Chipsmall Limited consists of a professional team with an average of over 10 year of expertise in the distribution of electronic components. Based in Hongkong, we have already established firm and mutual-benefit business relationships with customers from,Europe,America and south Asia,supplying obsolete and hard-to-find components to meet their specific needs.

With the principle of “Quality Parts,Customers Priority,Honest Operation,and Considerate Service”,our business mainly focus on the distribution of electronic components. Line cards we deal with include Microchip,ALPS,ROHM,Xilinx,Pulse,ON,Everlight and Freescale. Main products comprise IC,Modules,Potentiometer,IC Socket,Relay,Connector.Our parts cover such applications as commercial,industrial, and automotives areas.

We are looking forward to setting up business relationship with you and hope to provide you with the best service and solution. Let us make a better world for our industry!

Contact us

Tel: +86-755-8981 8866 Fax: +86-755-8427 6832

Email & Skype: info@chipsmall.com Web: www.chipsmall.com

Address: A1208, Overseas Decoration Building, #122 Zhenhua RD., Futian, Shenzhen, China

XUF232-1024-FB374 Datasheet

2016/01/05 Document Number: X009394,

Table of Contents

1 xCORE Multicore Microcontrollers . . . 2
2 XUF232-1024-FB374 Features . . . 5
3 Pin Configuration . . . 6
4 Signal Description . . . 7
5 Example Application Diagram . . . 13
6 Product Overview . . . 14
7 PLL . . . 17
8 Boot Procedure . . . 18
9 Memory . . . 19
10 USB PHY . . . 21
11 JTAG . . . 22
12 Board Integration . . . 23
13 DC and Switching Characteristics . . . 28
14 Package Information . . . 31
15 Ordering Information . . . 32
Appendices . . . 33
A Configuration of the XUF232-1024-FB374 . . . 33
B Processor Status Configuration . . . 36
C Tile Configuration . . . 47
D Node Configuration . . . 55
E USB Node Configuration . . . 63
F USB PHY Configuration . . . 65
G Device Errata . . . 72
H JTAG, xSCOPE and Debugging . . . 72
I Schematics Design Check List . . . 74
J PCB Layout Design Check List . . . 76
K Associated Design Documentation . . . 77
L Related Documentation . . . 77
M Revision History . . . 78

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/.

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.

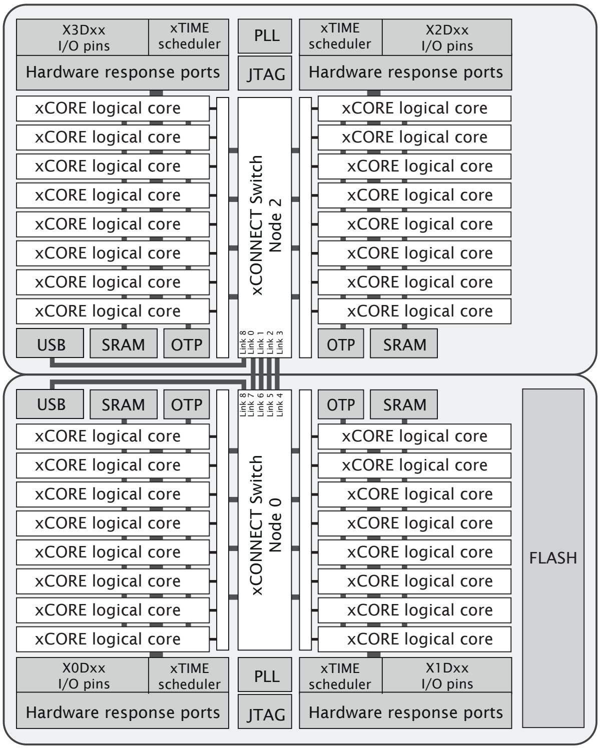


Figure 1: XUF232-1024-FB374 block diagram

Key features of the XUF232-1024-FB374 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
- **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
- **Flash** The device has a built-in 4MBflash. Section 8
- **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.

2 XUF232-1024-FB374 Features

- **Multicore Microcontroller with Advanced Multi-Core RISC Architecture**
  - 32 real-time logical cores on 4 xCORE tiles
  - Cores share up to 2000 MIPS
    — Up to 4000 MIPS in dual issue mode
  - Each logical core has:
    — Guaranteed throughput of between 1/5 and 1/8 of tile MIPS
    — 16x32bit dedicated registers
  - 167 high-density 16/32-bit instructions
    — All have single clock-cycle execution (except for divide)
    — 32x32→64-bit MAC instructions for DSP, arithmetic and user-definable cryptographic functions
- **Dual USB PHY, fully compliant with USB 2.0 specification**
- **Programmable I/O**
  - 176 general-purpose I/O pins, configurable as input or output
    — Up to 56 x 1bit port, 22 x 4bit port, 13 x 8bit port, 6 x 16bit port, 4 x 32bit port
    — 8 xCONNECT links
  - Port sampling rates of up to 60 MHz with respect to an external clock
  - 64 channel ends for communication with other cores, on or off-chip
- **Memory**
  - 1024KB internal single-cycle SRAM (max 256KB per tile) for code and data storage
  - 32KB internal OTP (max 8KB per tile) for application boot code
  - 4MB internal flash for application code and overlays
- **Hardware resources**
  - 24 clock blocks (6 per tile)
  - 40 timers (10 per tile)
  - 16 locks (4 per tile)
- **JTAG Module for On-Chip Debug**
- **Security Features**
  - Programming lock disables debug and prevents read-back of memory contents
  - AES bootloader ensures secrecy of IP held on external flash memory
- **Ambient Temperature Range**
  - Commercial qualification: 0 °C to 70 °C
  - Industrial qualification: -40 °C to 85 °C
- **Speed Grade**
  - 40: 2000 MIPS
- **Power Consumption**
  - 1140 mA (typical)
- **374-pin FBGA package 0.8 mm pitch**

