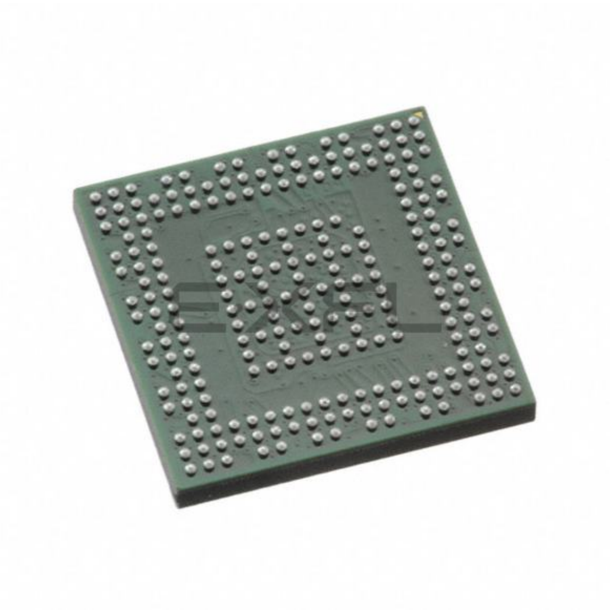

Welcome to E-XFL.COM

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 10-Core
Speed	2000MIPS
Connectivity	USB
Peripherals	-
Number of I/O	104
Program Memory Size	-
Program Memory Type	ROMless
EEPROM Size	-
RAM Size	512K x 8
Voltage - Supply (Vcc/Vdd)	0.95V ~ 3.6V
Data Converters	-
Oscillator Type	External
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	236-LFBGA
Supplier Device Package	236-FBGA (10x10)
Purchase URL	https://www.e-xfl.com/product-detail/xmos/xu210-512-fb236-i20

- **Ports** The I/O pins are connected to the processing cores by Hardware Response ports. The port logic can drive its pins high and low, or it can sample the value on its pins optionally waiting for a particular condition. Section 6.3
- **Clock blocks** xCORE devices include a set of programmable clock blocks that can be used to govern the rate at which ports execute. Section 6.4
- **Memory** Each xCORE Tile integrates a bank of SRAM for instructions and data, and a block of one-time programmable (OTP) memory that can be configured for system wide security features. Section 9
- **PLL** The PLL is used to create a high-speed processor clock given a low speed external oscillator. Section 7
- **USB** The USB PHY provides High-Speed and Full-Speed, device, host, and on-the-go functionality. Data is communicated through ports on the digital node. A library is provided to implement USB device functionality. Section 10
- **JTAG** The JTAG module can be used for loading programs, boundary scan testing, in-circuit source-level debugging and programming the OTP memory. Section 11

1.1 Software

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

1.2 xTIMEcomposer Studio

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

xTIMEcomposer can be driven from either a graphical development environment, or the command line. The tools are supported on Windows, Linux and MacOS X and available at no cost from xmos.com/downloads. Information on using the tools is provided in the xTIMEcomposer User Guide, X3766.

