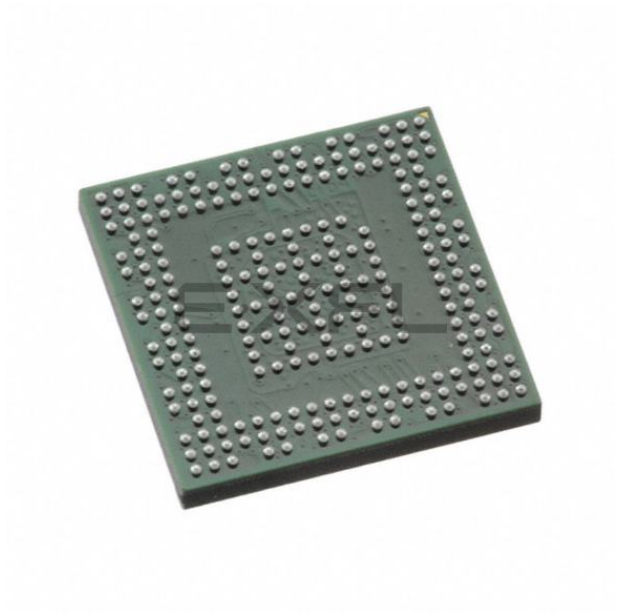


Welcome to E-XFL.COM

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	Obsolete
Core Processor	XCore
Core Size	32-Bit 16-Core
Speed	2000MIPS
Connectivity	RGMII, USB
Peripherals	-
Number of I/O	73
Program Memory Size	2MB (2M x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	512K x 8
Voltage - Supply (Vcc/Vdd)	0.95V ~ 3.6V
Data Converters	-
Oscillator Type	External
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	236-LFBGA
Supplier Device Package	236-FBGA (10x10)
Purchase URL	https://www.e-xfl.com/product-detail/xmos/xef216-512-fb236-i20

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_clk		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_ctl	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.

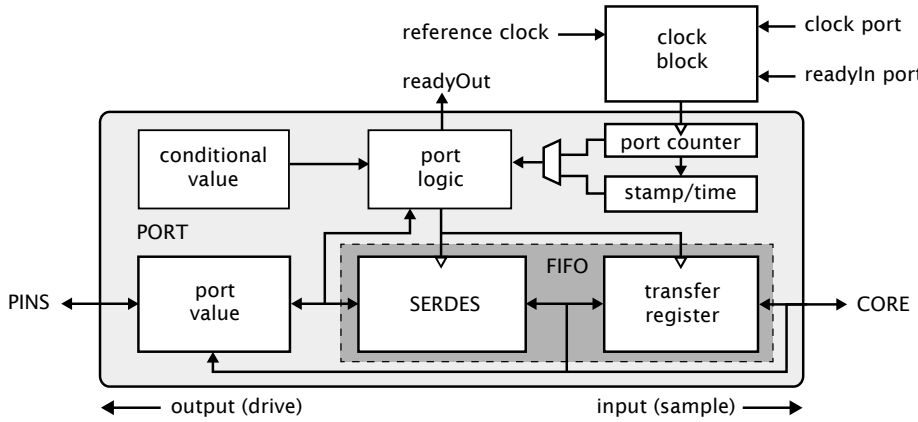


Figure 4: Port block diagram

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.

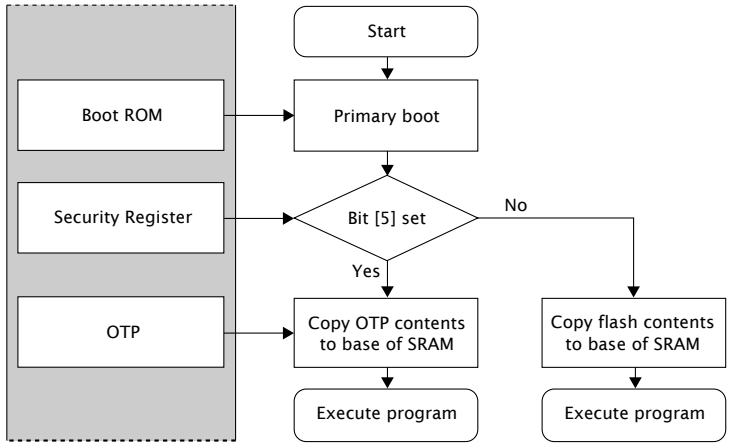


Figure 9: Boot procedure

- 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 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.

and data is communicated through ports on the digital node. A library, XUD, is provided to implement *USB-device* functionality.

