*

Application Note 217

Using a CT-R4F with the RealView™ Emulation Baseboard

Document number: ARM DAI 0217A
Issued: April 2009

ARM

Application Note 217
Using a CT-R4F with EB

Release information

The following changes have been made to this Application Note.

Change history

Date	Issue	Change
15 April 2009	A.01	First release

Proprietary notice

Confidentiality status

This document is Open Access. This document has no restriction on distribution.

Feedback on this Application Note

If you have any comments on this Application Note, please send email to errata@arm.com giving:

- the document title
- the document number
- the page number(s) to which your comments refer
- an explanation of your comments.

General suggestions for additions and improvements are also welcome.

ARM web address

http://www.arm.com

Table of Contents

1 Introduction

1.1 Purpose of this application note

This application note covers the operation of the Emulation Baseboard (EB) with a CT-R4F. It examines the contents of the baseboard FPGA, the system interconnect, the clock structure, and specifics of the programmer's model directly relevant to Core Tile operation.

After reading this Application Note the user should be in a position to make changes to the provided baseboard FPGA design, add their own AHB or AXI based peripherals to it, or debug and analyze the operation of the provided images.

1.2 EB and CT-R4F Tile overview

This application note is designed for a CT-R4F Tile fitted to Tile Site 1 on the EB as shown in **Figure 1-1 Core Tile and EB system**. This application note only works with CT-R4F. The FPGA image for the baseboard is provided as part of the Versatile CD installation.

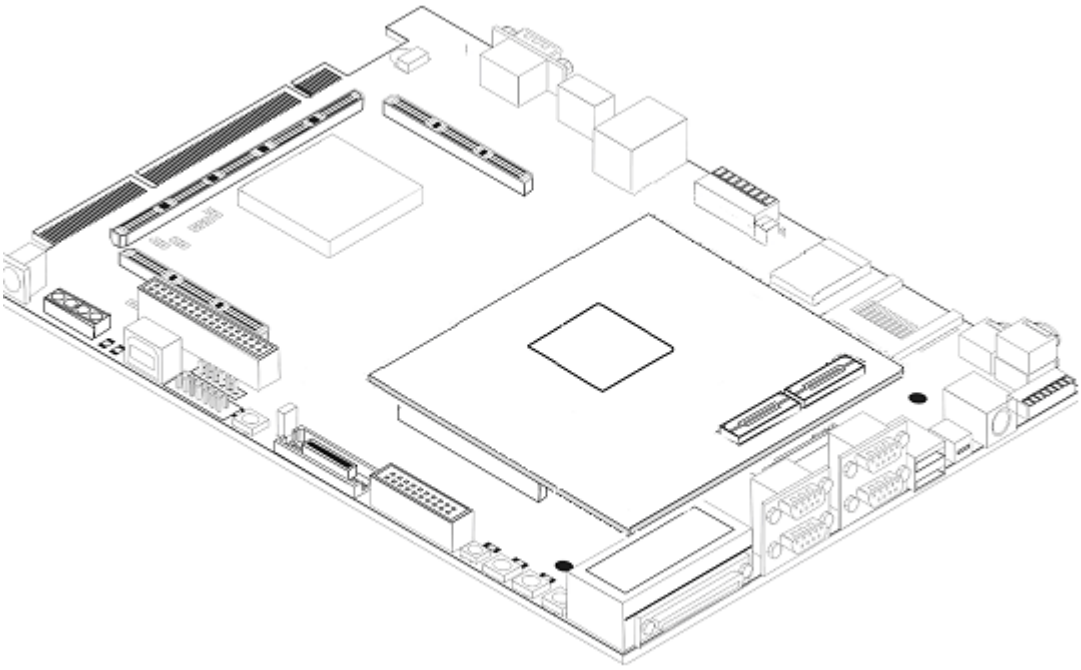


Figure 1-1 Core Tile and EB system

1.3 Optional Logic Tile

One or more logic tiles can be fitted to Tile Site 2. These Logic Tiles can contain both additional masters and slaves. See application note AN151 for an example and information about how to do this.

