that the performance of a logical core can be reduced below these predicted levels (unless *priority threads* are used: in this case the guaranteed minimum performance is computed based on the number of priority threads as defined in the architecture manual).

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.

6.2 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.

6.3 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 XU210-256-TQ128, 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.

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. xCORE-200 IO pins can

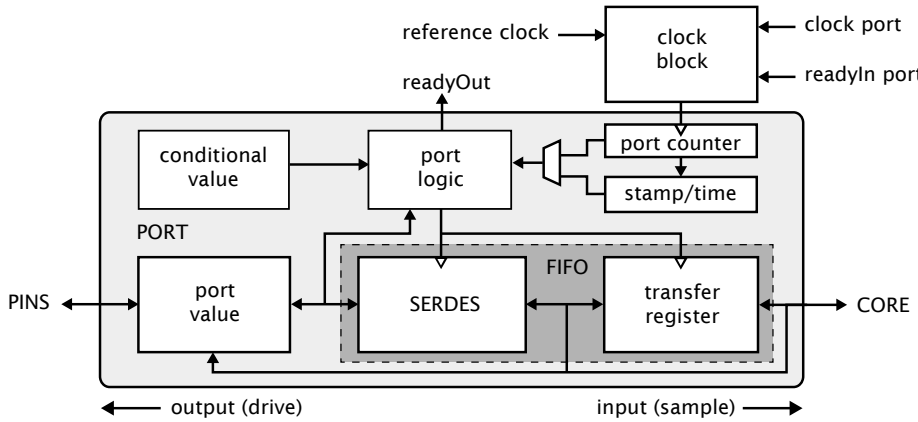


Figure 4: Port block diagram

be used as *open collector* outputs, where signals are driven low if a zero is output, but left high impedance if a one is output. This option is set on a per-port basis.

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.

6.4 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. xCORE-200 clock blocks optionally divide the clock input from a 1-bit port.

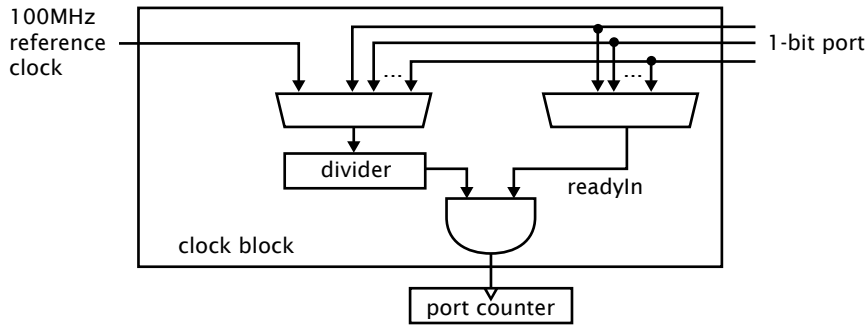


Figure 5: Clock block diagram

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.

6.5 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.

6.6 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.

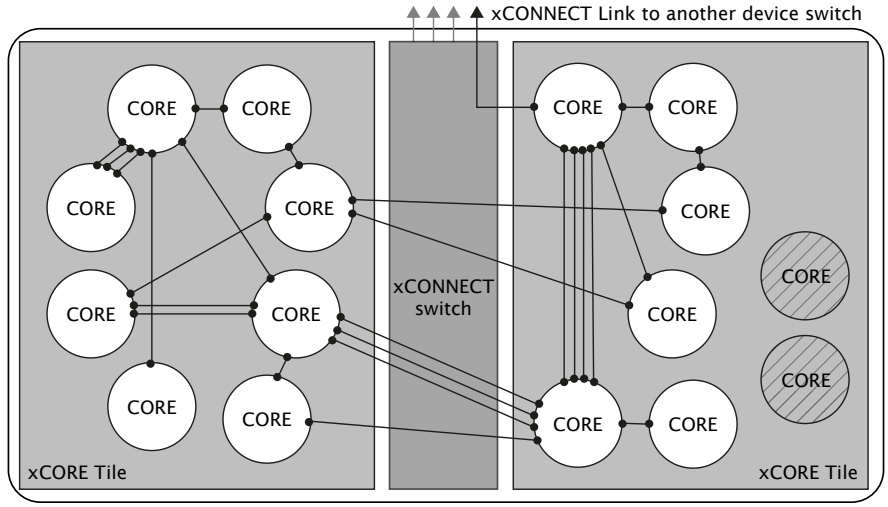


Figure 6: Switch, links and channel ends

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

7 PLL

The PLL creates a high-speed clock that is used for the switch, tile, and reference clock. The initial PLL multiplication value is shown in Figure 7:

Figure 7: The initial PLL multiplier values

Oscillator Frequency	Tile Frequency	PLL Ratio	PLL settings		
			OD	F	R
9-25 MHz	144-400 MHz	16	1	63	0

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

If the USB PHY is used, then either a 24 MHz or 12 MHz oscillator must be used.

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 xCORE-200 Clock Frequency Control document.

8 Boot Procedure

The device is kept in reset by driving RST_N low. When in reset, all GPIO pins have a pull-down enabled. 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) the processor boots.

The xCORE Tile boot procedure is illustrated in Figure 8. If bit 5 of the security register (*see* §9.1) is set, the device boots from OTP. To get a high value, a 3K3 pull-up resistor should be strapped onto the pin. To assure a low value, a pull-down resistor is required if other external devices are connected to this port.

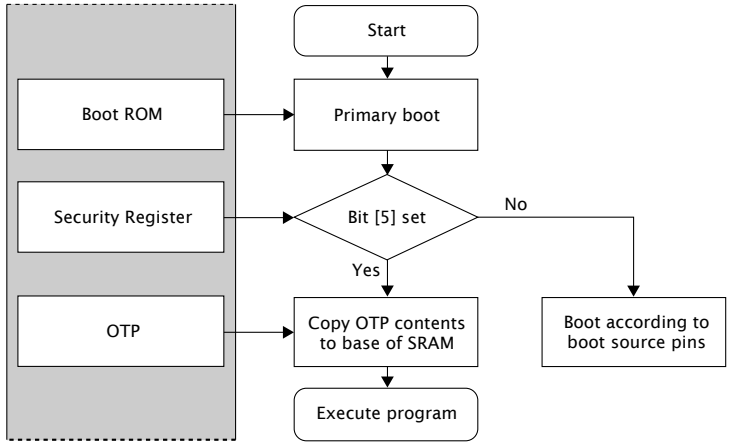


Figure 8: Boot procedure

X0D06	X0D05	X0D04	Tile 0 boot	Tile 1 boot	Enabled links
0	0	0	QSPI master	Channel end 0	None
0	0	1	SPI master	Channel end 0	None
0	1	0	SPI slave	Channel end 0	None
0	1	1	SPI slave	SPI slave	None
1	0	0	Channel end 0	Channel end 0	XL0 (2w)
1	0	1	Channel end 0	Channel end 0	XL4-XL7 (5w)
1	1	0	Channel end 0	Channel end 0	XL1, XL2, XL5, and XL6 (5w)
1	1	1	Channel end 0	Channel end 0	XL0-XL3 (5w)

Figure 9: Boot source pins

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.

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.

8.1 Boot from QSPI master

If set to boot from QSPI master, the processor enables the six pins specified in Figure 10, and drives the SPI clock at 50 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.

Figure 10: QSPI pins

Pin	Signal	Description
X0D01	SS	Slave Select
X0D04..X0D07	SPIO	Data
X0D10	SCLK	Clock

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

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

8.2 Boot from SPI master

If set to boot from SPI master, the processor enables the four pins specified in Figure 11, 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.

Figure 11: SPI master pins

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

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.

8.3 Boot from SPI slave

If set to boot from SPI slave, the processor enables the three pins specified in Figure 12 and expects a boot image to be clocked in. The supported clock polarity and phase are 0/0 and 1/1.

Figure 12: SPI slave pins

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

The xCORE Tile expects each byte to be transferred with the *least-significant bit first*. 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.

8.4 Boot from xConnect Link

If set to boot from an xConnect Link, the processor enables its link(s) around 2 us after the boot process starts. Enabling the Link switches off the pull-down resistors on the link, drives all the TX wires low (the initial state for the Link), and monitors the RX pins for boot-traffic; they 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.

8.5 Boot from OTP

If an xCORE tile is set to use secure boot (see Figure 8), 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.

8.6 Security register

The security register enables security features on the xCORE tile. The features shown in Figure 13 provide a strong level of protection and are sufficient for providing strong IP security.

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* §8).
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.
	21..15	General purpose software accessable security register available to end-users.
	31..22	General purpose user programmable JTAG UserID code extension.

Figure 13: Security register features

9 Memory

9.1 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 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`.

9.2 SRAM

Each xCORE Tile integrates a single 128KB 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.

