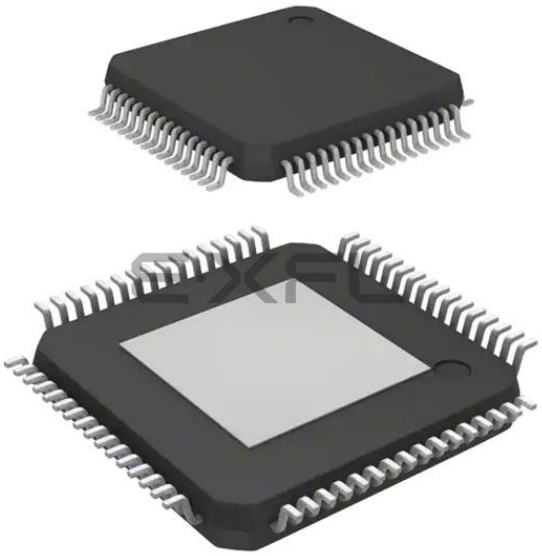


Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Active
Core Processor	XCore
Core Size	32-Bit 6-Core
Speed	400MIPS
Connectivity	Configurable
Peripherals	-
Number of I/O	36
Program Memory Size	64KB (16K x 32)
Program Memory Type	SRAM
EEPROM Size	-
RAM Size	-
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	64-LQFP Exposed Pad
Supplier Device Package	64-LQFP (10x10)
Purchase URL	https://www.e-xfl.com/product-detail/xmos/xs1-l6a-64-lq64-c4

- **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 5.5
- **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 5.6
- **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 5.3
- **Clock blocks** xCORE devices include a set of programmable clock blocks that can be used to govern the rate at which ports execute. Section 5.4
- **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 8
- **PLL** The PLL is used to create a high-speed processor clock given a low speed external oscillator. Section 6
- **JTAG** The JTAG module can be used for loading programs, boundary scan testing, in-circuit source-level debugging and programming the OTP memory. Section 9

1.1 Software

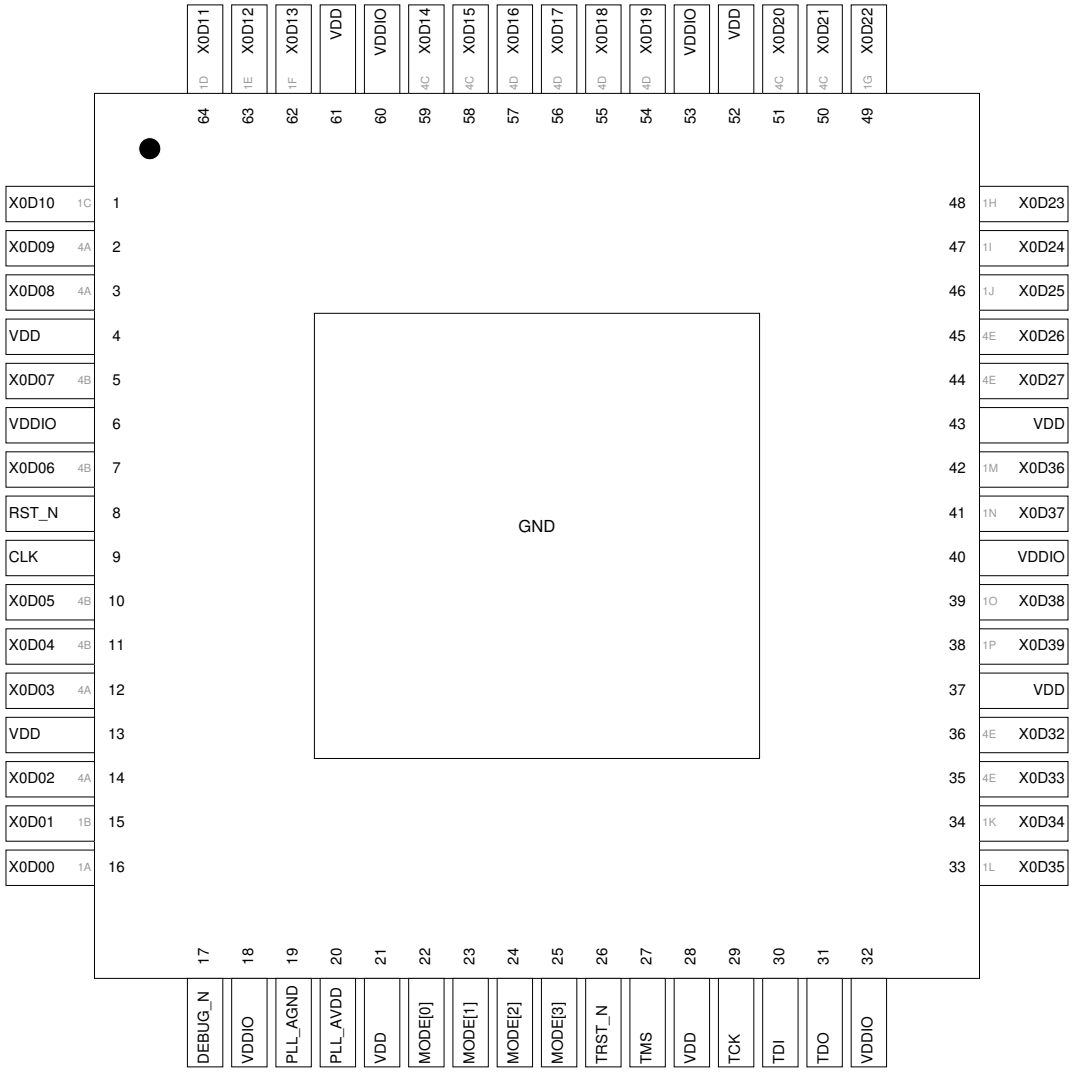
Devices are programmed using C, C++ or xC (C with multicore extensions). XMOS provides tested and proven software libraries, which allow you to quickly add interface and processor functionality such as USB, Ethernet, PWM, graphics driver, and audio EQ to your applications.