2 Getting started

Before you can use this application note, you will need to program the EB with the required FPGA image to enable the CT-R4F to function correctly. Follow these steps to program the FPGA image.

1. Plug the CT-R4F Tile onto TILE SITE 1 of the Emulation Baseboard.
2. Slide the CONFIG switch (S1) to the ON position.
3. Connect RVI to the Emulation Baseboard JTAG ICE connector (J18), or a USB cable to the USB Debug Port (J16).
4. Check the external supply voltage is +12V (positive on center pin, +/-10%, 35W), and connect it to the power connector (J28).
5. Power-up the boards. The '3V3 OK' LED and '5V OK' on the Emulation Baseboard should both be lit.
6. If using the USB connection, ensure that your PC has correctly identified an ARM® RealView™ ICE Micro Edition device is connected to the USB port. If the Windows operating system requires a USB driver to be installed please refer to the EB or PB926 \boardfiles\USB_Debug_driver\readme.txt.
7. If using Real View ICE (RVI), you must ensure that the RVI unit is powered and has completed its start-up sequence (check the LEDs on the front panel have stopped flashing).

8. You can now run the relevant 'progcards' utility for the connection you have prepared above.
- progcards_usb.exe for your USB connection
- progcards_rvi.exe for your RealView ICE connection

When using RVI select the target RVI box you are using.

9. Select the option for CT-R4F. The utility will report its progress. It may take several minutes to download. A successful configuration download will be terminated with the message "Programming Successful".
10. Power off the boards.
11. Set the configuration switches to load FPGA image 0. (S10 on the Emulation Baseboard set to all OFF).
12. Slide the CONFIG switch to the OFF position, and power on the boards. Ensure GLOBAL_DONE (D35) and the 'POWER' (D1) LEDs are lit. The Character LCD should show the Firmware and Hardware versions indicating that the Boot monitor firmware is running.
13. The system will now be fully configured and ready for use.

For configuring RVI for CT-R4F and AN217 debugging please refer to Using_CT-R4F_with_RVI.pdf file.

3 Architecture

This application note implements an AXI (AMBA 3.0) based system on the EB FPGA. The EB image exposes two master ports and two slave ports (all 64 bit muxed AXI). There are one slave port and one master routed to tile site1 and one slave and one master port routed to tile site 2.

Note that the direction of the arrows indicates the direction of control: it points from the Master to the Slave. An AXI bus contains signals going in both directions.

The PCI core is not included in the build as it is third party IP, it is an optional component.

