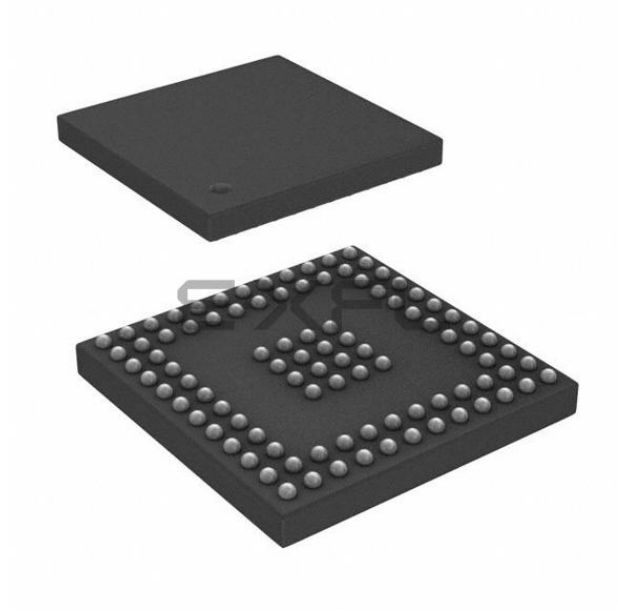

Welcome to E-XFL.COM

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 8-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	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	96-LFBGA
Supplier Device Package	96-FBGA (10x10)
Purchase URL	https://www.e-xfl.com/product-detail/xmos/xs1-u8a-64-fb96-i5

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/.

5 Example Application Diagram

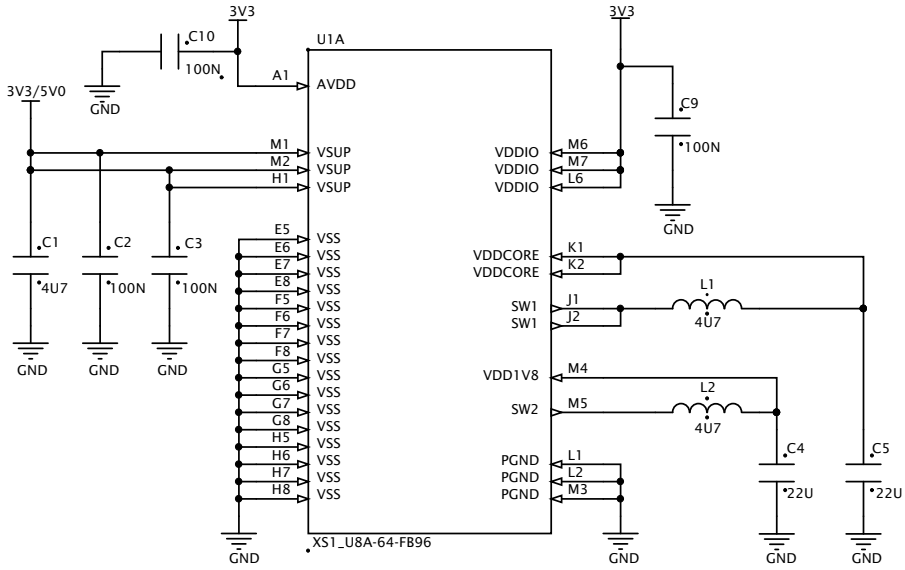

Figure 2: Simplified Reference Schematic

6 Product Overview

The XS1-U8A-64-FB96 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 the USB PHY, a multi-channel ADC (Analog to Digital Converter), deep sleep memory, an oscillator, a real-time counter, and power supply control.