Signal	Function						Type	Properties
X0D41	$X_0L0^{0}_{in}$			$8D^{5}$	$16B^{13}$		I/O	IOL, PD
X0D42	$X_0L0^{0}_{out}$			$8D^{6}$	$16B^{14}$		I/O	IOL, PD
X0D43	$X_0L0^{1}_{out}$			$8D^{7}$	$16B^{15}$		I/O	IOL, PD
X0D49	$X_0L5^{4}_{in}$					$32A^{0}$	I/O	IOR, PD
X0D50	$X_0L5^{3}_{in}$					$32A^{1}$	I/O	IOR, PD
X0D51	$X_0L5^{2}_{in}$					$32A^{2}$	I/O	IOR, PD
X0D52	$X_0L5^{1}_{in}$					$32A^{3}$	I/O	IOR, PD
X0D53	$X_0L5^{0}_{in}$					$32A^{4}$	I/O	IOR, PD
X0D54	$X_0L5^{0}_{out}$					$32A^{5}$	I/O	IOR, PD
X0D55	$X_0L5^{1}_{out}$					$32A^{6}$	I/O	IOR, PD
X0D56	$X_0L5^{2}_{out}$					$32A^{7}$	I/O	IOR, PD
X0D57	$X_0L5^{3}_{out}$					$32A^{8}$	I/O	IOR, PD
X0D58	$X_0L5^{4}_{out}$					$32A^{9}$	I/O	IOR, PD
X0D61	$X_0L6^{4}_{in}$					$32A^{10}$	I/O	IOR, PD
X0D62	$X_0L6^{3}_{in}$					$32A^{11}$	I/O	IOR, PD
X0D63	$X_0L6^{2}_{in}$					$32A^{12}$	I/O	IOR, PD
X0D64	$X_0L6^{1}_{in}$					$32A^{13}$	I/O	IOR, PD
X0D65	$X_0L6^{0}_{in}$					$32A^{14}$	I/O	IOR, PD
X0D66	$X_0L6^{0}_{out}$					$32A^{15}$	I/O	IOR, PD
X0D67	$X_0L6^{1}_{out}$					$32A^{16}$	I/O	IOR, PD
X0D68	$X_0L6^{2}_{out}$					$32A^{17}$	I/O	IOR, PD
X0D69	$X_0L6^{3}_{out}$					$32A^{18}$	I/O	IOR, PD
X0D70	$X_0L6^{4}_{out}$					$32A^{19}$	I/O	IOR, PD
X1D00	$X_0L7^{2}_{in}$	$1A^{0}$					I/O	IOR, PD
X1D01	$X_0L7^{1}_{in}$	$1B^{0}$					I/O	IOR, PD
X1D02	$X_0L4^{0}_{in}$		$4A^{0}$	$8A^{0}$	$16A^{0}$	$32A^{20}$	I/O	IOR, PD
X1D03	$X_0L4^{0}_{out}$		$4A^{1}$	$8A^{1}$	$16A^{1}$	$32A^{21}$	I/O	IOR, PD
X1D04	$X_0L4^{1}_{out}$		$4B^{0}$	$8A^{2}$	$16A^{2}$	$32A^{22}$	I/O	IOR, PD
X1D05	$X_0L4^{2}_{out}$		$4B^{1}$	$8A^{3}$	$16A^{3}$	$32A^{23}$	I/O	IOR, PD
X1D06	$X_0L4^{3}_{out}$		$4B^{2}$	$8A^{4}$	$16A^{4}$	$32A^{24}$	I/O	IOR, PD
X1D07	$X_0L4^{4}_{out}$		$4B^{3}$	$8A^{5}$	$16A^{5}$	$32A^{25}$	I/O	IOR, PD
X1D08	$X_0L7^{4}_{in}$		$4A^{2}$	$8A^{6}$	$16A^{6}$	$32A^{26}$	I/O	IOR, PD
X1D09	$X_0L7^{3}_{in}$		$4A^{3}$	$8A^{7}$	$16A^{7}$	$32A^{27}$	I/O	IOR, PD
X1D10		$1C^{0}$					I/O	IOT, PD
X1D11		$1D^{0}$					I/O	IOT, PD
X1D12		$1E^{0}$					I/O	IOL, PD
X1D13		$1F^{0}$					I/O	IOL, PD
X1D14			$4C^{0}$	$8B^{0}$	$16A^{8}$	$32A^{28}$	I/O	IOR, PD
X1D15			$4C^{1}$	$8B^{1}$	$16A^{9}$	$32A^{29}$	I/O	IOR, PD
X1D16	$X_0L3^{1}_{in}$		$4D^{0}$	$8B^{2}$	$16A^{10}$		I/O	IOL, PD
X1D17	$X_0L3^{0}_{in}$		$4D^{1}$	$8B^{3}$	$16A^{11}$		I/O	IOL, PD
X1D18	$X_0L3^{0}_{out}$		$4D^{2}$	$8B^{4}$	$16A^{12}$		I/O	IOL, PD
X1D19	$X_0L3^{1}_{out}$		$4D^{3}$	$8B^{5}$	$16A^{13}$		I/O	IOL, PD

(continued)

Signal	Function	Type	Properties
X1D70	$X_0L2^4_{out}$ $32A^{19}$	I/O	IOL, PD

System pins (3)			
Signal	**Function**	**Type**	**Properties**
CLK	PLL reference clock	Input	IOL, PD, ST
DEBUG_N	Multi-chip debug	I/O	IOL, PU
MODE[1:0]	Boot mode select	Input	PU