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

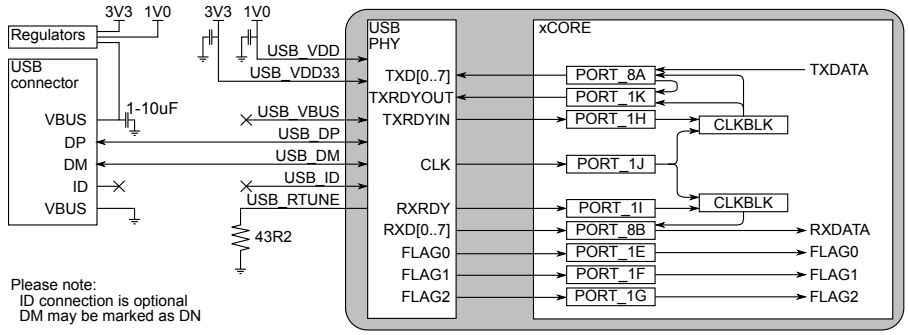


Figure 11: Bus powered *USB-device*

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

10.1 USB VBUS

USB_VBUS need not be connected if the device is wholly powered by USB, and the device is used to implement a *USB-device*.

If you use the USB PHY to design a self-powered *USB-device*, then the device must be able detect the presence of VBus on the USB connector (so the device can disconnect its pull-up resistors from D+/D- to ensure the device does not have any voltage on the D+/D- pins when VBus is not present, “USB Back Voltage Test”). This requires USB_VBUS to be connected to the VBUS pin of the USB connector as is shown in Figure 12.

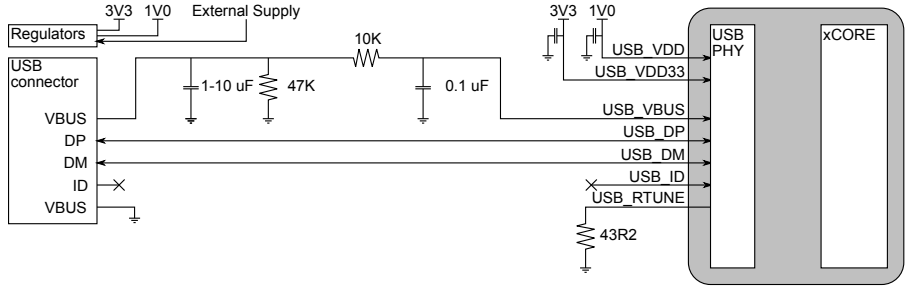


Figure 12: Self powered *USB-device*

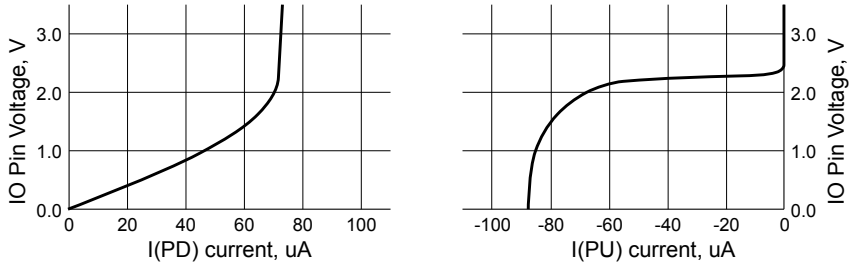


Figure 22: Typical internal pull-down and pull-up currents

14.3 ESD Stress Voltage

Figure 23: ESD stress voltage

Symbol	Parameter	MIN	TYP	MAX	UNITS	Notes
HBM	Human body model	-2.00		2.00	KV	
CDM	Charged Device Model	-500		500	V	