3 Pin Configuration

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
A	GND	VDDIO	X1D11	X1D32	X1D26	VDDIOT	X1D41	X0D31	X0D29	TDI	VDDIO	CLK	TDO	X3D32	X3D30	VDDIOT_2	X2D31	X2D29	X2D32	VDDIO	GND
B	X0D37	X0D36	X1D10	X1D33	X1D27	X1D42	X1D40	X0D30	X0D28	X2D36	GND	RST_N	TCK	X3D33	X3D31	X3D27	X2D30	X2D28	X2D27	X2D26	X2D35
C	X0D39	X0D38	VDD	X1D30	X1D28	X1D43	GND	X0D33	X0D32	MODE1	OTP_VCC	TRST_N	X3D10	X3D29	GND	X3D43	X3D41	X2D33	VDD	X2D25	X2D34
D	X0D41	X0D40	X1D34	X1D31	X1D29	GND	VDDIO	NC	DEBUG_N	MODE0		TMS	X3D11	X3D28	X3D26	X3D42	X3D40	X2D70	X3D00	X3D01	X2D24
E	X0D43	X0D42	X1D35	VDD	VDD	GND	VDDIO	VDD	VDD				VDD	VDD	VDDIO	GND	VDD	VDD	X2D69	X3D08	X3D09
F	X1D36	VDDIO	GND	VDD	VDD	VDD	VDD	VDD	VDD	PLL_AGND	PLL_AVDD	VDD	VDD	VDD	VDD	VDD	VDD	VDD	GND	VDDIO	X2D68
G	X1D49	X1D50	X1D51	NC	NC	NC	NC	NC	NC				NC	NC	NC	NC	NC	NC	X2D67	X2D66	X2D65
H	X1D53	X1D52	VDD																VDD	X2D63	X2D64
J	X1D54	X1D55	VDD		GND	GND	GND	GND	GND				GND	GND	GND	GND	GND		VDD	X2D62	X2D61
K	X1D58	X1D57	X1D56		GND	GND	GND	GND	GND				GND	GND	GND	GND	GND		X2D56	X2D57	X2D58
L	VDDIO	GND	X1D61		GND	GND	GND	GND	GND				GND	GND	GND	GND	GND		X2D55	GND	VDDIO
M	X1D64	X1D63	X1D62		GND	GND	GND	GND	GND				GND	GND	GND	GND	GND		X2D54	X2D53	X2D52
N	X1D65	X1D66	VDD		GND	GND	GND	GND	GND				GND	GND	GND	GND	GND		VDD	X2D50	X2D51
P	X1D68	X1D67	VDD																VDD	X3D06	X3D07
R	X1D69	X1D70	X1D37	NC	NC	NC	NC	NC	NC				NC	NC	NC	NC	NC	NC	X2D49	X3D04	X3D05
T	X1D38	VDDIO	GND	VDD	VDD	VDD	USB_VDD	VDD	VDD	VDD	GND	VDD	VDD	VDD	USB_2_VDD	VDD	VDD	VDD	GND	VDDIO	X3D03
U	X1D17	X1D16	X1D39	VDD	VDD	GND	VDDIO	NC	VDD		VDDIO		VDD	VDD	VDDIO	GND	VDD	VDD	NC	X2D19	X3D02
V	X1D19	X1D18	X0D01	X0D02	X0D08	X0D11	USB_ID	X1D14	X1D25	X0D21	NC	X3D23	X2D05	X2D07	USB_2_ID	NC	X3D15	X3D21	X2D12	X2D17	X2D18
W	X0D10	X1D22	USB_VDD33	X0D03	X0D09	USB_RTUNE	GND	X1D15	X0D14	X0D12	X0D23	X2D00	X2D04	X2D06	GND	USB_2_RTUNE	X3D14	X3D20	USB_2_VDD33	X2D23	X2D16
Y	X1D23	X0D00	X0D04	X0D06	X1D12	USB_VBUS	X1D24	X1D20	X0D15	X0D13	GND	X2D11	X2D02	X2D08	X3D13	USB_2_VBUS	X2D14	X2D20	X3D24	X2D13	X2D22
AA	GND	VDDIO	X0D05	X0D07	X1D13	USB_DM	USB_DP	X1D21	X0D20	X0D22	VDDIO	X3D12	X2D03	X2D09	USB_2_DM	USB_2_DP	X2D15	X2D21	X3D25	VDDIO	GND

4 Signal Description

This section lists the signals and I/O pins available on the XUF232-1024-FB374. The device provides a combination of 1bit, 4bit, 8bit and 16bit ports, as well as wider ports that are fully or partially (gray) bonded out. All pins of a port provide either output or input, but signals in different directions cannot be mapped onto the same port.

