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 10-Core
Speed	2000MIPS
Connectivity	USB
Peripherals	-
Number of I/O	81
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	0°C ~ 70°C (TA)
Mounting Type	Surface Mount
Package / Case	128-TQFP Exposed Pad
Supplier Device Package	128-TQFP (14x14)
Purchase URL	https://www.e-xfl.com/product-detail/xmos/xuf210-512-tq128-c20

- **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 2MBflash. 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 XUF210-512-TQ128 Features

- **Multicore Microcontroller with Advanced Multi-Core RISC Architecture**
  - 10 real-time logical cores on 2 xCORE tiles
  - Cores share up to 1000 MIPS
    - — Up to 2000 MIPS in dual issue mode
  - Each logical core has:
    - — Guaranteed throughput of between 1/5 and 1/5 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
- **USB PHY, fully compliant with USB 2.0 specification**
- **Programmable I/O**
  - 81 general-purpose I/O pins, configurable as input or output
    - — Up to 25 x 1bit port, 12 x 4bit port, 8 x 8bit port, 4 x 16bit port
    - — 4 xCONNECT links
  - Port sampling rates of up to 60 MHz with respect to an external clock
  - 64 channel endss (32 per tile) for communication with other cores, on or off-chip
- **Memory**
  - 512KB internal single-cycle SRAM (max 256KB per tile) for code and data storage
  - 16KB internal OTP (max 8KB per tile) for application boot code
  - 2MB internal flash for application code and overlays
- **Hardware resources**
  - 12 clock blocks (6 per tile)
  - 20 timers (10 per tile)
  - 8 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**
  - 20: 1000 MIPS
- **Power Consumption**
  - 570 mA (typical)
- **128-pin TQFP package 0.4 mm pitch**

6 Product Overview

The XUF210-512-TQ128 is a powerful device that consists of two xCORE Tiles, each comprising a flexible logical processing cores with tightly integrated I/O and on-chip memory.

6.1 Logical cores

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

Figure 3: Logical core performance

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

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

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

6.2 xTIME scheduler

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

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

6.3 Hardware Response Ports

Hardware Response ports connect an xCORE tile to one or more physical pins and as such define the interface between hardware attached to the XUF210-512-TQ128, and the software running on it. A combination of 1bit, 4bit, 8bit, 16bit and 32bit ports are available. All pins of a port provide either output or input. Signals in different directions cannot be mapped onto the same port.

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

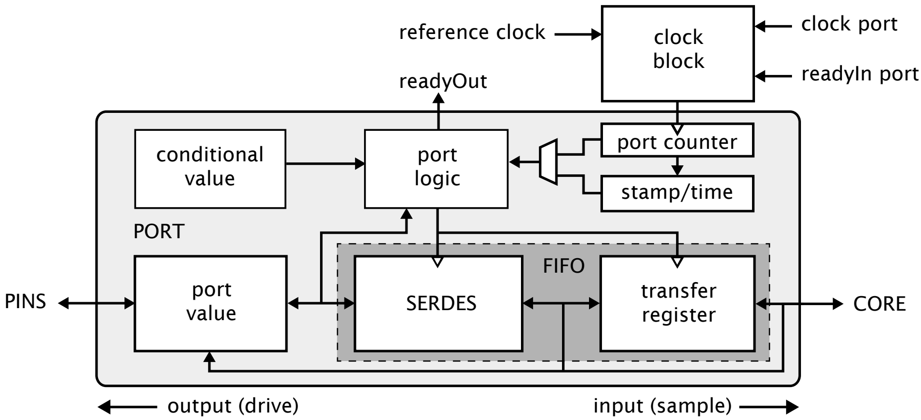

Figure 4: Port block diagram

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

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

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

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

6.4 Clock blocks

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

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

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.

The device boots from a QSPI flash (IS25LQ016B) 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.

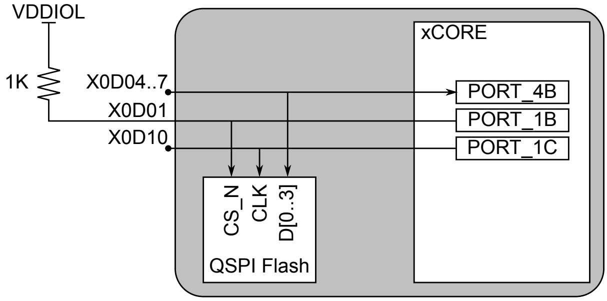

Figure 8: QSPI port connectivity

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.

