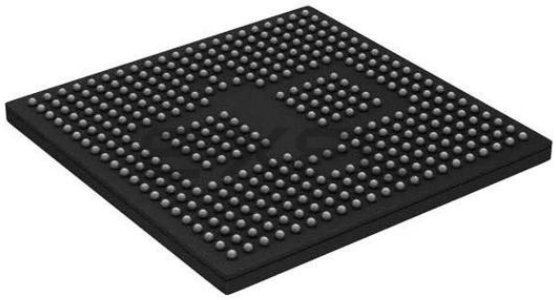


Welcome to E-XFL.COM

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 24-Core
Speed	4000MIPS
Connectivity	USB
Peripherals	-
Number of I/O	176
Program Memory Size	-
Program Memory Type	ROMless
EEPROM Size	-
RAM Size	1M 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	374-LFBGA
Supplier Device Package	374-FBGA (18x18)
Purchase URL	https://www.e-xfl.com/product-detail/xmos/xu224-1024-fb374-i40

Signal	Function					Type	Properties
X1D67	$X_0L2^1_{out}$				$32A^{16}$	I/O	IO, PD
X1D68	$X_0L2^2_{out}$				$32A^{17}$	I/O	IO, PD
X1D69	$X_0L2^3_{out}$				$32A^{18}$	I/O	IO, PD
X1D70	$X_0L2^4_{out}$				$32A^{19}$	I/O	IO, PD
X2D00		$1A^0$				I/O	IO, PD
X2D02			$4A^0$	$8A^0$	$16A^0$ $32A^{20}$	I/O	IO, PD
X2D03			$4A^1$	$8A^1$	$16A^1$ $32A^{21}$	I/O	IO, PD
X2D04			$4B^0$	$8A^2$	$16A^2$ $32A^{22}$	I/O	IO, PD
X2D05			$4B^1$	$8A^3$	$16A^3$ $32A^{23}$	I/O	IO, PD
X2D06			$4B^2$	$8A^4$	$16A^4$ $32A^{24}$	I/O	IO, PD
X2D07			$4B^3$	$8A^5$	$16A^5$ $32A^{25}$	I/O	IO, PD
X2D08			$4A^2$	$8A^6$	$16A^6$ $32A^{26}$	I/O	IO, PD
X2D09			$4A^3$	$8A^7$	$16A^7$ $32A^{27}$	I/O	IO, PD
X2D11		$1D^0$				I/O	IO, PD
X2D12		$1E^0$				I/O	IO, PD
X2D13		$1F^0$				I/O	IO, PD
X2D14			$4C^0$	$8B^0$	$16A^8$ $32A^{28}$	I/O	IO, PD
X2D15			$4C^1$	$8B^1$	$16A^9$ $32A^{29}$	I/O	IO, PD
X2D16	$X_2L4^4_{in}$		$4D^0$	$8B^2$	$16A^{10}$	I/O	IO, PD
X2D17	$X_2L4^3_{in}$		$4D^1$	$8B^3$	$16A^{11}$	I/O	IO, PD
X2D18	$X_2L4^2_{in}$		$4D^2$	$8B^4$	$16A^{12}$	I/O	IO, PD
X2D19	$X_2L4^1_{in}$		$4D^3$	$8B^5$	$16A^{13}$	I/O	IO, PD
X2D20			$4C^2$	$8B^6$	$16A^{14}$ $32A^{30}$	I/O	IO, PD
X2D21			$4C^3$	$8B^7$	$16A^{15}$ $32A^{31}$	I/O	IO, PD
X2D22		$1G^0$				I/O	IO, PD
X2D23		$1H^0$				I/O	IO, PD
X2D24	$X_2L7^0_{in}$	$1I^0$				I/O	IO, PD
X2D25	$X_2L7^0_{out}$	$1J^0$				I/O	IO, PD
X2D26	$X_2L7^3_{out}$		$4E^0$	$8C^0$	$16B^0$	I/O	IO, PD
X2D27	$X_2L7^4_{out}$		$4E^1$	$8C^1$	$16B^1$	I/O	IO, PD
X2D28			$4F^0$	$8C^2$	$16B^2$	I/O	IO, PD
X2D29			$4F^1$	$8C^3$	$16B^3$	I/O	IO, PD
X2D30			$4F^2$	$8C^4$	$16B^4$	I/O	IO, PD
X2D31			$4F^3$	$8C^5$	$16B^5$	I/O	IO, PD
X2D32			$4E^2$	$8C^6$	$16B^6$	I/O	IO, PD
X2D33			$4E^3$	$8C^7$	$16B^7$	I/O	IO, PD
X2D34	$X_2L7^1_{out}$	$1K^0$				I/O	IO, PD
X2D35	$X_2L7^2_{out}$	$1L^0$				I/O	IO, PD
X2D36		$1M^0$		$8D^0$	$16B^8$	I/O	IO, PD
X2D49	$X_2L5^4_{in}$				$32A^0$	I/O	IO, PD
X2D50	$X_2L5^3_{in}$				$32A^1$	I/O	IO, PD
X2D51	$X_2L5^2_{in}$				$32A^2$	I/O	IO, PD
X2D52	$X_2L5^1_{in}$				$32A^3$	I/O	IO, PD