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.

3 Pin Configuration



Signal	Function					Type	Properties
X0D11	1D[0]					I/O	PD_S, R_S
X0D12	1E[0]					I/O	PD_S, R_U
X0D13	XLB^4_{out}	1F[0]				I/O	PD_S, R_U
X0D14	XLB^3_{out}	4C[0]	8B[0]	16A[8]	32A[28]	I/O	PD_S, R_U
X0D15	XLB^2_{out}	4C[1]	8B[1]	16A[9]	32A[29]	I/O	PD_S, R_U
X0D16	XLB^1_{out}	4D[0]	8B[2]	16A[10]		I/O	PD_S, R_U
X0D17	XLB^0_{out}	4D[1]	8B[3]	16A[11]		I/O	PD_S, R_U
X0D18	XLB^0_{in}	4D[2]	8B[4]	16A[12]		I/O	PD_S, R_U
X0D19	XLB^1_{in}	4D[3]	8B[5]	16A[13]		I/O	PD_S, R_U
X0D20	XLB^2_{in}	4C[2]	8B[6]	16A[14]	32A[30]	I/O	PD_S, R_U
X0D21	XLB^3_{in}	4C[3]	8B[7]	16A[15]	32A[31]	I/O	PD_S, R_U
X0D22	XLB^4_{in}	1G[0]				I/O	PD_S, R_U
X0D23	1H[0]					I/O	PD_S, R_U
X0D24	1I[0]					I/O	PD_S
X0D25	1J[0]					I/O	PD_S
X0D26		4E[0]	8C[0]	16B[0]		I/O	PD_S, R_U
X0D27		4E[1]	8C[1]	16B[1]		I/O	PD_S, R_U
X0D32		4E[2]	8C[6]	16B[6]		I/O	PD_S, R_U
X0D33		4E[3]	8C[7]	16B[7]		I/O	PD_S, R_U
X0D34	1K[0]					I/O	PD_S
X0D35	1L[0]					I/O	PD_S
X0D36	1M[0]		8D[0]	16B[8]		I/O	PD_S
X0D37	1N[0]		8D[1]	16B[9]		I/O	PD_S, R_U
X0D38	1O[0]		8D[2]	16B[10]		I/O	PD_S, R_U
X0D39	1P[0]		8D[3]	16B[11]		I/O	PD_S, R_U

