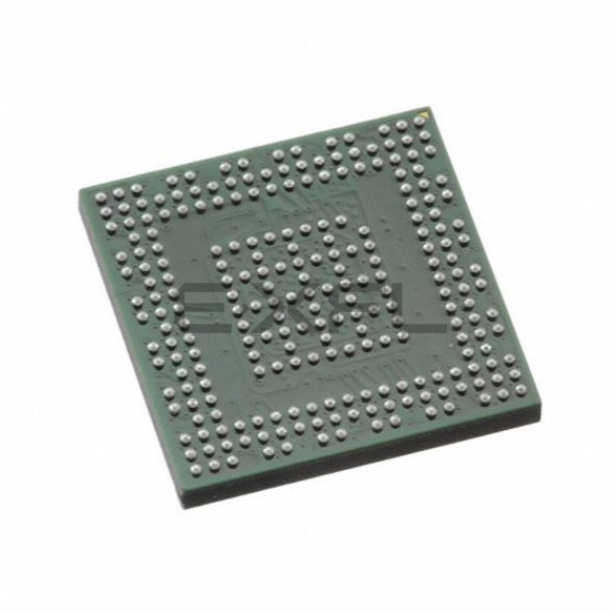


Welcome to E-XFL.COM

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 16-Core
Speed	2000MIPS
Connectivity	RGMII, USB
Peripherals	-
Number of I/O	73
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	0°C ~ 70°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/xe216-512-fb236-c20

1 xCORE Multicore Microcontrollers

The xCORE200 Series is a comprehensive range of 32-bit multicore microcontrollers that brings the low latency and timing determinism of the xCORE architecture to mainstream embedded applications. Unlike conventional microcontrollers, xCORE multicore microcontrollers execute multiple real-time tasks simultaneously and communicate between tasks using a high speed network. Because xCORE multicore microcontrollers are completely deterministic, you can write software to implement functions that traditionally require dedicated hardware.

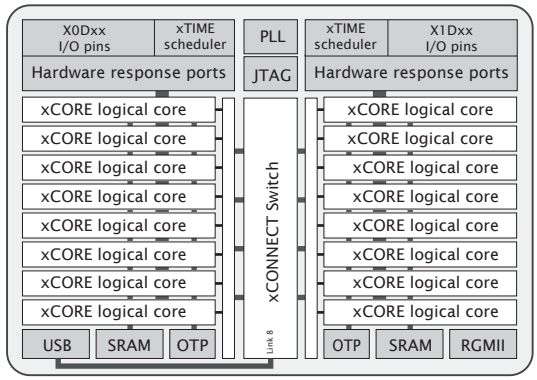


Figure 1: XE216-512-FB236 block diagram

Key features of the XE216-512-FB236 include:

- **Tiles**: Devices consist of one or more xCORE tiles. Each tile contains between five and eight 32-bit xCOREs with highly integrated I/O and on-chip memory.
- **Logical cores** Each logical core can execute tasks such as computational code, DSP code, control software (including logic decisions and executing a state machine) or software that handles I/O. Section 6.1
- **xTIME scheduler** The xTIME scheduler performs functions similar to an RTOS, in hardware. It services and synchronizes events in a core, so there is no requirement for interrupt handler routines. The xTIME scheduler triggers cores on events generated by hardware resources such as the I/O pins, communication channels and timers. Once triggered, a core runs independently and concurrently to other cores, until it pauses to wait for more events. Section 6.2
- **Channels and channel ends** Tasks running on logical cores communicate using channels formed between two channel ends. Data can be passed synchronously or asynchronously between the channel ends assigned to the communicating tasks. Section 6.5
- **xCONNECT Switch and Links** Between tiles, channel communications are implemented over a high performance network of xCONNECT Links and routed through a hardware xCONNECT Switch. Section 6.6