10 USB PHY

The USB PHY provides High-Speed and Full-Speed, device, host, and on-the-go functionality. The PHY is configured through a set of peripheral registers (Appendix F), and data is communicated through ports on the digital node. A library, libxud_s.a, is provided to implement USB device functionality.

The USB PHY is connected to the ports on Tile 0 and Tile 1 as shown in Figure 14. When the USB PHY is enabled on Tile 0, the ports shown can on Tile 0 only be used with the USB PHY. When the USB PHY is enabled on Tile 1, then the ports shown can on Tile 1 only be used with the USB PHY. All other IO pins and ports are unaffected. The USB PHY should not be enabled on both tiles.

An external resistor of 43.2 ohm (1% tolerance) should connect USB_TUNE to ground, as close as possible to the device.

Figure 14 shows how two clock blocks can be used to clock the USB ports. One clock block for the TXDATA path, and one clock block for the RXDATA path. Details on how to connect those ports are documented in an application note on USB for xCORE-200.

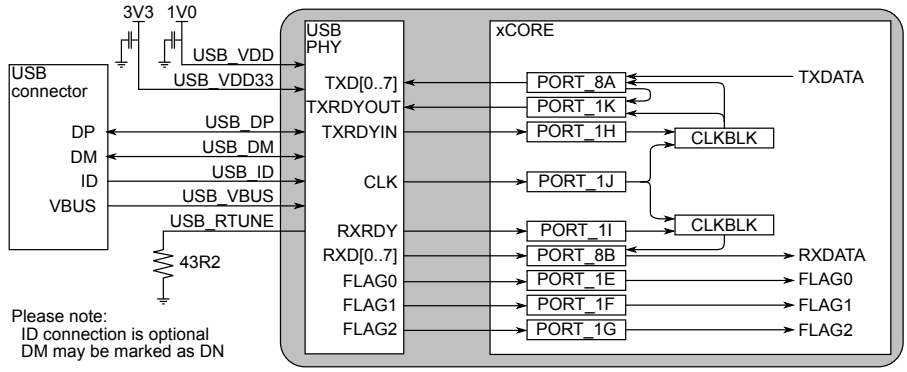


Figure 14: USB port functions

10.1 Logical Core Requirements

The XMOS XUD software component runs in a single logical core with endpoint and application cores communicating with it via a combination of channel communication and shared memory variables.

Each IN (host requests data from device) or OUT (data transferred from host to device) endpoint requires one logical core.

11 JTAG

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

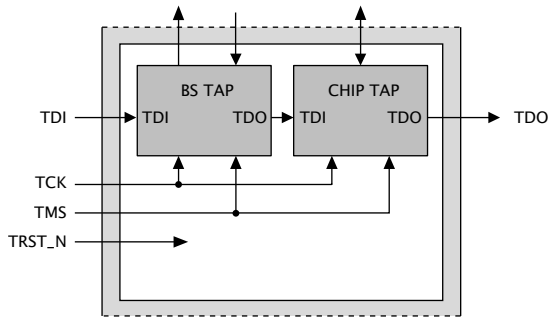


Figure 15: JTAG chain structure

The JTAG chain structure is illustrated in Figure 15. 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 JTAG device identification register can be read by using the IDCODE instruction. Its contents are specified in Figure 16.

Figure 16: IDCODE return value

Bit31				Device Identification Register																								Bit0
Version				Part Number																Manufacturer Identity								1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1	1	0	0	0	1	1	0	0	1	1
0				0				0				0				6				6				3				3			

The JTAG usercode register can be read by using the USERCODE instruction. Its contents are specified in Figure 17. The OTP User ID field is read from bits [22:31] of the security register on xCORE Tile 0, *see* §9.1 (all zero on unprogrammed devices).

Figure 17: USERCODE return value

Bit31										Usercode Register																				Bit0
OTP User ID										Unused				Silicon Revision																
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0				0				0				2				8				0				0				0			

12 Board Integration

The device has the following power supply pins:

- VDD pins for the xCORE Tile, including a USB_VDD pin that powers the USB PHY
- VDDIO pins for the I/O lines. Separate I/O supplies are provided for the left, top, and right side of the package; different I/O voltages may be supplied on those. The signal description (Section 4) specifies which I/O is powered from which power-supply
- PLL_AVDD pins for the PLL
- OTP_VCC pins for the OTP
- A USB_VDD33 pin for the analogue supply to the USB-PHY

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 PLLVDD 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 4.7 Ω resistor and multi-layer ceramic capacitor) is recommended on this pin.

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

12.1 USB connections

USB_VBUS should be connected to the VBUS pin of the USB connector. A 2.2 uF capacitor to ground is required on the VBUS pin. A ferrite bead may be used to reduce HF noise.

For self-powered systems, a bleeder resistor may be required to stop VBUS from floating when no USB cable is attached.

USB_DP and USB_DN should be connected to the USB connector. USB_ID does not need to be connected.

12.2 USB signal routing and placement

The USB_DP and USB_DN lines are the positive and negative data polarities of a high speed USB signal respectively. Their high-speed differential nature implies that they must be coupled and properly isolated. The board design must ensure that the board traces for USB_DP and USB_DN are tightly matched. In addition, according to the USB 2.0 specification, the USB_DP and USB_DN differential impedance must be 90 Ω.

12.2.1 General routing and placement guidelines

The following guidelines will help to avoid signal quality and EMI problems on high speed USB designs. They relate to a four-layer (Signal, GND, Power, Signal) PCB.

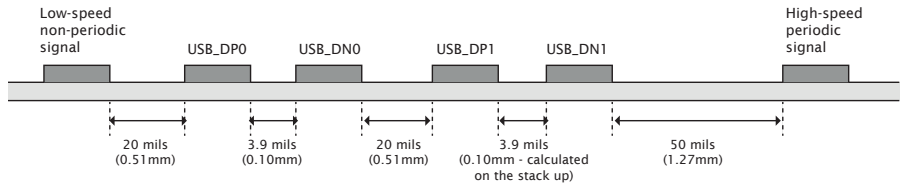


Figure 18: USB trace separation showing a low speed signal, two differential pairs and a high-speed clock

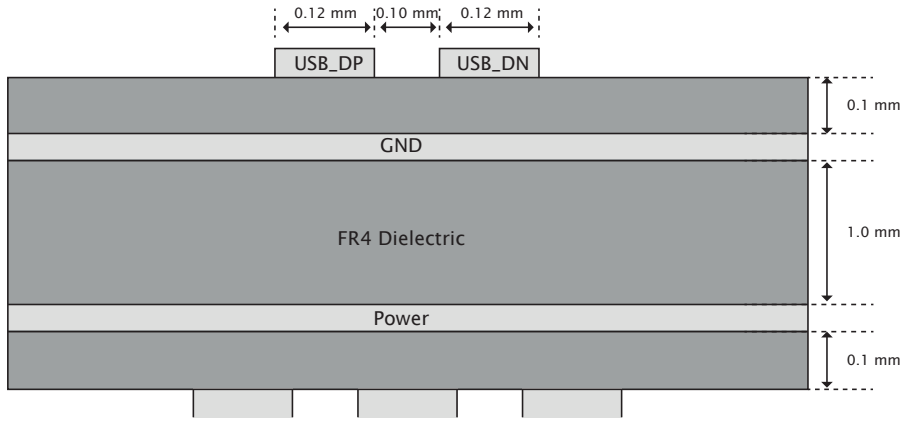


Figure 19: Example USB board stack

For best results, most of the routing should be done on the top layer (assuming the USB connector and XS2-U10A-256-TQ128 are on the top layer) closest to GND. Reference planes should be below the transmission lines in order to maintain control of the trace impedance.

We recommend that the high-speed clock and high-speed USB differential pairs are routed first before any other routing. When routing high speed USB signals, the following guidelines should be followed:

- High speed differential pairs should be routed together.
- High-speed USB signal pair traces should be trace-length matched. Maximum trace-length mismatch should be no greater than 4mm.
- Ensure that high speed signals (clocks, USB differential pairs) are routed as far away from off-board connectors as possible.
- High-speed clock and periodic signal traces that run parallel should be at least 1.27mm away from USB_DP/USB_DN (see Figure 18).

- Low-speed and non-periodic signal traces that run parallel should be at least 0.5mm away from USB_DP/USB_DN (see Figure 18).
- Route high speed USB signals on the top of the PCB wherever possible.
- Route high speed USB traces over continuous power planes, with no breaks. If a trade-off must be made, changing signal layers is preferable to crossing plane splits.
- Follow the $20 \times h$ rule; keep traces $20 \times h$ (the height above the power plane) away from the edge of the power plane.
- Use a minimum of vias in high speed USB traces.
- Avoid corners in the trace. Where necessary, rather than turning through a 90 degree angle, use two 45 degree turns or an arc.
- DO NOT route USB traces near clock sources, clocked circuits or magnetic devices.
- Avoid stubs on high speed USB signals.

