

Welcome to [E-XFL.COM](https://www.e-xfl.com)

### What is "[Embedded - Microcontrollers](#)"?

"[Embedded - Microcontrollers](#)" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

### Applications of "[Embedded - Microcontrollers](#)"

#### Details

Product Status	Active
Core Processor	XCore
Core Size	32-Bit 6-Core
Speed	500MIPS
Connectivity	Configurable
Peripherals	-
Number of I/O	38
Program Memory Size	64KB (16K x 32)
Program Memory Type	SRAM
EEPROM Size	-
RAM Size	-
Voltage - Supply (Vcc/Vdd)	3V ~ 3.6V
Data Converters	A/D 4x12b
Oscillator Type	Internal
Operating Temperature	0°C ~ 70°C (TA)
Mounting Type	Surface Mount
Package / Case	96-LFBGA
Supplier Device Package	96-FBGA (10x10)
Purchase URL	<a href="https://www.e-xfl.com/product-detail/xmos/xs1-u6a-64-fb96-c5">https://www.e-xfl.com/product-detail/xmos/xs1-u6a-64-fb96-c5</a>

---

## 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.

### 3 Pin Configuration

	1	2	3	4	5	6	7	8	9	10	11	12
A	AVDD	ADC0	ADC2	NC	USB_DP	USB_DN	USB_VBUS	<sup>1L</sup> X0D35	<sup>1A</sup> X0D00	<sup>1C</sup> X0D10	<sup>1E</sup> X0D12	<sup>32A</sup> X0D49
B	TDO	ADC1	ADC3	NC	MODE[2]	MODE[3]	USB_ID	<sup>1I</sup> X0D24	<sup>1B</sup> X0D01	<sup>1D</sup> X0D11	<sup>32A</sup> X0D50	<sup>32A</sup> X0D51
C	TCK	RST_N									<sup>32A</sup> X0D52	<sup>32A</sup> X0D53
D	TMS	TDI									<sup>32A</sup> X0D54	<sup>32A</sup> X0D55
E	XI/ CLK	DEBUG_N			AVSS	GND	GND	GND			<sup>32A</sup> X0D56	<sup>32A</sup> X0D57
F	XO	OSC_EXT_N			GND	GND	GND	GND			<sup>32A</sup> X0D58	<sup>32A</sup> X0D61
G	<sup>1D</sup> X0D43/ WAKE	NC			GND	GND	GND	GND			<sup>32A</sup> X0D62	<sup>32A</sup> X0D63
H	VSUP	NC			GND	GND	GND	GND			<sup>32A</sup> X0D64	<sup>32A</sup> X0D65
J	SW1	SW1									<sup>32A</sup> X0D66	<sup>32A</sup> X0D67
K	VDDCORE	VDDCORE									<sup>32A</sup> X0D68	<sup>32A</sup> X0D69
L	PGND	PGND	NC	MODE[1]	MODE[0]	VDDIO	<sup>1B</sup> X0D22	<sup>4C</sup> X0D20	<sup>4B</sup> X0D18	<sup>4B</sup> X0D16	<sup>4C</sup> X0D14	<sup>32A</sup> X0D70
M	VSUP	VSUP	PGND	VDD1V8	SW2	VDDIO	VDDIO	<sup>4C</sup> X0D21	<sup>4D</sup> X0D19	<sup>4D</sup> X0D17	<sup>4C</sup> X0D15	<sup>4F</sup> X0D13

The XMOS XUD software component performs all the low-level I/O operations required to meet the USB 2.0 specification, removing all low-level timing requirements from the application.

### 6.3 ADC and Power Management

Each XS1-U6A-64-FB96 device includes a set of analog components, including a 12b, 4-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

The tile has 6 active logical cores, which issue instructions down a shared four-stage pipeline. Instructions from the active cores are issued round-robin. If up to four logical cores are active, each core is allocated a quarter of the processing cycles. If more than four logical cores are active, each core is allocated at least  $1/n$  cycles (for  $n$  cores). Figure 4 shows the guaranteed core performance depending on the number of cores used.

**Figure 4:**  
Logical core performance

Speed grade	MIPS	Frequency	Minimum MIPS per core (for $n$ cores)							
			1	2	3	4	5	6		
5	500 MIPS	500 MHz	125	125	125	125	100	83		