Pins may have one or more of the following properties:

- PD/PU: The IO pin a weak pull-down or pull-up resistor. On GPIO pins this resistor can be enabled.
- ST: The IO pin has a Schmitt Trigger on its input.
- IOT: The IO pin is powered from VDDIOT (X1) or VDDIOT_2 (X3), not VDDIO
- IO: the pin is powered from VDDIO

Power pins (12)			
Signal	**Function**	**Type**	**Properties**
GND	Digital ground	GND	
OTP_VCC	OTP power supply	PWR	
PLL_AGND	Analog ground for PLL	PWR	
PLL_AVDD	Analog PLL power	PWR	
USB_2_VDD	Digital tile power	PWR	
USB_2_VDD33	USB Analog power	PWR	
USB_VDD	Digital tile power	PWR	
USB_VDD33	USB Analog power	PWR	
VDD	Digital tile power	PWR	
VDDIO	Digital I/O power	PWR	
VDDIOT	Digital I/O power (top)	PWR	
VDDIOT_2	Digital I/O power (top, X3)	PWR	

JTAG pins (6)			
Signal	**Function**	**Type**	**Properties**
RST_N	Global reset input	Input	IO, PU, ST
TCK	Test clock	Input	IO, PD, ST
TDI	Test data input	Input	IO, PU
TDO	Test data output	Output	IO, PD
TMS	Test mode select	Input	IO, PU
TRST_N	Test reset input	Input	IO, PU, ST

I/O pins (176)								
Signal	**Function**						**Type**	**Properties**
X0D00		$1A^0$					I/O	IO, PD
X0D01	$X_0L3^2_{out}$	$1B^0$					I/O—	IO, PD
X0D02			$4A^0$	$8A^0$	$16A^0$	$32A^{20}$	I/O	IO, PD
X0D03			$4A^1$	$8A^1$	$16A^1$	$32A^{21}$	I/O	IO, PD
X0D04			$4B^0$	$8A^2$	$16A^2$	$32A^{22}$	I/O—	IO, PD
X0D05			$4B^1$	$8A^3$	$16A^3$	$32A^{23}$	I/O—	IO, PD
X0D06			$4B^2$	$8A^4$	$16A^4$	$32A^{24}$	I/O—	IO, PD
X0D07			$4B^3$	$8A^5$	$16A^5$	$32A^{25}$	I/O—	IO, PD
X0D08			$4A^2$	$8A^6$	$16A^6$	$32A^{26}$	I/O	IO, PD
X0D09			$4A^3$	$8A^7$	$16A^7$	$32A^{27}$	I/O	IO, PD
X0D10	$X_0L3^3_{out}$	$1C^0$					I/O—	IO, PD
X0D11		$1D^0$					I/O	IO, PD
X0D12		$1E^0$					I/O	IO, PD
X0D13		$1F^0$					I/O	IO, PD
X0D14			$4C^0$	$8B^0$	$16A^8$	$32A^{28}$	I/O	IO, PD
X0D15			$4C^1$	$8B^1$	$16A^9$	$32A^{29}$	I/O	IO, PD
X0D20			$4C^2$	$8B^6$	$16A^{14}$	$32A^{30}$	I/O	IO, PD
X0D21			$4C^3$	$8B^7$	$16A^{15}$	$32A^{31}$	I/O	IO, PD
X0D22		$1G^0$					I/O	IO, PD
X0D23		$1H^0$					I/O	IO, PD
X0D28			$4F^0$	$8C^2$	$16B^2$		I/O	IO, PD
X0D29			$4F^1$	$8C^3$	$16B^3$		I/O	IO, PD
X0D30			$4F^2$	$8C^4$	$16B^4$		I/O	IO, PD
X0D31			$4F^3$	$8C^5$	$16B^5$		I/O	IO, PD
X0D32			$4E^2$	$8C^6$	$16B^6$		I/O	IO, PD
X0D33			$4E^3$	$8C^7$	$16B^7$		I/O	IO, PD
X0D36		$1M^0$		$8D^0$	$16B^8$		I/O	IO, PD
X0D37	$X_0L0^4_{in}$	$1N^0$		$8D^1$	$16B^9$		I/O	IO, PD
X0D38	$X_0L0^3_{in}$	$1O^0$		$8D^2$	$16B^{10}$		I/O	IO, PD
X0D39	$X_0L0^2_{in}$	$1P^0$		$8D^3$	$16B^{11}$		I/O	IO, PD
X0D40	$X_0L0^1_{in}$			$8D^4$	$16B^{12}$		I/O	IO, PD
X0D41	$X_0L0^0_{in}$			$8D^5$	$16B^{13}$		I/O	IO, PD
X0D42	$X_0L0^0_{out}$			$8D^6$	$16B^{14}$		I/O	IO, PD
X0D43	$X_0L0^1_{out}$			$8D^7$	$16B^{15}$		I/O	IO, PD
X1D10		$1C^0$					I/O	IOT, PD
X1D11		$1D^0$					I/O	IOT, PD
X1D12		$1E^0$					I/O	IO, PD
X1D13		$1F^0$					I/O	IO, PD
X1D14			$4C^0$	$8B^0$	$16A^8$	$32A^{28}$	I/O	IO, PD
X1D15			$4C^1$	$8B^1$	$16A^9$	$32A^{29}$	I/O	IO, PD
X1D16	$X_0L3^1_{in}$		$4D^0$	$8B^2$	$16A^{10}$		I/O	IO, PD

(continued)