12.3 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 128 pin Thin Quad Flat Pack package with exposed heat slug on a 0.4mm pitch. An example land pattern is shown in Figure 20.

Pad widths and spacings are such that solder mask can still be applied between the pads using standard design rules. This is recommended to reduce solder shorts.

The center pad solder paste level needs to be controlled so the device sits the correct height from the circuit board. For the 128 pin TQFP package, a 3x3 array of squares for solder paste is recommended as shown in Figure 21. This gives a paste level of 56%.

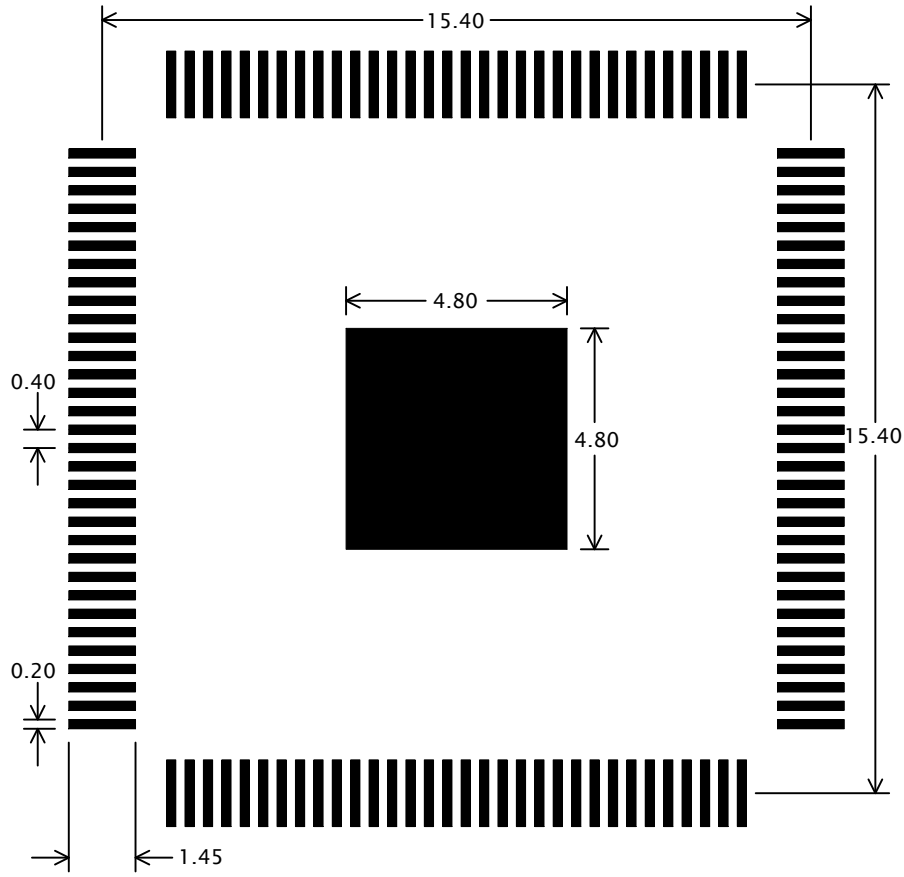


Figure 20: Example land pattern

12.4 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.

12.5 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.

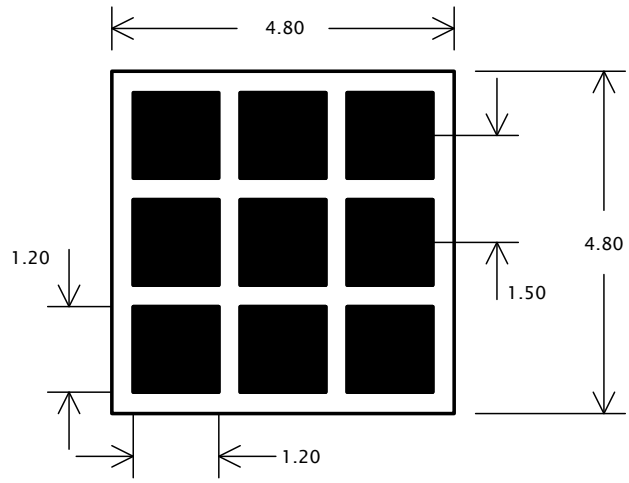


Figure 21: Solder stencil for centre pad

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.

13 DC and Switching Characteristics

13.1 Operating Conditions

Symbol	Parameter	MIN	TYP	MAX	UNITS	Notes
VDD	Tile DC supply voltage	0.95	1.00	1.05	V	
VDDIOL	I/O supply voltage	3.135	3.30	3.465	V	
VDDIOR	I/O supply voltage	3.135	3.30	3.465	V	
VDDIOT 3v3	I/O supply voltage	3.135	3.30	3.465	V	
VDDIOT 2v5	I/O supply voltage	2.375	2.50	2.625	V	
VDD33	Peripheral supply	3.135	3.30	3.465	V	
PLL_AVDD	PLL analog supply	0.95	1.00	1.05	V	
Cl	xCORE Tile I/O load capacitance			25	pF	
Ta	Ambient operating temperature (Commercial)	0		70	℃	
	Ambient operating temperature (Industrial)	-40		85	℃	
Tj	Junction temperature			125	℃	
Tstg	Storage temperature	-65		150	℃	

Figure 22: Operating conditions

13.2 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	2.20			V	B, C
V(OL)	Output low voltage			0.40	V	B, C
R(PU)	Pull-up resistance		35K		Ω	D
R(PD)	Pull-down resistance		35K		Ω	D

Figure 23: DC characteristics

A All pins except power supply pins.

B All general-purpose I/Os are nominal 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.

13.3 ESD Stress Voltage

Symbol	Parameter	MIN	TYP	MAX	UNITS	Notes
HBM	Human body model	-2.00		2.00	KV	
CDM	Charged Device Model	-500		500	V	

Figure 24: ESD stress voltage

13.4 Reset Timing

Figure 25: Reset timing

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

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

13.5 Power Consumption

Figure 26: xCORE Tile currents

Symbol	Parameter	MIN	TYP	MAX	UNITS	Notes
I(DDCQ)	Quiescent VDD current		45		mA	A, B, C
PD	Tile power dissipation		325		µW/MIPS	A, D, E, F
IDD	Active VDD current		570	700	mA	A, G
I(ADDPLL)	PLL_AVDD current			5	mA	H
I(VDD33)	VDD33 current		26.7		mA	I
I(USB_VDD)	USB_VDD current		8.27		mA	J

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, 500 MHz, average device resource usage.

H PLL_AVDD = 1.0 V

I HS mode transmitting while driving all 0's data (constant JKJK on DP/DM). Loading of 10 pF. Transfers do not include any interpacket delay.

J HS receive mode; no traffic.

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-U Power Consumption document,

13.6 Clock

Figure 27: Clock

Symbol	Parameter	MIN	TYP	MAX	UNITS	Notes
f	Frequency	9	24	25	MHz	
SR	Slew rate	0.10			V/ns	
TJ(LT)	Long term jitter (pk-pk)			2	%	A
f(MAX)	Processor clock frequency ()			400	MHz	B
	Processor clock frequency			500	MHz	B

A Percentage of CLK period.

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

Further details can be found in the XS1-U Clock Frequency Control document,

13.7 xCORE Tile I/O AC Characteristics

Figure 28: 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	

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.

13.8 xConnect Link Performance

Figure 29: 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.

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.

13.9 JTAG Timing

Figure 30: JTAG timing

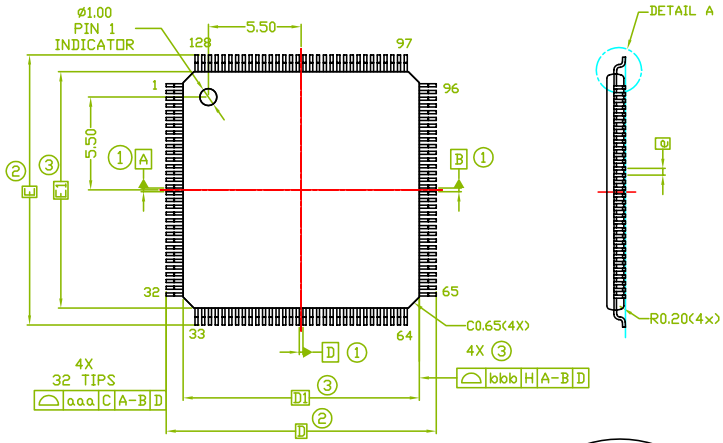
Symbol	Parameter	MIN	TYP	MAX	UNITS	Notes
f(TCK_D)	TCK frequency (debug)			18	MHz	
f(TCK_B)	TCK frequency (boundary scan)			10	MHz	
T(SETUP)	TDO to TCK setup time	5			ns	A
T(HOLD)	TDO to TCK hold time	5			ns	A
T(DELAY)	TCK to output delay			15	ns	B

A Timing applies to TMS and TDI inputs.