3 Pin Configuration

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
A	GND	VDDIOL	VDDIOL		TCK	CLK		4F X1D31 rx1	4F X1D29 rx_dll		8D X1D41 tx2	OTP_VCC		NC	MODE[0]		4F X0D29	VDDIOR	GND
B	1M X0D36	VDDIOL	VDDIOL	TDO	TMS	TRST_N	4E X1D33 rx3	4E X1D32 rx2	4F X1D28 rx_clk	4E X1D26 tx_clk	8D X1D42 tx1	OTP_VCC	NC	NC	MODE[1]	4E X0D33	4E X0D32	VDDIOR	VDDIOR
C	1N X0D37	1O X0D38	VDDIOL	TDI	DEBUG_N	RST_N	1C X1D10	1D X1D11	4F X1D30 rx0	4E X1D27 tx_dll	8D X1D43 tx0	8D X1D40 tx3	NC	NC	4F X0D31	4F X0D30	4F X0D28	4E X0D26	4E X0D27
D		1P X0D39	8D X0D40														1K X0D34	1L X0D35	
E	8D X0D43	8D X0D42	8D X0D41														1J X0D25	1I X0D24	1S X1D01
F	1K X1D34	1L X1D35	1M X1D36				NC	VDD	VDD	VDDIOT	VDD	VDD	PLL_AVDD	PLL_AGND			4A X1D08	4A X1D09	1A X1D00
G		32A X1D49	32A X1D50			VDD		GND		GND		GND		VDD			32A X0D69	32A X0D70	
H	32A X1D53	32A X1D52	32A X1D51			VDD	GND	GND	GND	GND	GND	GND	GND	VDD			32A X0D68	32A X0D67	32A X0D66
J	32A X1D54	32A X1D55	32A X1D56			VDD		GND		GND		GND		VDD			32A X0D63	32A X0D64	32A X0D65
K		32A X1D58	32A X1D57			VDD	GND	GND	GND	GND	GND	GND	GND	VDD			32A X0D62	32A X0D61	
L	32A X1D63	32A X1D62	32A X1D61			VDD		GND		GND		GND		VDD			32A X0D58	32A X0D57	32A X0D56
M	32A X1D64	32A X1D65	32A X1D66			VDD	GND	GND	GND	GND	GND	GND	GND	VDD			32A X0D53	32A X0D54	32A X0D55
N		32A X1D67	32A X1D68			VDD		GND		GND		GND		VDD			32A X0D51	32A X0D52	
P	32A X1D70	32A X1D69	1N X1D37			VDD	VDD	VDD	USB_VDD	USB_VDD	VDD	VDD	VDD	NC			4B X1D07	32A X0D50	32A X0D49
R	1O X1D38	1P X1D39	4D X1D17														4A X1D03	4B X1D05	4B X1D06
T		4D X1D16	4D X1D18														4A X1D02	4B X1D04	
U	1C X0D10	1B X0D01	4D X1D19	1A X0D00	1D X0D11	4B X0D07	1E X1D12	USB_VDD33	USB_VBUS	USB_ID	USB_VSSAC	NC	1I X1D24	1G X0D22	1F X0D13	1H X0D23	4D X0D19	4D X0D18	4D X0D17
V	1G X1D22	VDDIOL	VDDIOL	4B X0D04	4B X0D06	4A X0D03	4A X0D08	4A X0D09	USB_DM	USB_DP	4C X1D21	4C X1D14	1J X1D25	4C X0D21	4C X0D14	1E X0D12	VDDIOR	VDDIOR	4D X0D16
W	GND	VDDIOL	1H X1D23		4B X0D05	4A X0D02		1F X1D13	USB_RTUNE		4C X1D20	4C X1D15		4C X0D20	4C X0D15		VDDIOR	VDDIOR	GND

ports are available. All pins of a port provide either output or input. Signals in different directions cannot be mapped onto the same port.