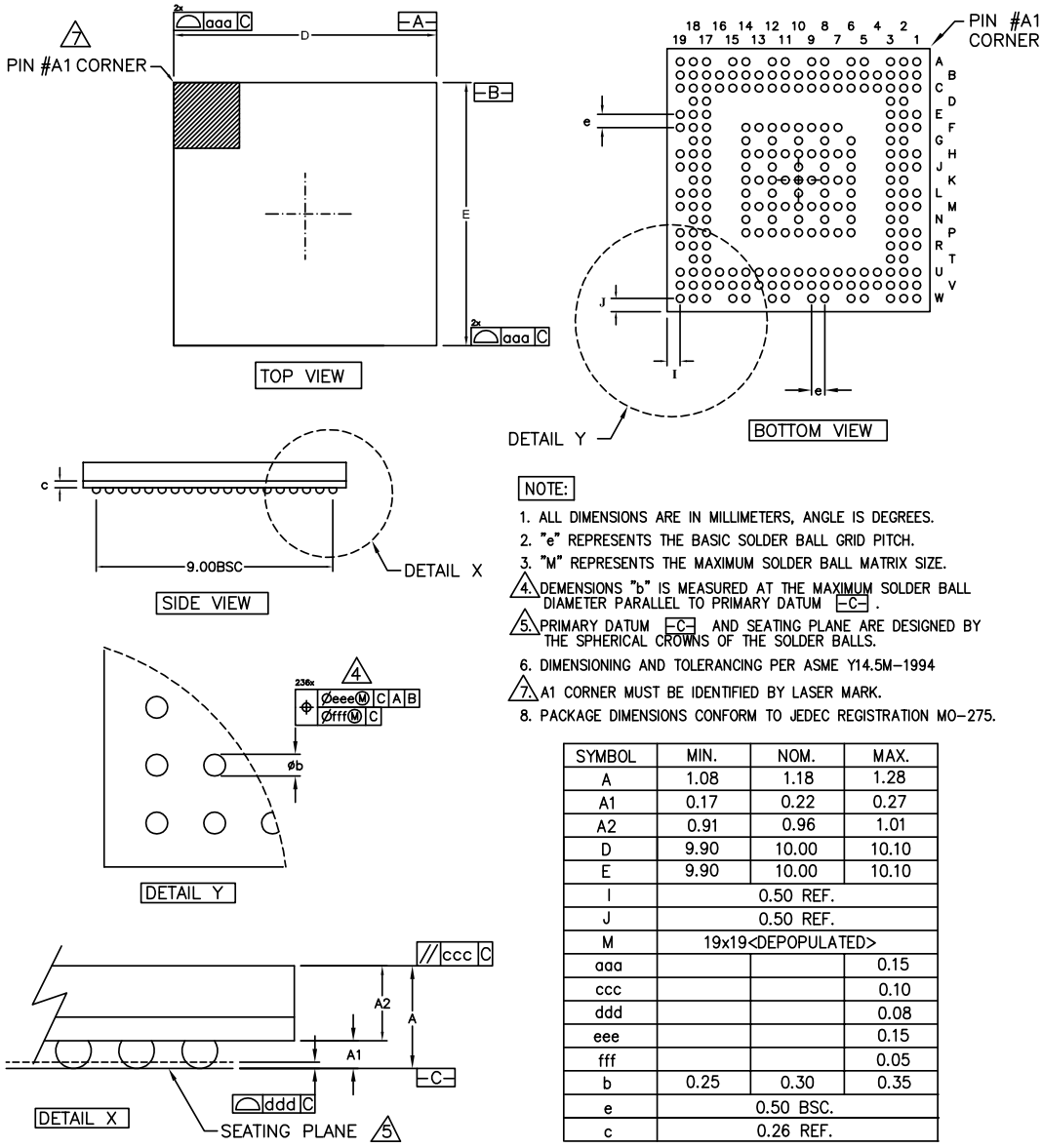
14.4 Reset Timing

Figure 24: Reset timing

Symbol	Parameters	MIN	TYP	MAX	UNITS	Notes
T(RST)	Reset pulse width	5			µs	
T(INIT)	Initialization time			150	µs	A

A Shows the time taken to start booting after RST_N has gone high.