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

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

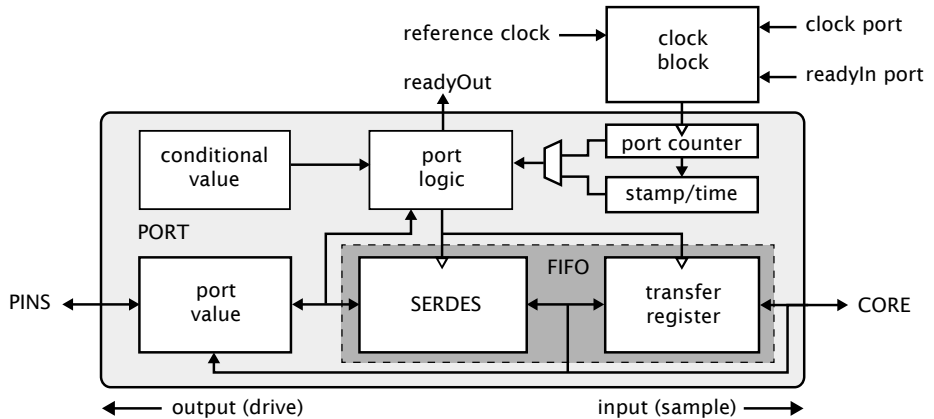
### 7.2 xTIME scheduler

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

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

### 7.3 Hardware Response Ports

Hardware Response ports connect an xCORE tile to one or more physical pins and as such define the interface between hardware attached to the XS1-U6A-64-FB96, and the software running on it. A combination of 1bit, 4bit, 8bit, 16bit and 32bit ports are available. All pins of a port provide either output or input. Signals in different directions cannot be mapped onto the same port.



**Figure 5:**  
Port block  
diagram

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

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

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

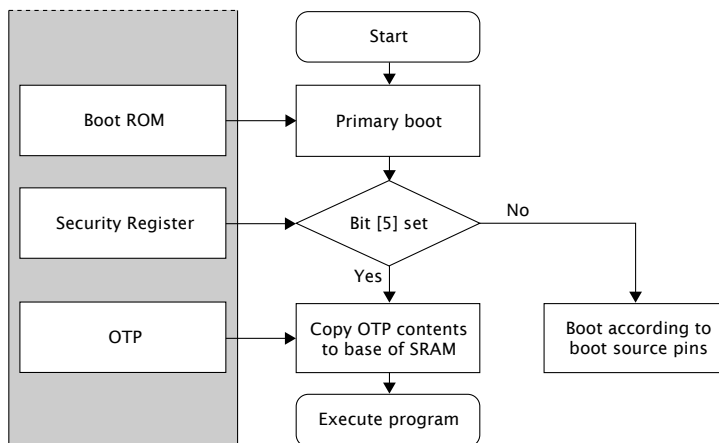
The ports and xCONNECT links are multiplexed onto the physical pins. If an xConnect Link is enabled, the pins of the underlying ports are disabled. If a port is enabled, it overrules ports with higher widths that share the same pins. The pins on the wider port that are not shared remain available for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

### 7.4 Clock blocks

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

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 9. In normal usage, MODE[3:2] controls the boot source according to the table in Figure 10. If bit 5 of the security register (see §10.1) is set, the device boots from OTP.



**Figure 9:**  
Boot procedure

MODE[3]	MODE[2]	Boot Source
0	0	None: Device waits to be booted via JTAG
0	1	Reserved
1	0	xConnect Link B
1	1	SPI

**Figure 10:**  
Boot source pins

The boot image has the following format:

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

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

### 9.1 Boot from SPI master

If set to boot from SPI master, the processor enables the four pins specified in Figure 11, and drives the SPI clock at 2.5 MHz (assuming a 400 MHz core clock). A

MIPS. This means that no more than six logical cores execute at any one time on a 500MHz device.

## 12 Analog-to-Digital Converter

The device has a 12-bit 1MSample/second Successive Approximation Register (SAR) Analogue to Digital Converter (ADC). It has 4 input pins which are multiplexed into the ADC. The sampling of the ADC is controlled using GPIO pin X0D24 that is triggered either by writing to port 11, 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 4 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 G.

The minimum latency for reading a value from the ADC into the xCORE register is shown in Figure 13:

**Figure 13:**  
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

## 13 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-U6A-64-FB96 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.

## 14 Energy management

XS1-U6A-64-FB96 devices can be powered by:

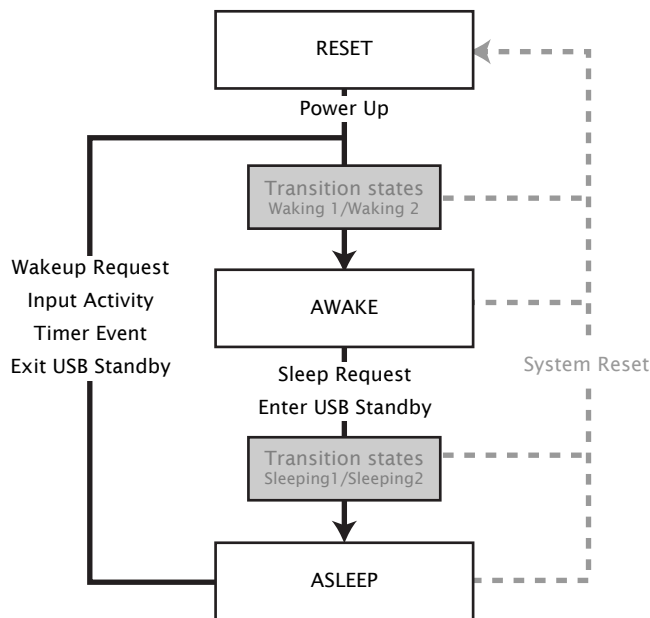
- ▶ An external 5v core and 3.3v I/O supply, increasing efficiency for USB bus powered applications.
- ▶ A single 3.3v supply.