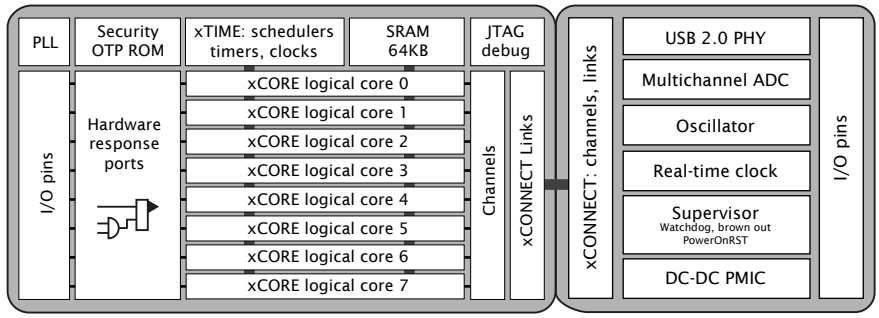

Figure 3: Block Diagram

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. Multiple devices can be deployed and connected using an integrated switching network, enabling more resources to be added 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 USB PHY

The USB PHY is fully compliant with the USB 2.0 specification. It supports high speed (480-Mbps) and full speed (12Mbps) operation.

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-U8A-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 8 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	7	8
5	500 MIPS	500 MHz	125	125	125	125	100	83	71	63

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-U8A-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.

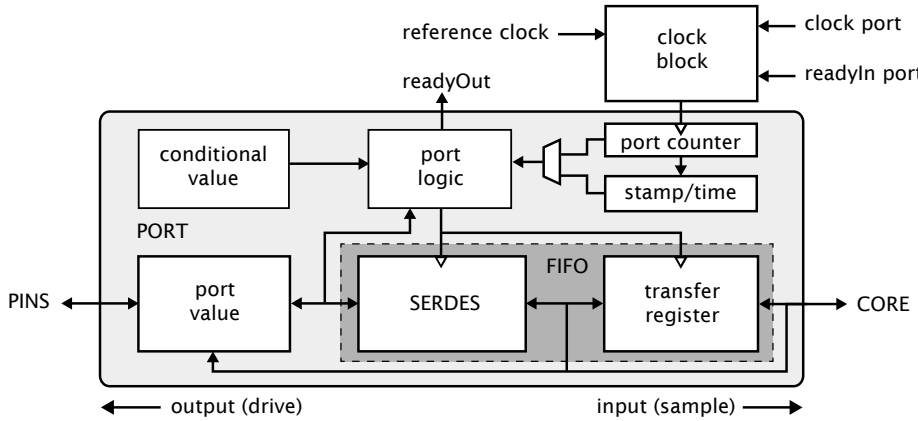

Figure 5: Port block diagram

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

The JTAG chain structure is illustrated in Figure 16. Directly after reset, three TAP controllers are present in the JTAG chain: the debug TAP, the boundary scan TAP and the processor TAP. The debug TAP provides access into the peripherals including the ADC and USB. 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 17.

Figure 17: IDCODE return value

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	1	1	0	1	1	0	0	0	1	1	0	0	1	1
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 18. The OTP User ID field is read from bits [22:31] of the security register , *see* §10.1 (all zero on unprogrammed devices).

Figure 18: USERCODE return value

Bit31																												Bit0
OTP User ID										Unused				Silicon Revision														
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0
0				0				0				2				C				0				0				0	

16 Board Integration

XS1-U8A-64-FB96 devices are optimized for layout on low cost 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-U8A-64-FB96 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 M1 and M2. 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.

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

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

B.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.

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
0x44	RO	PC of logical core 4
0x45	RO	PC of logical core 5
0x46	RO	PC of logical core 6
0x47	RO	PC of logical core 7
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
0x64	RO	SR of logical core 4
0x65	RO	SR of logical core 5
0x66	RO	SR of logical core 6
0x67	RO	SR of logical core 7
0x80 .. 0x9F	RO	Chanend status

Figure 45: Summary

C.21 SR of logical core 3: 0x63

0x63: SR of logical core 3

Bits	Perm	Init	Description
31:0	RO		Value.

C.22 SR of logical core 4: 0x64

0x64: SR of logical core 4

Bits	Perm	Init	Description
31:0	RO		Value.

C.23 SR of logical core 5: 0x65

0x65: SR of logical core 5

Bits	Perm	Init	Description
31:0	RO		Value.

C.24 SR of logical core 6: 0x66

0x66: SR of logical core 6

Bits	Perm	Init	Description
31:0	RO		Value.

C.25 SR of logical core 7: 0x67

0x67: SR of logical core 7

Bits	Perm	Init	Description
31:0	RO		Value.

C.26 Chanend status: 0x80 .. 0x9F

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

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 46: 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.