(continued)

5 Example Application Diagram

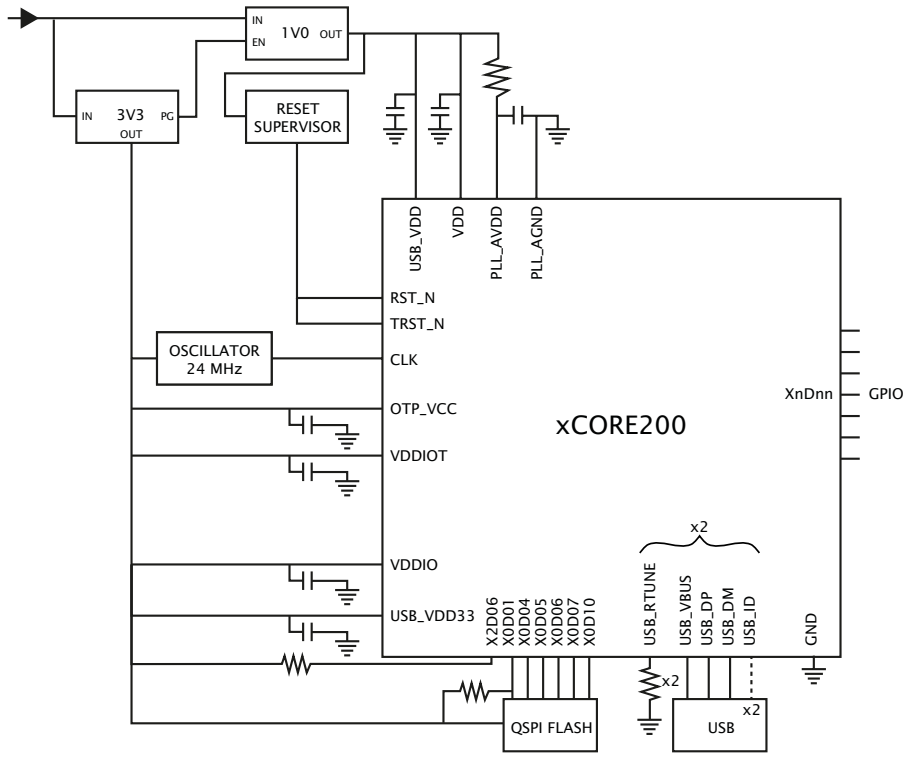


Figure 2: Simplified Reference Schematic

- 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 XU224-1024-FB374 is a powerful device that consists of four 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 6 active logical cores, which issue instructions down a shared five-stage pipeline. Instructions from the active cores are issued round-robin. If up to five logical cores are active, each core is allocated a fifth of the processing cycles. If more than five logical cores are active, each core is allocated at least $1/n$ cycles (for n cores). Figure 3 shows the guaranteed core performance depending on the number of cores used.

Figure 3: Logical core performance

Speed grade	MIPS	Frequency	Minimum MIPS per core (for *n* cores)							
			1	2	3	4	5	6		
20	2000 MIPS	500 MHz	100	100	100	100	100	83		

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). 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 five 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.

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 XU224-1024-FB374, and the software running on it. A combination of 1bit, 4bit, 8bit, 16bit and 32bit