### 14.1 DC-DC

XS1-U6A-64-FB96 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.

### 14.2 Power mode controller

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



**Figure 14:**  
XS1-U6A-64-FB96 Power Up States and Transitions



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

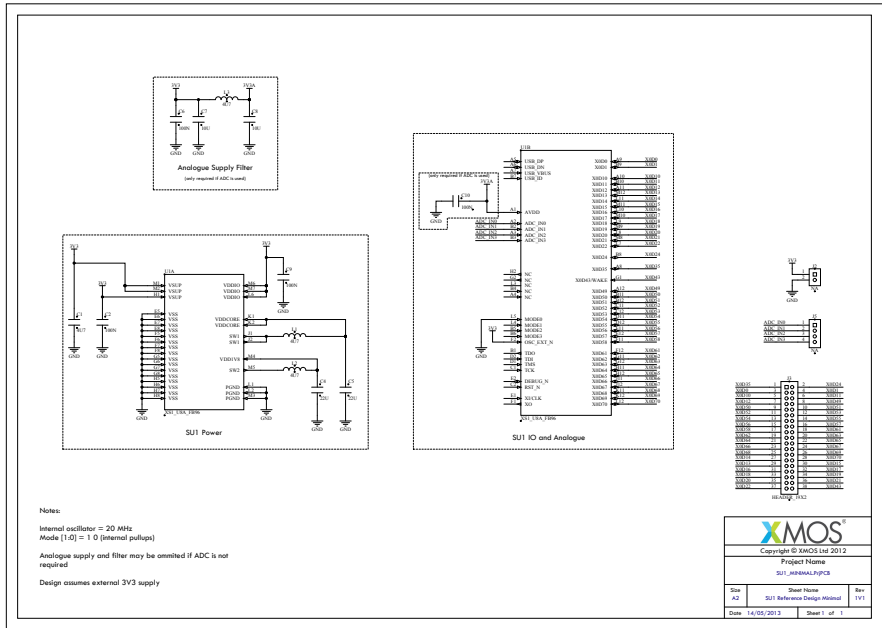
- ▶ High speed differential pairs should be routed together.
- ▶ High-speed USB signal pair traces should be trace-length matched. Maximum trace-length mismatch should be no greater than 4mm.
- ▶ Ensure that high speed signals (clocks, USB differential pairs) are routed as far away from off-board connectors as possible.
- ▶ High-speed clock and periodic signal traces that run parallel should be at least 1.27mm away from USB\_DP/USB\_DN (see Figure 20).
- ▶ Low-speed and non-periodic signal traces that run parallel should be at least 0.5mm away from USB\_DP/USB\_DN (see Figure 20).
- ▶ Route high speed USB signals on the top of the PCB wherever possible.
- ▶ Route high speed USB traces over continuous power planes, with no breaks. If a trade-off must be made, changing signal layers is preferable to crossing plane splits.
- ▶ Follow the  $20 \times h$  rule; keep traces  $20 \times h$  (the height above the power plane) away from the edge of the power plane.
- ▶ Use a minimum of vias in high speed USB traces.
- ▶ Avoid corners in the trace. Where necessary, rather than turning through a 90 degree angle, use two 45 degree turns or an arc.
- ▶ DO NOT route USB traces near clock sources, clocked circuits or magnetic devices.
- ▶ Avoid stubs on high speed USB signals.