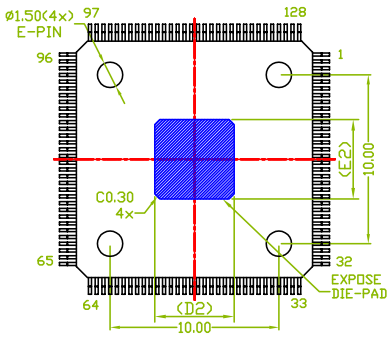
B Timing applies to TDO output from negative edge of TCK.

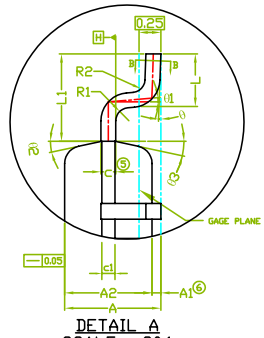
All JTAG operations are synchronous to TCK apart from the global asynchronous reset TRST_N.

14 Package Information

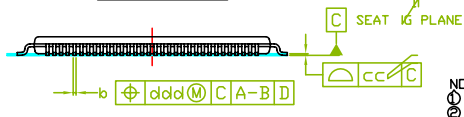


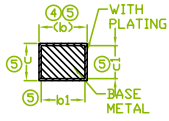
SYMBOL	Min.	Nom.	Max.
A	–	–	1.20
A1	0.05	–	0.15
A2	0.95	1.00	1.05
b	0.13	0.18	0.23
b1	0.13	0.16	0.19
D	16.00 BSC		
D1	14.00 BSC		
e	0.40 BSC		
E	16.00 BSC		
E1	14.00 BSC		
θ	0°	3.5°	7°
θ1	0°	–	–
θ2	11°	12°	13°
θ3	11°	12°	13°
c	0.09	–	0.20
c1	0.09	–	0.16
L	0.45	0.60	0.75
L1	1.00 REF		
R1	0.08	–	–
R2	0.08	–	0.20





REF	TOLERANCES OF FORM AND POSITION
aaa	0.20
bbb	0.20
ccc	0.08
ddd	0.07





LF Ref#	Symbol	Min	Nom	Max
L-17-09011	D2	4.60	4.70	4.80
	E2	4.60	4.70	4.80

NOTE :
① DATUM A-B AND D TO DETERMINE AT DATUM PLANE H.
② TO BE DETERMINED AT SEATING DATUM PLAN C.
③ DIMENSION D1 AND E1 DO NOT INCLUDE MOLD PROTRUSIONS. ALLOWABLE PROTRUSION IS 0.25 mm PER SIDE. S1 AND E1 ARE MAXIMUM PLASTIC BODY SIZE DIMENSION INCLUDING MOLD MISMATCH.
④ DIMENSION b DOSE NOT INCLUDE DAMBAR PROTRUSION. ALLOWABLE DAMBAR PROTRUSION SHALL NOT CAUSE THE LEAD WIDTH TO EXCEED THE MAXIMUM b DIMENSION BY MORE THAN 0.08 mm. DAMBAR CANNOT BE LOCATED ON THE LOWER RADIUS OR THE FOOT. MINIMUM SPACE BETWEEN PROTRUSION AND ADJACENT LEAD IS 0.07 mm FOR 0.4mm AND 0.5 mm PITCH PACKAGE.
⑤ THESE DIMENSIONS APPLY TO THE FLAT SECTION OF THE BETWEEN 0.10 mm AND 0.25 mm FROM THE LEAD TIP.
⑥ A1 IS THE DEFINED AS THE DISTANCE FROM THE SEATING PLAN TO THE LOWEST POINT ON TIIC PACKAGC BODY.
⑦ PACKAGE LEAD COUNT IS NON-JEDEC STANDARD.

14.1 Part Marking

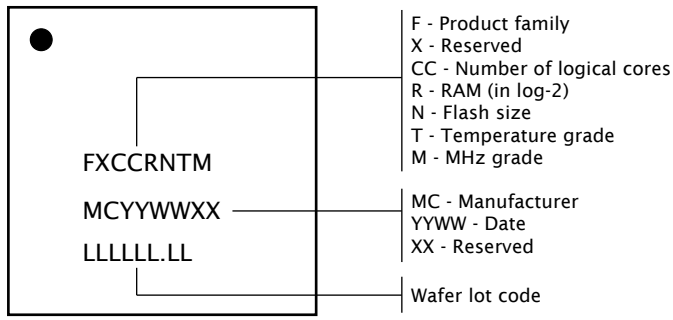


Figure 31: Part marking scheme

15 Ordering Information

Figure 32: Orderable part numbers

Product Code	Marking	Qualification	Speed Grade
XU210-256-TQ128-C20	U01070C5	Commercial	1000 MIPS
XU210-256-TQ128-I20	U01070I5	Industrial	1000 MIPS

Appendices

A Configuration of the XU210-256-TQ128

The device is configured through banks of registers, as shown in Figure 33.

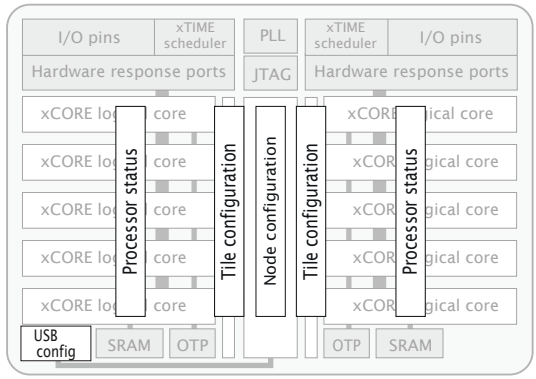


Figure 33: Registers

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).

A.4 Accessing a register of an analogue peripheral

Peripheral registers can be accessed through the interconnect using the functions `write_periph_32(device, peripheral, ...)`, `read_periph_32(device, peripheral, ...)`, `write_periph_8(device, peripheral, ...)`, and `read_periph_8(device, peripheral, ...)`; where device is the name of the analogue device, and peripheral is the number of the peripheral. These functions implement the protocols described below.

A channel-end should be allocated to communicate with the configuration registers. The destination of the channel-end should be set to `0xnnnnpp02` where `nnnn` is the node-identifier and `pp` is the peripheral identifier.

A write message comprises the following:

control-token 36	24-bit response channel-end identifier	8-bit register number	8-bit size	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 37	24-bit response channel-end identifier	8-bit register number	8-bit size	control-token 1

The response to the read message comprises either control token 3, 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	RW	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
0x0C	RO	RAM size
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 34: Summary

B.1 RAM base address: 0x00

This register contains the base address of the RAM. It is initialized to 0x00040000.

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
31:18	RW		The event and interrupt vectors.
17:0	RO	-	Reserved

B.3 xCORE Tile control: 0x02

Register to control features in the xCORE tile

Bits	Perm	Init	Description
31:26	RO	-	Reserved
25:18	RW	0	RGMII TX data delay value (in PLL output cycle increments)
17:9	RW	0	RGMII TX clock divider value. TX clk rises when counter (clocked by PLL output) reaches this value and falls when counter reaches (value»1). Value programmed into this field should be actual divide value required minus 1
8	RW	0	Enable RGMII interface periph ports
7:6	RO	-	Reserved
5	RW	0	Select the dynamic mode (1) for the clock divider when the clock divider is enabled. In dynamic mode the clock divider is only activated when all active threads are paused. In static mode the clock divider is always enabled.
4	RW	0	Enable the clock divider. This divides the output of the PLL to facilitate one of the low power modes.
3	RO	-	Reserved
2	RW		Select between UTMI (1) and ULPI (0) mode.
1	RW		Enable the ULPI Hardware support module
0	RO	-	Reserved

0x02: xCORE Tile control

B.4 xCORE Tile boot status: 0x03

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

Bits	Perm	Init	Description
31:24	RO	-	Reserved
23:16	RO		Processor number.
15:9	RO	-	Reserved
8	RO		Overwrite BOOT_MODE.
7:6	RO	-	Reserved
5	RO		Indicates if core1 has been powered off
4	RO		Cause the ROM to not poll the OTP for correct read levels
3	RO		Boot ROM boots from RAM
2	RO		Boot ROM boots from JTAG
1:0	RO		The boot PLL mode pin value.

0x03: xCORE Tile boot status

B.5 Security configuration: 0x05

Copy of the security register as read from OTP.