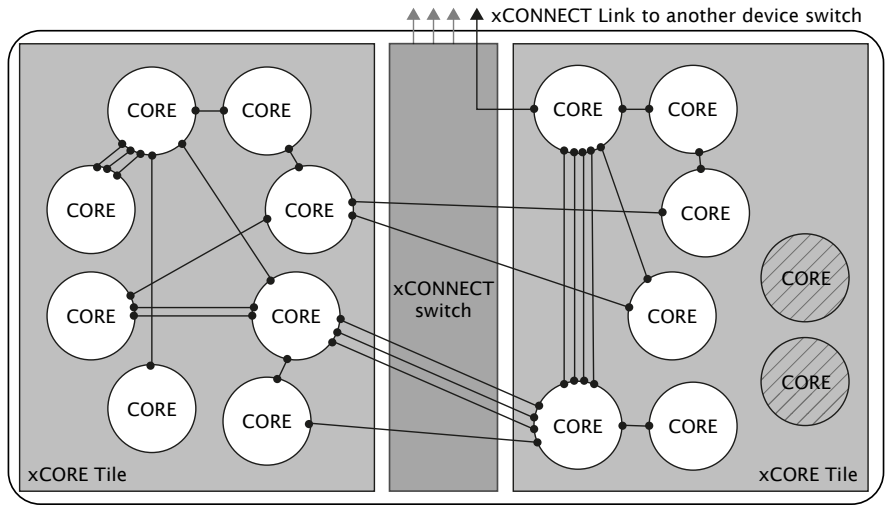


Figure 6: Switch, links and channel ends

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-U 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 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

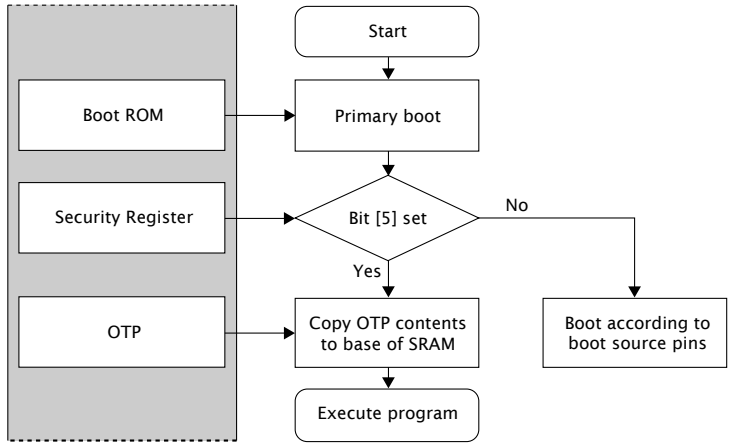


Figure 8: Boot procedure

X0D06	X0D05	X0D04	Tile 0 boot	Tile 1 boot	Enabled links
0	0	0	QSPI master	Channel end 0	None
0	0	1	SPI master	Channel end 0	None
0	1	0	SPI slave	Channel end 0	None
0	1	1	SPI slave	SPI slave	None
1	0	0	Channel end 0	Channel end 0	XL0 (2w)
1	0	1	Channel end 0	Channel end 0	XL4-XL7 (5w)
1	1	0	Channel end 0	Channel end 0	XL1, XL2, XL5, and XL6 (5w)
1	1	1	Channel end 0	Channel end 0	XL0-XL3 (5w)

Figure 9: Boot source pins

8.1 Boot from QSPI master

If set to boot from QSPI master, the processor enables the six pins specified in Figure 10, and drives the SPI clock at 50 MHz (assuming a 400 MHz core clock). A READ command is issued with a 24-bit address 0x000000. The clock polarity and phase are 0 / 0.

Pin	Signal	Description
X0D01	SS	Slave Select
X0D04..X0D07	SPIO	Data
X0D10	SCLK	Clock

Figure 10: QSPI pins

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

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.
6. Free channel-end 0.
7. Jump to the loaded code.

8.5 Boot from OTP

If an xCORE tile is set to use secure boot (see Figure 8), the boot image is read from address 0 of the OTP memory in the tile's security module.

This feature can be used to implement a secure bootloader which loads an encrypted image from external flash, decrypts and CRC checks it with the processor, and discontinues the boot process if the decryption or CRC check fails. XMOS provides a default secure bootloader that can be written to the OTP along with secret decryption keys.

Each tile has its own individual OTP memory, and hence some tiles can be booted from OTP while others are booted from SPI or the channel interface. This enables systems to be partially programmed, dedicating one or more tiles to perform a particular function, leaving the other tiles user-programmable.