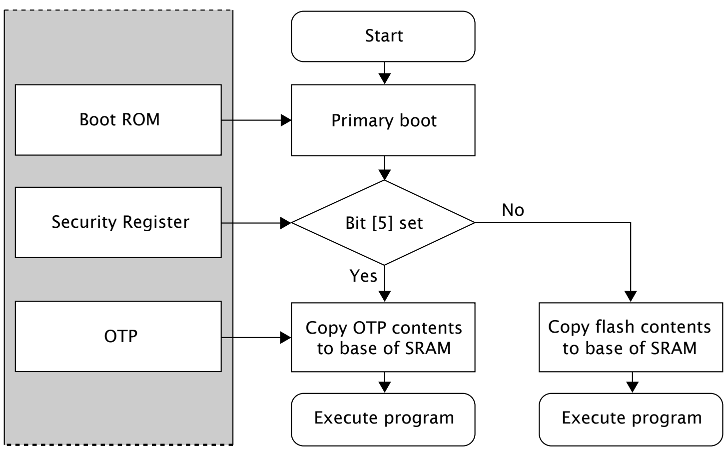

Figure 9: Boot procedure

13.7 xCORE Tile I/O AC Characteristics

Symbol	Parameter	MIN	TYP	MAX	UNITS	Notes
T(XOVALID)	Input data valid window	8			ns	
T(XOINVALID)	Output data invalid window	9			ns	
T(XIFMAX)	Rate at which data can be sampled with respect to an external clock			60	MHz	

Figure 25: I/O AC characteristics

The input valid window parameter relates to the capability of the device to capture data input to the chip with respect to an external clock source. It is calculated as the sum of the input setup time and input hold time with respect to the external clock as measured at the pins. The output invalid window specifies the time for which an output is invalid with respect to the external clock. Note that these parameters are specified as a window rather than absolute numbers since the device provides functionality to delay the incoming clock with respect to the incoming data.

Information on interfacing to high-speed synchronous interfaces can be found in the XS1 Port I/O Timing document, X5821.

13.8 xConnect Link Performance

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

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

B 7.5 ns symbol time.

Figure 26: Link performance

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

13.9 JTAG Timing

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

A Timing applies to TMS and TDI inputs.

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

Figure 27: JTAG timing

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

14.1 Part Marking

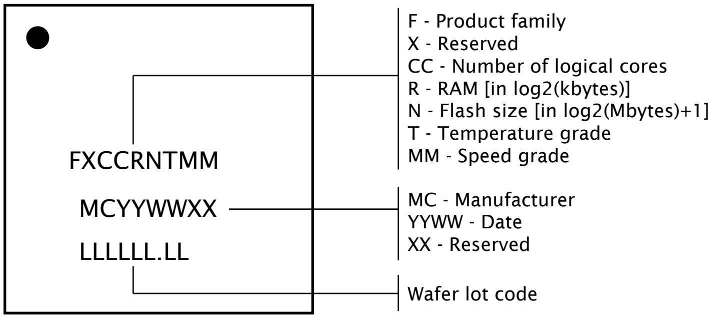

Figure 28: Part marking scheme

15 Ordering Information

Figure 29: Orderable part numbers

Product Code	Marking	Qualification	Speed Grade
XUF210-512-TQ128-C20	U11092C20	Commercial	1000 MIPS
XUF210-512-TQ128-I20	U11092I20	Industrial	1000 MIPS

Appendices

A Configuration of the XUF210-512-TQ128

The device is configured through banks of registers, as shown in Figure 30.

