

# XS1-A8A-128-FB217 Datasheet

---

preliminary

**Table of Contents**

1	xCORE Multicore Microcontrollers	2
2	XS1-A8A-128-FB217 Features	4
3	Pin Configuration	5
4	Signal Description	6
5	Example Application Diagram	9
6	Product Overview	10
7	xCORE Tile Resources	11
8	Oscillator	12
9	Boot Procedure	13
10	Memory	17
11	Analog-to-Digital Converter	17
12	Supervisor Logic	18
13	Energy management	18
14	JTAG	21
15	Board Integration	22
16	Example XS1-A8A-128-FB217 Board Designs	25
17	DC and Switching Characteristics	29
18	Package Information	34
19	Ordering Information	35
	Appendices	36
A	Configuring the device	36
B	Processor Status Configuration	39
C	xCORE Tile Configuration	48
D	Digital Node Configuration	54
E	Analogue Node Configuration	61
F	ADC Configuration	65
G	Deep sleep memory Configuration	69
H	Oscillator Configuration	70
I	Real time clock Configuration	72
J	Power control block Configuration	72
K	XMOS USB Interface	85
L	Device Errata	85
M	JTAG, xSCOPE and Debugging	86
N	Schematics Design Check List	88
O	PCB Layout Design Check List	90
P	Associated Design Documentation	91
Q	Related Documentation	91
R	Revision History	92

---

## TO OUR VALUED CUSTOMERS

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

XMOS Ltd. is the owner or licensee of the information in this document and is providing it to you "AS IS" with no warranty of any kind, express or implied and shall have no liability in relation to its use. XMOS Ltd. makes no representation that the information, or any particular implementation thereof, is or will be free from any claims of infringement and again, shall have no liability in relation to any such claims.

XMOS and the XMOS logo are registered trademarks of XMOS Ltd in the United Kingdom and other countries, and may not be used without written permission. Company and product names mentioned in this document are the trademarks or registered trademarks of their respective owners.

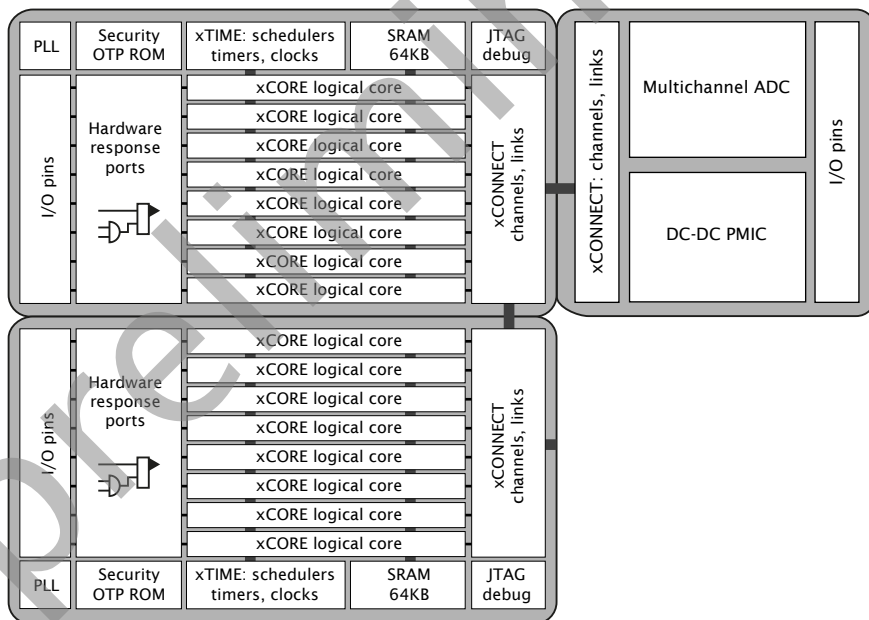
preliminary

# 1 xCORE Multicore Microcontrollers

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

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

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



**Figure 1:**  
XS1-A  
Series: 6-16  
core devices

XS1-A devices are available in a range of resource densities, package, performance and temperature grades depending on your needs. XS1-A devices range from 6-16 logical cores divided between one or two xCORE tiles, providing 400-1000 MIPS, up to 88 GPIO, and 64Kbytes or 128Kbytes of SRAM.

## 1.1 xSOFTip

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

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

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

## 1.2 xTIMEcomposer Studio

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

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

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

Information on using the tools is provided in a separate user guide, [X3766](#).

## 2 XS1-A8A-128-FB217 Features

- ▶ **Eight-Core Multicore Microcontroller with Advanced Multi-Core RISC Architecture**
  - Up to 1000 MIPS shared between up to 8 real-time logical cores across two tiles
  - Each logical core has:
    - Guaranteed throughput of 1/4 of tile MIPS
    - 16x32bit dedicated registers
  - 159 high-density 16/32-bit instructions
    - All have single clock-cycle execution (except for divide)
    - 32x32→64-bit MAC instructions for DSP, arithmetic and user-definable cryptographic functions
- ▶ **12b 1MSPS 8-channel SAR Analog-to-Digital Converter**
- ▶ **1 x LDO**
- ▶ **2 x DC-DC converters and Power Management Unit**
- ▶ **Watchdog Timer**
- ▶ **Onchip clocks/oscillators**
  - Crystal oscillator
  - 20MHz/31kHz silicon oscillators
- ▶ **Programmable I/O**
  - 90 general-purpose I/O pins, configurable as input or output
    - Up to 32 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
  - 64 channel ends for communication with other cores, on or off-chip
- ▶ **Memory**
  - 128KB internal single-cycle SRAM (max 64KB per tile) for code and data storage
  - 16KB internal OTP (max 8KB per tile) for application boot code
  - 128 bytes Deep Sleep Memory
- ▶ **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
- ▶ **Speed Grade**
  - 10: 1000 MIPS
  - 8: 800 MIPS
- ▶ **Power Consumption (typical)**
  - 600 mW at 500 MHz (typical)
  - Sleep Mode: 500  $\mu$ W
- ▶ **217-pin FBGA package 0.8 mm pitch**

### 3 Pin Configuration

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
A	X0D00	X0D02	X0D04	X0D06	X0D08	X0D10	X0D12	X0D14	X0D16	X0D18	X0D20	X0D22	X0D24	X0D25	X0D26	X0D28	X0D30	X0D32	VDDIO_OUT
B	X0D01	X0D03	X0D05	X0D07	X0D09	X0D11	X0D13	X0D15	X0D17	X0D19	X0D21	X0D23	X0D70	ADC_SAMPLE	X0D27	X0D29	X0D31	X0D33	X0D36
C	TDO	DEBUG_N																X0D38	X0D37
D	TCK	RST_N																X0D40	X0D39
E	TMS	TDI																X0D42	X0D41
F	MODE[2]	MODE[3]				GND	GND	GND	GND	GND	GND	GND	GND	GND				X0D34	X0D43/WAKE
G	AVDD	ADC7				GND	GND	GND	GND	GND	GND	GND	GND	GND				X1D00	X0D35
H	ADC5	ADC6				GND	GND	GND	GND	GND	GND	GND	GND	GND				X1D02	X1D01
J	ADC3	ADC4				GND	GND	GND	GND	GND	GND	GND	GND	GND				X1D04	X1D03
K	ADC1	ADC2				MODE[0]	MODE[1]	GND	GND	GND	GND	GND	GND	GND				X1D06	X1D05
L	NC	ADC0				OSC_EXT_N	MODE[4]	GND	GND	GND	GND	GND	GND	GND				X1D08	X1D07
M	XI/CLK	NC				NC	NC	GND	GND	GND	GND	GND	GND	GND				X1D10	X1D09
N	XO	NC				NC	NC	GND	GND	GND	GND	GND	GND	GND				X1D12	X1D11
P	NC	VSUP				GND	GND	GND	GND	GND	GND	GND	GND	GND				X1D14	X1D13
R	SW1	SW1																X1D16	X1D15
T	SW1	VDDCORE																X1D18	X1D17
U	VDDCORE	VDDCORE																X1D20	X1D19
V	PGND	PGND	VDDIO	PGND	VDD1V8	SW2	NC	X1D35	X1D43	X1D41	X1D39	X1D37	X1D33	X1D31	X1D29	X1D27	X1D25	X1D22	X1D21
W	VSUP	VSUP	VDDIO	PGND	VDD1V8	SW2	NC	X1D34	X1D42	X1D40	X1D38	X1D36	X1D32	X1D30	X1D28	X1D26	X1D24	X1D70	X1D23

### 4 Signal Description

Module	Signal	Function	Type	Active	Properties
PU=Pull Up, PD=Pull Down, ST=Schmitt Trigger Input, OT=Output Tristate, S=Switchable R <sub>S</sub> =Required for SPI boot (§9), R <sub>U</sub> =Required for USB-enabled devices (§K)					
Power	GND	Digital ground	GND	—	
	PGND	Power ground	GND	—	
	SW1	DCDC1 switched output voltage	PWR	—	
	SW2	DCDC2 switched output voltage	PWR	—	
	VDD1V8	1v8 voltage supply	PWR	—	
	VDDCORE	Core voltage supply	PWR	—	
	VDDIO	Digital I/O power	PWR	—	
	VDDIO_OUT	Digital I/O power out	PWR	—	
	VSUP	Power supply (3V3/5V0)	PWR	—	
Analog	ADC0	Analog input	Input	—	
	ADC1	Analog input	Input	—	
	ADC2	Analog input	Input	—	
	ADC3	Analog input	Input	—	
	ADC4	Analog input	Input	—	
	ADC5	Analog input	Input	—	
	ADC6	Analog input	Input	—	
	ADC7	Analog input	Input	—	
	ADC_SAMPLE	Sample Analog input	I/O	—	
	AVDD	Supply and reference voltage	PWR	—	
Clocks	MODE[4:0]	Boot mode select	Input	—	PU, ST
	OSC_EXT_N	Use Silicon Oscillator	Input	Low	ST
	XI/CLK	Crystal Oscillator/Clock Input	Input	—	
	XO	Crystal Oscillator Output	Output	—	
JTAG	DEBUG_N	Multi-chip debug	I/O	Low	PU
	TCK	Test clock	Input	—	PU, ST
	TDI	Test data input	Input	—	PU, ST
	TDO	Test data output	Output	—	PD, OT
	TMS	Test mode select	Input	—	PU, ST
Misc	RST_N	Global reset input	Input	Low	PU, ST
I/O	X0D00	P1A <sup>0</sup>	I/O	—	PD <sub>S</sub> , R <sub>S</sub>
	X0D01	XLA <sub>out</sub> <sup>4</sup> P1B <sup>0</sup>	I/O	—	PD <sub>S</sub> , R <sub>S</sub>
	X0D02	XLA <sub>out</sub> <sup>3</sup> P4A <sup>0</sup> P8A <sup>0</sup> P16A <sup>0</sup> P32A <sup>20</sup>	I/O	—	PD <sub>S</sub> , R <sub>U</sub>
	X0D03	XLA <sub>out</sub> <sup>2</sup> P4A <sup>1</sup> P8A <sup>1</sup> P16A <sup>1</sup> P32A <sup>21</sup>	I/O	—	PD <sub>S</sub> , R <sub>U</sub>
	X0D04	XLA <sub>out</sub> <sup>1</sup> P4B <sup>0</sup> P8A <sup>2</sup> P16A <sup>2</sup> P32A <sup>22</sup>	I/O	—	PD <sub>S</sub> , R <sub>U</sub>
	X0D05	XLA <sub>out</sub> <sup>0</sup> P4B <sup>1</sup> P8A <sup>3</sup> P16A <sup>3</sup> P32A <sup>23</sup>	I/O	—	PD <sub>S</sub> , R <sub>U</sub>
	X0D06	XLA <sub>in</sub> <sup>0</sup> P4B <sup>2</sup> P8A <sup>4</sup> P16A <sup>4</sup> P32A <sup>24</sup>	I/O	—	PD <sub>S</sub> , R <sub>U</sub>
	X0D07	XLA <sub>in</sub> <sup>1</sup> P4B <sup>3</sup> P8A <sup>5</sup> P16A <sup>5</sup> P32A <sup>25</sup>	I/O	—	PD <sub>S</sub> , R <sub>U</sub>
	X0D08	XLA <sub>in</sub> <sup>2</sup> P4A <sup>2</sup> P8A <sup>6</sup> P16A <sup>6</sup> P32A <sup>26</sup>	I/O	—	PD <sub>S</sub> , R <sub>U</sub>

(continued)