8.6 Security register

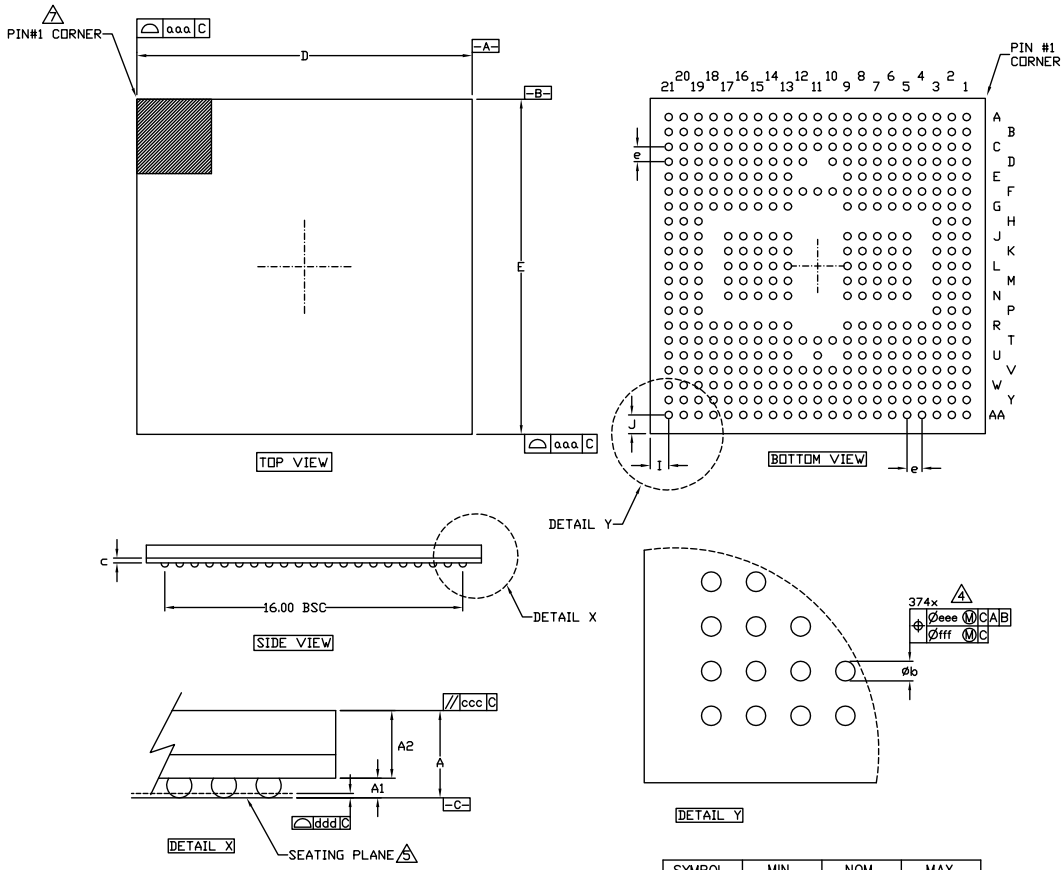
The security register enables security features on the xCORE tile. The features shown in Figure 13 provide a strong level of protection and are sufficient for providing strong IP security.

9 Memory

9.1 OTP

Each xCORE Tile integrates 8 KB one-time programmable (OTP) memory along with a security register that configures system wide security features. The OTP holds

14 Package Information



SYMBOL	MIN.	NOM.	MAX.
A	1.16	1.26	1.36
A1	0.25	0.30	0.35
A2	0.91	0.96	1.01
D	17.90	18.00	18.10
E	17.90	18.00	18.10
I	1.00 REF.		
J	1.00 REF.		
M	21x21<DEPOPULATED>		
aaa			0.15
ccc			0.20
ddd			0.10
eee			0.15
fff			0.08
b	0.35	0.40	0.45
e	0.80 BSC.		
c	0.26 REF.		

NOTE:

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

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 33: Summary

B.1 RAM base address: 0x00

This register contains the base address of the RAM. It is initialized to 0x00040000.

0x00: 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.

0x01: Vector base address

Bits	Perm	Init	Description
31:18	RW		The event and interrupt vectors.
17:0	RO	-	Reserved

B.3 xCORE Tile control: 0x02

Register to control features in the xCORE tile

Bits	Perm	Init	Description
31:26	RO	-	Reserved
25:18	RW	0	RGMII TX data delay value (in PLL output cycle increments)
17:9	RW	0	RGMII TX clock divider value. TX clk rises when counter (clocked by PLL output) reaches this value and falls when counter reaches (value»1). Value programmed into this field should be actual divide value required minus 1
8	RW	0	Enable RGMII interface periph ports
7:6	RO	-	Reserved
5	RW	0	Select the dynamic mode (1) for the clock divider when the clock divider is enabled. In dynamic mode the clock divider is only activated when all active threads are paused. In static mode the clock divider is always enabled.
4	RW	0	Enable the clock divider. This divides the output of the PLL to facilitate one of the low power modes.
3	RO	-	Reserved
2	RW		Select between UTMI (1) and ULPI (0) mode.
1	RW		Enable the ULPI Hardware support module
0	RO	-	Reserved

0x02: xCORE Tile control

B.4 xCORE Tile boot status: 0x03

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

Bits	Perm	Init	Description
31:24	RO	-	Reserved
23:16	RO		Processor number.
15:9	RO	-	Reserved
8	RO		Overwrite BOOT_MODE.
7:6	RO	-	Reserved
5	RO		Indicates if core1 has been powered off
4	RO		Cause the ROM to not poll the OTP for correct read levels
3	RO		Boot ROM boots from RAM
2	RO		Boot ROM boots from JTAG
1:0	RO		The boot PLL mode pin value.