0x0C: Directions 0-7

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

D.9 Directions 8-15: 0x0D

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

0x0D: Directions 8-15

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

D.10 DEBUG_N configuration: 0x10

Configures the behavior of the DEBUG_N pin.

0x10: DEBUG_N configuration

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

0x04: Node configuration register

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

E.3 Node identifier: 0x05

0x05: Node identifier

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

E.4 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.

E.8 Watchdog Disable: 0xD7

To enable the watchdog, write 0 to this register. To disable the watchdog, write the value 0x0D1SAB1E to this register.

0xD7: Watchdog Disable

Bits	Perm	Init	Description
31:0	RW	0x0D15AB1E	A value of 0x0D15AB1E written to this register resets and disables the watchdog timer.

F USB PHY Configuration

The USB PHY is connected to the following ports:

XS1_PORT_1J
Clk

XS1_PORT_1K
Tx ready out (Tx valid)

XS1_PORT_1H
Tx ready in

XS1_PORT_8A
Tx data

XS1_PORT_1M
Rx ready

XS1_PORT_8C
Rx data

XS1_PORT_1N
flag1

XS1_PORT_1O
flag2

XS1_PORT_1P
flag3

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

0x04: UIFM IFM control

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

F.3 UIFM Device Address: 0x08

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

0x08: UIFM Device Address

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

F.4 UIFM functional control: 0x0C

0x0C: UIFM functional control

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

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

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

0x10: UIFM on-the-go control

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

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

Status flags used for on-the-go negotiation

0x14: UIFM on-the-go flags

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

0x20: UIFM Sticky flags

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

F.10 UIFM port masks: 0x24

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

0x24: UIFM port masks

Bits	Perm	Init	Description
31:23	RO	-	Reserved
22:16	RW	0	Bit mask that determines which flags in UIFM_IFM_FLAG[6:0] contribute to port 1P. If any flag listed in this bitmask is high, port 1P will be high.
15	RO	-	Reserved
14:8	RW	0	Bit mask that determines which flags in UIFM_IFM_FLAG[6:0] contribute to port 1O. If any flag listed in this bitmask is high, port 1O will be high.
7	RO	-	Reserved
6:0	RW	0	Bit mask that determines which flags in UIFM_IFM_FLAG[6:0] contribute to port 1N. If any flag listed in this bitmask is high, port 1N will be high.

F.11 UIFM SOF value: 0x28

USB Start-Of-Frame counter

0x28: UIFM SOF value

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

F.12 UIFM PID: 0x2C

The last USB packet identifier received

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.

K.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

K.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.

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.

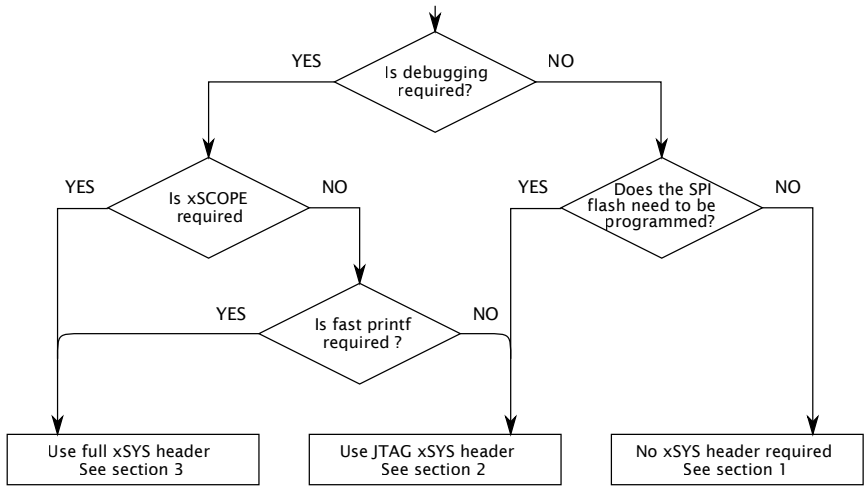

Figure 54: Decision diagram for the xSYS header

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.

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 XLB^{1}_{out}, XLB^{0}_{out}, XLB^{0}_{in}, XLB^{1}_{in} as follows:

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