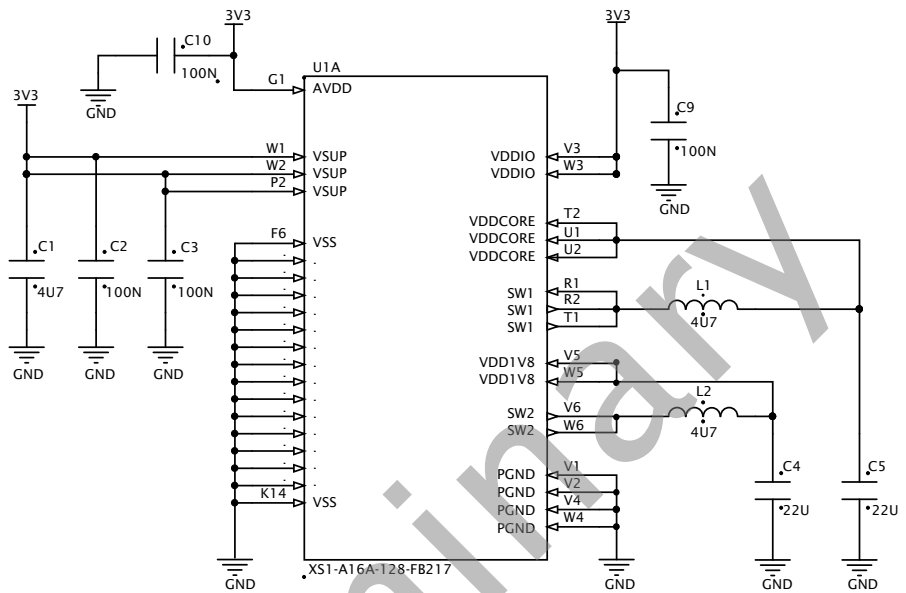


Module	Name	Function	Type	Active	Properties
I/O	X0D09	$XL A_{in}^3$ P4A <sup>3</sup> P8A <sup>7</sup> P16A <sup>7</sup> P32A <sup>27</sup>	I/O	—	PD <sub>S</sub> , RU
	X0D10	$XL A_{in}^4$ P1C <sup>0</sup>	I/O	—	PD <sub>S</sub> , R <sub>S</sub>
	X0D11	P1D <sup>0</sup>	I/O	—	PD <sub>S</sub> , R <sub>S</sub>
	X0D12	P1E <sup>0</sup>	I/O	—	PD <sub>S</sub> , RU
	X0D13	$XL B_{out}^4$ P1F <sup>0</sup>	I/O	—	PD <sub>S</sub> , RU
	X0D14	$XL B_{out}^3$ P4C <sup>0</sup> P8B <sup>0</sup> P16A <sup>8</sup> P32A <sup>28</sup>	I/O	—	PD <sub>S</sub> , RU
	X0D15	$XL B_{out}^2$ P4C <sup>1</sup> P8B <sup>1</sup> P16A <sup>9</sup> P32A <sup>29</sup>	I/O	—	PD <sub>S</sub> , RU
	X0D16	$XL B_{out}^1$ P4D <sup>0</sup> P8B <sup>2</sup> P16A <sup>10</sup>	I/O	—	PD <sub>S</sub> , RU
	X0D17	$XL B_{out}^0$ P4D <sup>1</sup> P8B <sup>3</sup> P16A <sup>11</sup>	I/O	—	PD <sub>S</sub> , RU
	X0D18	$XL B_{in}^0$ P4D <sup>2</sup> P8B <sup>4</sup> P16A <sup>12</sup>	I/O	—	PD <sub>S</sub> , RU
	X0D19	$XL B_{in}^1$ P4D <sup>3</sup> P8B <sup>5</sup> P16A <sup>13</sup>	I/O	—	PD <sub>S</sub> , RU
	X0D20	$XL B_{in}^2$ P4C <sup>2</sup> P8B <sup>6</sup> P16A <sup>14</sup> P32A <sup>30</sup>	I/O	—	PD <sub>S</sub> , RU
	X0D21	$XL B_{in}^3$ P4C <sup>3</sup> P8B <sup>7</sup> P16A <sup>15</sup> P32A <sup>31</sup>	I/O	—	PD <sub>S</sub> , RU
	X0D22	$XL B_{in}^4$ P1G <sup>0</sup>	I/O	—	PD <sub>S</sub> , RU
	X0D23	P1H <sup>0</sup>	I/O	—	PD <sub>S</sub> , RU
	X0D24	P1I <sup>0</sup>	I/O	—	PD <sub>S</sub>
	X0D25	P1J <sup>0</sup>	I/O	—	PD <sub>S</sub>
	X0D26	P4E <sup>0</sup> P8C <sup>0</sup> P16B <sup>0</sup>	I/O	—	PD <sub>S</sub> , RU
	X0D27	P4E <sup>1</sup> P8C <sup>1</sup> P16B <sup>1</sup>	I/O	—	PD <sub>S</sub> , RU
	X0D28	P4F <sup>0</sup> P8C <sup>2</sup> P16B <sup>2</sup>	I/O	—	PD <sub>S</sub> , RU
	X0D29	P4F <sup>1</sup> P8C <sup>3</sup> P16B <sup>3</sup>	I/O	—	PD <sub>S</sub> , RU
	X0D30	P4F <sup>2</sup> P8C <sup>4</sup> P16B <sup>4</sup>	I/O	—	PD <sub>S</sub> , RU
	X0D31	P4F <sup>3</sup> P8C <sup>5</sup> P16B <sup>5</sup>	I/O	—	PD <sub>S</sub> , RU
	X0D32	P4E <sup>2</sup> P8C <sup>6</sup> P16B <sup>6</sup>	I/O	—	PD <sub>S</sub> , RU
	X0D33	P4E <sup>3</sup> P8C <sup>7</sup> P16B <sup>7</sup>	I/O	—	PD <sub>S</sub> , RU
	X0D34	P1K <sup>0</sup>	I/O	—	PD <sub>S</sub>
	X0D35	P1L <sup>0</sup>	I/O	—	PD <sub>S</sub>
	X0D36	P1M <sup>0</sup> P8D <sup>0</sup> P16B <sup>8</sup>	I/O	—	PD <sub>S</sub>
	X0D37	P1N <sup>0</sup> P8D <sup>1</sup> P16B <sup>9</sup>	I/O	—	PD <sub>S</sub> , RU
	X0D38	P1O <sup>0</sup> P8D <sup>2</sup> P16B <sup>10</sup>	I/O	—	PD <sub>S</sub> , RU
	X0D39	P1P <sup>0</sup> P8D <sup>3</sup> P16B <sup>11</sup>	I/O	—	PD <sub>S</sub> , RU
	X0D40	P8D <sup>4</sup> P16B <sup>12</sup>	I/O	—	PD <sub>S</sub> , RU
	X0D41	P8D <sup>5</sup> P16B <sup>13</sup>	I/O	—	PD <sub>S</sub> , RU
	X0D42	P8D <sup>6</sup> P16B <sup>14</sup>	I/O	—	PD <sub>S</sub> , RU
	X0D43/WAKE	P8D <sup>7</sup> P16B <sup>15</sup>	I/O	—	PU <sub>S</sub> , RU
	X0D70	P32A <sup>19</sup>	I/O	—	PD <sub>S</sub>
	X1D00	P1A <sup>0</sup>	I/O	—	PD <sub>S</sub> , R <sub>S</sub>
	X1D01	$XL A_{out}^4$ P1B <sup>0</sup>	I/O	—	PD <sub>S</sub> , R <sub>S</sub>
	X1D02	$XL A_{out}^3$ P4A <sup>0</sup> P8A <sup>0</sup> P16A <sup>0</sup> P32A <sup>20</sup>	I/O	—	PD <sub>S</sub> , RU
	X1D03	$XL A_{out}^2$ P4A <sup>1</sup> P8A <sup>1</sup> P16A <sup>1</sup> P32A <sup>21</sup>	I/O	—	PD <sub>S</sub> , RU
	X1D04	$XL A_{out}^1$ P4B <sup>0</sup> P8A <sup>2</sup> P16A <sup>2</sup> P32A <sup>22</sup>	I/O	—	PD <sub>S</sub> , RU
	X1D05	$XL A_{out}^0$ P4B <sup>1</sup> P8A <sup>3</sup> P16A <sup>3</sup> P32A <sup>23</sup>	I/O	—	PD <sub>S</sub> , RU
	X1D06	$XL A_{in}^0$ P4B <sup>2</sup> P8A <sup>4</sup> P16A <sup>4</sup> P32A <sup>24</sup>	I/O	—	PD <sub>S</sub> , RU

(continued)

Module	Name	Function	Type	Active	Properties
I/O	X1D07	$XL A_{in}^1$ P4B <sup>3</sup> P8A <sup>5</sup> P16A <sup>5</sup> P32A <sup>25</sup>	I/O	—	PD <sub>S</sub> , RU
	X1D08	$XL A_{in}^2$ P4A <sup>2</sup> P8A <sup>6</sup> P16A <sup>6</sup> P32A <sup>26</sup>	I/O	—	PD <sub>S</sub> , RU
	X1D09	$XL A_{in}^3$ P4A <sup>3</sup> P8A <sup>7</sup> P16A <sup>7</sup> P32A <sup>27</sup>	I/O	—	PD <sub>S</sub> , RU
	X1D10	$XL A_{in}^4$ P1C <sup>0</sup>	I/O	—	PD <sub>S</sub> , R <sub>S</sub>
	X1D11	P1D <sup>0</sup>	I/O	—	PD <sub>S</sub> , R <sub>S</sub>
	X1D12	P1E <sup>0</sup>	I/O	—	PD <sub>S</sub> , RU
	X1D13	$XL B_{out}^4$ P1F <sup>0</sup>	I/O	—	PD <sub>S</sub> , RU
	X1D14	$XL B_{out}^3$ P4C <sup>0</sup> P8B <sup>0</sup> P16A <sup>8</sup> P32A <sup>28</sup>	I/O	—	PD <sub>S</sub> , RU
	X1D15	$XL B_{out}^2$ P4C <sup>1</sup> P8B <sup>1</sup> P16A <sup>9</sup> P32A <sup>29</sup>	I/O	—	PD <sub>S</sub> , RU
	X1D16	$XL B_{out}^1$ P4D <sup>0</sup> P8B <sup>2</sup> P16A <sup>10</sup>	I/O	—	PD <sub>S</sub> , RU
	X1D17	$XL B_{out}^0$ P4D <sup>1</sup> P8B <sup>3</sup> P16A <sup>11</sup>	I/O	—	PD <sub>S</sub> , RU
	X1D18	$XL B_{in}^0$ P4D <sup>2</sup> P8B <sup>4</sup> P16A <sup>12</sup>	I/O	—	PD <sub>S</sub> , RU
	X1D19	$XL B_{in}^1$ P4D <sup>3</sup> P8B <sup>5</sup> P16A <sup>13</sup>	I/O	—	PD <sub>S</sub> , RU
	X1D20	$XL B_{in}^2$ P4C <sup>2</sup> P8B <sup>6</sup> P16A <sup>14</sup> P32A <sup>30</sup>	I/O	—	PD <sub>S</sub> , RU
	X1D21	$XL B_{in}^3$ P4C <sup>3</sup> P8B <sup>7</sup> P16A <sup>15</sup> P32A <sup>31</sup>	I/O	—	PD <sub>S</sub> , RU
	X1D22	$XL B_{in}^4$ P1G <sup>0</sup>	I/O	—	PD <sub>S</sub> , RU
	X1D23	P1H <sup>0</sup>	I/O	—	PD <sub>S</sub> , RU
	X1D24	P1I <sup>0</sup>	I/O	—	PD <sub>S</sub>
	X1D25	P1J <sup>0</sup>	I/O	—	PD <sub>S</sub>
	X1D26	P4E <sup>0</sup> P8C <sup>0</sup> P16B <sup>0</sup>	I/O	—	PD <sub>S</sub> , RU
	X1D27	P4E <sup>1</sup> P8C <sup>1</sup> P16B <sup>1</sup>	I/O	—	PD <sub>S</sub> , RU
	X1D28	P4F <sup>0</sup> P8C <sup>2</sup> P16B <sup>2</sup>	I/O	—	PD <sub>S</sub> , RU
	X1D29	P4F <sup>1</sup> P8C <sup>3</sup> P16B <sup>3</sup>	I/O	—	PD <sub>S</sub> , RU
	X1D30	P4F <sup>2</sup> P8C <sup>4</sup> P16B <sup>4</sup>	I/O	—	PD <sub>S</sub> , RU
	X1D31	P4F <sup>3</sup> P8C <sup>5</sup> P16B <sup>5</sup>	I/O	—	PD <sub>S</sub> , RU
	X1D32	P4E <sup>2</sup> P8C <sup>6</sup> P16B <sup>6</sup>	I/O	—	PD <sub>S</sub> , RU
	X1D33	P4E <sup>3</sup> P8C <sup>7</sup> P16B <sup>7</sup>	I/O	—	PD <sub>S</sub> , RU
	X1D34	P1K <sup>0</sup>	I/O	—	PD <sub>S</sub>
	X1D35	P1L <sup>0</sup>	I/O	—	PD <sub>S</sub>
	X1D36	P1M <sup>0</sup> P8D <sup>0</sup> P16B <sup>8</sup>	I/O	—	PD <sub>S</sub>
	X1D37	P1N <sup>0</sup> P8D <sup>1</sup> P16B <sup>9</sup>	I/O	—	PD <sub>S</sub> , RU
	X1D38	P1O <sup>0</sup> P8D <sup>2</sup> P16B <sup>10</sup>	I/O	—	PD <sub>S</sub> , RU
	X1D39	P1P <sup>0</sup> P8D <sup>3</sup> P16B <sup>11</sup>	I/O	—	PD <sub>S</sub> , RU
	X1D40	P8D <sup>4</sup> P16B <sup>12</sup>	I/O	—	PD <sub>S</sub> , RU
	X1D41	P8D <sup>5</sup> P16B <sup>13</sup>	I/O	—	PD <sub>S</sub> , RU
	X1D42	P8D <sup>6</sup> P16B <sup>14</sup>	I/O	—	PD <sub>S</sub> , RU
	X1D43	P8D <sup>7</sup> P16B <sup>15</sup>	I/O	—	PU <sub>S</sub> , RU
	X1D70	P32A <sup>19</sup>	I/O	—	PD <sub>S</sub>

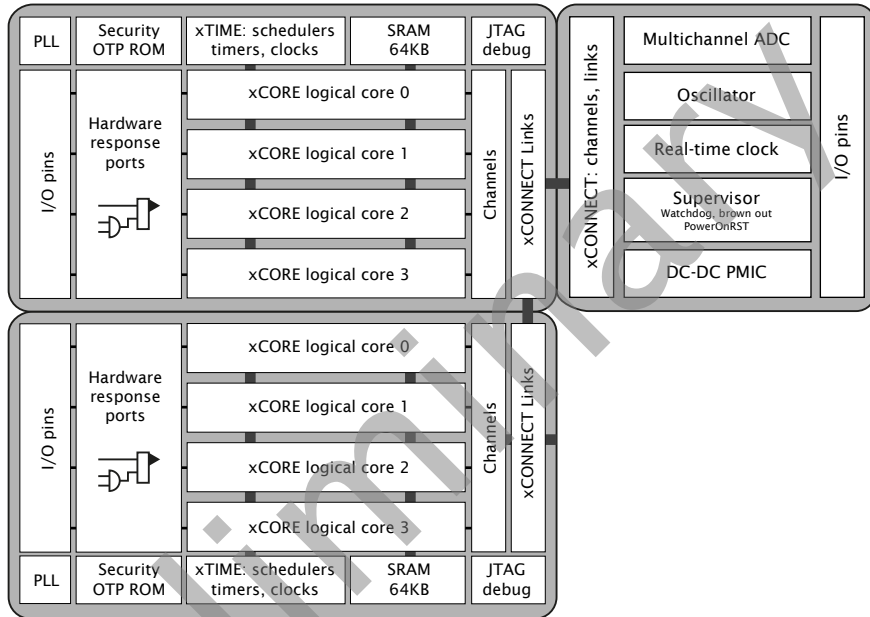
### 5 Example Application Diagram



**Figure 2:**  
Simplified  
Reference  
Schematic

## 6 Product Overview

The XS1-A8A-128-FB217 comprises a digital and an analog node, as shown in Figure 3. The digital node comprises an xCORE Tile, a Switch, and a PLL (Phase-locked-loop). The analog node comprises a multi-channel ADC (Analog to Digital Converter), deep sleep memory, an oscillator, a real-time counter, and power supply control.



**Figure 3:**  
Block Diagram

All communication between the digital and analog node takes place over a link that is connected to the Switch of the digital node. As such, the analog node can be controlled from any node on the system. The analog functions can be configured using a set of node configuration registers, and a set of registers for each of the peripherals.

The device can be programmed using high-level languages such as C/C++ and the XMOS-originated XC language, which provides extensions to C that simplify the control over concurrency, I/O and timing, or low-level assembler.