Signal	Function						Type	Properties
X1D17	$X_0L3^{0}_{in}$		$4D^1$	$8B^3$	$16A^{11}$		I/O	IO, PD
X1D18	$X_0L3^{0}_{out}$		$4D^2$	$8B^4$	$16A^{12}$		I/O	IO, PD
X1D19	$X_0L3^{1}_{out}$		$4D^3$	$8B^5$	$16A^{13}$		I/O	IO, PD
X1D20			$4C^2$	$8B^6$	$16A^{14}$	$32A^{30}$	I/O	IO, PD
X1D21			$4C^3$	$8B^7$	$16A^{15}$	$32A^{31}$	I/O	IO, PD
X1D22	$X_0L3^{4}_{out}$	$1G^0$					I/O	IO, PD
X1D23		$1H^0$					I/O	IO, PD
X1D24		$1I^0$					I/O	IO, PD
X1D25		$1J^0$					I/O	IO, PD
X1D26			$4E^0$	$8C^0$	$16B^0$		I/O	IOT, PD
X1D27			$4E^1$	$8C^1$	$16B^1$		I/O	IOT, PD
X1D28			$4F^0$	$8C^2$	$16B^2$		I/O	IOT, PD
X1D29			$4F^1$	$8C^3$	$16B^3$		I/O	IOT, PD
X1D30			$4F^2$	$8C^4$	$16B^4$		I/O	IOT, PD
X1D31			$4F^3$	$8C^5$	$16B^5$		I/O	IOT, PD
X1D32			$4E^2$	$8C^6$	$16B^6$		I/O	IOT, PD
X1D33			$4E^3$	$8C^7$	$16B^7$		I/O	IOT, PD
X1D34	$X_0L0^{2}_{out}$	$1K^0$					I/O	IO, PD
X1D35	$X_0L0^{3}_{out}$	$1L^0$					I/O	IO, PD
X1D36	$X_0L0^{4}_{out}$	$1M^0$		$8D^0$	$16B^8$		I/O	IO, PD
X1D37	$X_0L3^{4}_{in}$	$1N^0$		$8D^1$	$16B^9$		I/O	IO, PD
X1D38	$X_0L3^{3}_{in}$	$1O^0$		$8D^2$	$16B^{10}$		I/O	IO, PD
X1D39	$X_0L3^{2}_{in}$	$1P^0$		$8D^3$	$16B^{11}$		I/O	IO, PD
X1D40				$8D^4$	$16B^{12}$		I/O	IOT, PD
X1D41				$8D^5$	$16B^{13}$		I/O	IOT, PD
X1D42				$8D^6$	$16B^{14}$		I/O	IOT, PD
X1D43				$8D^7$	$16B^{15}$		I/O	IOT, PD
X1D49	$X_0L1^{4}_{in}$					$32A^0$	I/O	IO, PD
X1D50	$X_0L1^{3}_{in}$					$32A^1$	I/O	IO, PD
X1D51	$X_0L1^{2}_{in}$					$32A^2$	I/O	IO, PD
X1D52	$X_0L1^{1}_{in}$					$32A^3$	I/O	IO, PD
X1D53	$X_0L1^{0}_{in}$					$32A^4$	I/O	IO, PD
X1D54	$X_0L1^{0}_{out}$					$32A^5$	I/O	IO, PD
X1D55	$X_0L1^{1}_{out}$					$32A^6$	I/O	IO, PD
X1D56	$X_0L1^{2}_{out}$					$32A^7$	I/O	IO, PD
X1D57	$X_0L1^{3}_{out}$					$32A^8$	I/O	IO, PD
X1D58	$X_0L1^{4}_{out}$					$32A^9$	I/O	IO, PD
X1D61	$X_0L2^{4}_{in}$					$32A^{10}$	I/O	IO, PD
X1D62	$X_0L2^{3}_{in}$					$32A^{11}$	I/O	IO, PD
X1D63	$X_0L2^{2}_{in}$					$32A^{12}$	I/O	IO, PD
X1D64	$X_0L2^{1}_{in}$					$32A^{13}$	I/O	IO, PD
X1D65	$X_0L2^{0}_{in}$					$32A^{14}$	I/O	IO, PD
X1D66	$X_0L2^{0}_{out}$					$32A^{15}$	I/O	IO, PD

(continued)

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)