0x05: Security configuration

Bits	Perm	Init	Description
31	RW		Disables write permission on this register
30:15	RO	-	Reserved
14	RW		Disable access to XCore's global debug
13	RO	-	Reserved
12	RW		lock all OTP sectors
11:8	RW		lock bit for each OTP sector
7	RW		Enable OTP reduanacy
6	RO	-	Reserved
5	RW		Override boot mode and read boot image from OTP
4	RW		Disable JTAG access to the PLL/BOOT configuration registers
3:1	RO	-	Reserved
0	RW		Disable access to XCore's JTAG debug TAP

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 has been stopped for at least 10 core clock cycles (this can be achieved by inserting two nop instructions between the SETPS and GETPS). 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	Core ring oscillator enable.
0	RW	0	Peripheral ring oscillator enable.

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	0	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	0	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	0	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	0	Ring oscillator Counter data.

B.11 RAM size: 0x0C

The size of the RAM in bytes

0x0C: RAM size

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

B.12 Debug SSR: 0x10

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

0x10: Debug SSR

Bits	Perm	Init	Description
31:11	RO	-	Reserved
10	DRW		Address space indentifier
9	DRW		Determines the issue mode (DI bit) upon Kernel Entry after Exception or Interrupt.
8	RO		Determines the issue mode (DI bit).
7	DRW		When 1 the thread is in fast mode and will continually issue.
6	DRW		When 1 the thread is paused waiting for events, a lock or another resource.
5	RO	-	Reserved
4	DRW		1 when in kernel mode.
3	DRW		1 when in an interrupt handler.
2	DRW		1 when in an event enabling sequence.
1	DRW		When 1 interrupts are enabled for the thread.
0	DRW		When 1 events are enabled for the thread.

B.13 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.14 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.15 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.16 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.17 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		Number of the hardware breakpoint/watchpoint which caused the interrupt (always 0 for =HOST= and =DCALL=). If multiple breakpoints/watchpoints trigger at once, the lowest number is taken.
15:8	DRW		Number of thread which caused the debug interrupt (always 0 in the case of =HOST=).
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.18 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.19 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 threads are stopped when not in debug mode. Every bit which is set prevents the respective thread from running.

B.20 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.21 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.22 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 thread in the machine allowing the breakpoint to be enabled individually for each thread.
15:2	RO	-	Reserved
1	DRW	0	When 0 break when PC == IBREAK_ADDR. When 1 = break when PC != IBREAK_ADDR.
0	DRW	0	When 1 the instruction breakpoint is enabled.

B.23 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.24 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.25 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 thread in the machine allowing the breakpoint to be enabled individually for each thread.
15:3	RO	-	Reserved
2	DRW	0	When 1 the breakpoints will be be triggered on loads.
1	DRW	0	Determines the break condition: 0 = A AND B, 1 = A OR B.
0	DRW	0	When 1 the instruction breakpoint is enabled.

B.26 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.27 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.28 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 thread in the machine allowing the breakpoint to be enabled individually for each thread.
15:2	RO	-	Reserved
1	DRW	0	When 0 break when condition A is met. When 1 = break when condition B is met.
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	CRO	Device identification
0x01	CRO	xCORE Tile description 1
0x02	CRO	xCORE Tile description 2
0x04	CRW	Control PSwitch permissions to debug registers
0x05	CRW	Cause debug interrupts
0x06	CRW	xCORE Tile clock divider
0x07	CRO	Security configuration
0x20 .. 0x27	CRW	Debug scratch
0x40	CRO	PC of logical core 0
0x41	CRO	PC of logical core 1
0x42	CRO	PC of logical core 2
0x43	CRO	PC of logical core 3
0x44	CRO	PC of logical core 4
0x45	CRO	PC of logical core 5
0x46	CRO	PC of logical core 6
0x47	CRO	PC of logical core 7
0x60	CRO	SR of logical core 0
0x61	CRO	SR of logical core 1
0x62	CRO	SR of logical core 2
0x63	CRO	SR of logical core 3
0x64	CRO	SR of logical core 4
0x65	CRO	SR of logical core 5
0x66	CRO	SR of logical core 6
0x67	CRO	SR of logical core 7

Figure 35: Summary

C.1 Device identification: 0x00

This register identifies the xCORE Tile

0x00: Device identification

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

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	CRO		Number of channel ends.
23:16	CRO		Number of the locks.
15:8	CRO		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	CRO		Number of clock blocks.
7:0	CRO		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	CRW	0	When 1 the PSwitch is restricted to RO access to all CRW registers from SSwitch, XCore(PS_DBG_Scratch) and JTAG
30:1	RO	-	Reserved
0	CRW	0	When 1 the PSwitch is restricted to RO access to all CRW registers from SSwitch

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	CRW	0	1 when the processor is in debug mode.
0	CRW	0	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	CRW	0	Clock disable. Writing '1' will remove the clock to the tile.
30:16	RO	-	Reserved
15:0	CRW	0	Clock divider.

C.7 Security configuration: 0x07

Copy of the security register as read from OTP.

0x07: Security configuration

Bits	Perm	Init	Description
31	CRO		Disables write permission on this register
30:15	RO	-	Reserved
14	CRO		Disable access to XCore's global debug
13	RO	-	Reserved
12	CRO		lock all OTP sectors
11:8	CRO		lock bit for each OTP sector
7	CRO		Enable OTP reduanacy
6	RO	-	Reserved
5	CRO		Override boot mode and read boot image from OTP
4	CRO		Disable JTAG access to the PLL/BOOT configuration registers
3:1	RO	-	Reserved
0	CRO		Disable access to XCore's JTAG debug TAP

C.8 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.9 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	CRO		Value.

C.10 PC of logical core 1: 0x41

Value of the PC of logical core 1.

0x41: PC of logical core 1

Bits	Perm	Init	Description
31:0	CRO		Value.

C.11 PC of logical core 2: 0x42

Value of the PC of logical core 2.

0x42: PC of logical core 2

Bits	Perm	Init	Description
31:0	CRO		Value.

C.12 PC of logical core 3: 0x43

Value of the PC of logical core 3.

0x43: PC of logical core 3

Bits	Perm	Init	Description
31:0	CRO		Value.

C.13 PC of logical core 4: 0x44

Value of the PC of logical core 4.

0x44: PC of logical core 4

Bits	Perm	Init	Description
31:0	CRO		Value.

C.14 PC of logical core 5: 0x45

Value of the PC of logical core 5.

0x45: PC of logical core 5

Bits	Perm	Init	Description
31:0	CRO		Value.

C.15 PC of logical core 6: 0x46

Value of the PC of logical core 6.

0x46: PC of logical core 6

Bits	Perm	Init	Description
31:0	CRO		Value.

C.16 PC of logical core 7: 0x47

Value of the PC of logical core 7.

0x47: PC of logical core 7

Bits	Perm	Init	Description
31:0	CRO		Value.

C.17 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	CRO		Value.

C.18 SR of logical core 1: 0x61

Value of the SR of logical core 1

0x61: SR of logical core 1

Bits	Perm	Init	Description
31:0	CRO		Value.

C.19 SR of logical core 2: 0x62

Value of the SR of logical core 2

0x62: SR of logical core 2

Bits	Perm	Init	Description
31:0	CRO		Value.

C.20 SR of logical core 3: 0x63

Value of the SR of logical core 3

0x63: SR of logical core 3

Bits	Perm	Init	Description
31:0	CRO		Value.

C.21 SR of logical core 4: 0x64

Value of the SR of logical core 4

0x64: SR of logical core 4

Bits	Perm	Init	Description
31:0	CRO		Value.

C.22 SR of logical core 5: 0x65

Value of the SR of logical core 5

0x65: SR of logical core 5

Bits	Perm	Init	Description
31:0	CRO		Value.

C.23 SR of logical core 6: 0x66

Value of the SR of logical core 6

0x66: SR of logical core 6

Bits	Perm	Init	Description
31:0	CRO		Value.

C.24 SR of logical core 7: 0x67

Value of the SR of logical core 7

0x67: SR of logical core 7

Bits	Perm	Init	Description
31:0	CRO		Value.

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
0x09	R	System JTAG device ID register
0x0A	R	System USERCODE register
0x0C	RW	Directions 0-7
0x0D	RW	Directions 8-15
0x10	RW	Reserved
0x11	RW	Reserved.
0x1F	RO	Debug source
0x20 .. 0x28	RW	Link status, direction, and network
0x40 .. 0x47	RO	PLink status and network
0x80 .. 0x88	RW	Link configuration and initialization
0xA0 .. 0xA7	RW	Static link configuration

Figure 36: 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	-	Reserved
23:16	RO		Sampled values of BootCtl pins on Power 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 SLinks on the SSwitch.
15:8	RO		Number of processors on the SSwitch.
7:0	RO		Number of processors on the device.