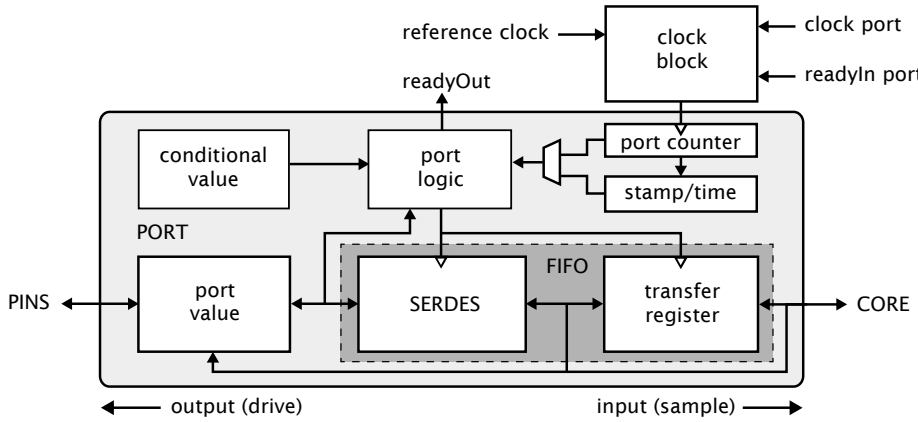


Figure 4: Port block diagram

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. xCORE-200 IO pins can 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.

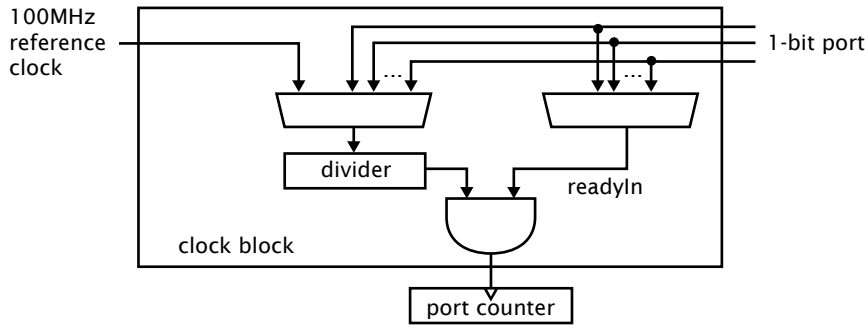


Figure 5: Clock block diagram

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.

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

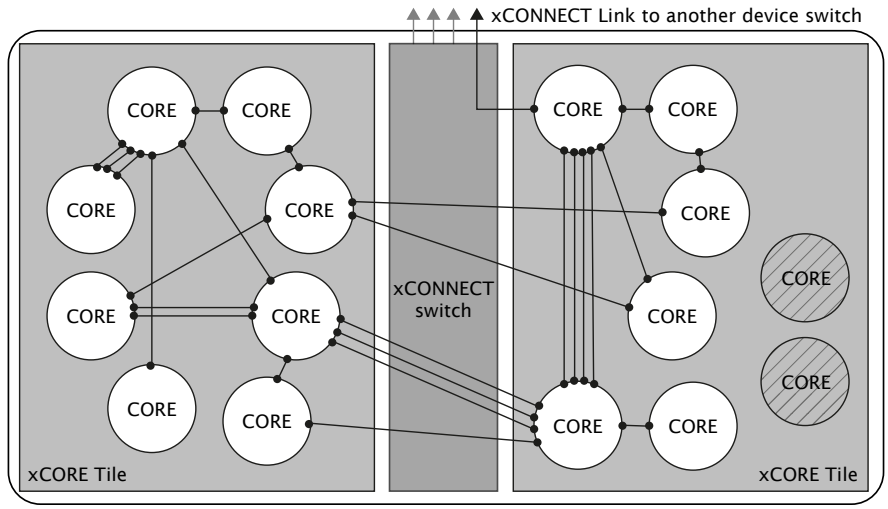


Figure 6: Switch, links and channel ends

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.

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

7 PLL

The PLL creates a high-speed clock that is used for the switch, tile, and reference clock. The 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 7:

Figure 7: PLL multiplier values and MODE pins

Oscillator Frequency	MODE 1	MODE 0	Tile Boot Frequency	PLL Ratio	PLL settings OD	F	R
3.25-10 MHz	0	0	130-400 MHz	40	1	159	0
9-25 MHz	1	1	144-400 MHz	16	1	63	0
25-50 MHz	1	0	167-400 MHz	8	1	31	0
50-100 MHz	0	1	196-400 MHz	4	1	15	0

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

The OTP memory is programmed using three special I/O ports: the OTP address port is a 16-bit port with resource ID 0x100200, the OTP data is written via a 32-bit port with resource ID 0x200100, and the OTP control is on a 16-bit port with ID 0x100300. Programming is performed through `libotp` and `xburn`.