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

5 Example Application Diagram

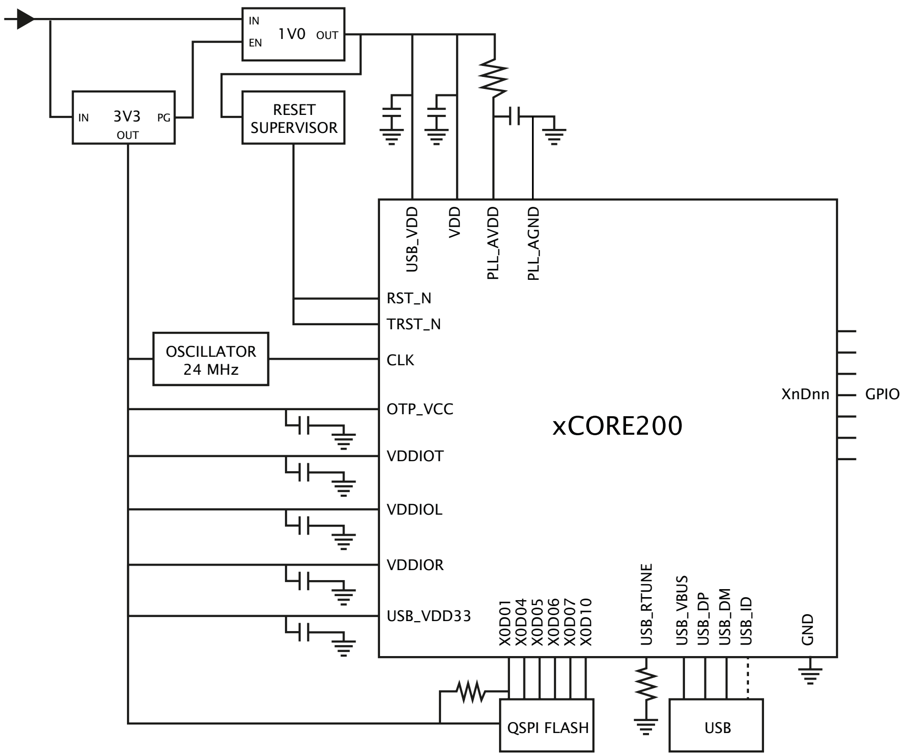

Figure 2: Simplified Reference Schematic

- see Section 10 for details on the USB PHY
- see Section 12 for details on the power supplies and PCB design

6 Product Overview

The XU210-512-FB236 is a powerful device that consists of two xCORE Tiles, each comprising a flexible logical processing cores with tightly integrated I/O and on-chip memory.

6.1 Logical cores

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

Figure 3: Logical core performance

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

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

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

6.2 xTIME scheduler

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

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

6.3 Hardware Response Ports

Hardware Response ports connect an xCORE tile to one or more physical pins and as such define the interface between hardware attached to the XU210-512-FB236, and the software running on it. A combination of 1bit, 4bit, 8bit, 16bit and 32bit ports are available. All pins of a port provide either output or input. Signals in different directions cannot be mapped onto the same port.

The port logic can drive its pins high or low, or it can sample the value on its pins, optionally waiting for a particular condition. Ports are accessed using dedicated instructions that are executed in a single processor cycle. xCORE-200 IO pins can

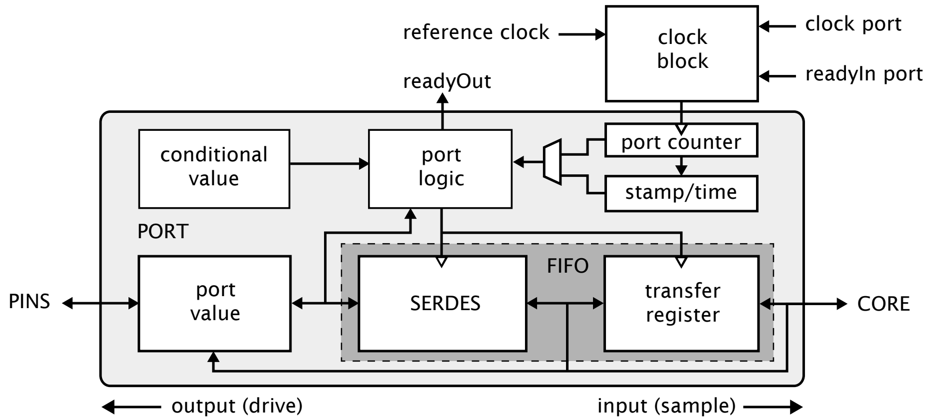

Figure 4: Port block diagram

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

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

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

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

6.4 Clock blocks

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

A clock block can use a 1-bit port as its clock source allowing external application clocks to be used to drive the input and output interfaces. xCORE-200 clock blocks optionally divide the clock input from a 1-bit port.