### 6.1 xCore Tile

The xCORE Tile is a flexible multicore microcontroller component with tightly integrated I/O and on-chip memory. The tile contains multiple logical cores that run simultaneously, each of which is guaranteed a slice of processing power and can execute computational code, control software and I/O interfaces. The logical cores use channels to exchange data within a tile or across tiles. The tiles are connected via an integrated switch network, called xCONNECT, which uses a

proprietary physical layer protocol and can also be used to add additional resources to a design. The I/O pins are driven using intelligent ports that can serialize data, interpret strobe signals and wait for scheduled times or events, making the device ideal for real-time control applications.

## 6.2 ADC and Power Management

Each XS1-A8A-128-FB217 device includes a set of analog components, including a 12b, 8-channel ADC, power management unit, watchdog timer, real-time counter and deep sleep memory. The device reduces the number of additional external components required and allows designs to be implemented using simple 2-layer boards.

## 7 xCORE Tile Resources

### 7.1 Logical cores, Synchronizers and Locks

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

**Figure 4:**  
Logical core performance

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

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

### 7.2 Channel Ends, Links and Switch

Logical cores communicate using point-to-point connections formed between two channel ends. Between tiles, channel communications are implemented over xConnect Links and routed through switches. The links operate in either 2 wires per direction or 5 wires per direction mode, depending on the amount of bandwidth required. Circuit switched, streaming and packet switched data can both be supported efficiently. Streams provide the fastest possible data rates between tiles, 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.

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

### 7.3 Ports and Clock Blocks

Ports provide an interface between the logical cores and I/O pins. The XS1-A8A-128-FB217 includes a combination of 1bit, 4bit and 8bit ports. In addition the wider ports are partially or fully bonded out making the connected pins available for I/O or xCONNECT links. All pins of a port provide either output or input. Signals in different directions cannot be mapped onto the same port.

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

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

### 7.4 Processor Timers

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

## 8 Oscillator

The oscillator block provides:

- ▶ An oscillator circuit. Together with an external resonator (crystal or ceramic), the oscillator circuit can provide a clock-source for both the real-time counter and the xCORE Tile. The external resonator can be chosen by the designer to have the appropriate frequency and accuracy. If desired, an external oscillator can be used on the XI/CLK input pin.
- ▶ A 20 MHz silicon oscillator. This enables the device to boot and execute code without requiring an external crystal. The silicon oscillator is not as accurate as an external crystal.
- ▶ A 31,250 Hz oscillator. This enables the real-time counter to operate whilst the device is in low-power mode. This oscillator is not as accurate as an external crystal.

The oscillator can be controlled through package pins, a set of peripheral registers, and a digital node control register.

A package pin OSC\_EXT\_N is used to select the oscillator to use on boot. It must be grounded to select an external resonator or connected to VDDIO to select the on-chip 20 MHz oscillator. If an external resonator is used, then it must be in the

range 5-100 MHz. Two more package pins, MODE0 and MODE1 are used to inform the node of the frequency.

The analog node runs at the frequency provided by the oscillator. Hence, increasing the clock frequency will speed up operation of the analog node, and will speed up communicating data with the digital node. The digital node has a PLL.

The PLL creates a high-speed clock that is used for the switch, tile, and reference clock. The PLL multiplication value is selected through the two MODE pins, and can be changed by software to speed up the tile or use less power. The MODE pins are set as shown in Figure 5:

**Figure 5:**  
PLL multiplier values and MODE pins

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

Figure 5 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.

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

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

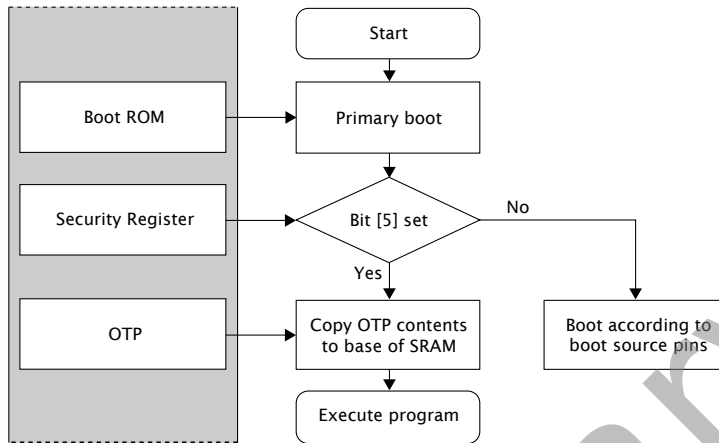
## 9 Boot Procedure

The device is kept in reset by driving RST\_N low. When in reset, all GPIO pins are high impedance. When the device is taken out of reset by releasing RST\_N the processor starts its internal reset process. After approximately 750,000 input clocks, all GPIO pins have their internal pull-resistor enabled, and the processor boots at a clock speed that depends on MODE0 and MODE1.

The processor boot procedure is illustrated in Figure 6. In normal usage, MODE[4:2] controls the boot source according to the table in Figure 7. If bit 5 of the security register (see §10.1) is set, the device boots from OTP.

The boot image has the following format:

- ▶ A 32-bit program size *s* in words.



**Figure 6:**  
Boot procedure

MODE [4]	MODE [3]	MODE [2]	Boot Source
X	0	0	None: Device waits to be booted via JTAG
X	0	1	Reserved
0	1	0	Tile0 boots from link B, Tile1 from channel end 0 via Tile0
0	1	1	Tile0 boots from SPI, Tile1 from channel end 0 via Tile0
1	1	0	Tile0 and Tile1 independently enable link B and internal links (E, F, G, H), and boot from channel end 0
1	1	1	Tile0 and Tile 1 boot from SPI independently

**Figure 7:**  
Boot source pins

- ▶ 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.

### 9.1 Boot from SPI

If set to boot from SPI, the processor enables the four pins specified in Figure 8, 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 8:**  
SPI 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.

## 9.2 Boot from xConnect Link

If set to boot from an xConnect Link, the processor enables Link B around 200 ns after the boot process starts. Enabling the Link switches off the pull-down on resistors X0D16..X0D19, drives X0D16 and X0D17 low (the initial state for the Link), and monitors pins X0D18 and X0D19 for boot-traffic. X0D18 and X0D19 must be low at this stage. If the internal pull-down is too weak to drain any residual charge, external pull-downs of 10K may be required on those pins.

The boot-rom on the core will then:

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

## 9.3 Boot from OTP

If an xCORE tile is set to use secure boot (see Figure 6), 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.

#### 9.4 Security register

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

**Figure 9:**  
Security register features

## 10 Memory

### 10.1 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`.

### 10.2 SRAM

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

### 10.3 Deep Sleep Memory

The XS1-A8A-128-FB217 device includes 128 bytes of deep sleep memory for state storage during sleep mode. Data stored in the memory is lost if the device is powered down.

## 11 Analog-to-Digital Converter

The device has a 12-bit 1MSample/second Successive Approximation Register (SAR) Analogue to Digital Converter (ADC). It has 8 input pins which are multiplexed into the ADC. The sampling of the ADC is controlled using the ADC\_SAMPLE pin that should be wired to a GPIO pin, for example X0D24 (port 11). The sampling is triggered either by writing to the port, or by driving the pin externally. On each rising edge of the sample pin the ADC samples, holds and converts the data value from one of the analog input pins. Each of the 8 inputs can be enabled individually. Each of the enabled analog inputs is sampled in turn, on successive rising edges of the sample pin. The data is transmitted to the channel-end that the user configures during initialization of the ADC. Data is transmitted over the channel in individual packets, or in packets that contain multiple consecutive samples. The ADC uses an external reference voltage, nominally 3V3, which represents the full range of the ADC. The ADC configuration registers are documented in Appendix F.

The minimum latency for reading a value from the ADC into the xCORE register is shown in Figure 10:

**Figure 10:**  
Minimum  
latency to  
read sample  
from ADC to  
xCORE

Sample	Tile clock frequency	Start of packet	Subsequent samples
32-bit	500 MHz	840 ns	710 ns
32-bit	400 MHz	870 ns	740 ns
16-bit	500 MHz	770 ns	640 ns
16-bit	400 MHz	800 ns	670 ns

## 12 Supervisor Logic

An independent supervisor circuit provides power-on-reset, brown-out, and watchdog capabilities. This facilitates the design of systems that fail gracefully, whilst keeping BOM costs down.

The reset supervisor holds the chip in reset until all power supplies are good. This provides a power-on-reset (POR). An external reset is optional and the pin RST\_N can be left not-connected.

If at any time any of the power supplies drop because of too little supply or too high a demand, the power supervisor will bring the chip into reset until the power supplies have been restored. This will reboot the system as if a cold-start has happened.

The 16-bit watchdog timer provides 1ms accuracy and runs independently of the real-time counter. It can be programmed with a time-out of between 1 ms and 65 seconds (Appendix E). If the watchdog is not set before it times out, the XS1-A8A-128-FB217 is reset. On boot, the program can read a register to test whether the reset was due to the watchdog. The watchdog timer is only enabled and clocked whilst the processor is in the AWAKE power state.

## 13 Energy management

XS1-A8A-128-FB217 devices can be powered by:

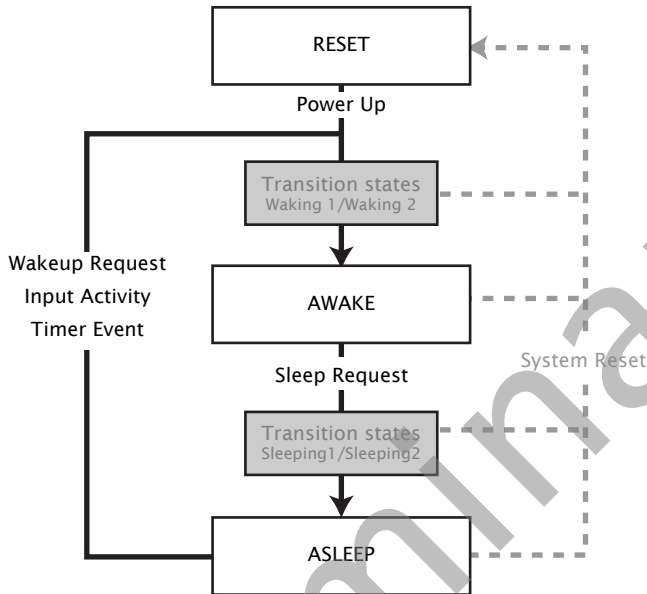
- ▶ An external 5v core and 3.3v I/O supply.
- ▶ A single 3.3v supply.

### 13.1 DC-DC

XS1-A8A-128-FB217 devices include two DC-DC buck converters which can be configured to take input voltages between 3.3-5V power supply and output circuit voltages (nominally 1.8V and 1.0V) required by the analog peripherals and digital node.

### 13.2 Power mode controller

The device transitions through multiple states during the power-up and powerdown process.



**Figure 11:**  
XS1-A8A-128-FB217 Power Up States and Transitions

The device is quiescent in the ASLEEP state, and is running in the AWAKE state. The other states allow a controlled transition between AWAKE and ASLEEP.

A transition from AWAKE state to ASLEEP state is instigated by a write to the general control register. Sleep requests must only be made in the AWAKE state.

A transition from the ASLEEP state into the AWAKE state is instigated by a wakeup request triggered by an input, or a timer. The device only responds to a wakeup stimulus in the ASLEEP state. If wakeup stimulus occurs whilst transitioning from AWAKE to ASLEEP, the appropriate response occurs when the ASLEEP state is reached.

Configuration is through a set of registers documented in Appendix J.

### 13.3 Deep Sleep Modes and Real-Time Counter

The normal mode in which the XS1-A8A-128-FB217 operates is the AWAKE mode. In this mode, all cores, memory, and peripherals operate as normal. To save power, the XS1-A8A-128-FB217 can be put into a deep sleep mode, called ASLEEP, where the digital node is powered down, and most peripherals are powered down. The XS1-A8A-128-FB217 will stay in the ASLEEP mode until one of two conditions:

1. An external pin is asserted or deasserted (set by the program);

2. The 64-bit real-time counter reaches a value set by the program; or

When the chip is awake, the real-time counter counts the number of clock ticks on the oscillator. As such, the real-time counter will run at a fixed ratio, but synchronously with the 100 MHz timers on the xCORE Tile. When asleep, the real-time counter can be automatically switched to the 31,250 Hz silicon oscillator to save power (see Appendix H). To ensure that the real-time counter increases linearly over time, a programmable value is added to the counter on every 31,250 Hz clock-tick. This means that the clock will run at a granularity of 31,250 Hz but still maintain real-time in terms of the frequency of the main oscillator. If an accurate clock is required, even whilst asleep, then an external crystal or oscillator shall be provided that is used in both AWAKE and ASLEEP state.

The designer has to make a trade-off between accuracy of clocks when asleep and awake, costs, and deep-sleep power consumption. Four example designs are shown in Figure 12.

**Figure 12:**  
Example trade-offs in oscillator selection

Clocks used		Power Asleep	BOM costs	Accuracy	
Awake	Asleep			Awake	Asleep
20 Mhz SiOsc	31,250 SiOsc	lowest	lowest	lowest	lowest
24 MHz Crystal	31,250 SiOsc	lowest	medium	highest	lowest
5 MHz ext osc	5 MHz ext osc	medium	highest	highest	highest
24 MHz Crystal	24 MHz crystal	highest	medium	highest	highest

During deep-sleep, the program can store some state in 128 bytes of Deep Sleep Memory.

### 13.4 Requirements during sleep mode

Whilst in sleep mode, the device must still be powered as normal over 3V3 or 5V0 on VSUP, and 3V3 on VDDIO; however it will draw less power on both VSUP and VDDIO.

For best results (lowest power):

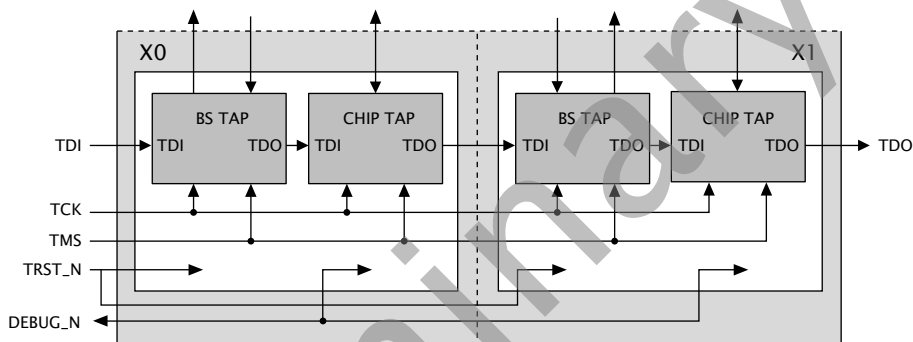
- ▶ The XTAL bias and XTAL oscillators should be switched off.
- ▶ The sleep register should be configured to
  - ▶ Disable all power supplies except DCDC2.
  - ▶ Set all power supplies to PFM mode
  - ▶ Mask the clock
  - ▶ Assert reset
- ▶ All GPIO and JTAG pins should be quiescent, and none should be driven against a pull-up or pull-down.
- ▶ 3V3 should be supplied as the input voltage to VSUP.