Signal	Function						Type	Properties
X2D53	$X_2L5^0_{in}$					$32A^4$	I/O	IO, PD
X2D54	$X_2L5^0_{out}$					$32A^5$	I/O	IO, PD
X2D55	$X_2L5^1_{out}$					$32A^6$	I/O	IO, PD
X2D56	$X_2L5^2_{out}$					$32A^7$	I/O	IO, PD
X2D57	$X_2L5^3_{out}$					$32A^8$	I/O	IO, PD
X2D58	$X_2L5^4_{out}$					$32A^9$	I/O	IO, PD
X2D61	$X_2L6^4_{in}$					$32A^{10}$	I/O	IO, PD
X2D62	$X_2L6^3_{in}$					$32A^{11}$	I/O	IO, PD
X2D63	$X_2L6^2_{in}$					$32A^{12}$	I/O	IO, PD
X2D64	$X_2L6^1_{in}$					$32A^{13}$	I/O	IO, PD
X2D65	$X_2L6^0_{in}$					$32A^{14}$	I/O	IO, PD
X2D66	$X_2L6^0_{out}$					$32A^{15}$	I/O	IO, PD
X2D67	$X_2L6^1_{out}$					$32A^{16}$	I/O	IO, PD
X2D68	$X_2L6^2_{out}$					$32A^{17}$	I/O	IO, PD
X2D69	$X_2L6^3_{out}$					$32A^{18}$	I/O	IO, PD
X2D70	$X_2L6^4_{out}$					$32A^{19}$	I/O	IO, PD
X3D00	$X_2L7^2_{in}$	$1A^0$					I/O	IO, PD
X3D01	$X_2L7^1_{in}$	$1B^0$					I/O	IO, PD
X3D02	$X_2L4^0_{in}$		$4A^0$	$8A^0$	$16A^0$	$32A^{20}$	I/O	IO, PD
X3D03	$X_2L4^0_{out}$		$4A^1$	$8A^1$	$16A^1$	$32A^{21}$	I/O	IO, PD
X3D04	$X_2L4^1_{out}$		$4B^0$	$8A^2$	$16A^2$	$32A^{22}$	I/O	IO, PD
X3D05	$X_2L4^2_{out}$		$4B^1$	$8A^3$	$16A^3$	$32A^{23}$	I/O	IO, PD
X3D06	$X_2L4^3_{out}$		$4B^2$	$8A^4$	$16A^4$	$32A^{24}$	I/O	IO, PD
X3D07	$X_2L4^4_{out}$		$4B^3$	$8A^5$	$16A^5$	$32A^{25}$	I/O	IO, PD
X3D08	$X_2L7^4_{in}$		$4A^2$	$8A^6$	$16A^6$	$32A^{26}$	I/O	IO, PD
X3D09	$X_2L7^3_{in}$		$4A^3$	$8A^7$	$16A^7$	$32A^{27}$	I/O	IO, PD
X3D10		$1C^0$					I/O	IOT, PD
X3D11		$1D^0$					I/O	IOT, PD
X3D12		$1E^0$					I/O	IO, PD
X3D13		$1F^0$					I/O	IO, PD
X3D14			$4C^0$	$8B^0$	$16A^8$	$32A^{28}$	I/O	IO, PD
X3D15			$4C^1$	$8B^1$	$16A^9$	$32A^{29}$	I/O	IO, PD
X3D20			$4C^2$	$8B^6$	$16A^{14}$	$32A^{30}$	I/O	IO, PD
X3D21			$4C^3$	$8B^7$	$16A^{15}$	$32A^{31}$	I/O	IO, PD
X3D23		$1H^0$					I/O	IO, PD
X3D24		$1I^0$					I/O	IO, PD
X3D25		$1J^0$					I/O	IO, PD
X3D26			$4E^0$	$8C^0$	$16B^0$		I/O	IOT, PD
X3D27			$4E^1$	$8C^1$	$16B^1$		I/O	IOT, PD
X3D28			$4F^0$	$8C^2$	$16B^2$		I/O	IOT, PD
X3D29			$4F^1$	$8C^3$	$16B^3$		I/O	IOT, PD
X3D30			$4F^2$	$8C^4$	$16B^4$		I/O	IOT, PD
X3D31			$4F^3$	$8C^5$	$16B^5$		I/O	IOT, PD

(continued)

Signal	Function	Type	Properties
X3D32	4E[2] 8C[6] 16B[6]	I/O	IOT, PD
X3D33	4E[3] 8C[7] 16B[7]	I/O	IOT, PD
X3D40	8D[4] 16B[12]	I/O	IOT, PD
X3D41	8D[5] 16B[13]	I/O	IOT, PD
X3D42	8D[6] 16B[14]	I/O	IOT, PD
X3D43	8D[7] 16B[15]	I/O	IOT, PD

System pins (4)			
Signal	**Function**	**Type**	**Properties**
CLK	PLL reference clock	Input	IO, PD, ST
DEBUG_N	Multi-chip debug	I/O	IO, PU
MODE0	Boot mode select	Input	PU
MODE1	Boot mode select	Input	PU

usb pins (10)			
Signal	**Function**	**Type**	**Properties**
USB_2_DM	USB Serial Data Inverted, node 2	I/O	
USB_2_DP	USB Serial Data, node 2	I/O	
USB_2_ID	USB Device ID (OTG) - Reserved, node 2	I/O	
USB_2_RTUNE	USB resistor, node 2	I/O	
USB_2_VBUS	USB Power Detect Pin, node 2	I/O	
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

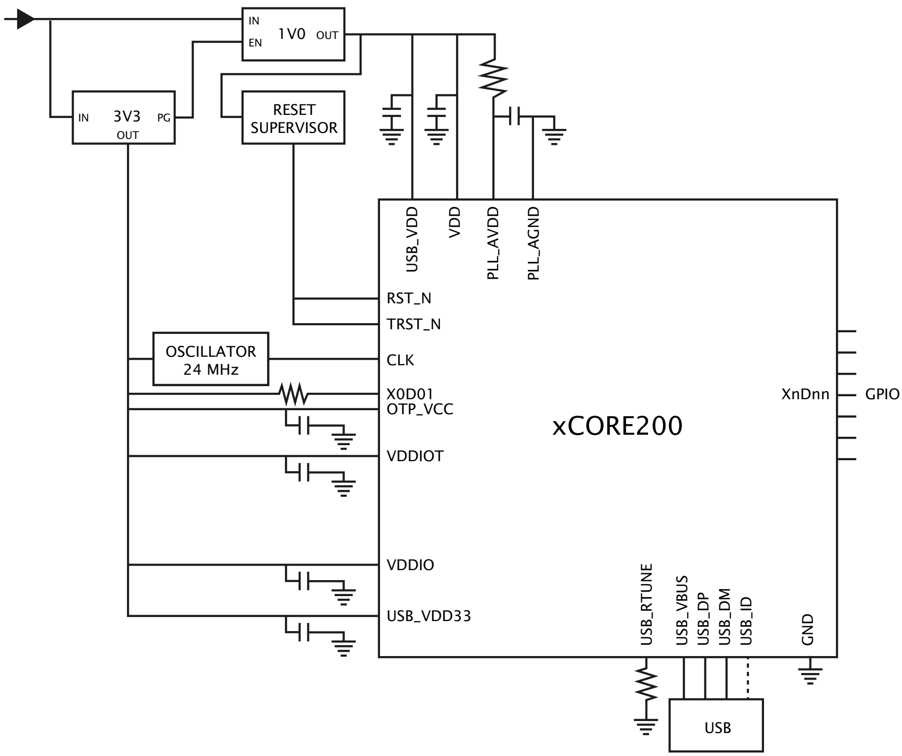


Figure 2: Simplified Reference Schematic

6 Product Overview

The XUF232-1024-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 8 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	7	8
20	2000 MIPS	500 MHz	100	100	100	100	100	83	71	63

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 XUF232-1024-FB374, 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.