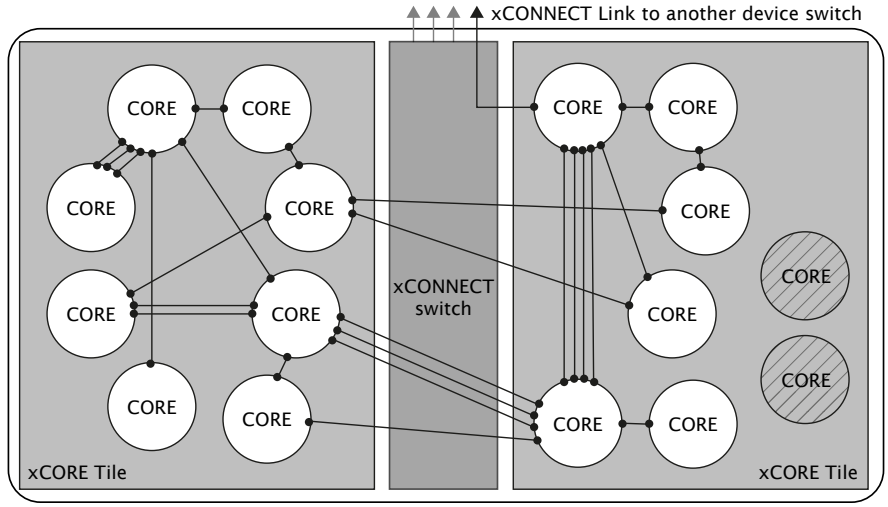


Figure 5: Switch, links and channel ends

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

6 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 6:

Oscillator Frequency	MODE 1	MODE 0	Tile Frequency	PLL Ratio	PLL settings OD	F	R
5-13 MHz	0	0	130-399.75 MHz	30.75	1	122	0
13-20 MHz	1	1	260-400.00 MHz	20	2	119	0
20-48 MHz	1	0	167-400.00 MHz	8.33	2	49	0
48-100 MHz	0	1	196-400.00 MHz	4	2	23	0

Figure 6: PLL multiplier values and MODE pins

Figure 6 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:

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

1. Allocate channel-end 0.
2. Input a word on channel-end 0. It will use this word as a channel to acknowledge the boot. Provide the null-channel-end 0x0000FF02 if no acknowledgment is required.
3. Input the boot image specified above, including the CRC.
4. Input an END control token.
5. Output an END control token to the channel-end received in step 2.
6. Free channel-end 0.
7. Jump to the loaded code.

7.3 Boot from OTP

If an xCORE tile is set to use secure boot (see Figure 7), the boot image is read from address 0 of the OTP memory in the tile's security module.

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

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

7.4 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.

8 Memory

8.1 OTP

The 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.

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

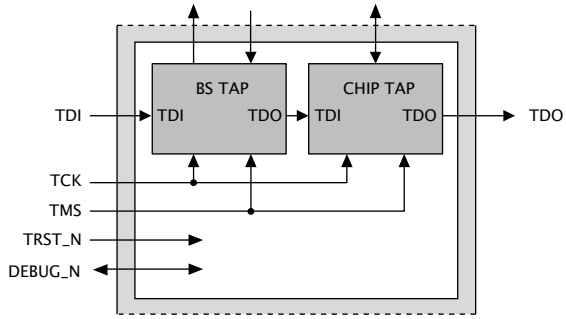


Figure 11: JTAG chain structure

The JTAG chain structure is illustrated in Figure 11. Directly after reset, two TAP controllers are present in the JTAG chain: 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 4K7-47KΩ 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 12.

Figure 12: IDCODE return value

Bit31																												Bit0
Version				Part Number																Manufacturer Identity								1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	1	0	0	0	1	1	1
0				0				0				0				2				6				3				3

The JTAG usercode register can be read by using the USERCODE instruction. Its contents are specified in Figure 13. The OTP User ID field is read from bits [22:31] of the security register , *see* §8.1 (all zero on unprogrammed devices).

Figure 13: USERCODE return value

Bit31																												Bit0
OTP User ID										Unused				Silicon Revision														
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0
0				0				0				2				8				0				0				0

10 Board Integration

The device has the following power supply pins:

- VDD pins for the xCORE Tile
- VDDIO pins for the I/O lines
- PLL_AVDD pins for the PLL

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 supply must ramp to its final value before VDD reaches 0.4 V.

The PLL_AVDD 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 100 nF 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* §7). RST_N and must be asserted low during and after power up for 100 ns.

10.1 Land patterns and solder stencils

The land pattern recommendations in this document are based on a RoHS compliant process and derived, where possible, from the nominal *Generic Requirements for Surface Mount Design and Land Pattern Standards* IPC-7351B specifications. This standard aims to achieve desired targets of heel, toe and side fillets for solder-joints.