This will result in a power consumption of less than 100 uA on both VSUP and VDDIO.

If any power supply loses power-good status during the asleep-to-awake or awake-to-asleep transitions, a system reset is issued.

## 14 JTAG

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



**Figure 13:**  
JTAG chain structure

The JTAG chain structure is illustrated in Figure 13. Directly after reset, three TAP controllers are present in the JTAG chain for each xCORE Tile: the debug TAP, the boundary scan TAP and the processor TAP. The debug TAP provides access into the peripherals including the ADC. The boundary scan TAP is a standard 1149.1 compliant TAP that can be used for boundary scan of the I/O pins. The processor TAP provides access into the xCORE Tile, switch and OTP for loading code and debugging.

The JTAG module can be reset by holding TMS high for five clock cycles.

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

The JTAG device identification register can be read by using the IDCODE instruction. Its contents are specified in Figure 14.

**Figure 14:**  
IDCODE return value

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

The JTAG usercode register can be read by using the USERCODE instruction. Its contents are specified in Figure 15. The OTP User ID field is read from bits [22:31] of the security register on xCORE Tile 0, see §10.1 (all zero on unprogrammed devices).

**Figure 15:**  
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	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0										2				C										0													

## 15 Board Integration

XS1-A8A-128-FB217 devices are optimized for layout on low cost, 2 layer PCBs using standard design rules. Careful layout is required to maximize the device performance. XMOS therefore recommends that the guidelines in this section are followed when laying out boards using the device.

The XS1-A8A-128-FB217 includes two DC-DC buck converters that take input voltages between 3.3-5V and output the 1.8V and 1.0V circuits required by the digital core and analogue peripherals. The DC-DC converters should have a 4.7uF X5R or X7R ceramic capacitor and a 100nF X5R or X7R ceramic capacitor on the VSUP input pins W1 and W2. These capacitors must be placed as close as possible to the those pins (within a maximum of 5mm), with the routing optimized to minimize the inductance and resistance of the traces.

The SW output pin must have an LC filter on the output with a 4.7uH inductor and 22uF X5R capacitor. The capacitor must have maximum ESR value of 0.015R, and the inductor should have a maximum DCR value of 0.07R. A list of suggested inductors is in Figure 16.

**Figure 16:**  
Example 4.7  
µH inductors

	Part number	Current	Max DCR	Package
Wurth	744043004	1550 mA	70 mΩ	4.8 x 4.8 mm
Murata	LQH55DN4R7M03L	2700 mA	57 mΩ	5750 (2220)

The traces from the SW output pins to the inductor and from the output capacitor back to the VDD pins must be routed to minimize the coupling between them.

The power supplies must be brought up monotonically and input voltages must not exceed specification at any time.

The VDDIO supply to the XS1-A8A-128-FB217 requires a 100nF X5R or X7R ceramic decoupling capacitor placed as close as possible to the supply pins. VDDIO\_OUT is the switched IO supply; it is only supplied when the chip is AWAKE. This pin can be used to provide extra decoupling, or it can be used to switch other devices off during sleep mode, for example a SPI flash. No more than 240 mA should be drawn on VDDIO at any time: this includes any supply sourced statically (e.g., driving a LED from a GPIO pin), any dynamic power consumption (e.g., toggling a GPIO pin at a high frequency) and any devices powered through VDDIO\_OUT.



If the ADC Is used, it requires a 100nF X5R or X7R ceramic decoupling capacitor placed as close as possible to the AVDD pin. Care should be taken to minimize noise on these inputs, and if necessary an extra 10uF decoupling capacitor and ferrite bead can be used to remove noise from this supply.

The crystal oscillator requires careful routing of the XI / XO nodes as these are high impedance and very noise sensitive. Hence, the traces should be as wide and short as possible, and routed over a continuous ground plane. They should not be routed near noisy supply lines or clocks. The device has a load capacitance of 18pF for the crystal. Care must be taken, so that the inductance and resistance of the ground returns from the capacitors to the ground of the device is minimized.

### 15.1 Land patterns and solder stencils

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

Solder paste and ground via recommendations are based on our engineering and development kit board production. They have been found to work and optimized as appropriate to achieve a high yield. 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 217 pin Fine Ball Grid Array package on a 0.8mm pitch with 0.4mm balls.

An example land pattern is shown in Figure 17.

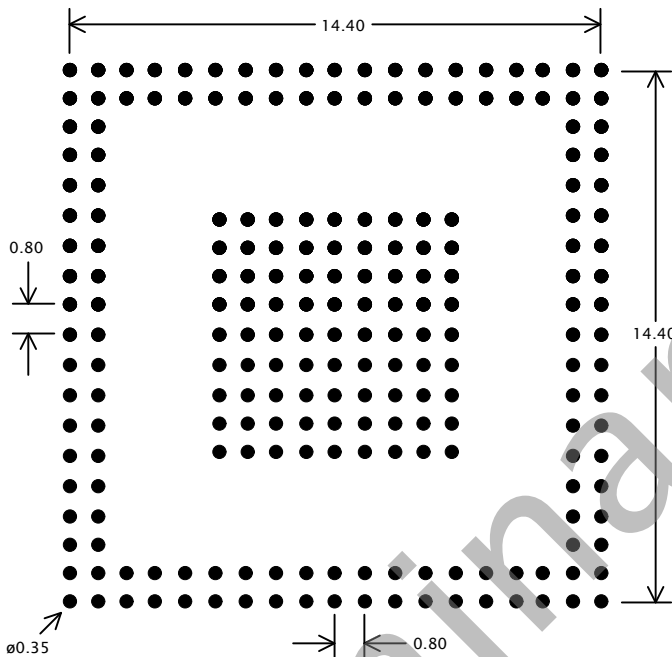
Pad widths and spacings are such that solder mask can still be applied between the pads using standard design rules. This is highly recommended to reduce solder shorts.

### 15.2 Ground and Thermal Vias

Vias next to every other ground ball into the ground plane of the PCB are recommended for a low inductance ground connection and good thermal performance. Vias with with a 0.6mm diameter annular ring and a 0.3mm drill would be suitable.

### 15.3 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



**Figure 17:**  
Example land pattern

pressure of moisture. Example damage can include bond wire damage, die lifting, internal or external package cracks and/or delamination.

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

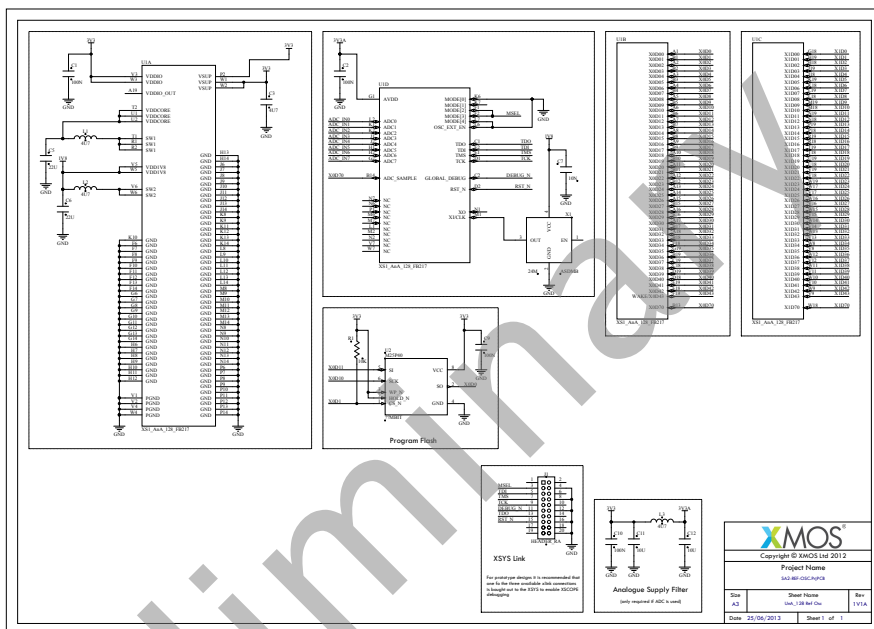
## 16 Example XS1-A8A-128-FB217 Board Designs

This section shows example schematics and layout for a 2-layer PCB.

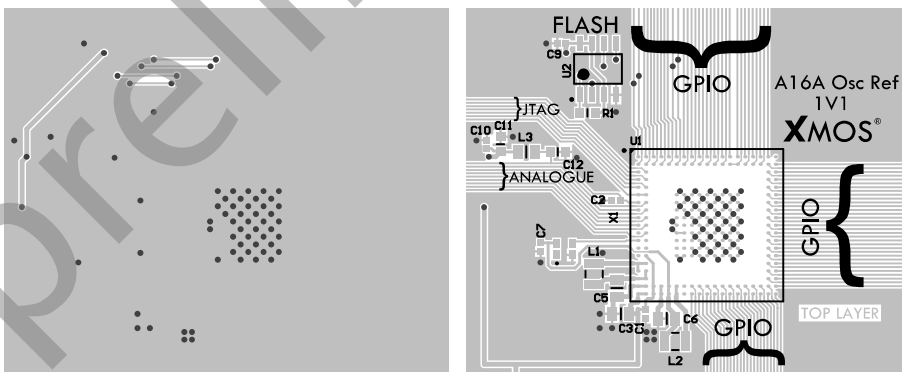
- ▶ Figures 18 shows example schematics and layout. It uses a 24 MHz crystal for the clock, and an SPI flash for booting. XS1-A8A-128-FB217 is powered directly from 3V3.
- ▶ Figures 19 shows example schematics and layout for a design that uses an oscillator rather than a crystal. If required a 3V3 oscillator can be used (for example when sharing an oscillator with other parts of the design), but a resistor bridge must be included to reduce the XI/CLK input from 3V3 to 1V8.
- ▶ Figure 20 shows example schematics and layout for a design that runs off the internal 20 MHz oscillator. The XS1-A8A-128-FB217 is powered directly from 3V3.

Flash, AVDD, RST, and JTAG connectivity are all optional. Flash can be removed if the processor boots from OTP. The AVDD decoupler and wiring can be removed if the ADC is not used. RST\_N and all JTAG wiring can be removed if debugging is not required (see Appendix M)





**Figure 19:**  
Example Oscillator schematic, with top and bottom layout of a 2-layer PCB



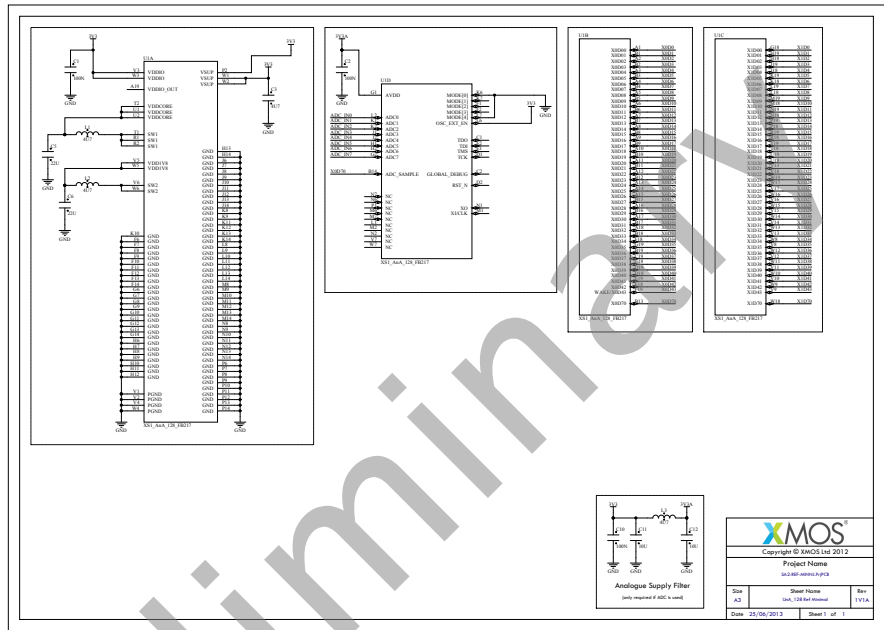
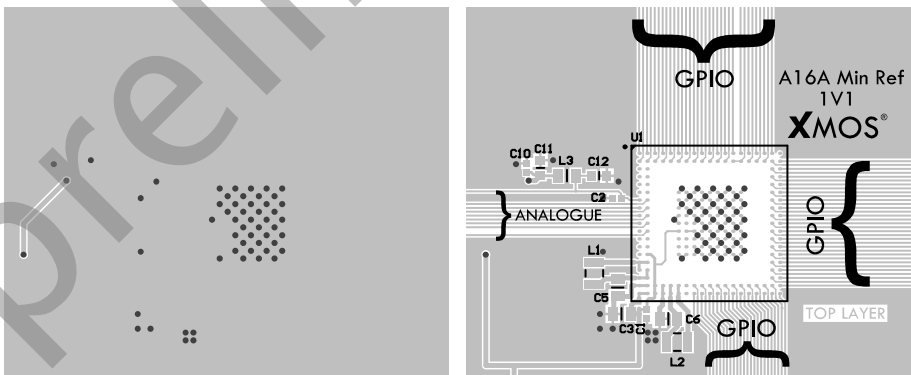


Figure 20: Example minimal system schematic, with top and bottom layout of a 2-layer PCB



## 17 DC and Switching Characteristics

### 17.1 Operating Conditions

Symbol	Parameter	MIN	TYP	MAX	UNITS	Notes
VSUP	Power Supply (3.3V Mode) (Speed Grade 10)	3.00	3.30	3.60	V	
	Power Supply (5V Mode) (Speed Grade 10)	4.50	5.00	5.50	V	
VDDIO	I/O supply voltage	3.00	3.30	3.60	V	
AVDD	Analog Supply and Reference Voltage	3.00	3.30	3.60	V	
Cl	xCORE Tile I/O load capacitance			25	pF	
Ta	Ambient operating temperature	0		70	°C	
Tj	Junction temperature			125	°C	
Tstg	Storage temperature	-65		150	°C	