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.	

15.1 Part Marking

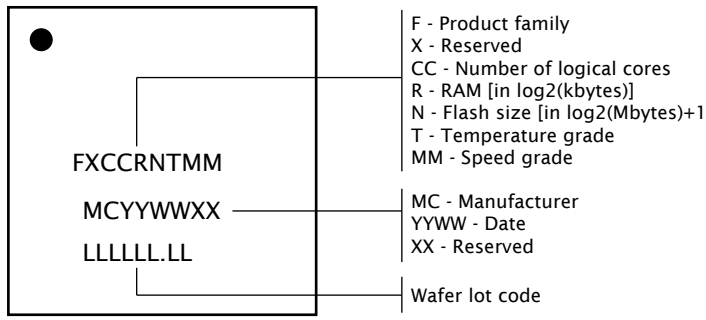


Figure 30: Part marking scheme

16 Ordering Information

Figure 31: Orderable part numbers

Product Code	Marking	Qualification	Speed Grade
XEF216-512-FB236-C20	E01692C20	Commercial	1000 MIPS
XEF216-512-FB236-I20	E01692I20	Industrial	1000 MIPS

control-token 192	24-bit response channel-end identifier	16-bit register number	32-bit data	control-token 1

The response to a write message comprises either control tokens 3 and 1 (for success), or control tokens 4 and 1 (for failure).

A read message comprises the following:

control-token 193	24-bit response channel-end identifier	16-bit register number	control-token 1

The response to the read message comprises either control token 3, 32-bit of data, and control-token 1 (for success), or control tokens 4 and 1 (for failure).

A.3 Accessing node configuration

Node configuration registers can be accessed through the interconnect using the functions `write_node_config_reg(device, ...)` and `read_node_config_reg(device, ...)`, where `device` is the name of the node. These functions implement the protocols described below.

Instead of using the functions above, a channel-end can be allocated to communicate with the node configuration registers. The destination of the channel-end should be set to `0xnnnnC30C` where `nnnn` is the node-identifier.

A write message comprises the following:

control-token 192	24-bit response channel-end identifier	16-bit register number	32-bit data	control-token 1

The response to a write message comprises either control tokens 3 and 1 (for success), or control tokens 4 and 1 (for failure).

A read message comprises the following:

control-token 193	24-bit response channel-end identifier	16-bit register number	control-token 1

The response to a read message comprises either control token 3, 32-bit of data, and control-token 1 (for success), or control tokens 4 and 1 (for failure).

A.4 Accessing a register of an analogue peripheral

Peripheral registers can be accessed through the interconnect using the functions `write_periph_32(device, peripheral, ...)`, `read_periph_32(device, peripheral, ...)`, `write_periph_8(device, peripheral, ...)`, and `read_periph_8(device, peripheral, ...)`; where device is the name of the analogue device, and peripheral is the number of the peripheral. These functions implement the protocols described below.

A channel-end should be allocated to communicate with the configuration registers. The destination of the channel-end should be set to `0xnnnnpp02` where `nnnn` is the node-identifier and `pp` is the peripheral identifier.

A write message comprises the following:

control-token 36	24-bit response channel-end identifier	8-bit register number	8-bit size	data	control-token 1

The response to a write message comprises either control tokens 3 and 1 (for success), or control tokens 4 and 1 (for failure).

A read message comprises the following:

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

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

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

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

C.24 SR of logical core 7: 0x67

Value of the SR of logical core 7

0x67: SR of logical core 7

Bits	Perm	Init	Description
31:0	CRO		Value.

D Node Configuration

The digital node control registers can be accessed using configuration reads and writes (use `write_node_config_reg(device, ...)` and `read_node_config_reg(device, ...)` for reads and writes).