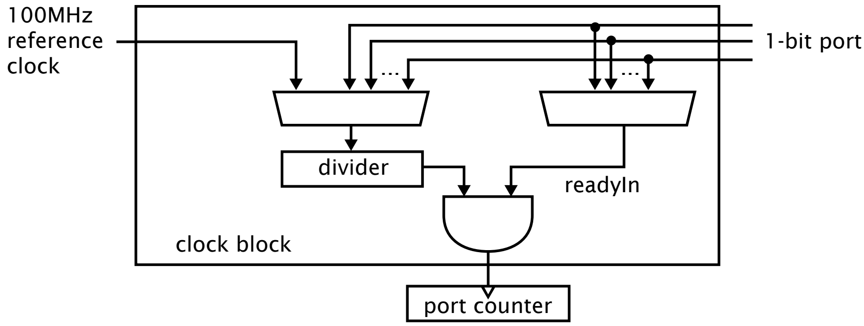

Figure 5: Clock block diagram

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

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

6.5 Channels and Channel Ends

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

6.6 xCONNECT Switch and Links

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

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

The links operate in either 2 wires per direction or 5 wires per direction mode, depending on the amount of bandwidth required. Circuit switched, streaming and packet switched data can both be supported efficiently. Streams provide the fastest possible data rates between xCORE Tiles (up to 250 MBit/s), but each stream requires a single link to be reserved between switches on two tiles. All packet communications can be multiplexed onto a single link.

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

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

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

8.3 Boot from SPI slave

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

Figure 12: SPI slave pins

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

The xCORE Tile expects each byte to be transferred with the *least-significant bit first*. The pins used for SPI boot are hardcoded in the boot ROM and cannot be changed. If required, an SPI boot program can be burned into OTP that uses different pins.

8.4 Boot from xConnect Link

If set to boot from an xConnect Link, the processor enables its link(s) around 2 us after the boot process starts. Enabling the Link switches off the pull-down resistors on the link, drives all the TX wires low (the initial state for the Link), and monitors the RX pins for boot-traffic; they must be low at this stage. If the internal pull-down is too weak to drain any residual charge, external pull-downs of 10K may be required on those pins.

The boot-rom on the core will then:

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

Feature	Bit	Description
Disable JTAG	0	The JTAG interface is disabled, making it impossible for the tile state or memory content to be accessed via the JTAG interface.
Disable Link access	1	Other tiles are forbidden access to the processor state via the system switch. Disabling both JTAG and Link access transforms an xCORE Tile into a "secure island" with other tiles free for non-secure user application code.
Secure Boot	5	The xCORE Tile is forced to boot from address 0 of the OTP, allowing the xCORE Tile boot ROM to be bypassed (*see* §8).
Redundant rows	7	Enables redundant rows in OTP.
Sector Lock 0	8	Disable programming of OTP sector 0.
Sector Lock 1	9	Disable programming of OTP sector 1.
Sector Lock 2	10	Disable programming of OTP sector 2.
Sector Lock 3	11	Disable programming of OTP sector 3.
OTP Master Lock	12	Disable OTP programming completely: disables updates to all sectors and security register.
Disable JTAG-OTP	13	Disable all (read & write) access from the JTAG interface to this OTP.
Disable Global Debug	14	Disables access to the DEBUG_N pin.
	21..15	General purpose software accessable security register available to end-users.
	31..22	General purpose user programmable JTAG UserID code extension.

Figure 13: Security register features

interface, although data memory can be expanded through appropriate use of the ports.

10 USB PHY

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

The USB PHY is connected to the ports on Tile 0 and Tile 1 as shown in Figure 14. When the USB PHY is enabled on Tile 0, the ports shown can on Tile 0 only be used with the USB PHY. When the USB PHY is enabled on Tile 1, then the ports shown can on Tile 1 only be used with the USB PHY. All other IO pins and ports are unaffected. The USB PHY should not be enabled on both tiles. Two clock blocks can be used to clock the USB ports. One clock block for the TXDATA path, and one clock block for the RXDATA path. Details on how to connect those ports are documented in an application note on USB for xCORE-200.

In any case, extra components (such as a ferrite bead and diodes) may be required for EMC compliance and ESD protection. Different wiring is required for USB-host and USB-OTG.