### 16.3 Land patterns and solder stencils

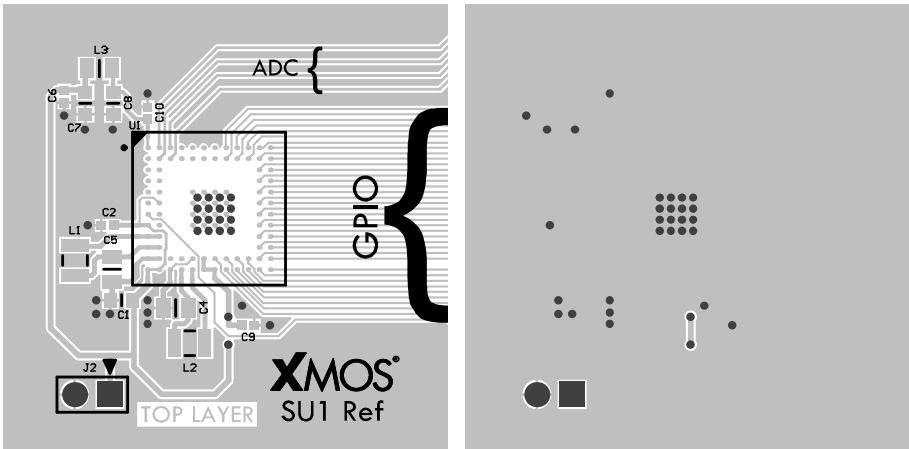
The land pattern recommendations in this document are based on a RoHS compliant process and derived, where possible, from the nominal *Generic Requirements for Surface Mount Design and Land Pattern Standards IPC-7351B* specifications. This standard aims to achieve desired targets of heel, toe and side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

Solder paste and ground via recommendations are based on our engineering and development kit board production. They have been found to work and optimized as appropriate to achieve a high yield. 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.



**Figure 25:**  
 Example minimal system schematic, with top and bottom layout of a 2-layer PCB



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 `0xnnnnC30C` where `nnnn` is the node-identifier.

A write message comprises the following:

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

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

A read message comprises the following:

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

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

### A.4 Accessing a register of an analogue peripheral

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

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

A write message comprises the following:

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

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

A read message comprises the following:

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

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

<b>0x00:</b> RAM base address	Bits	Perm	Init	Description
	31:2	RW		Most significant 16 bits of all addresses.
	1:0	RO	-	Reserved

**B.2 Vector base address: 0x01**

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

<b>0x01:</b> Vector base address	Bits	Perm	Init	Description
	31:16	RW		The most significant bits for all event and interrupt vectors.
	15:0	RO	-	Reserved

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

Register to control features in the xCORE tile

<b>0x02:</b> xCORE Tile control	Bits	Perm	Init	Description
	31:6	RO	-	Reserved
	5	RW	0	Set to 1 to select the dynamic mode for the clock divider when the clock divider is enabled. In dynamic mode the clock divider is only activated when all active logical cores are paused. In static mode the clock divider is always enabled.
	4	RW	0	Set to 1 to enable the clock divider. This slows down the xCORE tile clock in order to use less power.
	3:0	RO	-	Reserved

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

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

**C.22 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

### 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

**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).

#### K.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

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

## K.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

### K.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

### K.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

### K.11 DCDC control: 0x2C

This register controls the two DC-DC converters.

## L Device Errata

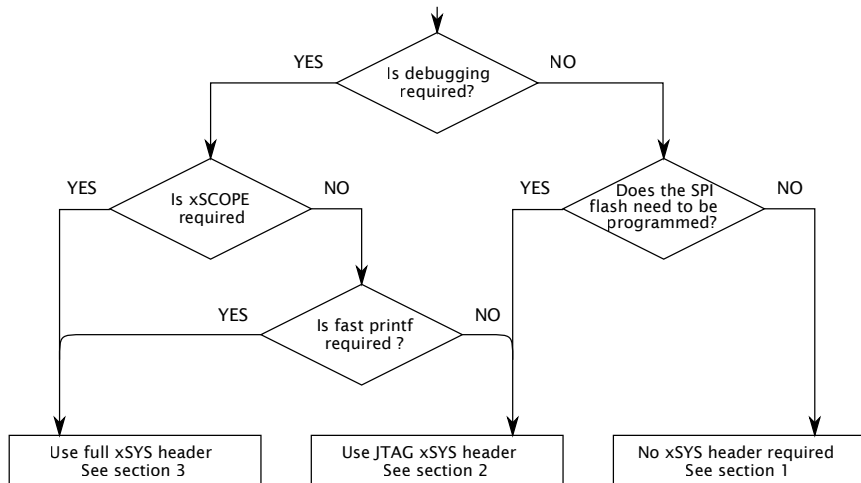
This section describes minor operational differences from the data sheet and recommended workarounds. As device and documentation issues become known, this section will be updated the document revised.

To guarantee a logic low is seen on the pins DEBUG\_N, MODE[3:0], TMS, TCK and TDI, the driving circuit should present an impedance of less than 100 Ω to ground. Usually this is not a problem for CMOS drivers driving single inputs. If one or more of these inputs are placed in parallel, however, additional logic buffers may be required to guarantee correct operation.

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

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



**Figure 54:**  
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.

- If you have not included an XSYS header, you have devised a method to program the SPI-flash or OTP (Section [M](#)).

#### N.4 GPIO

- You have not mapped both inputs and outputs to the same multi-bit port.

#### 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](#)).