3.1 System overview

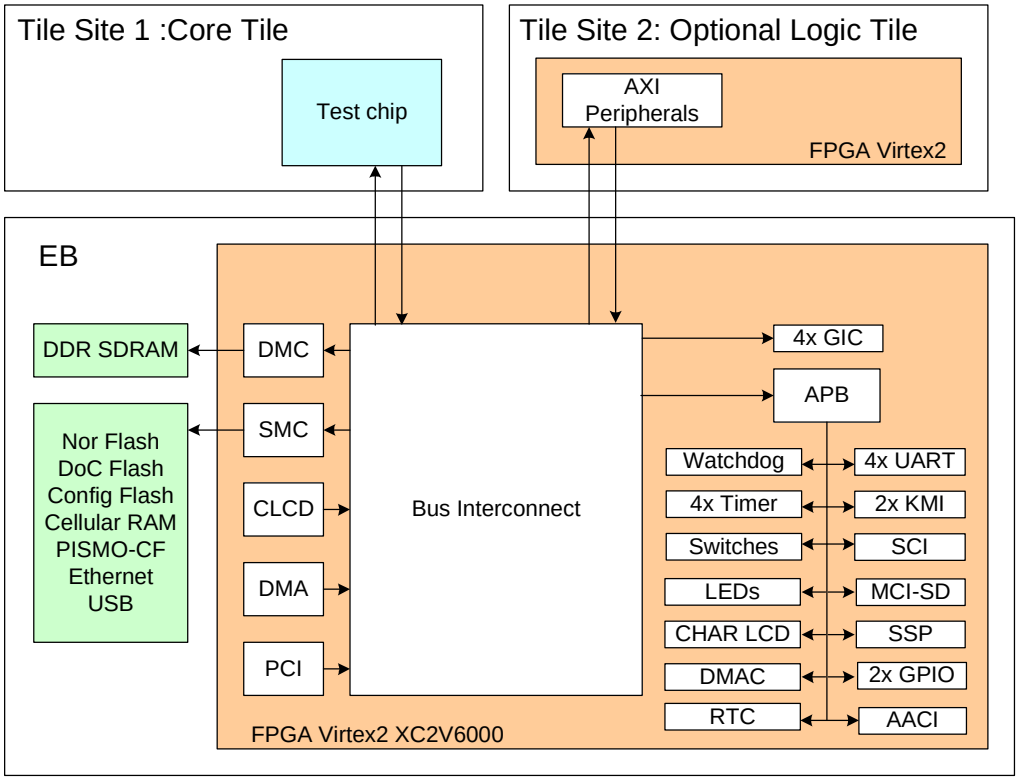


Figure 3-1 System interconnect

3.2 System architecture

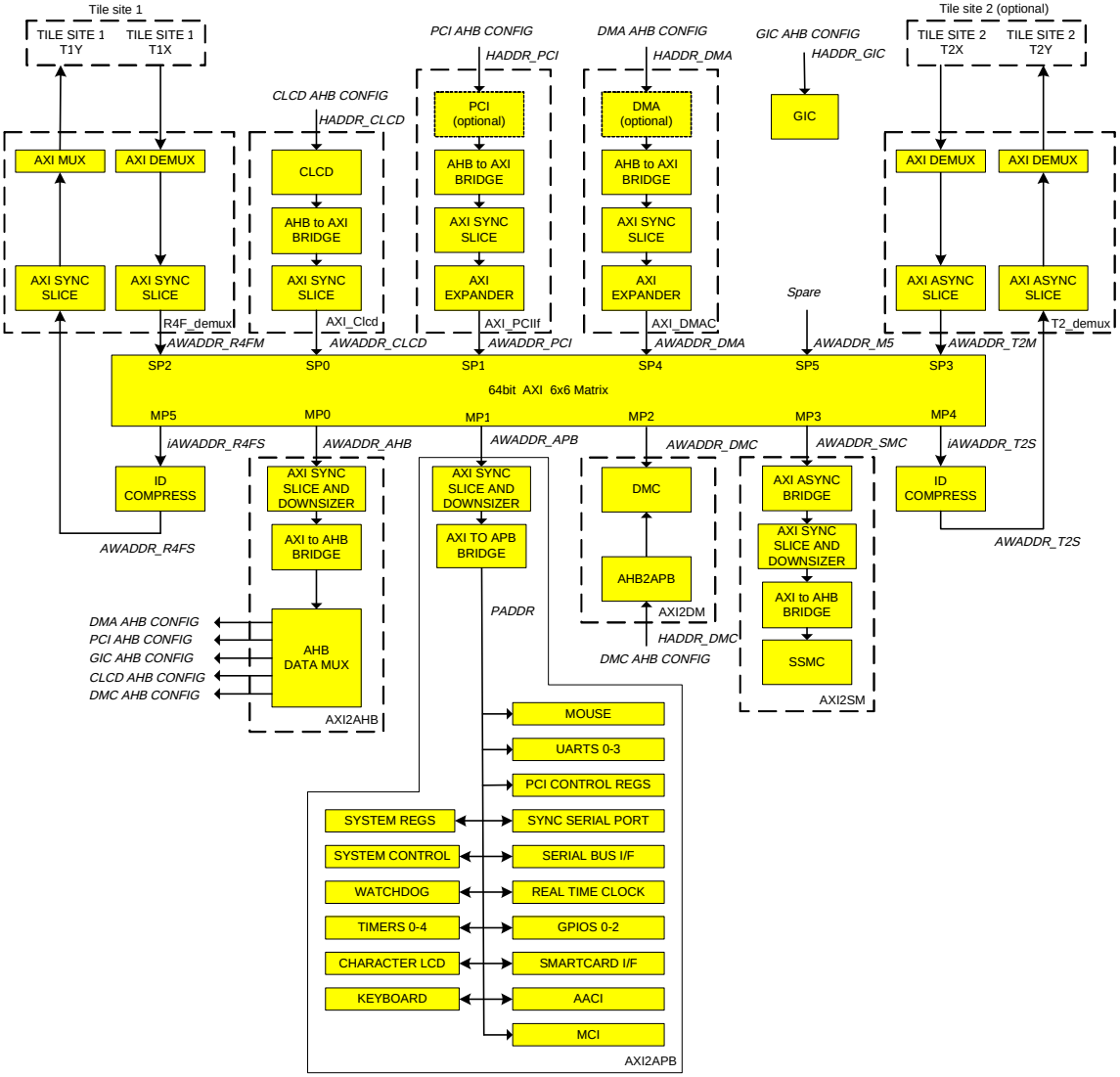


Figure 3-2 AN217 example design block diagram

3.2.1 EB Module functionality

The function of each of these blocks is as follows:

AXI SYNC SLICE

This is a synchronous register slice that is added to the incoming master ports R4FM, CLCD, PCI, and DMAC on the bus matrix to ensure that data is available for a complete cycle prior to AXI matrix address decoding. It also helps to increase the operating frequency.

AXI ASYNC SLICE

This is an asynchronous bridge that is added to the incoming master port from tile site 2 to ensure that data is available for a complete cycle prior to AXI matrix address decoding and allows that site to operate in a different clock domain. It increases the operating frequency and introduces one clock cycle of latency.

AXI DEMUX

This block demultiplexes the multiplexes signals from the master ports on tile site 1 and 2. The muxing scheme is described in a section **Error! Reference source not found.**.

ID COMPRESS

This block merges the ID fields on the AXI bus to reduce the width of each ID channel. The ID is compressed using a simple algorithm to save pins.

AXI MUX

This block multiplexes the signals from the master port Mp4. The muxing scheme is described in a section Error! Reference source not found. Error! Reference source not found..

64-bit AXI 6x6 Bus Matrix