10.2 Logical Core Requirements

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

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

11 JTAG

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

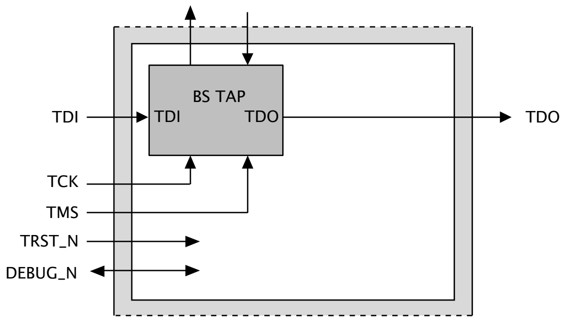

Figure 16: JTAG chain structure

The JTAG chain structure is illustrated in Figure 16. It comprises a single 1149.1 compliant TAP that can be used for boundary scan of the I/O pins. It has a 4-bit IR and 32-bit DR. It also provides access to a chip TAP that in turn can access the xCORE Tile for loading code and debugging.

The TRST_N pin must be asserted low during and after power up for 100 ns. If JTAG is not required, the TRST_N pin can be tied to ground to hold the JTAG module in reset.

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

- Use a minimum of vias in high speed USB traces.
- Avoid corners in the trace. Where necessary, rather than turning through a 90 degree angle, use two 45 degree turns or an arc.
- DO NOT route USB traces near clock sources, clocked circuits or magnetic devices.
- Avoid stubs on high speed USB signals.

12.3 Land patterns and solder stencils

The package is a 236 ball Fine Ball Grid Array (FBGA) on a 0.5 mm pitch. We recommend you use HDI or better PCB technology. The missing balls in the outer rows can be used to route the first inner row out over the top layer. The missing balls in the center can be used for ground vias. The missing rows four and five can be used for VDD vias if required.

The land patterns and solder stencils will depend on the PCB manufacturing process. We recommend you design them with using the IPC specifications *"Generic Requirements for Surface Mount Design and Land Pattern Standards"* IPC-7351B. This standard aims to achieve desired targets of heel, toe and side fillets for solder-joints. The mechanical drawings in Section 14 specify the dimensions and tolerances.

12.4 Ground and Thermal Vias

Vias from the ground balls into the ground plane of the PCB are recommended for a low inductance ground connection and good thermal performance. Typical designs could use 16 vias in a 4 x 4 grid, equally spaced amongst the ground balls.

12.5 Moisture Sensitivity

XMOS devices are, like all semiconductor devices, susceptible to moisture absorption. When removed from the sealed packaging, the devices slowly absorb moisture from the surrounding environment. If the level of moisture present in the device is too high during reflow, damage can occur due to the increased internal vapour pressure of moisture. Example damage can include bond wire damage, die lifting, internal or external package cracks and/or delamination.

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

13 DC and Switching Characteristics

13.1 Operating Conditions

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

Figure 21: Operating conditions

13.2 DC Characteristics, VDDIO=3V3

Symbol	Parameter	MIN	TYP	MAX	UNITS	Notes
V(IH)	Input high voltage	2.00		3.60	V	A
V(IL)	Input low voltage	-0.30		0.70	V	A
V(OH)	Output high voltage	2.20			V	B, C
V(OL)	Output low voltage			0.40	V	B, C
I(PU)	Internal pull-up current (Vin=0V)	-100			µA	D
I(PD)	Internal pull-down current (Vin=3.3V)			100	µA	D
I(LC)	Input leakage current	-10		10	µA	

Figure 22: DC characteristics

A All pins except power supply pins.

B Pins X1D40, X1D41, X1D42, X1D43, X1D26, and X1D27 are nominal 8 mA drivers, the remainder of the general-purpose I/Os are 4 mA.

C Measured with 4 mA drivers sourcing 4 mA, 8 mA drivers sourcing 8 mA.

D Used to guarantee logic state for an I/O when high impedance. The internal pull-ups/pull-downs should not be used to pull external circuitry. In order to pull the pin to the opposite state, a 4K7 resistor is recommended to overome the internal pull current.