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	RW	0	0 = SSCTL registers have write access. 1 = SSCTL registers can not be written to.
30:9	RO	-	Reserved
8	RW	0	0 = PLL_CTL_REG has write access. 1 = PLL_CTL_REG can not be written to.
7:1	RO	-	Reserved
0	RW	0	0 = 2-byte headers, 1 = 1-byte headers (reset as 0).

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 ID of this node.

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	RW		If set to 1, the chip will not be reset
30	RW		If set to 1, the chip will not wait for the PLL to re-lock. Only use this if a gradual change is made to the PLL
29	DW		If set to 1, set the PLL to be bypassed
28	DW		If set to 1, set the boot mode to boot from JTAG
27:26	RO	-	Reserved
25:23	RW		Output divider value range from 1 (8'h0) to 250 (8'hF9). P value.
22:21	RO	-	Reserved
20:8	RW		Feedback multiplication ratio, range from 1 (8'h0) to 255 (8'hFE). M value.
7	RO	-	Reserved
6:0	RW		Oscilator input divider value range from 1 (8'h0) to 32 (8'h0F). N value.

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	SSwitch clock generation

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	Software ref. clock divider

D.8 System JTAG device ID register: 0x09

0x09: System JTAG device ID register

Bits	Perm	Init	Description
31:28	RO		
27:12	RO		
11:1	RO		
0	RO		

D.9 System USERCODE register: 0x0A

0x0A: System USERCODE register

Bits	Perm	Init	Description
31:18	RO		JTAG USERCODE value programmed into OTP SR
17:0	RO		metal fixable ID code

D.10 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 dimension is 7.
27:24	RW	0	The direction for packets whose dimension is 6.
23:20	RW	0	The direction for packets whose dimension is 5.
19:16	RW	0	The direction for packets whose dimension is 4.
15:12	RW	0	The direction for packets whose dimension is 3.
11:8	RW	0	The direction for packets whose dimension is 2.
7:4	RW	0	The direction for packets whose dimension is 1.
3:0	RW	0	The direction for packets whose dimension is 0.

D.11 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 dimension is F.
27:24	RW	0	The direction for packets whose dimension is E.
23:20	RW	0	The direction for packets whose dimension is D.
19:16	RW	0	The direction for packets whose dimension is C.
15:12	RW	0	The direction for packets whose dimension is B.
11:8	RW	0	The direction for packets whose dimension is A.
7:4	RW	0	The direction for packets whose dimension is 9.
3:0	RW	0	The direction for packets whose dimension is 8.

D.12 Reserved: 0x10

Reserved.

0x10: Reserved

Bits	Perm	Init	Description
31:2	RO	-	Reserved
1	RW	0	Reserved.
0	RW	0	Reserved.

D.13 Reserved.: 0x11

Reserved.

0x11: Reserved.

Bits	Perm	Init	Description
31:2	RO	-	Reserved
1	RW	0	Reserved.
0	RW	0	Reserved.

D.14 Debug source: 0x1F

Contains the source of the most recent debug event.

0x1F: Debug source

Bits	Perm	Init	Description
31:5	RO	-	Reserved
4	RW		Reserved.
3:2	RO	-	Reserved
1	RW		If set, XCore1 is the source of last GlobalDebug event.
0	RW		If set, XCore0 is the source of last GlobalDebug event.

D.15 Link status, direction, and network: 0x20 .. 0x28

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. The registers control links 0..7.

0x20 .. 0x28: Link status, direction, and network

Bits	Perm	Init	Description
31:26	RO	-	Reserved
25:24	RO		Identify the SRC_TARGET type 0 - SLink, 1 - PLink, 2 - SSCTL, 3 - Undefine.
23:16	RO		When the link is in use, this is the destination link number to which all packets are sent.
15:12	RO	-	Reserved
11:8	RW	0	The direction that this link operates in.
7:6	RO	-	Reserved
5:4	RW	0	Determines the network to which this link belongs, reset as 0.
3	RO	-	Reserved
2	RO		1 when the current packet is considered junk and will be thrown away.
1	RO		1 when the dest side of the link is in use.
0	RO		1 when the source side of the link is in use.

D.16 PLink status and network: 0x40 .. 0x47

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

Bits	Perm	Init	Description
31:26	RO	-	Reserved
25:24	RO		Identify the SRC_TARGET type 0 - SLink, 1 - PLink, 2 - SSCTL, 3 - Undefine.
23:16	RO		When the link is in use, this is the destination link number to which all packets are sent.
15:6	RO	-	Reserved
5:4	RW	0	Determines the network to which this link belongs, reset as 0.
3	RO	-	Reserved
2	RO		1 when the current packet is considered junk and will be thrown away.
1	RO		1 when the dest side of the link is in use.
0	RO		1 when the source side of the link is in use.

0x40 .. 0x47: PLink status and network

D.17 Link configuration and initialization: 0x80 .. 0x88

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. The registers control links 0..7.

Bits	Perm	Init	Description
31	RW		Write to this bit with '1' will enable the XLink, writing '0' will disable it. This bit controls the muxing of ports with overlapping xlinks.
30	RW	0	0: operate in 2 wire mode; 1: operate in 5 wire mode
29:28	RO	-	Reserved
27	RO		Rx buffer overflow or illegal token encoding received.
26	RO	0	This end of the xlink has issued credit to allow the remote end to transmit
25	RO	0	This end of the xlink has credit to allow it to transmit.
24	WO		Clear this end of the xlink's credit and issue a HELLO token.
23	WO		Reset the receiver. The next symbol that is detected will be the first symbol in a token.
22	RO	-	Reserved
21:11	RW	0	Specify min. number of idle system clocks between two continuous symbols witin a transmit token -1.
10:0	RW	0	Specify min. number of idle system clocks between two continuous transmit tokens -1.

0x80 .. 0x88: Link configuration and initialization

D.18 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. The registers control links C, D, A, B, G, H, E, and F in that order.

0xA0 .. 0xA7: Static link configuration

Bits	Perm	Init	Description
31	RW	0	Enable static forwarding.
30:9	RO	-	Reserved
8	RW	0	The destination processor on this node that packets received in static mode are forwarded to.
7:5	RO	-	Reserved
4:0	RW	0	The destination channel end on this node that packets received in static mode are forwarded to.

E USB Node Configuration

The USB 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).

Figure 37: Summary

Number	Perm	Description
0x00	RO	Device identification register
0x04	RW	Node configuration register
0x05	RW	Node identifier
0x51	RW	System clock frequency
0x80	RW	Link Control and Status

E.1 Device identification register: 0x00

This register contains version information, and information on power-on behavior.

0x00: Device identification register

Bits	Perm	Init	Description
31:24	RO	0x0F	Chip identifier
23:16	RO	-	Reserved
15:8	RO	0x02	Revision number of the USB block
7:0	RO	0x00	Version number of the USB block

E.2 Node configuration register: 0x04

This register is used to set the communication model to use (1 or 3 byte headers), and to prevent any further updates.

0x04: Node configuration register

Bits	Perm	Init	Description
31	RW	0	Set to 1 to disable further updates to the node configuration and link control and status registers.
30:1	RO	-	Reserved
0	RW	0	Header mode. 0: 3-byte headers; 1: 1-byte headers.

E.3 Node identifier: 0x05

0x05: Node identifier

Bits	Perm	Init	Description
31:16	RO	-	Reserved
15:0	RW	0	16-bit node identifier. This does not need to be set, and is present for compatibility with XS1-switches.

E.4 System clock frequency: 0x51

0x51: System clock frequency

Bits	Perm	Init	Description
31:7	RO	-	Reserved
6:0	RW	25	Oscillator clock frequency in MHz rounded up to the nearest integer value. Only values between 5 and 100 MHz are valid - writes outside this range are ignored and will be NACKed. This field must be set on start up of the device and any time that the input oscillator clock frequency is changed. It must contain the system clock frequency in MHz rounded up to the nearest integer value.

E.5 Link Control and Status: 0x80

0x80: Link Control and Status

Bits	Perm	Init	Description
31:28	RO	-	Reserved
27	RO		Rx buffer overflow or illegal token encoding received.
26	RO	0	This end of the xlink has issued credit to allow the remote end to transmit
25	RO	0	This end of the xlink has credit to allow it to transmit.
24	WO		Clear this end of the xlink's credit and issue a HELLO token.
23	WO		Reset the receiver. The next symbol that is detected will be the first symbol in a token.
22	RO	-	Reserved
21:11	RW	1	Specify min. number of idle system clocks between two continuous symbols witin a transmit token -1.
10:0	RW	1	Specify min. number of idle system clocks between two continuous transmit tokens -1.

F USB PHY Configuration

The USB PHY is connected to the ports shown in section 10.

The *USB PHY* is peripheral 1. The control registers are accessed using 32-bit reads and writes (use `write_periph_32(device, 1, ...)` and `read_periph_32(device, 1, ...)` for reads and writes).