0x03: xCORE Tile boot status

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.

B.18 Debug interrupt data: 0x16

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

0x16: Debug interrupt data

Bits	Perm	Init	Description
31:0	DRW		Value.

B.19 Debug core control: 0x18

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

0x18: Debug core control

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

B.20 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.21 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.

0x9C .. 0x9F: Resources 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.
15:2	RO	-	Reserved
1	DRW	0	When 0 break when condition A is met. When 1 = break when condition B is met.
0	DRW	0	When 1 the instruction breakpoint is enabled.

0x00: Device identification

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

C.2 xCORE Tile description 1: 0x01

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

0x01: xCORE Tile description 1

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

C.3 xCORE Tile description 2: 0x02

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

0x02: xCORE Tile description 2

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

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

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

0x04: Control PSwitch permissions to debug registers

Bits	Perm	Init	Description
31	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.

C.15 PC of logical core 6: 0x46

Value of the PC of logical core 6.

0x46: PC of logical core 6

Bits	Perm	Init	Description
31:0	CRO		Value.

C.16 PC of logical core 7: 0x47

Value of the PC of logical core 7.

0x47: PC of logical core 7

Bits	Perm	Init	Description
31:0	CRO		Value.

C.17 SR of logical core 0: 0x60

Value of the SR of logical core 0

0x60: SR of logical core 0

Bits	Perm	Init	Description
31:0	CRO		Value.

C.18 SR of logical core 1: 0x61

Value of the SR of logical core 1

0x61: SR of logical core 1

Bits	Perm	Init	Description
31:0	CRO		Value.

C.19 SR of logical core 2: 0x62

Value of the SR of logical core 2

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 System clock frequency: 0x51

0x51: System clock frequency

Bits	Perm	Init	Description
31:7	RO	-	Reserved
6:0	RW	25	Oscillator clock frequency in MHz rounded up to the nearest integer value. Only values between 5 and 100 MHz are valid - writes outside this range are ignored and will be NACKed. This field must be set on start up of the device and any time that the input oscillator clock frequency is changed. It must contain the system clock frequency in MHz rounded up to the nearest integer value.

E.5 Link Control and Status: 0x80

0x80: Link Control and Status

Bits	Perm	Init	Description
31:28	RO	-	Reserved
27	RO		Rx buffer overflow or illegal token encoding received.
26	RO	0	This end of the xlink has issued credit to allow the remote end to transmit
25	RO	0	This end of the xlink has credit to allow it to transmit.
24	WO		Clear this end of the xlink's credit and issue a HELLO token.
23	WO		Reset the receiver. The next symbol that is detected will be the first symbol in a token.
22	RO	-	Reserved
21:11	RW	1	Specify min. number of idle system clocks between two continuous symbols witin a transmit token -1.
10:0	RW	1	Specify min. number of idle system clocks between two continuous transmit tokens -1.

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

H Schematics Design Check List

☑ This section is a checklist for use by schematics designers using the XU224-1024-FB374. 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 1400 mA (Section 12 and Figure 21).

☐ 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.

H.5 Boot

- ☐ The device is connected to a QSPI flash for booting, connected to X0D01, X0D04..X0D07, and X0D10 (Section 8). If not, you must boot the device through OTP or JTAG, or set it to boot from SPI and connect a SPI flash.
- ☐ The Flash that you have chosen is supported by **xflash**, or you have created a specification file for it.

H.6 JTAG, XScope, and debugging

- ☐ You have decided as to whether you need an XSYS header or not (Section G)
- ☐ If you have not included an XSYS header, you have devised a method to program the SPI-flash or OTP (Section G).

H.7 GPIO

- ☐ You have not mapped both inputs and outputs to the same multi-bit port.
- ☐ Pins X0D04, X0D05, X0D06, and X0D07 are output only and are, during and after reset, pulled high and low appropriately (Section 8)
- ☐ Pins X2D04, X2D05, X2D06 and X2D07 are output only and during and after reset, X2D06 is pulled high and X2D04, X2D05, and X2D07 are pulled low (Section 8)

H.8 Multi device designs

Skip this section if your design only includes a single XMOS device.

- ☐ One device is connected to a QSPI or SPI flash for booting.
- ☐ Devices that boot from link have, for example, X0D06 pulled high and have link XL0 connected to a device to boot from (Section 8).