Solder paste and ground via recommendations are based on our engineering and development kit board production. They have been found to work and optimized as appropriate to achieve a high yield. The size, type and number of vias used in the center pad affects how much solder wicks down the vias during reflow. This in turn, along with solder paster coverage, affects the final assembled package height. These factors should be taken into account during design and manufacturing of the PCB.

The following land patterns and solder paste contains recommendations. Final land pattern and solder paste decisions are the responsibility of the customer. These should be tuned during manufacture to suit the manufacturing process.

The package is a 64 pin Low profile Quad Flat Pack package with exposed heat slug on a 0.5mm pitch. An example land pattern is shown in Figure 14.

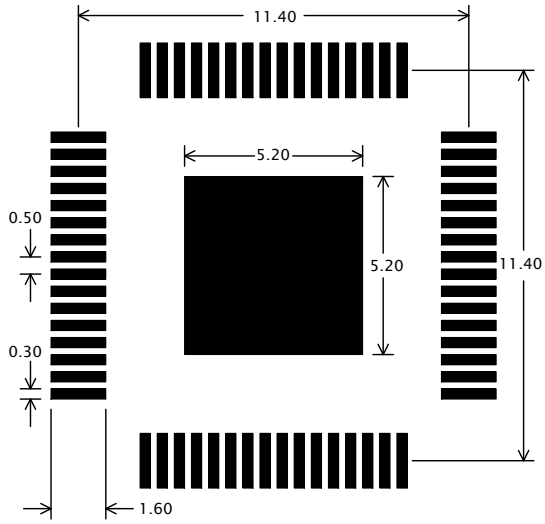


Figure 14: Example land pattern

For the 64 pin LQFP package, a 3x3 array of squares for solder paste is recommended as shown in Figure 15. This gives a paste level of 48%.

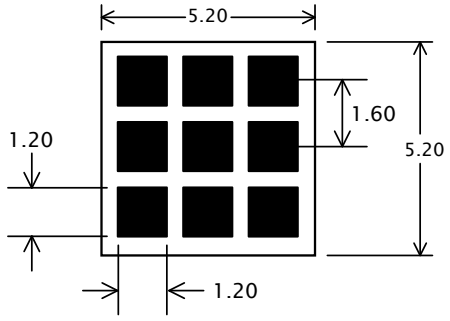
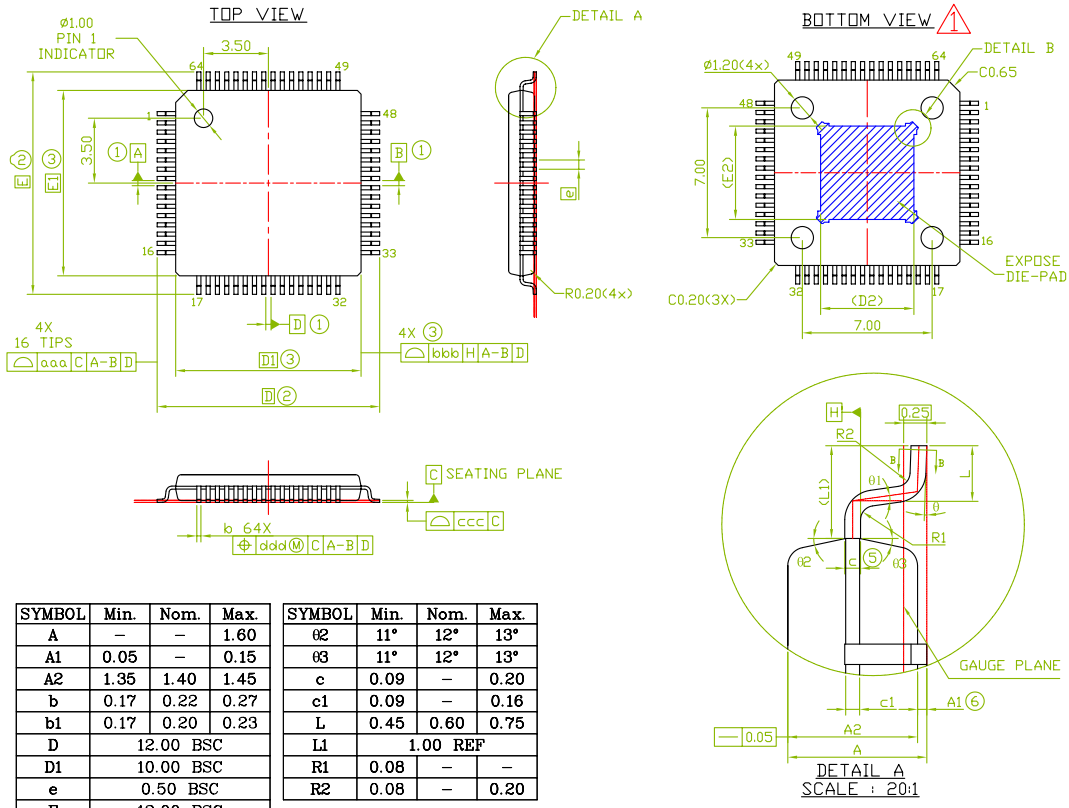