9.2 SRAM

Each xCORE Tile integrates a single 256KB SRAM bank for both instructions and data. All internal memory is 32 bits wide, and instructions are either 16-bit or 32-bit. Byte (8-bit), half-word (16-bit) or word (32-bit) accesses are supported and are executed within one tile clock cycle. There is no dedicated external memory interface, although data memory can be expanded through appropriate use of the ports.

10 USB PHY

The USB PHY provides High-Speed and Full-Speed, device, host, and on-the-go functionality. The PHY is configured through a set of peripheral registers (Appendix F),

The RGMII block is connected to the ports on Tile 1 as shown in Figure 17. When the RGMII block is enabled, the ports shown can only be used with the RGMII block, and IO pins X1D26..X1D33/X1D40..X1D43 can only be used with the RGMII block. Ports and pins not used in Figure 17 can be used as normal.

The RGMII block generates a clock (configured using processor status register 2), and has the facility to delay the outgoing clock edge, putting it out of phase with the data. The RGMII block translates the double data-rate 4-wire data signals and 1-wire control signal into single-data rate 8-wire TX and DX signals and two control signals. Figure 17 shows how four clock blocks can be used to clock the RGMII ports. One clock block for the TXDATA path, one clock block for the RXDATA path, one clock block to delay the TX_CLK, and one clock block clocked on a negative valid signal to enable mode switching between 10/100/1000 speeds. Details on how to connect those ports are documented in an application note on RGMII for xCORE200. The XMOS RGMII software component runs a MAC layer on Tile 1.

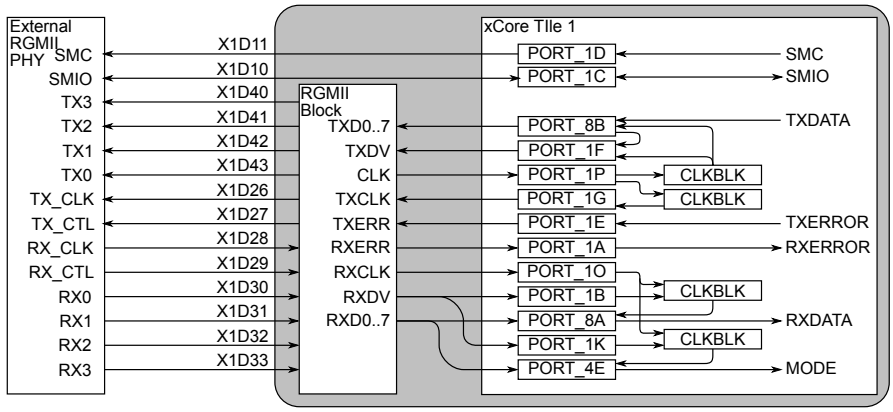


Figure 17: RGMII port functions on Tile 1

The SMI interface should be connected to two one-bit ports that are configured as open-drain IOs, using external pull-ups to 2.5V. Ports 1C and 1D are notionally allocated for this, but any GPIO can be used for this purpose.

The bundles of RX and TX pins should be wired using matched trace-lengths over an uninterrupted ground-plane. The RGMII pins are supplied through the VDDIOT supply pins, which should be provided with 2.5V. Decouplers should be placed with a short path to VDDIOT and ground. If the PHY supports a 3.3V IO voltage, then a 3.3V supply can be used for VDDIOT.

The RGMII PHY should be configured so that RX_CLK is low during reset of the xCORE. This may be achieved by putting a pull-down resistor on the reset of the PHY, keeping the PHY in reset until the RGMII layer on the xCORE takes the PHY out of reset.

14.7 xCORE Tile I/O AC Characteristics

Figure 30: 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.