**Figure 21:**  
Operating conditions

### 17.2 DC1 Characteristics

Symbol	Parameter	MIN	TYP	MAX	UNITS	Notes
VDDCORE	Tile Supply Voltage	0.90	1.00	1.10	V	
V(RIPPLE)	Ripple Voltage (peak to peak)		10	40	mV	
V(ACC)	Voltage Accuracy	-1		1	%	
F(S)	Switching Frequency		1		MHz	
F(SVAR)	Variation in Switching Frequency	-10		10	%	
Effic	Efficiency		80		%	
PGT(HIGH)	Powergood Threshold (High)		95		%/VDDCORE	
PGT(LOW)	Powergood Threshold (Low)		80		%/VDDCORE	

**Figure 22:**  
DC1 characteristics

### 17.3 DC2 Characteristics

Symbol	Parameter	MIN	TYP	MAX	UNITS	Notes
VDD1V8	1V8 Supply Voltage		1.80		V	
V(RIPPLE)	Ripple Voltage (peak to peak)		10	40	mV	
V(ACC)	Voltage Accuracy	-1		1	%	
F(S)	Switching Frequency		1		MHz	
F(SVAR)	Variation in Switching Frequency	-10		10	%	
Effic	Efficiency		80		%	
PGT(HIGH)	Powergood Threshold (High)		95		%/VDD1V8	
PGT(LOW)	Powergood Threshold (Low)		80		%/VDD1V8	

**Figure 23:**  
DC2 characteristics

### 17.4 ADC Characteristics

Symbol	Parameter	MIN	TYP	MAX	UNITS	Notes
N	Resolution		12		bits	
Fs	Conversion Speed			1	MSPS	
Nch	Number of Channels		8			
Vin	Input Range	0		AVDD	V	
DNL	Differential Non Linearity	-1		1.5	LSB	
INL	Integral Non Linearity	-4		4	LSB	
E(GAIN)	Gain Error	-10		10	LSB	
E(OFFSET)	Offset Error	-3		3	mV	
T(PWRUP)	Power time for ADC Clock Fclk			7	1/Fclk	
ENOB	Effective Number of bits		10			

**Figure 24:**  
ADC characteristics

### 17.5 Digital I/O Characteristics

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

**Figure 25:**  
Digital I/O characteristics



### 17.6 ESD Stress Voltage

**Figure 26:**  
ESD stress voltage

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

### 17.7 Device Timing Characteristics

**Figure 27:**  
Device timing characteristics

Symbol	Parameter	MIN	TYP	MAX	UNITS	Notes
T(RST)	Reset pulse width	5			µs	
T(INIT)	Initialisation (On Silicon Oscillator) (Speed Grade 10)			TBC	ms	
	Initialisation (Crystal Oscillator) (Speed Grade 10)			TBC	ms	
T(WAKE)	Wake up time (Sleep to Active)			TBC	ms	
T(SLEEP)	Sleep Time (Active to Sleep)			TBC	ms	

### 17.8 Crystal Oscillator Characteristics

**Figure 28:**  
Crystal oscillator characteristics

Symbol	Parameter	MIN	TYP	MAX	UNITS	Notes
F(FO)	Input Frequency	5		30	MHz	

### 17.9 External Oscillator Characteristics

**Figure 29:**  
External oscillator characteristics

Symbol	Parameter	MIN	TYP	MAX	UNITS	Notes
F(EXT)	External Frequency			100	MHz	
V(IH)	Input high voltage	1.60	1.80	2.00	V	
V(IL)	Input low voltage			0.4	V	
V(ACC)	Voltage Accuracy	TBC		TBC	%	

### 17.10 Power Consumption

**Figure 30:**  
xCORE Tile currents

Symbol	Parameter	MIN	TYP	MAX	UNITS	Notes
P(AWAKE)	Active Power for awake states	TBC	600	TBC	mW	
P(SLEEP)	Power when asleep	TBC	500	TBC	µW	

### 17.11 Clock

**Figure 31:**  
Clock

Symbol	Parameter	MIN	TYP	MAX	UNITS	Notes
f(MAX)	Processor clock frequency			500	MHz	

### 17.12 Processor I/O AC Characteristics

**Figure 32:**  
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](#).

### 17.13 xConnect Link Performance

**Figure 33:**  
Link performance

Symbol	Parameter	MIN	TYP	MAX	UNITS	Notes
B(2blinkP)	2wire link bandwidth (packetized)			103	MBit/s	
B(5blinkP)	5wire link bandwidth (packetized)			271	MBit/s	
B(2blinkS)	2wire link bandwidth (streaming)			125	MBit/s	
B(5blinkS)	5wire link bandwidth (streaming)			313	MBit/s	

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.

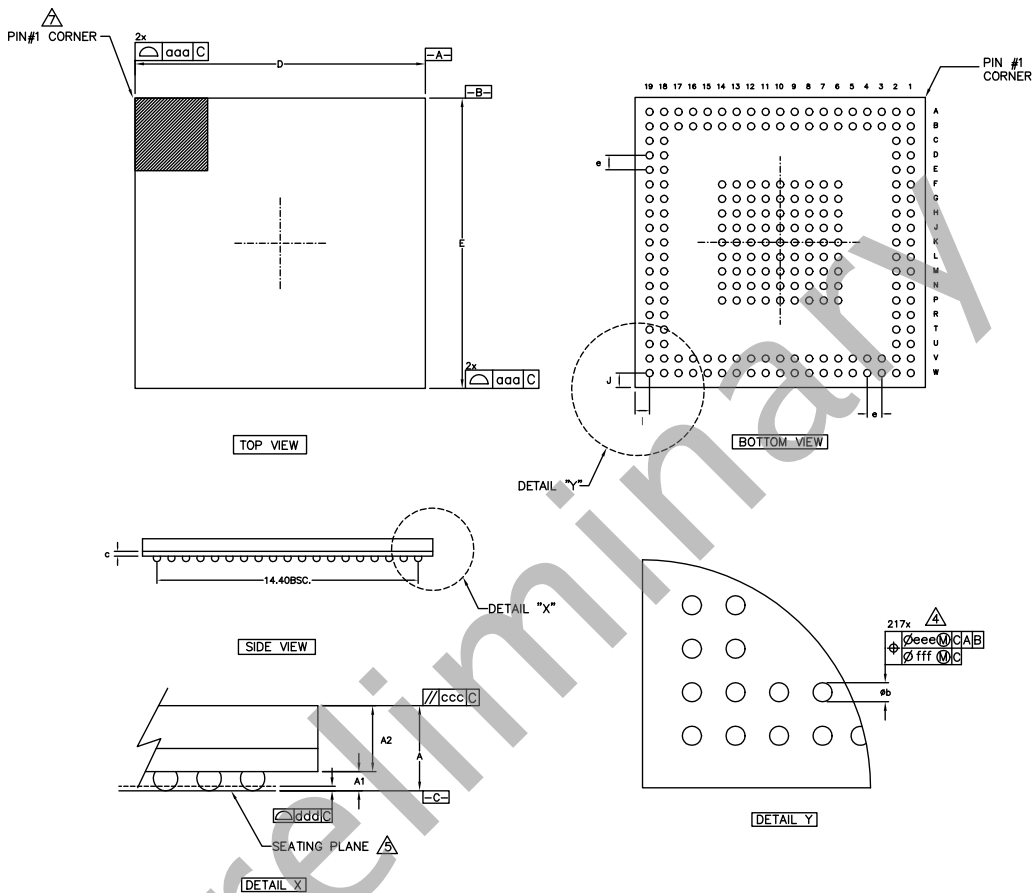
### 17.14 JTAG Timing

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

**Figure 34:**  
JTAG timing

Preliminary

### 18 Package Information

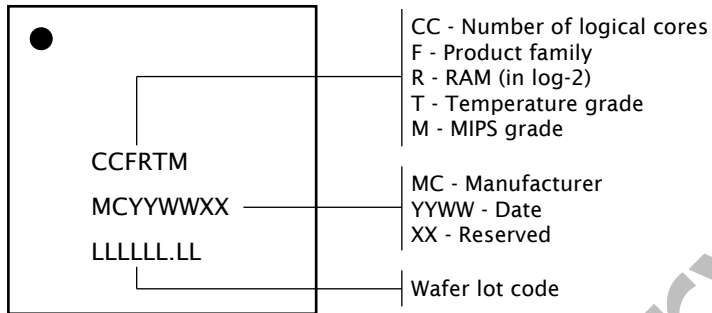


**NOTE:**

1. ALL DIMENSIONS ARE IN MILLIMETERS.
2. "e" REPRESENTS THE BASIC SOLDER BALL GRID PITCH.
3. "m" REPRESENTS THE MAXIMUM SOLDER BALL MATRIX SIZE.
4. DIMENSIONS "b" IS MEASURED AT THE MAXIMUM SOLDER BALL DIAMETER PARALLEL TO PRIMARY DATUM [C].
5. PRIMARY DATUM [C] AND SEATING PLANE ARE DESIGNED BY THE SPHERICAL CROWNS OF THE SOLDER BALLS.
6. DIMENSIONING AND TOLERANCING PER ASME Y14.5M-1994
7. A1 CORNER MUST BE IDENTIFIED BY INK OR LASER MARK.
8. PACKAGE DIMENSIONS CONFORM TO JEDEC REGISTRATION MO-275.

SYMBOL	MIN.	NOM.	MAX.
A	1.16	1.26	1.36
A1	0.25	0.30	0.35
A2	0.91	0.96	1.01
D	15.90	16.00	16.10
E	15.90	16.00	16.10
J	0.80 REF.		
M	19 X 19<DEPOPULATED>		
aaa			0.15
ccc			0.20
ddd			0.10
eee			0.15
fff			0.08
b	0.35	0.40	0.45
e	0.80 BSC.		
c	0.26 REF.		

### 18.1 Part Marking



**Figure 35:**  
Part marking scheme

## 19 Ordering Information

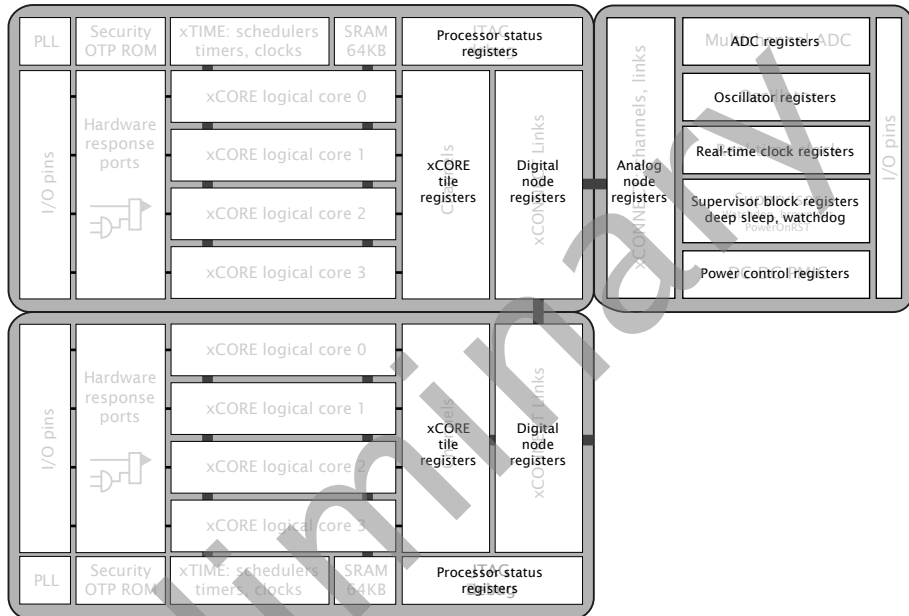
Product Code	Marking	Qualification	Speed Grade
XS1-A8A-128-FB217-C10	8A7C10	Commercial	1000 MIPS
XS1-A8A-128-FB217-I10	8A7I10	Industrial	1000 MIPS
XS1-A8A-128-FB217-C8	8A7C8	Commercial	800 MIPS
XS1-A8A-128-FB217-I8	8A7I8	Industrial	800 MIPS

**Figure 36:**  
Orderable part numbers

## Appendices

### A Configuring the device

The device is configured through ten banks of registers, as shown in Figure 37.



**Figure 37:**  
Registers

#### 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 `nnnnn` 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 digital and analogue node configuration registers

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 `0xnxxxnC30C` where `xxxx` 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).

Preliminary



## B Processor Status Configuration

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

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

**Figure 38:**  
Summary

### B.1 RAM base address: 0x00

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

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

**0x00:**  
RAM base  
address

### 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.

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

**0x01:**  
Vector base  
address

### B.3 xCORE Tile control: 0x02

Register to control features in the xCORE tile

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

**0x02:**  
xCORE Tile  
control

### B.4 xCORE Tile boot status: 0x03

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

**0x03:**  
xCORE Tile  
boot status

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

### B.5 Security configuration: 0x05

Copy of the security register as read from OTP.

**0x05:**  
Security  
configuration

Bits	Perm	Init	Description
31:0	RO		Value.

### B.6 Ring Oscillator Control: 0x06

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

**0x06:**  
Ring  
Oscillator  
Control

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

### B.7 Ring Oscillator Value: 0x07

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

**0x07:**  
Ring  
Oscillator  
Value

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

### B.8 Ring Oscillator Value: 0x08

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

0x08: Ring Oscillator Value	Bits	Perm	Init	Description
	31:16	RO	-	Reserved
	15:0	RO	-	Ring oscillator counter data.

### B.9 Ring Oscillator Value: 0x09

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

0x09: Ring Oscillator Value	Bits	Perm	Init	Description
	31:16	RO	-	Reserved
	15:0	RO	-	Ring oscillator counter data.

### B.10 Ring Oscillator Value: 0x0A

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

0x0A: Ring Oscillator Value	Bits	Perm	Init	Description
	31:16	RO	-	Reserved
	15:0	RO	-	Ring oscillator counter data.

### B.11 Debug SSR: 0x10

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

0x10: Debug SSR	Bits	Perm	Init	Description
	31:0	RO	-	Reserved

### B.12 Debug SPC: 0x11

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

0x11: Debug SPC	Bits	Perm	Init	Description
	31:0	DRW		Value.

### B.13 Debug SSP: 0x12

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

0x12: Debug SSP	Bits	Perm	Init	Description
	31:0	DRW		Value.

### B.14 DGETREG operand 1: 0x13

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

0x13: DGETREG operand 1	Bits	Perm	Init	Description
	31:8	RO	-	Reserved
	7:0	DRW		Thread number to be read

### B.15 DGETREG operand 2: 0x14

Register number to be read by DGETREG

0x14: DGETREG operand 2	Bits	Perm	Init	Description
	31:5	RO	-	Reserved
	4:0	DRW		Register number to be read

### B.16 Debug interrupt type: 0x15

Register that specifies what activated the debug interrupt.

**0x15:**  
Debug  
interrupt type

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

**B.17 Debug interrupt data: 0x16**

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

**0x16:**  
Debug  
interrupt data

Bits	Perm	Init	Description
31:0	DRW		Value.