Figure 15: Solder stencil for centre pad

12 Package Information



SYMBOL	Min.	Nom.	Max.
A	–	–	1.60
A1	0.05	–	0.15
A2	1.35	1.40	1.45
b	0.17	0.22	0.27
b1	0.17	0.20	0.23
D	12.00 BSC		
D1	10.00 BSC		
e	0.50 BSC		
E	12.00 BSC		
E1	10.00 BSC		
θ	0°	3.5°	7°
θ1	0°	–	–

SYMBOL	Min.	Nom.	Max.
θ2	11°	12°	13°
θ3	11°	12°	13°
c	0.09	–	0.20
c1	0.09	–	0.16
L	0.45	0.60	0.75
L1	1.00 REF		
R1	0.08	–	–
R2	0.08	–	0.20

REF	TOLERANCES OF FORM AND POSITION
aaa	0.20
bbb	0.20
ccc	0.08
ddd	0.08

NOTE :
① DATUM A-B AND D TO DETERMINE AT DATUM PLANE H.
② TO BE DETERMINED AT SEATING DATUM PLANE C.
③ DIMENSION D1 AND E1 DO NOT INCLUDE MOLD PROTRUSIONS. ALLOWABLE PROTRUSION IS 0.25mm PER SIDE. D1 AND E1 ARE MAXIMUM PLASTIC BODY SIZE DIMENSION INCLUDING MOLD MISMATCH.
④ DIMENSION b DOES NOT INCLUDE DAMBAR PROTRUSION. ALLOWABLE DAMBAR PROTRUSION SHALL NOT CAUSE THE LEAD WIDTH TO EXCEED THE MAXIMUM b DIMENSION BY MORE THAN 0.08mm. DAMBAR CANNOT BE LOCATED ON THE LOWER RADIUS OR THE FOOT. MINIMUM SPACE BETWEEN PROTRUSION AND ADJACENT LEAD IS 0.07mm FOR 0.4mm AND 0.5mm PITCH PACKAGE.
⑤ THESE DIMENSIONS APPLY TO THE FLAT SECTION OF THE BETWEEN 0.10mm AND 0.25mm FROM THE LEAD TIP.
⑥ A1 IS THE DEFINED AS THE DISTANCE FROM THE SEATING PLANE TO THE LOWEST POINT ON THE PACKAGE BODY.
7 PACKAGE DIMENSIONS REFERENCE TO JEDEC MS-026 Rev.D, except D2 and E2

LF Ref#	Symbol	Min	Nom	Max
L-16-09006	D2	4.93	5.03	5.13
	E2	4.93	5.03	5.13

Appendices

A Configuration of the XS1

The device is configured through three banks of registers, as shown in Figure 27.

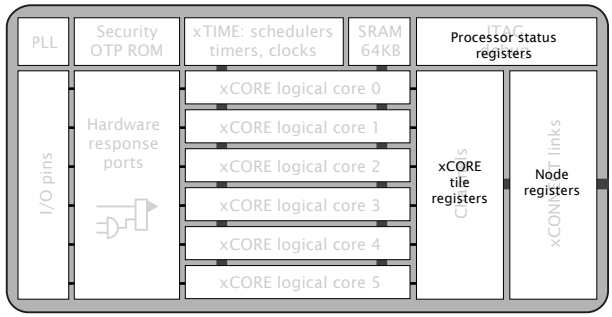


Figure 27: Registers

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 0x0C. 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:

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).

B.1 RAM base address: 0x00

This register contains the base address of the RAM. It is initialized to 0x00010000.