Number	Perm	Description
0x00	WO	UIFM reset
0x04	RW	UIFM IFM control
0x08	RW	UIFM Device Address
0x0C	RW	UIFM functional control
0x10	RW	UIFM on-the-go control
0x14	RO	UIFM on-the-go flags
0x18	RW	UIFM Serial Control
0x1C	RW	UIFM signal flags
0x20	RW	UIFM Sticky flags
0x24	RW	UIFM port masks
0x28	RW	UIFM SOF value
0x2C	RO	UIFM PID
0x30	RO	UIFM Endpoint
0x34	RW	UIFM Endpoint match
0x38	RW	OTG Flags mask
0x3C	RW	UIFM power signalling
0x40	RW	UIFM PHY control

Figure 38: Summary

F.1 UIFM reset: 0x00

A write to this register with any data resets all UIFM state, but does not otherwise affect the phy.

0x00: UIFM reset

Bits	Perm	Init	Description
31:0	WO		Value.

F.2 UIFM IFM control: 0x04

General settings of the UIFM IFM state machine.

0x04: UIFM IFM control

Bits	Perm	Init	Description
31:8	RO	-	Reserved
7	RW	0	Set to 1 to enable XEVACKMODE mode.
6	RW	0	Set to 1 to enable SOFISTOKEN mode.
5	RW	0	Set to 1 to enable UIFM power signalling mode.
4	RW	0	Set to 1 to enable IF timing mode.
3	RO	-	Reserved
2	RW	0	Set to 1 to enable UIFM linestate decoder.
1	RW	0	Set to 1 to enable UIFM CHECKTOKENS mode.
0	RW	0	Set to 1 to enable UIFM DOTOKENS mode.

F.3 UIFM Device Address: 0x08

The device address whose packets should be received. 0 until enumeration, it should be set to the assigned value after enumeration.

0x08: UIFM Device Address

Bits	Perm	Init	Description
31:7	RO	-	Reserved
6:0	RW	0	The enumerated USB device address must be stored here. Only packets to this address are passed on.

F.4 UIFM functional control: 0x0C

0x0C: UIFM functional control

Bits	Perm	Init	Description
31:5	RO	-	Reserved
4:2	RW	1	Set to 0 to disable UIFM to UTMI+ OPMODE mode.
1	RW	1	Set to 1 to switch UIFM to UTMI+ TERMSELECT mode.
0	RW	1	Set to 1 to switch UIFM to UTMI+ XCVRSELECT mode.

F.5 UIFM on-the-go control: 0x10

This register is used to negotiate an on-the-go connection.

0x10: UIFM on-the-go control

Bits	Perm	Init	Description
31:8	RO	-	Reserved
7	RW	0	Set to 1 to switch UIFM to EXTVBUSIND mode.
6	RW	0	Set to 1 to switch UIFM to DRVVBUSEXT mode.
5	RO	-	Reserved
4	RW	0	Set to 1 to switch UIFM to UTMI+ CHRGVBUS mode.
3	RW	0	Set to 1 to switch UIFM to UTMI+ DISCHRGVBUS mode.
2	RW	0	Set to 1 to switch UIFM to UTMI+ DMPULLDOWN mode.
1	RW	0	Set to 1 to switch UIFM to UTMI+ DPPULLDOWN mode.
0	RW	0	Set to 1 to switch UIFM to IDPULLUP mode.

F.6 UIFM on-the-go flags: 0x14

Status flags used for on-the-go negotiation

0x14: UIFM on-the-go flags

Bits	Perm	Init	Description
31:6	RO	-	Reserved
5	RO	0	Value of UTMI+ Bvalid flag.
4	RO	0	Value of UTMI+ IDGND flag.
3	RO	0	Value of UTMI+ HOSTDIS flag.
2	RO	0	Value of UTMI+ VBUSVLD flag.
1	RO	0	Value of UTMI+ SESSVLD flag.
0	RO	0	Value of UTMI+ SESSEND flag.

F.7 UIFM Serial Control: 0x18

0x18: UIFM Serial Control

Bits	Perm	Init	Description
31:7	RO	-	Reserved
6	RO	0	1 if UIFM is in UTMI+ RXRCV mode.
5	RO	0	1 if UIFM is in UTMI+ RXDM mode.
4	RO	0	1 if UIFM is in UTMI+ RXDP mode.
3	RW	0	Set to 1 to switch UIFM to UTMI+ TXSE0 mode.
2	RW	0	Set to 1 to switch UIFM to UTMI+ TXDATA mode.
1	RW	1	Set to 0 to switch UIFM to UTMI+ TXENABLE mode.
0	RW	0	Set to 1 to switch UIFM to UTMI+ FSLSSERIAL mode.

F.8 UIFM signal flags: 0x1C

Set of flags that monitor line and error states. These flags normally clear on the next packet, but they may be made sticky by using PER_UIFM_FLAGS_STICKY, in which they must be cleared explicitly.

0x1C: UIFM signal flags

Bits	Perm	Init	Description
31:7	RO	-	Reserved
6	RW	0	Set to 1 when the UIFM decodes a token successfully (e.g. it passes CRC5, PID check and has matching device address).
5	RW	0	Set to 1 when linestate indicates an SE0 symbol.
4	RW	0	Set to 1 when linestate indicates a K symbol.
3	RW	0	Set to 1 when linestate indicates a J symbol.
2	RW	0	Set to 1 if an incoming datapacket fails the CRC16 check.
1	RW	0	Set to the value of the UTMI_RXACTIVE input signal.
0	RW	0	Set to the value of the UTMI_RXERROR input signal

F.9 UIFM Sticky flags: 0x20

These bits define the sticky-ness of the bits in the UIFM IFM FLAGS register. A 1 means that bit will be sticky (hold its value until a 1 is written to that bitfield), or normal, in which case signal updates to the UIFM IFM FLAGS bits may be over-written by subsequent changes in those signals.

0x20: UIFM Sticky flags

Bits	Perm	Init	Description
31:7	RO	-	Reserved
6:0	RW	0	Stickyness for each flag.

F.10 UIFM port masks: 0x24

Set of masks that identify how port 1N, port 1O and port 1P are affected by changes to the flags in FLAGS

0x24: UIFM port masks

Bits	Perm	Init	Description
31:24	RW	0	Bit mask that determines which flags in UIFM_IFM_FLAG[6:0] contribute to port 1?. If any flag listed in this bitmask is high, port 1? will be high.
23:16	RW	0	Bit mask that determines which flags in UIFM_IFM_FLAG[6:0] contribute to port 1P. If any flag listed in this bitmask is high, port 1P will be high.
15:8	RW	0	Bit mask that determines which flags in UIFM_IFM_FLAG[6:0] contribute to port 1O. If any flag listed in this bitmask is high, port 1O will be high.
7:0	RW	0	Bit mask that determines which flags in UIFM_IFM_FLAG[6:0] contribute to port 1N. If any flag listed in this bitmask is high, port 1N will be high.

F.11 UIFM SOF value: 0x28

USB Start-Of-Frame counter

0x28: UIFM SOF value

Bits	Perm	Init	Description
31:11	RO	-	Reserved
10:8	RW	0	Most significant 3 bits of SOF counter
7:0	RW	0	Least significant 8 bits of SOF counter

F.12 UIFM PID: 0x2C

The last USB packet identifier received

0x2C: UIFM PID

Bits	Perm	Init	Description
31:4	RO	-	Reserved
3:0	RO	0	Value of the last received PID.

F.13 UIFM Endpoint: 0x30

The last endpoint seen

0x30: UIFM Endpoint

Bits	Perm	Init	Description
31:5	RO	-	Reserved
4	RO	0	1 if endpoint contains a valid value.
3:0	RO	0	A copy of the last received endpoint.

F.14 UIFM Endpoint match: 0x34

This register can be used to mark UIFM endpoints as special.

0x34: UIFM Endpoint match

Bits	Perm	Init	Description
31:16	RO	-	Reserved
15:0	RW	0	This register contains a bit for each endpoint. If its bit is set, the endpoint will be supplied on the RX port when ORed with 0x10.

F.15 OTG Flags mask: 0x38

0x38: OTG Flags mask

Bits	Perm	Init	Description
31:0	RW	0	Data

F.16 UIFM power signalling: 0x3C

0x3C: UIFM power signalling

Bits	Perm	Init	Description
31:9	RO	-	Reserved
8	RW	0	Valid
7:0	RW	0	Data

F.17 UIFM PHY control: 0x40

0x40: UIFM PHY control

Bits	Perm	Init	Description
31:19	RO	-	Reserved
18	RW	0	Set to 1 to disable pulldowns on ports 8A and 8B.
17:14	RO	-	Reserved
13	RW	0	After an auto-resume, this bit is set to indicate that the resume signalling was for reset (se0). Set to 0 to clear.
12	RW	0	After an auto-resume, this bit is set to indicate that the resume signalling was for resume (K). Set to 0 to clear.
11:8	RW	0	Log-2 number of clocks before any linestate change is propagated.
7	RW	0	Set to 1 to use the suspend controller handle to resume from suspend. Otherwise, the program has to poll the linestate_filt field in phy_teststatus.
6:4	RW	0	Control the the conf1,2,3 input pins of the PHY.
3:0	RO	-	Reserved

G 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, TRST_N, TMS, 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.