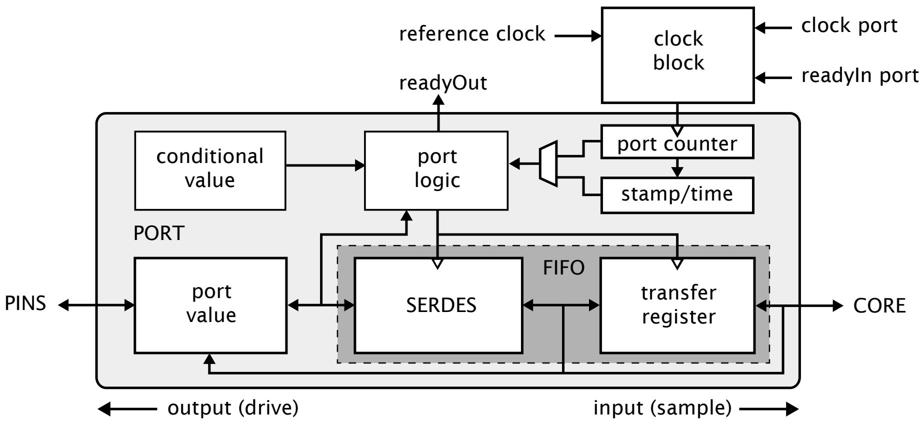


Figure 4: Port block diagram

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.

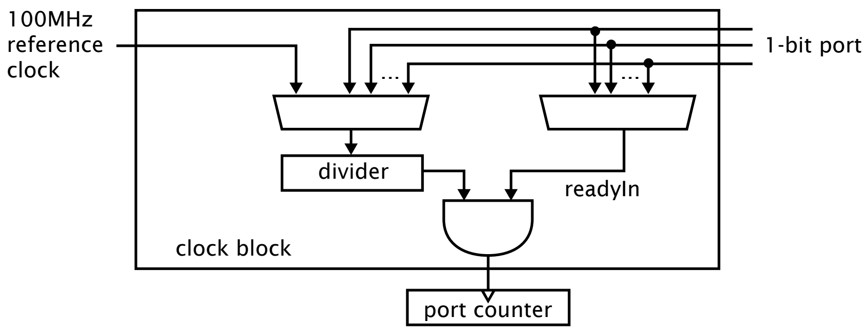


Figure 5: Clock block diagram

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

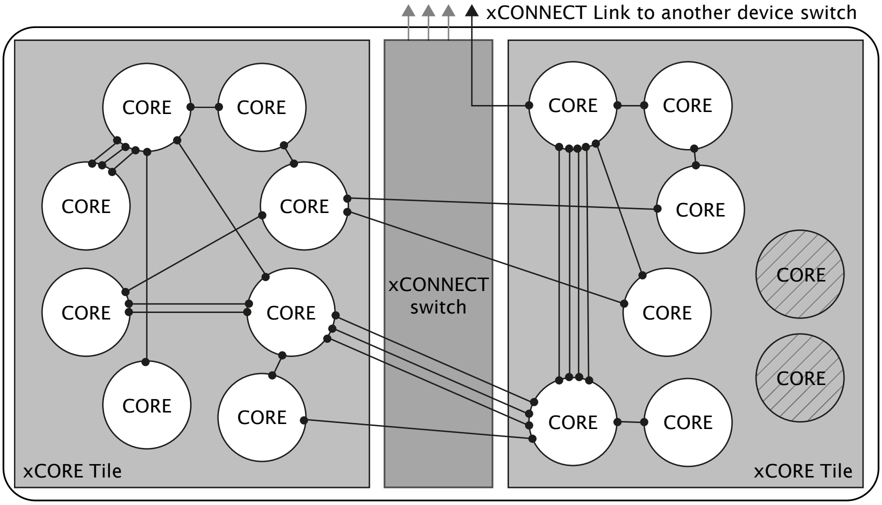


Figure 6: Switch, links and channel ends

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-UF 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 7 also lists the values of OD, F and R, which are the registers that define the ratio of the tile frequency to the oscillator frequency:

$$F_{core} = F_{osc} \times \frac{F+1}{2} \times \frac{1}{R+1} \times \frac{1}{OD+1}$$

OD, F and R must be chosen so that $0 \le R \le 63$, $0 \le F \le 4095$, $0 \le OD \le 7$, and $260MHz \le F_{osc} \times \frac{F+1}{2} \times \frac{1}{R+1} \le 1.3GHz$. The OD, F, and R values can be modified by writing to the digital node PLL configuration register.

The MODE pins must be held at a static value during and after deassertion of the system reset. If the USB PHY is used, then either a 24 MHz or 12 MHz oscillator must be used.

If a different tile frequency is required (eg, 500 MHz), then the PLL must be reprogrammed after boot to provide the required tile frequency. The XMOS tools perform this operation by default. Further details on configuring the clock can be found in the xCORE-200 Clock Frequency Control document.

8 Boot Procedure

The device is kept in reset by driving RST_N low. When in reset, all GPIO pins have a pull-down enabled. The processor must be held in reset until VDDIOL is in spec for at least 1 ms. When the device is taken out of reset by releasing RST_N the processor starts its internal reset process. After 15-150 μs (depending on the input clock) the processor boots.