0x00: RAM base address

Bits	Perm	Init	Description
31:2	RW		Most significant 16 bits of all addresses.
1:0	RO	-	Reserved

B.2 Vector base address: 0x01

Base address of event vectors in each resource. On an interrupt or event, the 16 most significant bits of the destination address are provided by this register; the least significant 16 bits come from the event vector.

0x01: Vector base address

Bits	Perm	Init	Description
31:16	RW		The most significant bits for all event and interrupt vectors.
15:0	RO	-	Reserved

B.3 xCORE Tile control: 0x02

Register to control features in the xCORE tile

0x02: xCORE Tile control

Bits	Perm	Init	Description
31:6	RO	-	Reserved
5	RW	0	Set to 1 to select the dynamic mode for the clock divider when the clock divider is enabled. In dynamic mode the clock divider is only activated when all active logical cores are paused. In static mode the clock divider is always enabled.
4	RW	0	Set to 1 to enable the clock divider. This slows down the xCORE tile clock in order to use less power.
3:0	RO	-	Reserved

B.4 xCORE Tile boot status: 0x03

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

0x15: Debug interrupt type

Bits	Perm	Init	Description
31:18	RO	-	Reserved
17:16	DRW		If the debug interrupt was caused by a hardware breakpoint or hardware watchpoint, this field contains the number of the breakpoint or watchpoint. If multiple breakpoints or watchpoints trigger at once, the lowest number is taken.
15:8	DRW		If the debug interrupt was caused by a logical core, this field contains the number of that core. Otherwise this field is 0.
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

B.17 Debug interrupt data: 0x16

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

0x16: Debug interrupt data

Bits	Perm	Init	Description
31:0	DRW		Value.

B.18 Debug core control: 0x18

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

0x18: Debug core control

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

0x50 .. 0x53: Data watchpoint address 1

Bits	Perm	Init	Description
31:0	DRW		Value.

B.23 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.24 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 logical core in the tile allowing the breakpoint to be enabled individually for each logical core.
15:3	RO	-	Reserved
2	DRW	0	Set to 1 to enable breakpoints to be triggered on loads. Breakpoints always trigger on stores.
1	DRW	0	By default, data watchpoints trigger if memory in the range [Address1..Address2] is accessed (the range is inclusive of Address1 and Address2). If set to 1, data watchpoints trigger if memory outside the range (Address2..Address1) is accessed (the range is exclusive of Address2 and Address1).
0	DRW	0	When 1 the instruction breakpoint is enabled.

B.25 Resources breakpoint mask: 0x80 .. 0x83

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

C Tile Configuration

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

Number	Perm	Description
0x00	RO	Device identification
0x01	RO	xCORE Tile description 1
0x02	RO	xCORE Tile description 2
0x04	CRW	Control PSwitch permissions to debug registers
0x05	CRW	Cause debug interrupts
0x06	RW	xCORE Tile clock divider
0x07	RO	Security configuration
0x10 .. 0x13	RO	PLink status
0x20 .. 0x27	CRW	Debug scratch
0x40	RO	PC of logical core 0
0x41	RO	PC of logical core 1
0x42	RO	PC of logical core 2
0x43	RO	PC of logical core 3
0x44	RO	PC of logical core 4
0x45	RO	PC of logical core 5
0x60	RO	SR of logical core 0
0x61	RO	SR of logical core 1
0x62	RO	SR of logical core 2
0x63	RO	SR of logical core 3
0x64	RO	SR of logical core 4
0x65	RO	SR of logical core 5
0x80 .. 0x9F	RO	Chanend status

Figure 29: Summary

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
0x0C	RW	Directions 0-7
0x0D	RW	Directions 8-15
0x10	RW	DEBUG_N configuration
0x1F	RO	Debug source
0x20 .. 0x27	RW	Link status, direction, and network
0x40 .. 0x43	RW	PLink status and network
0x80 .. 0x87	RW	Link configuration and initialization
0xA0 .. 0xA7	RW	Static link configuration

Figure 30: 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	0x00	Chip identifier.
23:16	RO		Sampled values of pins MODE0, MODE1, ... on reset.
15:8	RO		SSwitch revision.
7:0	RO		SSwitch version.

0x00: Device identification

D.2 System switch description: 0x01

This register specifies the number of processors and links that are connected to this switch.

0x01: System switch description

Bits	Perm	Init	Description
31:24	RO	-	Reserved
23:16	RO		Number of links on the switch.
15:8	RO		Number of cores that are connected to this switch.
7:0	RO		Number of links per processor.