**B.18 Debug core control: 0x18**

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

**0x18:**  
Debug core  
control

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

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

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

**0x20 .. 0x27:**  
Debug  
scratch

Bits	Perm	Init	Description
31:0	DRW		Value.

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

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

**0x30 .. 0x33:**  
Instruction  
breakpoint  
address

Bits	Perm	Init	Description
31:0	DRW		Value.

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

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

**0x40 .. 0x43:**  
Instruction  
breakpoint  
control

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

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

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

**0x50 .. 0x53:**  
Data  
watchpoint  
address 1

Bits	Perm	Init	Description
31:0	DRW		Value.

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

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

**0x60 .. 0x63:**  
Data  
watchpoint  
address 2

Bits	Perm	Init	Description
31:0	DRW		Value.

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

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

**0x70 .. 0x73:**  
Data  
breakpoint  
control  
register

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

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

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



**0x80 .. 0x83:**  
Resources  
breakpoint  
mask

Bits	Perm	Init	Description
31:0	DRW		Value.

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

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

**0x90 .. 0x93:**  
Resources  
breakpoint  
value

Bits	Perm	Init	Description
31:0	DRW		Value.

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

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

**0x9C .. 0x9F:**  
Resources  
breakpoint  
control  
register

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

## C xCORE Tile Configuration

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

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

**Figure 39:**  
Summary

### C.1 Device identification: 0x00

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

**0x00:**  
Device  
identification

### C.2 xCORE Tile description 1: 0x01

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

**0x01:**  
xCORE Tile  
description 1

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

### C.3 xCORE Tile description 2: 0x02

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

**0x02:**  
xCORE Tile  
description 2

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

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

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

**0x04:**  
Control  
PSwitch  
permissions  
to debug  
registers

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

### C.5 Cause debug interrupts: 0x05

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

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

**0x05:**  
Cause debug  
interrupts

### 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](#)

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

**0x06:**  
xCORE Tile  
clock divider

### C.7 Security configuration: 0x07

Copy of the security register as read from OTP.

Bits	Perm	Init	Description
31:0	RO		Value.

**0x07:**  
Security  
configuration

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

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

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

**0x10 .. 0x13:**  
PLink status

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

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

**0x20 .. 0x27:**  
Debug scratch

Bits	Perm	Init	Description
31:0	CRW		Value.

### C.10 PC of logical core 0: 0x40

Value of the PC of logical core 0.

**0x40:**  
PC of logical core 0

Bits	Perm	Init	Description
31:0	RO		Value.

**C.11 PC of logical core 1: 0x41**

**0x41:**  
PC of logical  
core 1

Bits	Perm	Init	Description
31:0	RO		Value.

**C.12 PC of logical core 2: 0x42**

**0x42:**  
PC of logical  
core 2

Bits	Perm	Init	Description
31:0	RO		Value.

**C.13 PC of logical core 3: 0x43**

**0x43:**  
PC of logical  
core 3

Bits	Perm	Init	Description
31:0	RO		Value.

**C.14 SR of logical core 0: 0x60**

Value of the SR of logical core 0

**0x60:**  
SR of logical  
core 0

Bits	Perm	Init	Description
31:0	RO		Value.

**C.15 SR of logical core 1: 0x61**

**0x61:**  
SR of logical  
core 1

Bits	Perm	Init	Description
31:0	RO		Value.

### C.16 SR of logical core 2: 0x62

**0x62:**  
SR of logical  
core 2

Bits	Perm	Init	Description
31:0	RO		Value.

### C.17 SR of logical core 3: 0x63

**0x63:**  
SR of logical  
core 3

Bits	Perm	Init	Description
31:0	RO		Value.

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

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

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

**0x80 .. 0x9F:**  
Chanend  
status

## D Digital Node Configuration

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

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

**Figure 40:**  
Summary

### D.1 Device identification: 0x00

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

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

**0x00:**  
Device  
identification

### D.2 System switch description: 0x01

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



**0x01:**  
System  
switch  
description

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

### D.3 Switch configuration: 0x04

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

**0x04:**  
Switch  
configuration

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

### D.4 Switch node identifier: 0x05

This register contains the node identifier.

**0x05:**  
Switch node  
identifier

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

### D.5 PLL settings: 0x06

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

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

**0x06:**  
PLL settings

### D.6 System switch clock divider: 0x07

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

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

**0x07:**  
System  
switch clock  
divider

### D.7 Reference clock: 0x08

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

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

**0x08:**  
Reference  
clock

### D.8 Directions 0-7: 0x0C

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

**0x0C:**  
Directions  
0-7

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

### D.9 Directions 8-15: 0x0D

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

**0x0D:**  
Directions  
8-15

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

### D.10 DEBUG\_N configuration: 0x10

Configures the behavior of the DEBUG\_N pin.

**0x10:**  
DEBUG\_N  
configuration

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

### D.11 Debug source: 0x1F

Contains the source of the most recent debug event.

**0x1F:**  
Debug source

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

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

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

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

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

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

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

**0x40 .. 0x43:**  
PLink status  
and network

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

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

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

**0x80 .. 0x87:**  
Link configuration and initialization

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

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

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

**0xA0 .. 0xA7:**  
Static link configuration

## E Analogue Node Configuration

The analogue 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	<a href="#">Device identification register</a>
0x04	RW	<a href="#">Node configuration register</a>
0x05	RW	<a href="#">Node identifier</a>
0x50	RW	<a href="#">Reset and Mode Control</a>
0x51	RW	<a href="#">System clock frequency</a>
0x80	RW	<a href="#">Link Control and Status</a>
0xD6	RW	<a href="#">1 KHz Watchdog Control</a>
0xD7	RW	<a href="#">Watchdog Disable</a>

**Figure 41:**  
Summary

### E.1 Device identification register: 0x00

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

Bits	Perm	Init	Description
31:24	RO	0x0F	Chip identifier
23:17	RO	-	Reserved
16	RO	pin	Oscillator used on power-up. This is set by the OSC_EXT_N pin: 0: boot from crystal; 1: boot from on-silicon 20 MHz oscillator.
15:8	RO	0x02	Revision number of the analogue block
7:0	RO	0x00	Version number of the analogue block

**0x00:**  
Device  
identification  
register

### 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.

	Bits	Perm	Init	Description
<b>0x04:</b> Node configuration register	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**

	Bits	Perm	Init	Description
<b>0x05:</b> Node identifier	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 Reset and Mode Control: 0x50**

The XS1-S has two main reset signals: a system-reset and an xCORE Tile-reset. System-reset resets the whole system including external devices, whilst xCORE Tile-reset resets the xCORE Tile(s) only. The resets are induced either by software (by a write to the register below) or by one of the following:

- \* External reset on RST\_N (System reset)
- \* Brown out on one of the power supplies (System reset)
- \* Watchdog timer (System reset)
- \* Sleep sequence (xCORE Tile reset)
- \* Clock source change (xCORE Tile reset)

The minimum system reset duration is achieved when the fastest permissible clock is used. The reset durations will be proportionately longer when a slower clock is used. Note that the minimum system reset duration allows for all power rails except the VOUT2 to turn off, and decay.

The length of the system reset comes from an internal counter, counting 524,288 oscillator clock cycles which gives the maximum time allowable for the supply rails to discharge. The system reset duration is a balance between leaving a long time for the supply rails to discharge, and a short time for the system to boot. Example reset times are 44 ms with a 12 MHz oscillator or 5.5 ms with a 96 MHz oscillator.



**0x50:**  
Reset and  
Mode Control

Bits	Perm	Init	Description
31:25	RO	-	Reserved
24	RW		Tristate processor mode pins.
23:18	RO	-	Reserved
17:16	RW		Processor mode pins.
15:2	RO	-	Reserved
1	WO	0	xCORE Tile reset. Set to 1 to initiate a reset of the xCORE Tile. This bit is self clearing. A write to this configuration register with this bit asserted results in no response packet being sent to the sender regardless of whether or not a response was requested.
0	WO	0	System reset. Set to 1 to initiate a reset whose scope includes most configuration and peripheral control registers. This bit is self clearing. A write to this configuration register with this bit asserted results in no response packet being sent to the sender regardless of whether or not a response was requested.

**E.5 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. The following functions depend on the correct frequency settings: * Processor reset delay * The watchdog clock * The real-time clock when running in sleep mode

## E.6 Link Control and Status: 0x80

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

**0x80:**  
Link Control  
and Status

## E.7 1 KHz Watchdog Control: 0xD6

The watchdog provides a mechanism to prevent programs from hanging by resetting the xCORE Tile after a pre-set time. The watchdog should be periodically “kicked” by the application, causing the count-down to be restarted. If the watchdog expires, it may be due to a program hanging, for example because of a (transient) hardware issue.

The watchdog timeout is measured in 1 ms clock ticks, meaning that a time between 1 ms and 65 seconds can be set for the timeout. The watchdog timer is only clocked during the AWAKE power state. When writing the timeout value, both the timeout and its one's complement should be written. This reduces the chances of accidentally setting kicking the watchdog. If the written value does not comprise a 16-bit value with a 16-bit one's complement, the request will be NACKed, otherwise an ACK will be sent.

If the watchdog expires, the xCORE Tile is reset.

Bits	Perm	Init	Description
31:16	RO	0	Current value of watchdog timer.
15:0	RW	1000	Number of 1kHz cycles after which the watchdog should expire and initiate a system reset.

**0xD6:**  
1 KHz  
Watchdog  
Control

## E.8 Watchdog Disable: 0xD7

To enable the watchdog, write 0 to this register. To disable the watchdog, write the value 0x0D15AB1E to this register.

<b>0xD7:</b> Watchdog Disable	<b>Bits</b>	<b>Perm</b>	<b>Init</b>	<b>Description</b>
	31:0	RW	0x0D15AB1E	A value of 0x0D15AB1E written to this register resets and disables the watchdog timer.

## F ADC Configuration

The device has a 12-bit Analogue to Digital Converter (ADC). It has multiple input pins, and on each positive clock edge on port 11, it samples and converts a value on the next input pin. The data is transmitted to a channel-end that must be set on enabling the ADC input pin.

The ADC is peripheral 2. The control registers are accessed using 32-bit reads and writes (use `write_periph_32(device, 2, ...)` and `read_periph_32(device, 2, ...)` for reads and writes).

Number	Perm	Description
0x00	RW	<a href="#">ADC Control input pin 0</a>
0x04	RW	<a href="#">ADC Control input pin 1</a>
0x08	RW	<a href="#">ADC Control input pin 2</a>
0x0C	RW	<a href="#">ADC Control input pin 3</a>
0x10	RW	<a href="#">ADC Control input pin 4</a>
0x14	RW	<a href="#">ADC Control input pin 5</a>
0x18	RW	<a href="#">ADC Control input pin 6</a>
0x1C	RW	<a href="#">ADC Control input pin 7</a>
0x20	RW	<a href="#">ADC General Control</a>

**Figure 42:**  
Summary

### F.1 ADC Control input pin 0: 0x00

Controls specific to ADC input pin 0.

**0x00:**  
ADC Control  
input pin 0

Bits	Perm	Init	Description
31:8	RW	0	The node and channel-end identifier to which data for this ADC input pin should be send to. This is the top 24 bits of the channel-end identifier as allocated on an xCORE Tile.
7:1	RO	-	Reserved
0	RW	0	Set to 1 to enable this input pin on the ADC.

## F.2 ADC Control input pin 1: 0x04

Controls specific to ADC input pin 1.

**0x04:**  
ADC Control  
input pin 1

Bits	Perm	Init	Description
31:8	RW	0	The node and channel-end identifier to which data for this ADC input pin should be send to. This is the top 24 bits of the channel-end identifier as allocated on an xCORE Tile.
7:1	RO	-	Reserved
0	RW	0	Set to 1 to enable this input pin on the ADC.

## F.3 ADC Control input pin 2: 0x08

Controls specific to ADC input pin 2.

**0x08:**  
ADC Control  
input pin 2

Bits	Perm	Init	Description
31:8	RW	0	The node and channel-end identifier to which data for this ADC input pin should be send to. This is the top 24 bits of the channel-end identifier as allocated on an xCORE Tile.
7:1	RO	-	Reserved
0	RW	0	Set to 1 to enable this input pin on the ADC.

## F.4 ADC Control input pin 3: 0x0C

Controls specific to ADC input pin 3.

**0x0C:**  
ADC Control  
input pin 3

Bits	Perm	Init	Description
31:8	RW	0	The node and channel-end identifier to which data for this ADC input pin should be send to. This is the top 24 bits of the channel-end identifier as allocated on an xCORE Tile.
7:1	RO	-	Reserved
0	RW	0	Set to 1 to enable this input pin on the ADC.

#### F.5 ADC Control input pin 4: 0x10

Controls specific to ADC input pin 4.

**0x10:**  
ADC Control  
input pin 4

Bits	Perm	Init	Description
31:8	RW	0	The node and channel-end identifier to which data for this ADC input pin should be send to. This is the top 24 bits of the channel-end identifier as allocated on an xCORE Tile.
7:1	RO	-	Reserved
0	RW	0	Set to 1 to enable this input pin on the ADC.

#### F.6 ADC Control input pin 5: 0x14

Controls specific to ADC input pin 5.

**0x14:**  
ADC Control  
input pin 5

Bits	Perm	Init	Description
31:8	RW	0	The node and channel-end identifier to which data for this ADC input pin should be send to. This is the top 24 bits of the channel-end identifier as allocated on an xCORE Tile.
7:1	RO	-	Reserved
0	RW	0	Set to 1 to enable this input pin on the ADC.

#### F.7 ADC Control input pin 6: 0x18

Controls specific to ADC input pin 6.

**0x18:**  
ADC Control  
input pin 6

Bits	Perm	Init	Description
31:8	RW	0	The node and channel-end identifier to which data for this ADC input pin should be send to. This is the top 24 bits of the channel-end identifier as allocated on an xCORE Tile.
7:1	RO	-	Reserved
0	RW	0	Set to 1 to enable this input pin on the ADC.

### F.8 ADC Control input pin 7: 0x1C

Controls specific to ADC input pin 7.

**0x1C:**  
ADC Control  
input pin 7

Bits	Perm	Init	Description
31:8	RW	0	The node and channel-end identifier to which data for this ADC input pin should be send to. This is the top 24 bits of the channel-end identifier as allocated on an xCORE Tile.
7:1	RO	-	Reserved
0	RW	0	Set to 1 to enable this input pin on the ADC.

### F.9 ADC General Control: 0x20

General ADC control.