14.8 xConnect Link Performance

Figure 31: Link performance

Symbol	Parameter	MIN	TYP	MAX	UNITS	Notes
B(2blinkP)	2b link bandwidth (packetized)			87	MBit/s	A, B
B(5blinkP)	5b link bandwidth (packetized)			217	MBit/s	A, B
B(2blinkS)	2b link bandwidth (streaming)			100	MBit/s	B
B(5blinkS)	5b link bandwidth (streaming)			250	MBit/s	B

A Assumes 32-byte packet in 3-byte header mode. Actual performance depends on size of the header and payload.

B 7.5 ns symbol time.

The asynchronous nature of links means that the relative phasing of CLK clocks is not important in a multi-clock system, providing each meets the required stability criteria.

14.9 JTAG Timing

Figure 32: JTAG timing

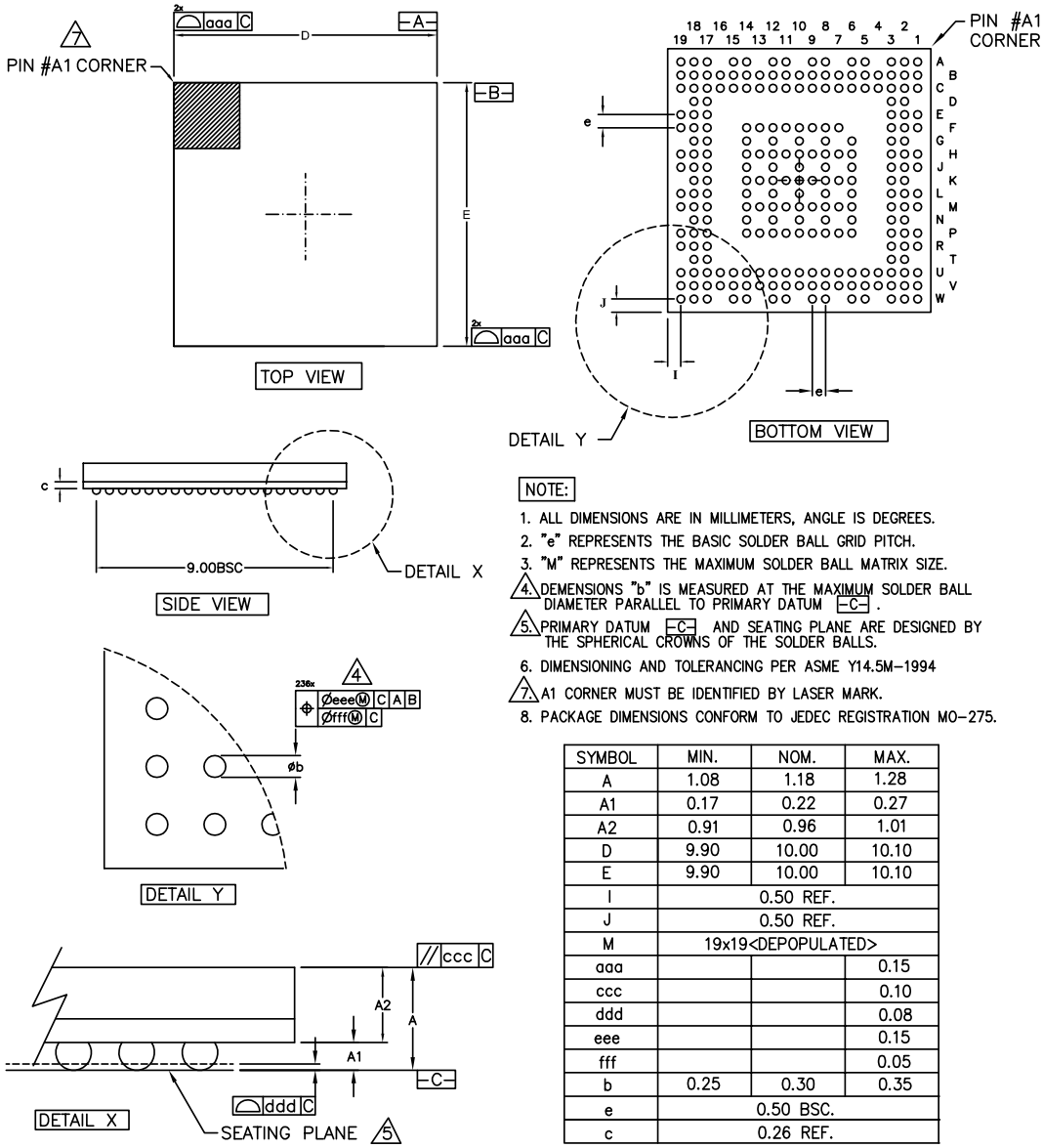
Symbol	Parameter	MIN	TYP	MAX	UNITS	Notes
f(TCK_D)	TCK frequency (debug)			18	MHz	
f(TCK_B)	TCK frequency (boundary scan)			10	MHz	
T(SETUP)	TDO to TCK setup time	5			ns	A
T(HOLD)	TDO to TCK hold time	5			ns	A
T(DELAY)	TCK to output delay			15	ns	B