Pin X2D06 must be pulled high with an external pull-up whilst the chip comes out of reset, to ensure that tile 2 will boot from link. X2D04, X2D05, and X2D07 should be kept low whilst the chip comes out of reset.

The device boots from a QSPI flash that is embedded in the device. The QSPI flash is connected to the ports on Tile 0 as shown in Figure 8. An external 1K resistor must connect X0D01 to VDDIOL. X0D10 should ideally not be connected. If X0D10 is connected, then a 150 ohm series resistor close to the device is recommended. X0D04..X0D07 should be not connected.

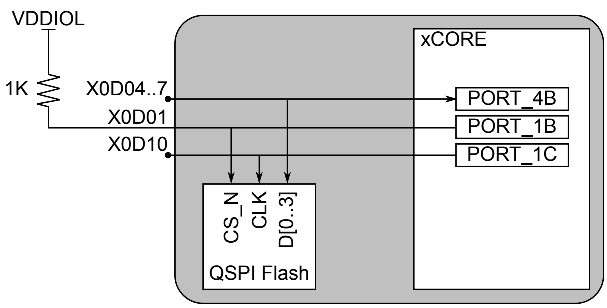


Figure 8: QSPI port connectivity

The xCORE Tile boot procedure is illustrated in Figure 9. If bit 5 of the security register (*see* §9.1) is set, the device boots from OTP. Otherwise, the device boots from the internal flash.

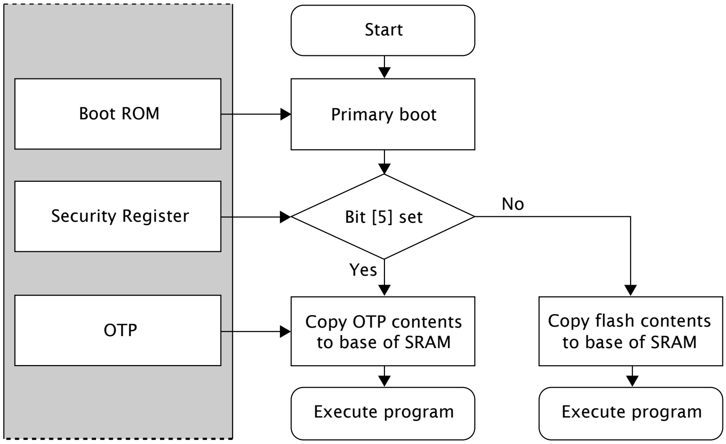


Figure 9: Boot procedure

The boot image has the following format:

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

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

8.1 Security register

The security register enables security features on the xCORE tile. The features shown in Figure 10 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 data in four sectors each containing 512 rows of 32 bits which can be used to

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 10: Security register features

implement secure bootloaders and store encryption keys. Data for the security register is loaded from the OTP on power up. All additional data in OTP is copied from the OTP to SRAM and executed first on the processor.

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), and data is communicated through ports on the digital node. A library, libxud_s.a, is provided to implement USB device functionality.

This device has two USB PHYs. One PHY is part of Node 0 and can be connected to either Tile 0 or Tile 1; it uses pins USB_DM, USB_DP, USB_VUBS, USB_ID, and USB_RTUNE. The other PHY is part of Node 2 and can be connected to either Tile 2 or Tile 3; it uses pins USB_2_DM, USB_2_DP, USB_2_VUBS, USB_2_ID, and USB_2_RTUNE. Below we present the configuration of the USB PHY connected to Node 0 (Tiles 0 and 1) an identical configuration is available for Node 2.

The USB PHY is connected to the ports on Tile 0 and Tile 1 as shown in Figure 11. 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.

An external resistor of 43.2 ohm (1% tolerance) should connect USB_RTUNE to ground, as close as possible to the device.

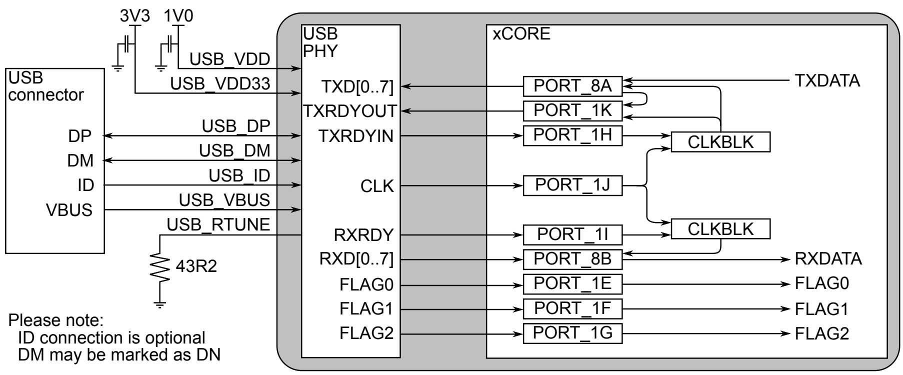


Figure 11: USB port functions

Figure 11 shows how 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.

10.1 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.