Bits	Perm	Init	Description
31:25	RO	-	Reserved
24	RO	1	Indicates that an ADC sample has been dropped. This bit is cleared on a read.
23:18	RO	-	Reserved
17:16	RW	1	Number of bits per ADC sample. The ADC values are always left aligned: 0: 8 bits samples - the least significant four bits of each sample are discarded. 1: 16 bits samples - the sample is padded with four zero bits in bits 3..0. The most significant byte is transmitted first. 2: reserved 3: 32 bits samples - the sample is padded with 20 zero bits in bits 19..0. The most significant byte is transmitted first, hence the word can be input with a single 32-bit IN instruction.
15:8	RW	1	Number of samples to be transmitted per packet. The value 0 indicates that the packet will not be terminated until interrupted by an ADC control register access.
7:2	RO	-	Reserved
1	RW	0	Set to 1 to switch the ADC to sample a 0.8V signal rather than the external voltage. This can be used to calibrate the ADC. When switching to and from calibration mode, one sample value should be discarded. If a sample value $x$ is measured in calibration mode, then a scale factor $800000/x$ can be used to translate subsequent measurements into microvolts (using integer arithmetic).
0	RW	0	Set to 1 to enable the ADC. Note that when enabled, the ADC control registers above are read-only. The ADC must be disabled whilst setting up the per-input-pin control. On enabling the ADC, six pulses must be generated to calibrate the ADC. These pulses will not generate packets on the selected channel-end. The seventh and further pulses will deliver samples to the selected channel-end.

**0x20:**  
ADC General  
Control

## G Deep sleep memory Configuration

This peripheral contains a 128 byte RAM that retains state whilst the main processor is put to sleep.

The *Deep sleep memory* is peripheral 3. The control registers are accessed using 8-bit reads and writes (use `write_periph_8(device, 3, ...)` and `read_periph_8(device, 3, ...)` for reads and writes).

**Figure 43:**  
Summary

Number	Perm	Description
0x00 .. 0x7F	RW	<a href="#">Deep sleep memory</a>
0xFF	RW	<a href="#">Deep sleep memory valid</a>

### G.1 Deep sleep memory: 0x00 .. 0x7F

128 bytes of memory that can be used to hold data when the xCORE Tile is powered down.

**0x00 .. 0x7F:**  
Deep sleep memory

Bits	Perm	Init	Description
7:0	RW		User defined data

### G.2 Deep sleep memory valid: 0xFF

One byte of memory that is reset to 0. The program can write a non zero value in this register to indicate that the data in deep sleep memory is valid.

**0xFF:**  
Deep sleep memory valid

Bits	Perm	Init	Description
7:0	RW	0	User defined data, reset to 0.

## H Oscillator Configuration

The *Oscillator* is peripheral 4. The control registers are accessed using 8-bit reads and writes (use `write_periph_8(device, 4, ...)` and `read_periph_8(device, 4, ...)` for reads and writes).

**Figure 44:**  
Summary

Number	Perm	Description
0x00	RW	<a href="#">General oscillator control</a>
0x01	RW	<a href="#">On-silicon-oscillator control</a>
0x02	RW	<a href="#">Crystal-oscillator control</a>



**H.1 General oscillator control: 0x00**

**0x00:**  
General oscillator control

Bits	Perm	Init	Description
7:2	RO	-	Reserved
1	RW	0	Set to 1 to reset the xCORE Tile when the value of the oscillator select control register (bit 0) is changed.
0	RW	pin	Selects the oscillator to use: 0: Crystal oscillator 1: On-silicon oscillator

**H.2 On-silicon-oscillator control: 0x01**

This register controls the on-chip logic that implements an on-chip oscillator. The on-chip oscillator does not require an external crystal, but does not provide an accurate timing source. The nominal frequency of the on-silicon-oscillator is given below, but the actual frequency are temperature, voltage, and chip dependent.

**0x01:**  
On-silicon-oscillator control

Bits	Perm	Init	Description
7:2	RO	-	Reserved
1	RW	0	Selects the clock speed of the on-chip oscillator: 0: approximately 20 Mhz (fast clock) 1: approximately 31,250 Hz (slow clock)
0	RW	1	Set to 0 to disable the on-chip oscillator. Do not do this unless the xCORE Tile is running off the crystal oscillator.

**H.3 Crystal-oscillator control: 0x02**

This register controls the on-chip logic that implements the crystal oscillator; the crystal-oscillator requires an external crystal.

**0x02:**  
Crystal-oscillator control

Bits	Perm	Init	Description
7:2	RO	-	Reserved
1	RW	1	Set to 0 to disable the crystal bias circuit. Only switch the bias off if an external oscillator rather than a crystal is connected.
0	RW	1	Set to 0 to disable the crystal oscillator. Do not do this unless the xCORE Tile is running off the on-silicon oscillator.

## I Real time clock Configuration

The *Real time clock* is peripheral 5. The control registers are accessed using 32-bit reads and writes (use `write_periph_32(device, 5, ...)` and `read_periph_32(device, ↪ 5, ...)` for reads and writes).

**Figure 45:**  
Summary

Number	Perm	Description
0x00	RW	Real time counter least significant 32 bits
0x04	RW	Real time counter most significant 32 bits

### I.1 Real time counter least significant 32 bits: 0x00

This registers contains the lower 32-bits of the real-time counter.

**0x00:**  
Real time  
counter least  
significant 32  
bits

Bits	Perm	Init	Description
31:0	RO	0	Least significant 32 bits of real-time counter.

### I.2 Real time counter most significant 32 bits: 0x04

This registers contains the upper 32-bits of the real-time counter.

**0x04:**  
Real time  
counter most  
significant 32  
bits

Bits	Perm	Init	Description
31:0	RO	0	Most significant 32 bits of real-time counter.

## J Power control block Configuration

The *Power control block* is peripheral 6. The control registers are accessed using 32-bit reads and writes (use `write_periph_32(device, 6, ...)` and `read_periph_32(↪ device, 6, ...)` for reads and writes).

Number	Perm	Description
0x00	RW	General control
0x04	RW	Time to wake-up, least significant 32 bits
0x08	RW	Time to wake-up, most significant 32 bits
0x0C	RW	Power supply states whilst ASLEEP
0x10	RW	Power supply states whilst WAKING1
0x14	RW	Power supply states whilst WAKING2
0x18	RW	Power supply states whilst AWAKE
0x1C	RW	Power supply states whilst SLEEPING1
0x20	RW	Power supply states whilst SLEEPING2
0x24	RW	Power sequence status
0x2C	RW	DCDC control
0x30	RW	Power supply status
0x34	RW	VDDCORE level control
0x40	RW	LDO5 level control

**Figure 46:**  
Summary

### J.1 General control: 0x00

This register controls the basic settings for power modes.

Bits	Perm	Init	Description
31:8	RO	-	Reserved
7	RW	0	By default, when waking up, the voltage levels stored in the LEVEL CONTROL registers are used. Set to 1 to use the power-on voltage levels.
6	WO		Set to 1 to re-apply the current contents of the AWAKE state. Use this when the program has changed the contents of the AWAKE state register. Self clearing.
5	RW	0	Set to 1 to use a 64-bit timer.
4	RW	0	Set to 1 to wake-up on the timer.
3	RW	1	If waking on the WAKE pin is enabled (see above), then by default the device wakes up when the WAKE pin is pulled high. Set to 0 to wake-up when the WAKE pin is pulled low.
2	RW	0	Set to 1 to wake-up when the WAKE pin is at the right level.
1	RW	0	Set to 1 to initiate sleep sequence - self clearing. Only set this bit when in AWAKE state.
0	RW	0	Sleep clock select. Set to 1 to use the default clock rather than the internal 31.25 kHz oscillator. Note: this bit is only effective in the ASLEEP state.

**0x00:**  
General control

**J.2 Time to wake-up, least significant 32 bits: 0x04**

This register stores the time to wake-up. The value is only used if wake-up from the real-time clock is enabled, and the device is asleep.

**0x04:**  
Time to wake-up, least significant 32 bits

Bits	Perm	Init	Description
31:0	RW	0	Least significant 32 bits of time to wake-up.

**J.3 Time to wake-up, most significant 32 bits: 0x08**

This register stores the time to wake-up. The value is only used if wake-up from the real-time clock is enabled, if 64-bit comparisons are enabled, and the device is asleep. In most cases, 32-bit comparisons suffice.

**0x08:**  
Time to  
wake-up,  
most  
significant 32  
bits

Bits	Perm	Init	Description
31:0	RW	0	Most significant 32 bits of time to wake-up (ignored unless 64-bit timer comparison is enabled).

#### J.4 Power supply states whilst ASLEEP: 0x0C

This register controls the state the power control block should be in when in the ASLEEP state. It also defines the minimum time that the system shall stay in this state. When the minimum time is expired, the next state may be entered if either of the wake conditions (real-time counter or WAKE pin) happens. Note that the minimum number of cycles is counted in according to the currently enabled clock, which may be the slow 31 KHz clock.

Bits	Perm	Init	Description
31:21	RO	-	Reserved
20:16	RW	16	Log2 number of cycles to stay in this state: 0: 1 clock cycles 1: 2 clock cycles 2: 4 clock cycles ... 31: 2147483648 clock cycles
15	RO	-	Reserved
14	RW	0	Set to 1 to disable clock to the xCORE Tile.
13:10	RO	-	Reserved
9	RW	0	Sets modulation used by DCDC2: 0: PWM modulation (max 475 mA) 1: PFM modulation (max 50 mA)
8	RW	0	Sets modulation used by DCDC1: 0: PWM modulation (max 700 mA) 1: PFM modulation (max 50 mA)
7:6	RO	-	Reserved
5	RW	0	Set to 1 to enable VOUT6 (IO supply).
4	RW	0	Set to 1 to enable LDO5 (core PLL supply).
3:2	RO	-	Reserved
1	RO	0	Set to 1 to enable DCDC2 (analogue supply).
0	RW	0	Set to 1 to enable DCDC1 (core supply).

**0x0C:**  
Power supply  
states whilst  
ASLEEP

**J.5 Power supply states whilst WAKING1: 0x10**

This register controls what state the power control block should be in when in the WAKING1 state. It also defines the minimum time that the system shall stay in this state. When the minimum time is expired, the next state is entered if all enabled power supplies are good.

Bits	Perm	Init	Description
31:21	RO	-	Reserved
20:16	RW	16	Log2 number of cycles to stay in this state: 0: 1 clock cycles 1: 2 clock cycles 2: 4 clock cycles ... 31: 2147483648 clock cycles
15	RO	-	Reserved
14	RW	0	Set to 1 to disable clock to the xCORE Tile.
13:10	RO	-	Reserved
9	RW	0	Sets modulation used by DCDC2: 0: PWM modulation (max 475 mA) 1: PFM modulation (max 50 mA)
8	RW	0	Sets modulation used by DCDC1: 0: PWM modulation (max 700 mA) 1: PFM modulation (max 50 mA)
7:6	RO	-	Reserved
5	RW	1	Set to 1 to enable VOUT6 (IO supply).
4	RW	0	Set to 1 to enable LDO5 (core PLL supply).
3:2	RO	-	Reserved
1	RO	0	Set to 1 to enable DCDC2 (analogue supply).
0	RW	0	Set to 1 to enable DCDC1 (core supply).

**0x10:**  
Power supply  
states whilst  
WAKING1

**J.6 Power supply states whilst WAKING2: 0x14**

This register controls what state the power control block should be in when in the WAKING2 state. It also defines the minimum time that the system shall stay in this state. When the minimum time is expired, the next state is entered if all enabled power supplies are good.

Bits	Perm	Init	Description
31:21	RO	-	Reserved
20:16	RW	16	Log2 number of cycles to stay in this state: 0: 1 clock cycles 1: 2 clock cycles 2: 4 clock cycles ... 31: 2147483648 clock cycles
15	RO	-	Reserved
14	RW	0	Set to 1 to disable clock to the xCORE Tile.
13:10	RO	-	Reserved
9	RW	0	Sets modulation used by DCDC2: 0: PWM modulation (max 475 mA) 1: PFM modulation (max 50 mA)
8	RW	0	Sets modulation used by DCDC1: 0: PWM modulation (max 700 mA) 1: PFM modulation (max 50 mA)
7:6	RO	-	Reserved
5	RW	1	Set to 1 to enable VOUT6 (IO supply).
4	RW	1	Set to 1 to enable LDO5 (core PLL supply).
3:2	RO	-	Reserved
1	RO	1	Set to 1 to enable DCDC2 (analogue supply).
0	RW	1	Set to 1 to enable DCDC1 (core supply).

**0x14:**  
Power supply  
states whilst  
WAKING2

### J.7 Power supply states whilst AWAKE: 0x18

This register controls what state the power control block should be in when in the AWAKE state.

Bits	Perm	Init	Description
31:15	RO	-	Reserved
14	RW	0	Set to 1 to disable clock to the xCORE Tile.
13:10	RO	-	Reserved
9	RW	0	Sets modulation used by DCDC2: 0: PWM modulation (max 475 mA) 1: PFM modulation (max 50 mA)
8	RW	0	Sets modulation used by DCDC1: 0: PWM modulation (max 700 mA) 1: PFM modulation (max 50 mA)
7:6	RO	-	Reserved
5	RW	1	Set to 1 to enable VOUT6 (IO supply).
4	RW	1	Set to 1 to enable LDO5 (core PLL supply).
3:2	RO	-	Reserved
1	RO	1	Set to 1 to enable DCDC2 (analogue supply).
0	RW	1	Set to 1 to enable DCDC1 (core supply).

**0x18:**  
Power supply  
states whilst  
AWAKE

### J.8 Power supply states whilst SLEEPING1: 0x1C

This register controls what state the power control block should be in when in the SLEEPING1 state. It also defines the time that the system shall stay in this state.



Bits	Perm	Init	Description
31:21	RO	-	Reserved
20:16	RW	16	Log2 number of cycles to stay in this state: 0: 1 clock cycles 1: 2 clock cycles 2: 4 clock cycles ... 31: 2147483648 clock cycles
15	RO	-	Reserved
14	RW	0	Set to 1 to disable clock to the xCORE Tile.
13:10	RO	-	Reserved
9	RW	0	Sets modulation used by DCDC2: 0: PWM modulation (max 475 mA) 1: PFM modulation (max 50 mA)
8	RW	0	Sets modulation used by DCDC1: 0: PWM modulation (max 700 mA) 1: PFM modulation (max 50 mA)
7:6	RO	-	Reserved
5	RW	1	Set to 1 to enable VOUT6 (IO supply).
4	RW	0	Set to 1 to enable LDO5 (core PLL supply).
3:2	RO	-	Reserved
1	RO	1	Set to 1 to enable DCDC2 (analogue supply).
0	RW	0	Set to 1 to enable DCDC1 (core supply).