A Timing applies to TMS and TDI inputs.

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

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

15 Package Information



SYMBOL	MIN.	NOM.	MAX.
A	1.08	1.18	1.28
A1	0.17	0.22	0.27
A2	0.91	0.96	1.01
D	9.90	10.00	10.10
E	9.90	10.00	10.10
I		0.50 REF.	
J		0.50 REF.	
M		19x19<DEPOPULATED>	
aaa			0.15
ccc			0.10
ddd			0.08
eee			0.15
fff			0.05
b	0.25	0.30	0.35
e		0.50 BSC.	
c		0.26 REF.	

0x07: Ring Oscillator Value

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

B.8 Ring Oscillator Value: 0x08

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

0x08: Ring Oscillator Value

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

B.9 Ring Oscillator Value: 0x09

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

0x09: Ring Oscillator Value

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

B.10 Ring Oscillator Value: 0x0A

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

0x0A: Ring Oscillator Value

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

B.11 RAM size: 0x0C

The size of the RAM in bytes

B.25 Data breakpoint control register: 0x70 .. 0x73

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

0x70 .. 0x73: Data breakpoint control register

Bits	Perm	Init	Description
31:24	RO	-	Reserved
23:16	DRW	0	A bit for each thread in the machine allowing the breakpoint to be enabled individually for each thread.
15:3	RO	-	Reserved
2	DRW	0	When 1 the breakpoints will be be triggered on loads.
1	DRW	0	Determines the break condition: 0 = A AND B, 1 = A OR B.
0	DRW	0	When 1 the instruction breakpoint is enabled.

B.26 Resources breakpoint mask: 0x80 .. 0x83

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

0x80 .. 0x83: Resources breakpoint mask

Bits	Perm	Init	Description
31:0	DRW		Value.

B.27 Resources breakpoint value: 0x90 .. 0x93

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

0x90 .. 0x93: Resources breakpoint value

Bits	Perm	Init	Description
31:0	DRW		Value.

B.28 Resources breakpoint control register: 0x9C .. 0x9F

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

C Tile Configuration

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

Number	Perm	Description
0x00	CRO	Device identification
0x01	CRO	xCORE Tile description 1
0x02	CRO	xCORE Tile description 2
0x04	CRW	Control PSwitch permissions to debug registers
0x05	CRW	Cause debug interrupts
0x06	CRW	xCORE Tile clock divider
0x07	CRO	Security configuration
0x20 .. 0x27	CRW	Debug scratch
0x40	CRO	PC of logical core 0
0x41	CRO	PC of logical core 1
0x42	CRO	PC of logical core 2
0x43	CRO	PC of logical core 3
0x44	CRO	PC of logical core 4
0x45	CRO	PC of logical core 5
0x46	CRO	PC of logical core 6
0x47	CRO	PC of logical core 7
0x60	CRO	SR of logical core 0
0x61	CRO	SR of logical core 1
0x62	CRO	SR of logical core 2
0x63	CRO	SR of logical core 3
0x64	CRO	SR of logical core 4
0x65	CRO	SR of logical core 5
0x66	CRO	SR of logical core 6
0x67	CRO	SR of logical core 7