H JTAG, xSCOPE and Debugging

If you intend to design a board that can be used with the XMOS toolchain and xTAG debugger, you will need an xSYS header on your board. Figure 39 shows a decision diagram which explains what type of xSYS connectivity you need. The three subsections below explain the options in detail.

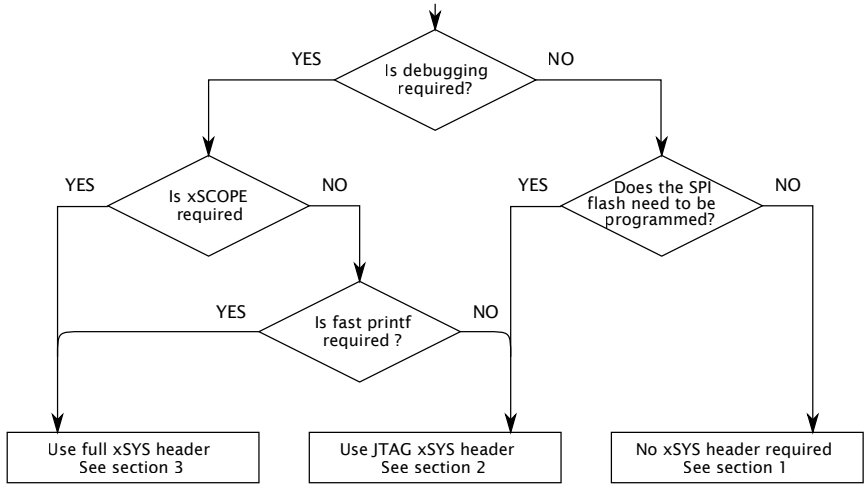


Figure 39: Decision diagram for the xSYS header

H.1 No xSYS header

The use of an xSYS header is optional, and may not be required for volume production designs. However, the XMOS toolchain expects the xSYS header; if you do not have an xSYS header then you must provide your own method for writing to flash/OTP and for debugging.

H.2 JTAG-only xSYS header

The xSYS header connects to an xTAG debugger, which has a 20-pin 0.1" female IDC header. The design will hence need a male IDC header. We advise to use a boxed header to guard against incorrect plug-ins. If you use a 90 degree angled header, make sure that pins 2, 4, 6, ..., 20 are along the edge of the PCB.

Connect pins 4, 8, 12, 16, 20 of the xSYS header to ground, and then connect:

- TDI to pin 5 of the xSYS header
- TMS to pin 7 of the xSYS header
- TCK to pin 9 of the xSYS header
- TDO to pin 13 of the xSYS header

The RST_N net should be open-drain, active-low, and have a pull-up to VDDIO.

H.3 Full xSYS header

For a full xSYS header you will need to connect the pins as discussed in Section H.2, and then connect a 2-wire xCONNECT Link to the xSYS header. The links can be found in the Signal description table (Section 4): they are labelled XL0, XL1, etc in the function column. The 2-wire link comprises two inputs and outputs, labelled ${}^{1}_{out}$, ${}^{0}_{out}$, ${}^{0}_{in}$, and ${}^{1}_{in}$. For example, if you choose to use XL0 for xSCOPE I/O, you need to connect up $XL0^{1}_{out}$, $XL0^{0}_{out}$, $XL0^{0}_{in}$, $XL0^{1}_{in}$ as follows:

- $XL0^{1}_{out}$ (X0D43) to pin 6 of the xSYS header with a 33R series resistor close to the device.
- $XL0^{0}_{out}$ (X0D42) to pin 10 of the xSYS header with a 33R series resistor close to the device.
- $XL0^{0}_{in}$ (X0D41) to pin 14 of the xSYS header.
- $XL0^{1}_{in}$ (X0D40) to pin 18 of the xSYS header.

I Schematics Design Check List

☑ This section is a checklist for use by schematics designers using the XU210-256-TQ128. Each of the following sections contains items to check for each design.

I.1 Power supplies

☐ VDDIO and OTP_VCC supply is within specification before the VDD (core) supply is turned on. Specifically, the VDDIO and OTP_VCC supply is within specification before VDD (core) reaches 0.4V (Section 12).

☐ The VDD (core) supply ramps monotonically (rises constantly) from 0V to its final value (0.95V - 1.05V) within 10ms (Section 12).

☐ The VDD (core) supply is capable of supplying 600mA (Section 12).

☐ PLL_AVDD is filtered with a low pass filter, for example an RC filter, see Section 12 .

I.2 Power supply decoupling

☐ The design has multiple decoupling capacitors per supply, for example at least four0402 or 0603 size surface mount capacitors of 100nF in value, per supply (Section 12).

☐ A bulk decoupling capacitor of at least 10uF is placed on each supply (Section 12).

I.3 Power on reset

☐ The RST_N and TRST_N pins are asserted (low) during or after power up. The device is not used until these resets have taken place.
As the errata in the datasheets show, the internal pull-ups on these two pins can occasionally provide stronger than normal pull-up currents. For this reason, an RC type reset circuit is discouraged as behavior would be unpredictable. A voltage supervisor type reset device is recommended to guarantee a good reset. This also has the benefit of resetting the system should the relevant supply go out of specification.

I.4 Clock

☐ The CLK input pin is supplied with a clock with monotonic rising edges and low jitter.

- ☐ You have chosen an input clock frequency that is supported by the device (Section 7).

I.5 Boot

- ☐ The device is connected to a QSPI flash for booting, connected to X0D01, X0D04..X0D07, and X0D10 (Section 8). If not, you must boot the device through OTP or JTAG, or set it to boot from SPI and connect a SPI flash.
- ☐ The Flash that you have chosen is supported by **xflash**, or you have created a specification file for it.

I.6 JTAG, XScope, and debugging

- ☐ You have decided as to whether you need an XSYS header or not (Section H)
- ☐ If you have not included an XSYS header, you have devised a method to program the SPI-flash or OTP (Section H).

I.7 GPIO

- ☐ You have not mapped both inputs and outputs to the same multi-bit port.
- ☐ Pins X0D04, X0D05, X0D06, and X0D07 are output only and are, after reset, pulled high and low appropriately (Section 8)

I.8 Multi device designs

Skip this section if your design only includes a single XMOS device.

- ☐ One device is connected to a SPI flash for booting.
- ☐ Devices that boot from link have MODE2 grounded and MODE3 NC. These device must have link XLB connected to a device to boot from (see 8).
- ☐ If you included an XSYS header, you have included buffers for RST_N, TRST_N, TMS, TCK, MODE2, and MODE3 (Section G).

J PCB Layout Design Check List

☑ This section is a checklist for use by PCB designers using the XS2-U10A-256-TQ128. Each of the following sections contains items to check for each design.

J.1 Ground Plane

☐ Multiple vias (eg, 9) have been used to connect the center pad to the PCB ground plane. These minimize impedance and conduct heat away from the device. (Section 12.4).

☐ Other than ground vias, there are no (or only a few) vias underneath or closely around the device. This create a good, solid, ground plane.

J.2 Power supply decoupling

☐ The decoupling capacitors are all placed close to a supply pin (Section 12).

☐ The decoupling capacitors are spaced around the device (Section 12).

☐ The ground side of each decoupling capacitor has a direct path back to the center ground of the device.

J.3 PLL_AVDD

☐ The PLL_AVDD filter (especially the capacitor) is placed close to the PLL_AVDD pin (Section 12).

K Associated Design Documentation

Document Title	Information	Document Number
Estimating Power Consumption For XS1-U Devices	Power consumption	
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

L Related Documentation

Document Title	Information	Document Number
The XMOS XS1 Architecture	ISA manual	X7879
XS1 Port I/O Timing	Port timings	X5821
xCONNECT Architecture	Link, switch and system information	X4249
XS1-U Link Performance and Design Guidelines	Link timings	
XS1-U Clock Frequency Control	Advanced clock control	

M Revision History

Date	Description
2015-03-20	Preliminary release
2015-04-14	Added RST to pins to be pulled hard, and removed reference to TCK from Errata Removed TRST_N references in packages that have no TRST_N
2015-05-06	Removed references tro DEBUG_N
2015-07-09	Updated electrical characteristics - Section 13
2015-08-19	Added I(USB_VDD) - Section 13 Added USB layout guidelines - Section 12