Number	Perm	Description
0x00	RO	Device identification
0x01	RO	System switch description
0x04	RW	Switch configuration
0x05	RW	Switch node identifier
0x06	RW	PLL settings
0x07	RW	System switch clock divider
0x08	RW	Reference clock
0x09	R	System JTAG device ID register
0x0A	R	System USERCODE register
0x0C	RW	Directions 0-7
0x0D	RW	Directions 8-15
0x10	RW	DEBUG_N configuration, tile 0
0x11	RW	DEBUG_N configuration, tile 1
0x1F	RO	Debug source
0x20 .. 0x28	RW	Link status, direction, and network
0x40 .. 0x47	RO	PLink status and network
0x80 .. 0x88	RW	Link configuration and initialization
0xA0 .. 0xA7	RW	Static link configuration

Figure 35: Summary

D.1 Device identification: 0x00

This register contains version and revision identifiers and the mode-pins as sampled at boot-time.

Bits	Perm	Init	Description
31:24	RO	-	Reserved
23:16	RO		Sampled values of BootCtl pins on Power On Reset.
15:8	RO		SSwitch revision.
7:0	RO		SSwitch version.

0x00: Device identification

F USB PHY Configuration

The USB PHY is connected to the ports shown in section 10.

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

Number	Perm	Description
0x00	WO	UIFM reset
0x04	RW	UIFM IFM control
0x08	RW	UIFM Device Address
0x0C	RW	UIFM functional control
0x10	RW	UIFM on-the-go control
0x14	RO	UIFM on-the-go flags
0x18	RW	UIFM Serial Control
0x1C	RW	UIFM signal flags
0x20	RW	UIFM Sticky flags
0x24	RW	UIFM port masks
0x28	RW	UIFM SOF value
0x2C	RO	UIFM PID
0x30	RO	UIFM Endpoint
0x34	RW	UIFM Endpoint match
0x38	RW	OTG Flags mask
0x3C	RW	UIFM power signalling
0x40	RW	UIFM PHY control

Figure 37: Summary

F.1 UIFM reset: 0x00

A write to this register with any data resets all UIFM state, but does not otherwise affect the phy.

0x00: UIFM reset

Bits	Perm	Init	Description
31:0	WO		Value.

F.2 UIFM IFM control: 0x04

General settings of the UIFM IFM state machine.

0x10: UIFM on-the-go control

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

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

Status flags used for on-the-go negotiation

0x14: UIFM on-the-go flags

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

F.7 UIFM Serial Control: 0x18

0x18: UIFM Serial Control

Bits	Perm	Init	Description
31:7	RO	-	Reserved
6	RO	0	1 if UIFM is in UTMI+ RXRCV mode.
5	RO	0	1 if UIFM is in UTMI+ RXDM mode.
4	RO	0	1 if UIFM is in UTMI+ RXDP mode.
3	RW	0	Set to 1 to switch UIFM to UTMI+ TXSE0 mode.
2	RW	0	Set to 1 to switch UIFM to UTMI+ TXDATA mode.
1	RW	1	Set to 0 to switch UIFM to UTMI+ TXENABLE mode.
0	RW	0	Set to 1 to switch UIFM to UTMI+ FSLSSERIAL mode.

F.8 UIFM signal flags: 0x1C

Set of flags that monitor line and error states. These flags normally clear on the next packet, but they may be made sticky by using PER_UIFM_FLAGS_STICKY, in which they must be cleared explicitly.

0x1C: UIFM signal flags

Bits	Perm	Init	Description
31:7	RO	-	Reserved
6	RW	0	Set to 1 when the UIFM decodes a token successfully (e.g. it passes CRC5, PID check and has matching device address).
5	RW	0	Set to 1 when linestate indicates an SE0 symbol.
4	RW	0	Set to 1 when linestate indicates a K symbol.
3	RW	0	Set to 1 when linestate indicates a J symbol.
2	RW	0	Set to 1 if an incoming datapacket fails the CRC16 check.
1	RW	0	Set to the value of the UTMI_RXACTIVE input signal.
0	RW	0	Set to the value of the UTMI_RXERROR input signal

F.9 UIFM Sticky flags: 0x20

These bits define the sticky-ness of the bits in the UIFM IFM FLAGS register. A 1 means that bit will be sticky (hold its value until a 1 is written to that bitfield), or normal, in which case signal updates to the UIFM IFM FLAGS bits may be over-written by subsequent changes in those signals.