B Processor Status Configuration

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

Number	Perm	Description
0x00	RW	RAM base address
0x01	RW	Vector base address
0x02	RW	xCORE Tile control
0x03	RO	xCORE Tile boot status
0x05	RW	Security configuration
0x06	RW	Ring Oscillator Control
0x07	RO	Ring Oscillator Value
0x08	RO	Ring Oscillator Value
0x09	RO	Ring Oscillator Value
0x0A	RO	Ring Oscillator Value
0x0C	RO	RAM size
0x10	DRW	Debug SSR
0x11	DRW	Debug SPC
0x12	DRW	Debug SSP
0x13	DRW	DGETREG operand 1
0x14	DRW	DGETREG operand 2
0x15	DRW	Debug interrupt type
0x16	DRW	Debug interrupt data
0x18	DRW	Debug core control
0x20 .. 0x27	DRW	Debug scratch
0x30 .. 0x33	DRW	Instruction breakpoint address
0x40 .. 0x43	DRW	Instruction breakpoint control
0x50 .. 0x53	DRW	Data watchpoint address 1
0x60 .. 0x63	DRW	Data watchpoint address 2
0x70 .. 0x73	DRW	Data breakpoint control register
0x80 .. 0x83	DRW	Resources breakpoint mask
0x90 .. 0x93	DRW	Resources breakpoint value
0x9C .. 0x9F	DRW	Resources breakpoint control register

Figure 34: Summary

0x0C: RAM size

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

B.12 Debug SSR: 0x10

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

0x10: Debug SSR

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

B.13 Debug SPC: 0x11

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

0x11: Debug SPC

Bits	Perm	Init	Description
31:0	DRW		Value.

B.14 Debug SSP: 0x12

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

0x1F: Debug source

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

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

These registers contain status information for low level debugging (read-only), the network number that each link belongs to, and the direction that each link is part of. The registers control links 0..7.

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

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

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

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

0x2C: UIFM PID

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

F.13 UIFM Endpoint: 0x30

The last endpoint seen

0x30: UIFM Endpoint

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

F.14 UIFM Endpoint match: 0x34

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

0x34: UIFM Endpoint match

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

F.15 OTG Flags mask: 0x38

0x38: OTG Flags mask

Bits	Perm	Init	Description
31:0	RW	0	Data

F.16 UIFM power signalling: 0x3C

0x3C: UIFM power signalling

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

F.17 UIFM PHY control: 0x40

0x40: UIFM PHY control

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

G JTAG, xSCOPE and Debugging

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

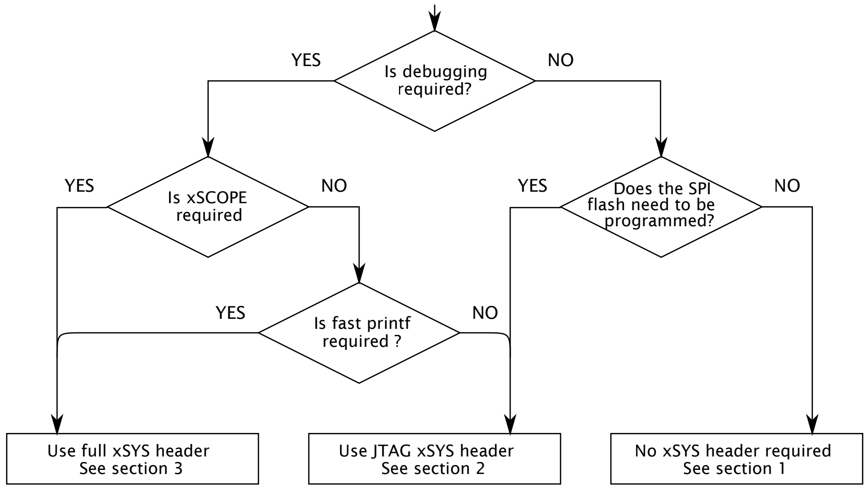

Figure 39: Decision diagram for the xSYS header

G.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.

G.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

H Schematics Design Check List

☑ This section is a checklist for use by schematics designers using the XU210-512-FB236. Each of the following sections contains items to check for each design.

H.1 Power supplies

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

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

☐ The VDD (core) supply is capable of supplying 700 mA (Section 12 and Figure 22).

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

H.2 Power supply decoupling

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

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

H.3 Power on reset

☐ The RST_N and TRST_N pins are asserted (low) during or after power up. The device is not used until these resets have taken place.

H.4 Clock

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

☐ Pins MODE0 and MODE1 are set to the correct value for the chosen oscillator frequency. The MODE settings are shown in the Oscillator section, Section 7. If you have a choice between two values, choose the value with the highest multiplier ratio since that will boot faster.

J Associated Design Documentation

Document Title	Information	Document Number
Estimating Power Consumption For XS1-U Devices	Power consumption	
Programming XC on XMOS Devices	Timers, ports, clocks, cores and channels	X9577
xTIMEcomposer User Guide	Compilers, assembler and linker/mapper Timing analyzer, xScope, debugger Flash and OTP programming utilities	X3766

K Related Documentation

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