Figure 37: Summary

C.1 Device identification: 0x00

This register identifies the xCORE Tile

0x04: Control PSwitch permissions to debug registers

Bits	Perm	Init	Description
31	CRW	0	When 1 the PSwitch is restricted to RO access to all CRW registers from SSwitch, XCore(PS_DBG_Scratch) and JTAG
30:1	RO	-	Reserved
0	CRW	0	When 1 the PSwitch is restricted to RO access to all CRW registers from SSwitch

C.5 Cause debug interrupts: 0x05

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

0x05: Cause debug interrupts

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

C.6 xCORE Tile clock divider: 0x06

This register contains the value used to divide the PLL clock to create the xCORE tile clock. The divider is enabled under control of the tile control register

0x06: xCORE Tile clock divider

Bits	Perm	Init	Description
31	CRW	0	Clock disable. Writing '1' will remove the clock to the tile.
30:16	RO	-	Reserved
15:0	CRW	0	Clock divider.

C.7 Security configuration: 0x07

Copy of the security register as read from OTP.

0x07: Security configuration

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

C.8 Debug scratch: 0x20 .. 0x27

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

0x20 .. 0x27: Debug scratch

Bits	Perm	Init	Description
31:0	CRW		Value.

C.9 PC of logical core 0: 0x40

Value of the PC of logical core 0.

0x40: PC of logical core 0

Bits	Perm	Init	Description
31:0	CRO		Value.

C.10 PC of logical core 1: 0x41

Value of the PC of logical core 1.

D.8 System JTAG device ID register: 0x09

0x09: System JTAG device ID register

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

D.9 System USERCODE register: 0x0A

0x0A: System USERCODE register

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

D.10 Directions 0-7: 0x0C

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

0x0C: Directions 0-7

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

D.11 Directions 8-15: 0x0D

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

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.

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

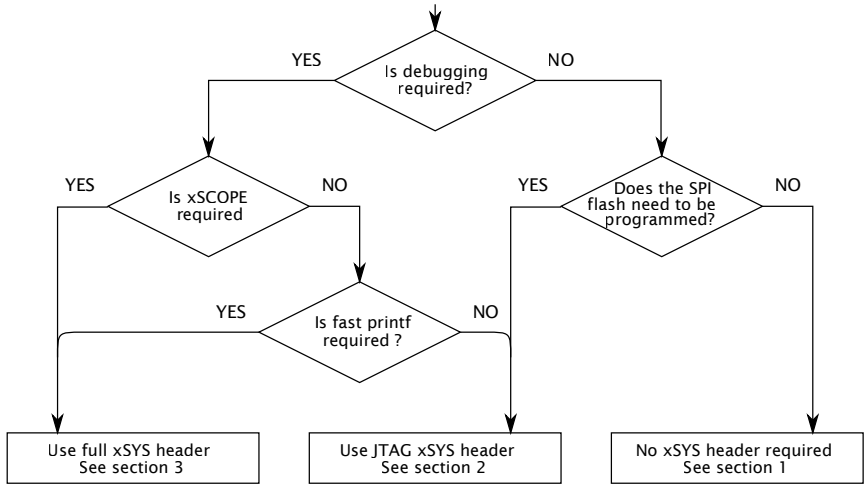


Figure 41: Decision diagram for the xSYS header

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

- TDO to pin 13 of the xSYS header

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

G.3 Full xSYS header

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

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

I PCB Layout Design Check List

☑ This section is a checklist for use by PCB designers using the XS2-UE16A-512-FB236. Each of the following sections contains items to check for each design.

I.1 Ground Plane

☐ Each ground ball has a via to minimize impedance and conduct heat away from the device. (Section 13.4)

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

I.2 RGMII interface

This section can be skipped if you do not have any device connected to the RGMII interface.

☐ The RGMII traces are length and impedance matched.

I.3 Power supply decoupling

☐ The decoupling capacitors are all placed close to a supply pin (Section 13).

☐ The decoupling capacitors are spaced around the device (Section 13).

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

I.4 PLL_AVDD

☐ The PLL_AVDD filter (especially the capacitor) is placed close to the PLL_AVDD pin (Section 13).