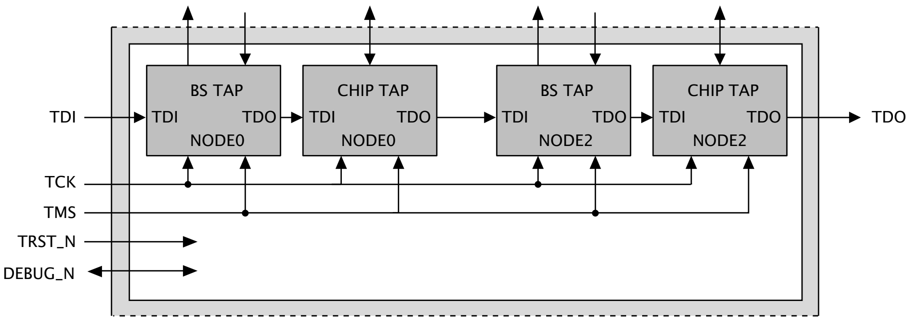


Figure 12: JTAG chain structure

The JTAG chain structure is illustrated in Figure 12. Directly after reset, two TAP controllers are present in the JTAG chain for each xCORE Tile: the boundary scan TAP and the chip TAP. The boundary scan TAP is a standard 1149.1 compliant TAP that can be used for boundary scan of the I/O pins. The chip TAP provides access into the xCORE Tile, switch and OTP 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.

The JTAG device identification register can be read by using the IDCODE instruction. Its contents are specified in Figure 13.

Figure 13: IDCODE return value

Bit31																Device Identification Register															Bit0
Version				Part Number																Manufacturer Identity											1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1	1	0	0	0	1	1	0	0	1	1
0				0				0				0				6				6				3				3			

The JTAG usercode register can be read by using the USERCODE instruction. Its contents are specified in Figure 14. The OTP User ID field is read from bits [22:31] of the security register on xCORE Tile 0, *see* §9.1 (all zero on unprogrammed devices).

Figure 14: USERCODE return value

<table>
<tr><td colspan="4">Bit31</td><td colspan="24">Usercode Register</td><td colspan="4">Bit0</td></tr>
<tr><td colspan="10">OTP User ID</td><td colspan="4">Unused</td><td colspan="18">Silicon Revision</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td colspan="4">0</td><td colspan="4">0</td><td colspan="4">0</td><td colspan="4">2</td><td colspan="4">8</td><td colspan="4">0</td><td colspan="4">0</td><td colspan="4">0</td></tr>
</table>

12 Board Integration

The device has the following power supply pins:

- VDD pins for the xCORE Tile, including USB_VDD and USB_2_VDD pins that power the USB PHY
- VDDIO pins for the I/O lines
- PLL_AVDD pins for the PLL
- OTP_VCC pins for the OTP
- USB_VDD33 and USB_2_VDD33 pins for the analogue supply to the USB-PHY

Several pins of each type are provided to minimize the effect of inductance within the package, all of which must be connected. The power supplies must be brought up monotonically and input voltages must not exceed specification at any time.

The VDD supply must ramp from 0 V to its final value within 10 ms to ensure correct startup.

The VDDIO and OTP_VCC supply must ramp to its final value before VDD reaches 0.4 V.

The PLLVDD supply should be separated from the other noisier supplies on the board. The PLL requires a very clean power supply, and a low pass filter (for example, a 4.7 Ω resistor and multi-layer ceramic capacitor) is recommended on this pin.

The following ground pins are provided:

- PLL_AGND for PLL_AVDD
- GND for all other supplies

All ground pins must be connected directly to the board ground.

The VDD and VDDIO supplies should be decoupled close to the chip by several 100 nF low inductance multi-layer ceramic capacitors between the supplies and GND (for example, 4x100nF 0402 low inductance MLCCs per supply rail). The ground side of the decoupling capacitors should have as short a path back to the

GND pins as possible. A bulk decoupling capacitor of at least 10 uF should be placed on each of these supplies.

RST_N is an active-low asynchronous-assertion global reset signal. Following a reset, the PLL re-establishes lock after which the device boots up according to the boot mode (*see* §8). RST_N and must be asserted low during and after power up for 100 ns.

12.1 USB connections

USB_VBUS should be connected to the VBUS pin of the USB connector. A 2.2 uF capacitor to ground is required on the VBUS pin. A ferrite bead may be used to reduce HF noise.

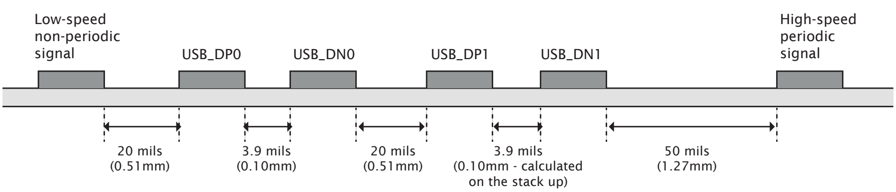
For self-powered systems, a bleeder resistor may be required to stop VBUS from floating when no USB cable is attached.

USB_DP and USB_DN should be connected to the USB connector. USB_ID does not need to be connected.

12.2 USB signal routing and placement

The USB_DP and USB_DN lines are the positive and negative data polarities of a high speed USB signal respectively. Their high-speed differential nature implies that they must be coupled and properly isolated. The board design must ensure that the board traces for USB_DP and USB_DN are tightly matched. In addition, according to the USB 2.0 specification, the USB_DP and USB_DN differential impedance must be 90 Ω.

Figure 15: USB trace separation showing a low speed signal, two differential pairs and a high-speed clock



12.2.1 General routing and placement guidelines

The following guidelines will help to avoid signal quality and EMI problems on high speed USB designs. They relate to a four-layer (Signal, GND, Power, Signal) PCB.

For best results, most of the routing should be done on the top layer (assuming the USB connector and XS2-UF32A-1024-FB374 are on the top layer) closest to GND. Reference planes should be below the transmission lines in order to maintain control of the trace impedance.