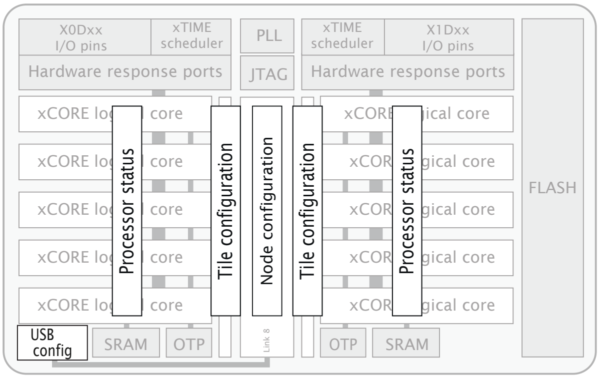

Figure 30: Registers

The following communication sequences specify how to access those registers. Any messages transmitted contain the most significant 24 bits of the channel-end to which a response is to be sent. This comprises the node-identifier and the channel number within the node. if no response is required on a write operation, supply 24-bits with the last 8-bits set, which suppresses the reply message. Any multi-byte data is sent most significant byte first.

A.1 Accessing a processor status register

The processor status registers are accessed directly from the processor instruction set. The instructions GETPS and SETPS read and write a word. The register number should be translated into a processor-status resource identifier by shifting the register number left 8 places, and ORing it with 0x0B. Alternatively, the functions `getps(reg)` and `setps(reg,value)` can be used from XC.

A.2 Accessing an xCORE Tile configuration register

xCORE Tile configuration registers can be accessed through the interconnect using the functions `write_tile_config_reg(tileref, ...)` and `read_tile_config_reg(tile ↪ ref, ...)`, where `tileref` is the name of the xCORE Tile, e.g. `tile[1]`. These functions implement the protocols described below.

Instead of using the functions above, a channel-end can be allocated to communicate with the xCORE tile configuration registers. The destination of the channel-end should be set to `0xnnnnC20C` where `nnnnnn` is the tile-identifier.

A write message comprises the following:

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).

0x12: Debug SSP

Bits	Perm	Init	Description
31:0	DRW		Value.

B.15 DGETREG operand 1: 0x13

The resource ID of the logical core whose state is to be read.

0x13: DGETREG operand 1

Bits	Perm	Init	Description
31:8	RO	-	Reserved
7:0	DRW		Thread number to be read

B.16 DGETREG operand 2: 0x14

Register number to be read by DGETREG

0x14: DGETREG operand 2

Bits	Perm	Init	Description
31:5	RO	-	Reserved
4:0	DRW		Register number to be read

B.17 Debug interrupt type: 0x15

Register that specifies what activated the debug interrupt.

0x15: Debug interrupt type

Bits	Perm	Init	Description
31:18	RO	-	Reserved
17:16	DRW		Number of the hardware breakpoint/watchpoint which caused the interrupt (always 0 for =HOST= and =DCALL=). If multiple breakpoints/watchpoints trigger at once, the lowest number is taken.
15:8	DRW		Number of thread which caused the debug interrupt (always 0 in the case of =HOST=).
7:3	RO	-	Reserved
2:0	DRW	0	Indicates the cause of the debug interrupt 1: Host initiated a debug interrupt through JTAG 2: Program executed a DCALL instruction 3: Instruction breakpoint 4: Data watch point 5: Resource watch point

0x30 .. 0x33: Instruction breakpoint address

Bits	Perm	Init	Description
31:0	DRW		Value.

B.22 Instruction breakpoint control: 0x40 .. 0x43

This register controls which logical cores may take an instruction breakpoint, and under which condition.

0x40 .. 0x43: Instruction breakpoint control

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 PC == IBREAK_ADDR. When 1 = break when PC != IBREAK_ADDR.
0	DRW	0	When 1 the instruction breakpoint is enabled.

B.23 Data watchpoint address 1: 0x50 .. 0x53

This set of registers contains the first address for the four data watchpoints.

0x50 .. 0x53: Data watchpoint address 1

Bits	Perm	Init	Description
31:0	DRW		Value.

B.24 Data watchpoint address 2: 0x60 .. 0x63

This set of registers contains the second address for the four data watchpoints.

0x60 .. 0x63: Data watchpoint address 2

Bits	Perm	Init	Description
31:0	DRW		Value.

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

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

0x70 .. 0x73: Data breakpoint control register

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

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

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

0x80 .. 0x83: Resources breakpoint mask

Bits	Perm	Init	Description
31:0	DRW		Value.