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

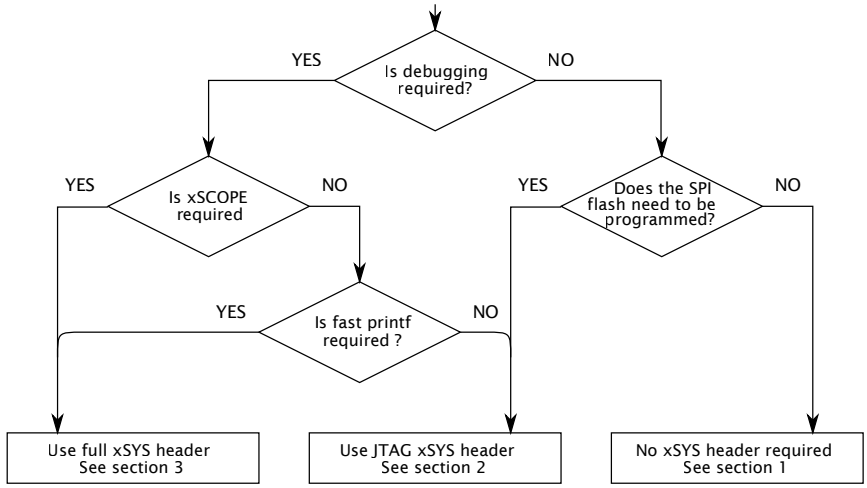


Figure 38: Decision diagram for the xSYS header

G.1 No xSYS header

The use of an xSYS header is optional, and may not be required for volume production designs. However, the XMOS toolchain expects the xSYS header; if you do not have an xSYS header then you must provide your own method for writing to flash/OTP and for debugging.

G.2 JTAG-only xSYS header

The xSYS header connects to an xTAG debugger, which has a 20-pin 0.1" female IDC header. The design will hence need a male IDC header. We advise to use a boxed header to guard against incorrect plug-ins. If you use a 90 degree angled header, make sure that pins 2, 4, 6, ..., 20 are along the edge of the PCB.

Connect pins 4, 8, 12, 16, 20 of the xSYS header to ground, and then connect:

- TDI to pin 5 of the xSYS header
- TMS to pin 7 of the xSYS header
- TCK to pin 9 of the xSYS header
- DEBUG_N to pin 11 of the xSYS header

H Schematics Design Check List

☑ This section is a checklist for use by schematics designers using the XEF216-512-FB236. Each of the following sections contains items to check for each design.

H.1 Power supplies

☐ VDDIO and OTP_VCC supply is within specification before the VDD (core) supply is turned on. Specifically, the VDDIO and OTP_VCC supply is within specification before VDD (core) reaches 0.4V (Section 13).

☐ The VDD (core) supply ramps monotonically (rises constantly) from 0V to its final value (0.95V - 1.05V) within 10ms (Section 13).

☐ The VDD (core) supply is capable of supplying 700 mA (Section 13 and Figure 21).

☐ PLL_AVDD is filtered with a low pass filter, for example an RC filter, see Section 13 .

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 13).

☐ A bulk decoupling capacitor of at least 10uF is placed on each supply (Section 13).

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.

L Revision History

Date	Description
2015-03-20	Preliminary release
2015-04-14	Added RST to pins to be pulled hard, and removed reference to TCK from Errata Removed TRST_N references in packages that have no TRST_N New diagram for boot from embedded flash showing ports Pull up requirements for shared clock and external resistor for QSPI
2015-05-06	Removed references to DEBUG_N
2015-07-09	Updated electrical characteristics - Section 14
2015-08-19	Added I(USB_VDD) - Section 14 Added USB layout guidelines - Section 13
2015-08-27	Updated part marking - Section 16
2016-04-20	Typical internal pull-up and pull down current diagrams added - Section 14
2017-02-02	Updated USB VBUS wiring description with bus-powered usb-device instructions - Section 10 Clarified available RGMII ports/pins - Section 11

XMOS®