**0x1C:**  
Power supply  
states whilst  
SLEEPING1

### J.9 Power supply states whilst SLEEPING2: 0x20

This register controls what state the power control block should be in when in the SLEEPING2 state. It also defines the time that the system shall stay in this state.

Bits	Perm	Init	Description
31:21	RO	-	Reserved
20:16	RW	16	Log2 number of cycles to stay in this state: 0: 1 clock cycles 1: 2 clock cycles 2: 4 clock cycles ... 31: 2147483648 clock cycles
15	RO	-	Reserved
14	RW	0	Set to 1 to disable clock to the xCORE Tile.
13:10	RO	-	Reserved
9	RW	0	Sets modulation used by DCDC2: 0: PWM modulation (max 475 mA) 1: PFM modulation (max 50 mA)
8	RW	0	Sets modulation used by DCDC1: 0: PWM modulation (max 700 mA) 1: PFM modulation (max 50 mA)
7:6	RO	-	Reserved
5	RW	0	Set to 1 to enable VOUT6 (IO supply).
4	RW	0	Set to 1 to enable LDO5 (core PLL supply).
3:2	RO	-	Reserved
1	RO	1	Set to 1 to enable DCDC2 (analogue supply).
0	RW	0	Set to 1 to enable DCDC1 (core supply).

**0x20:**  
Power supply  
states whilst  
SLEEPING2

**J.10 Power sequence status: 0x24**

This register defines the current status of the power supply controller.

Bits	Perm	Init	Description
31:30	RO	-	Reserved
29	RO	0	1 if VOUT6 was enabled in the previous state.
28	RO	0	1 if LDO5 was enabled in the previous state.
27:26	RO	-	Reserved
25	RO	1	1 if DCDC2 was enabled in the previous state.
24	RO	0	1 if DCDC1 was enabled in the previous state.
23:19	RO	-	Reserved
18:16	RO		Current state of the power sequence state machine 0: Reset 1: Asleep 2: Waking 1 3: Waking 2 4: Awake Wait 5: Awake 6: Sleeping 1 7: Sleeping 2
15	RO	-	Reserved
14	RO	0	Set to 1 to disable clock to the xCORE Tile.
13:10	RO	-	Reserved
9	RO	0	Sets modulation used by DCDC2: 0: PWM modulation (max 475 mA) 1: PFM modulation (max 50 mA)
8	RO	0	Sets modulation used by DCDC1: 0: PWM modulation (max 700 mA) 1: PFM modulation (max 50 mA)
7:6	RO	-	Reserved
5	RO	0	Set to 1 to enable VOUT6 (IO supply).
4	RO	0	Set to 1 to enable LDO5 (core PLL supply).
3:2	RO	-	Reserved
1	RO	0	Set to 1 to enable DCDC2 (analogue supply).
0	RO	0	Set to 1 to enable DCDC1 (core supply).

**0x24:**  
Power  
sequence  
status

**J.11 DCDC control: 0x2C**

This register controls the two DC-DC converters.

Bits	Perm	Init	Description
31:26	RO	-	Reserved
25:24	RW	2	Sets the power good level for VDDCORE and VDD1V8: 0: 0.80 x VDDCORE, 0.80 x VDD1V8 1: 0.85 x VDDCORE, 0.85 x VDD1V8 2: 0.90 x VDDCORE, 0.90 x VDD1V8 3: 0.75 x VDDCORE, 0.75 x VDD1V8
23:17	RO	-	Reserved
16	RW	0	Clear DCDC1 and DCDC2 error flags, not self clearing.
15	RO	-	Reserved
14:13	RW	0	Sets the DCDC2 current limit: 0: 1A 1: 1.5A 2: 2A 3: 0.5A
12:10	RO	-	Reserved
9:8	RW	1	Sets the clock used by DCDC2 to generate VDD1V8: 0: 0.9 MHz 1: 1.0 MHz 2: 1.1 MHz 3: 1.2 MHz
7	RO	-	Reserved
6:5	RW	0	Sets the DCDC1 current limit: 0: 1.2A 1: 1.8A 2: 2.5A 3: 0.8A
4:2	RO	-	Reserved
1:0	RW	1	Sets the clock used by DCDC1 to generate VDDCORE: 0: 0.9 MHz 1: 1.0 MHz 2: 1.1 MHz 3: 1.2 MHz

**0x2C:**  
DCDC control

### J.12 Power supply status: 0x30

This register provides the current status of the power supplies.

**0x30:**  
Power supply  
status

Bits	Perm	Init	Description
31:25	RO	-	Reserved
24	RO		1 if on-silicon oscillator is stable.
23:20	RO	-	Reserved
19	RO		1 if VDDPLL is good.
18:17	RO	-	Reserved
16	RO		1 if VDDCORE is good.
15:10	RO	-	Reserved
9	RO		1 if DCDC2 is in current limiting mode.
8	RO		1 if DCDC1 is in current limiting mode.
7:2	RO	-	Reserved
1	RO		1 if DCDC2 is in soft-start mode.
0	RO		1 if DCDC1 is in soft-start mode.

**J.13 VDDCORE level control: 0x34**

This register can be used to set the desired voltage on VDDCORE. If the level is to be raised or lowered, it should be raised in steps of no more than 10 mV per microsecond in order to prevent overshoot and undershoot. The default value depends on the MODE pins.

**0x34:**  
VDDCORE  
level control

Bits	Perm	Init	Description
31:7	RO	-	Reserved
6:0	RW	pin	The required voltage in 10 mV steps: 0: 0.60V 1: 0.61V 2: 0.62V ... 69: 1.29V 70: 1.30V

**J.14 LDO5 level control: 0x40**

This register can be used to set the desired voltage on LDO5. If the level is to be raised, it should be raised in steps of 1 (100 mV). The default value depends on the MODE pins.

**0x40:**  
LDO5 level  
control

Bits	Perm	Init	Description
31:3	RO	-	Reserved
2:0	RW	pin	The required voltage in 100 mV steps: 0: 0.6V 1: 0.7V 2: 0.8V ... 6: 1.2V 7: 1.3V

preliminary

## K XMOS USB Interface

XMOS provides a low-level USB interface for connecting the device to a USB transceiver using the UTMI+ Low Pin Interface (ULPI). The ULPI signals must be connected to the pins named in Figure 47. Note also that some ports on the same tile are used internally and are not available for use when the USB driver is active (they are available otherwise).

Pin	Signal	Pin	Signal	Pin	Signal
XnD02	Unavailable when USB active	XnD12	ULPI_STP	XnD26	Unavailable when USB active
XnD03		XnD13	ULPI_NXT	XnD27	
XnD04		XnD14	ULPI_DATA[0]	XnD28	
XnD05		XnD15	ULPI_DATA[1]	XnD29	
XnD06		XnD16	ULPI_DATA[2]	XnD30	
XnD07		XnD17	ULPI_DATA[3]	XnD31	
XnD08		XnD18	ULPI_DATA[4]	XnD32	
XnD09		XnD19	ULPI_DATA[5]	XnD33	
		XnD20	ULPI_DATA[6]		
	XnD21	ULPI_DATA[7]	XnD37	Unavailable when USB active	
	XnD22	ULPI_DIR	XnD38		
	XnD23	ULPI_CLK	XnD39		
			XnD40		
			XnD41		
			XnD42		
			XnD43		

**Figure 47:**  
ULPI signals provided by the XMOS USB driver

## L Device Errata

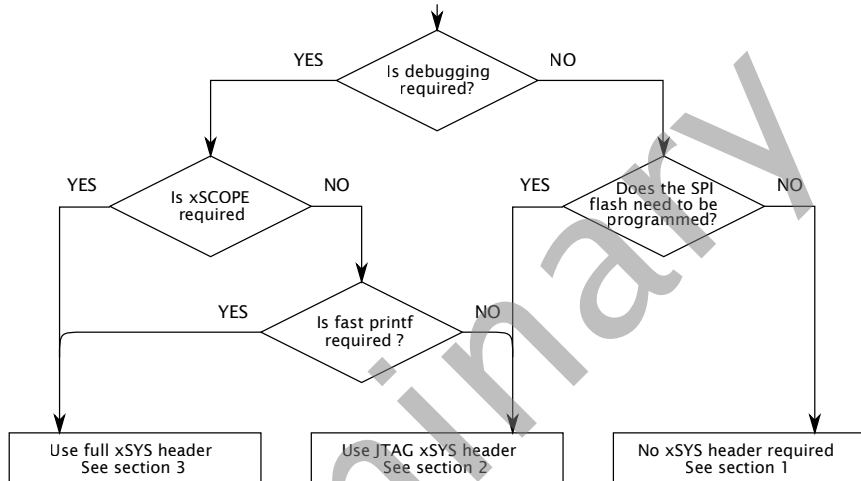
This section describes minor operational differences from the data sheet and recommended workarounds. As device and documentation issues become known, this section will be updated the document revised.

To guarantee a logic low is seen on the pins DEBUG\_N, MODE[4:0], TMS, TCK and TDI, the driving circuit should present an impedance of less than 100Ω 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.

## M 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 48 shows a decision diagram which explains what type of xSYS connectivity you need. The three subsections below explain the options in detail.



**Figure 48:**  
Decision diagram for the xSYS header

### M.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.

### M.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
- ▶ DEBUG\_N to pin 11 of the xSYS header



- ▶ TDO to pin 13 of the xSYS header
- ▶ RST\_N to pin 15 of the xSYS header
- ▶ If MODE2 is configured high, connect MODE2 to pin 3 of the xSYS header. Do not connect to VDDIO.
- ▶ If MODE3 is configured high, connect MODE3 to pin 3 of the xSYS header. Do not connect to VDDIO.

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

### M.3 Full xSYS header

For a full xSYS header you will need to connect the pins as discussed in Section M.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 XLA, XLB, 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 XLB of tile 0 for xSCOPE I/O, you need to connect up  ${}^1_{out}$ ,  ${}^0_{out}$ ,  ${}^0_{in}$ ,  ${}^1_{in}$  as follows:

- ▶  ${}^1_{out}$  (X0D16) to pin 6 of the xSYS header with a 33R series resistor close to the device.
- ▶  ${}^0_{out}$  (X0D17) to pin 10 of the xSYS header with a 33R series resistor close to the device.
- ▶  ${}^0_{in}$  (X0D18) to pin 14 of the xSYS header.
- ▶  ${}^1_{in}$  (X0D19) to pin 18 of the xSYS header.

## N Schematics Design Check List

- This section is a checklist for use by schematics designers using the XS1-A8A-128-FB217. Each of the following sections contains items to check for each design.

### N.1 Clock

- Pins MODE0 and MODE1 are set to the correct value for the chosen frequency. The MODE settings are shown in the Oscillator section, Section 8. If you have a choice between two values, choose the value with the highest multiplier ratio since that will boot faster.
- OSC\_EXT\_N is tied to ground (for use with a crystal or oscillator) or tied to VDDIO (for use with the internal oscillator). If using the internal oscillator, set MODE0 and MODE1 to be for the 20-48 MHz range (Section 8).
- If you have used an oscillator, it is a 1V8 oscillator. (Section 16)

### N.2 USB ULPI Mode

This section can be skipped if you do not have an external USB PHY.

- If using ULPI, the ULPI signals are connected to specific ports as shown in Section K.
- If using ULPI, the ports that are used internally are not connected, see Section K. (Note that this limitation only applies when the ULPI is enabled, they can still be used before or after the ULPI is being used.)

### N.3 Boot

- The device is connected to a SPI flash for booting, connected to X0D0, X0D01, X0D10, and X0D11 (Section 9). If not, you must boot the device through OTP or JTAG.
- The device that is connected to flash has both MODE2 and MODE3 NC (Section 9).
- The SPI flash that you have chosen is supported by **xflash**, or you have created a specification file for it.

#### N.4 JTAG, XScope, and debugging

- You have decided as to whether you need an XSYS header or not (Section [M](#))
- If you included an XSYS header, you connected pin 3 to any MODE2/MODE3 pin that would otherwise be NC (Section [M](#)).
- If you have not included an XSYS header, you have devised a method to program the SPI-flash or OTP (Section [M](#)).

#### N.5 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 [9](#)).
- If you included an XSYS header, you have included buffers for RST\_N, TMS, TCK, MODE2, and MODE3 (Section [L](#)).

## O PCB Layout Design Check List

- ✓ This section is a checklist for use by PCB designers using the XS1-A8A-128-FB217. Each of the following sections contains items to check for each design.

### O.1 Ground Balls and Ground Plane

- There is one via for every other ground ball to minimize impedance and conduct heat away from the device (Section 15.1).
- There are only few non-ground vias around the square of ground balls, to creating a good, solid, ground plane. Examples of a good and bad designs are shown in Figure XXX.

### O.2 Power supply decoupling

- VSUP has a ceramic X5R or X7R bulk decoupler as close as possible to the VSUP and PGND (VDDCORE) pins; right next to the device (Section 15).
- The 1V0 decoupling cap is close to the VDDCORE and PGND pins (Section 15).
- The 1V8 decoupling cap is close to the VDD1V8 and PGND pins (Section 15).
- All PGND nets are connected together prior to connection to the main ground plane (Section 15).

An example PCB layout is shown in Section 16. Placing the decouplers too far away may lead to the device not coming up, or not operating properly.

## P Associated Design Documentation

Document Title	Information	Document Number
Programming XC on XMOS Devices	Timers, ports, clocks, cores and channels	<a href="#">X9577</a>
xTIMEcomposer User Guide	Compilers, assembler and linker/mapper Timing analyzer, xScope, debugger Flash and OTP programming utilities	<a href="#">X3766</a>

## Q Related Documentation

Document Title	Information	Document Number
The XMOS XS1 Architecture	ISA manual	<a href="#">X7879</a>
XS1 Port I/O Timing	Port timings	<a href="#">X5821</a>
XS1-L System Specification	Link, switch and system information	<a href="#">X1151</a>
XS1-L Link Performance and Design Guidelines	Link timings	<a href="#">X2999</a>
XS1-L Clock Frequency Control	Advanced clock control	<a href="#">X1433</a>

## R Revision History

Date	Description
2013-04-16	First release
2013-07-19	Updated Features list with available ports and links - Section <a href="#">2</a> Simplified link bits in Signal Description - Section <a href="#">4</a> New JTAG, xSCOPE and Debugging appendix - Section <a href="#">M</a> New Schematics Design Check List - Section <a href="#">N</a> New PCB Layout Design Check List - Section <a href="#">O</a>



Copyright © 2013, All Rights Reserved.

Xmos Ltd. is the owner or licensee of this design, code, or Information (collectively, the "Information") and is providing it to you "AS IS" with no warranty of any kind, express or implied and shall have no liability in relation to its use. Xmos Ltd. makes no representation that the Information, or any particular implementation thereof, is or will be free from any claims of infringement and again, shall have no liability in relation to any such claims.