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

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

0x90 .. 0x93: Resources breakpoint value

Bits	Perm	Init	Description
31:0	DRW		Value.

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

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

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.

0x41: PC of logical core 1

Bits	Perm	Init	Description
31:0	CRO		Value.

C.11 PC of logical core 2: 0x42

Value of the PC of logical core 2.

0x42: PC of logical core 2

Bits	Perm	Init	Description
31:0	CRO		Value.

C.12 PC of logical core 3: 0x43

Value of the PC of logical core 3.

0x43: PC of logical core 3

Bits	Perm	Init	Description
31:0	CRO		Value.

C.13 PC of logical core 4: 0x44

Value of the PC of logical core 4.

0x44: PC of logical core 4

Bits	Perm	Init	Description
31:0	CRO		Value.

C.14 PC of logical core 5: 0x45

Value of the PC of logical core 5.

0x45: PC of logical core 5

Bits	Perm	Init	Description
31:0	CRO		Value.

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

0x06: PLL settings

Bits	Perm	Init	Description
31	RW		If set to 1, the chip will not be reset
30	RW		If set to 1, the chip will not wait for the PLL to re-lock. Only use this if a gradual change is made to the PLL
29	DW		If set to 1, set the PLL to be bypassed
28	DW		If set to 1, set the boot mode to boot from JTAG
27:26	RO	-	Reserved
25:23	RW		Output divider value range from 1 (8'h0) to 250 (8'hF9). P value.
22:21	RO	-	Reserved
20:8	RW		Feedback multiplication ratio, range from 1 (8'h0) to 255 (8'hFE). M value.
7	RO	-	Reserved
6:0	RW		Oscilator input divider value range from 1 (8'h0) to 32 (8'h0F). N value.

D.6 System switch clock divider: 0x07

Sets the ratio of the PLL clock and the switch clock.

0x07: System switch clock divider

Bits	Perm	Init	Description
31:16	RO	-	Reserved
15:0	RW	0	SSwitch clock generation

D.7 Reference clock: 0x08

Sets the ratio of the PLL clock and the reference clock used by the node.

0x08: Reference clock

Bits	Perm	Init	Description
31:16	RO	-	Reserved
15:0	RW	3	Software ref. clock divider

0x04: UIFM IFM control

Bits	Perm	Init	Description
31:8	RO	-	Reserved
7	RW	0	Set to 1 to enable XEVACKMODE mode.
6	RW	0	Set to 1 to enable SOFISTOKEN mode.
5	RW	0	Set to 1 to enable UIFM power signalling mode.
4	RW	0	Set to 1 to enable IF timing mode.
3	RO	-	Reserved
2	RW	0	Set to 1 to enable UIFM linestate decoder.
1	RW	0	Set to 1 to enable UIFM CHECKTOKENS mode.
0	RW	0	Set to 1 to enable UIFM DOTOKENS mode.

F.3 UIFM Device Address: 0x08

The device address whose packets should be received. 0 until enumeration, it should be set to the assigned value after enumeration.

0x08: UIFM Device Address

Bits	Perm	Init	Description
31:7	RO	-	Reserved
6:0	RW	0	The enumerated USB device address must be stored here. Only packets to this address are passed on.

F.4 UIFM functional control: 0x0C

0x0C: UIFM functional control

Bits	Perm	Init	Description
31:5	RO	-	Reserved
4:2	RW	1	Set to 0 to disable UIFM to UTMI+ OPMODE mode.
1	RW	1	Set to 1 to switch UIFM to UTMI+ TERMSELECT mode.
0	RW	1	Set to 1 to switch UIFM to UTMI+ XCVRSELECT mode.

F.5 UIFM on-the-go control: 0x10

This register is used to negotiate an on-the-go connection.

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.

0x2C: UIFM PID

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

F.13 UIFM Endpoint: 0x30

The last endpoint seen

0x30: UIFM Endpoint

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

F.14 UIFM Endpoint match: 0x34

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

0x34: UIFM Endpoint match

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

F.15 OTG Flags mask: 0x38

0x38: OTG Flags mask

Bits	Perm	Init	Description
31:0	RW	0	Data

F.16 UIFM power signalling: 0x3C

0x3C: UIFM power signalling

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

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

G.3 Full xSYS header

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

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