D.3 Switch configuration: 0x04

This register enables the setting of two security modes (that disable updates to the PLL or any other registers) and the header-mode.

0x04: Switch configuration

Bits	Perm	Init	Description
31	RO	0	Set to 1 to disable any write access to the configuration registers in this switch.
30:9	RO	-	Reserved
8	RO	0	Set to 1 to disable updates to the PLL configuration register.
7:1	RO	-	Reserved
0	RO	0	Header mode. Set to 1 to enable 1-byte headers. This must be performed on all nodes in the system.

D.4 Switch node identifier: 0x05

This register contains the node identifier.

0x05: Switch node identifier

Bits	Perm	Init	Description
31:16	RO	-	Reserved
15:0	RW	0	The unique 16-bit ID of this node. This ID is matched most-significant-bit first with incoming messages for routing purposes.

D.5 PLL settings: 0x06

An on-chip PLL multiplies the input clock up to a higher frequency clock, used to clock the I/O, processor, and switch, see Oscillator. Note: a write to this register will cause the tile to be reset.

0x06: PLL settings

Bits	Perm	Init	Description
31:26	RO	-	Reserved
25:23	RW		OD: Output divider value The initial value depends on pins MODE0 and MODE1.
22:21	RO	-	Reserved
20:8	RW		F: Feedback multiplication ratio The initial value depends on pins MODE0 and MODE1.
7	RO	-	Reserved
6:0	RW		R: Oscilator input divider value The initial value depends on pins MODE0 and MODE1.

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	Switch clock divider. The PLL clock will be divided by this value plus one to derive the switch clock.

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	Architecture reference clock divider. The PLL clock will be divided by this value plus one to derive the 100 MHz reference clock.

D.8 Directions 0-7: 0x0C

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

D.13 PLink status and network: 0x40 .. 0x43

These registers contain status information and the network number that each processor-link belongs to.

Bits	Perm	Init	Description
31:26	RO	-	Reserved
25:24	RO		If this link is currently routing data into the switch, this field specifies the type of link that the data is routed to: 0: plink 1: external link 2: internal control link
23:16	RO	0	If the link is routing data into the switch, this field specifies the destination link number to which all tokens are sent.
15:6	RO	-	Reserved
5:4	RW	0	Determines the network to which this link belongs, set for quality of service.
3	RO	-	Reserved
2	RO	0	Set to 1 if the current packet is junk and being thrown away. A packet is considered junk if, for example, it is not routable.
1	RO	0	Set to 1 if the switch is routing data into the link, and if a route exists from another link.
0	RO	0	Set to 1 if the link is routing data into the switch, and if a route is created to another link on the switch.

0x40 .. 0x43: PLink status and network

D.14 Link configuration and initialization: 0x80 .. 0x87

These registers contain configuration and debugging information specific to external links. The link speed and width can be set, the link can be initialized, and the link status can be monitored. The registers control links C, D, A, B, G, H, E, and F in that order.

- ☐ 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 6. If you have a choice between two values, choose the value with the highest multiplier ratio since that will boot faster.

H.5 USB ULPI Mode

This section can be skipped if you do not have an external USB PHY.

- ☐ If using ULPI, the ULPI signals are connected to specific ports as shown in Section E.
- ☐ If using ULPI, the ports that are used internally are not connected, see Section E. (Note that this limitation only applies when the ULPI is enabled, they can still be used before or after the ULPI is being used.)

H.6 Boot

- ☐ The device is connected to a SPI flash for booting, connected to X0D0, X0D01, X0D10, and X0D11 (Section 7). If not, you must boot the device through OTP or JTAG.
- ☐ The device that is connected to flash has both MODE2 and MODE3 connected to pin 3 on the xSYS Header (MSEL). If no debug adapter connection is supported (not recommended) MODE2 and MODE3 are to be left NC (Section 7).
- ☐ The SPI flash that you have chosen is supported by **xflash**, or you have created a specification file for it.

H.7 JTAG, XScope, and debugging

- ☐ You have decided as to whether you need an XSYS header or not (Section G)
- ☐ If you included an XSYS header, you connected pin 3 to any MODE2/MODE3 pin that would otherwise be NC (Section G).
- ☐ If you have not included an XSYS header, you have devised a method to program the SPI-flash or OTP (Section G).

H.8 GPIO

- ☐ You have not mapped both inputs and outputs to the same multi-bit port.