This provides the bulk of the interconnect structure. It allows any of the 6 slave ports to connect to any of the 6 master ports without blocking the other masters. It also contains the decoder mapping to determine the address map, and a scheme to determine priority of competing masters to a single slave.

SSMC

This is a Synchronous Static Memory Controller. ARM PrimeCell PL093 is used in this design. For more information please refer to the PrimeCell documentation.

AHBtoAXI BRIDGE

This is a bridge component to change from an AHB bus to an AXI bus. Where nn is the input width and mm is the output width in decimal.

DMC

This is a Dynamic Memory Controller. ARM PrimeCell PL340 is used in this design. For more information please refer to the PrimeCell documentation. The default frequency for the DMC is 55MHz (OSCCLK1 divided by 2)

Since the Dynamic Memory Controller connects to high speed double data rate clocked devices it is necessary to make use of special pad and signal types to and from the Dynamic Memory. This block instantiated within the DMC

AXItoAPB BRIDGE

This is a bridge component to change from an AXI bus to an APB3 bus. This component contains decoding scheme for the bus, allowing 25 APB peripherals to be connected.

DMA

This is a direct memory access component. This design allows for the PrimeCell PL081 controllers to be added. It also allows for no DMA. For more information on the DMA blocks refer to the PrimeCell documentation.

PCI

The ARM provided image includes the option of including a Xilinx PCI component (version 3.0). This is a 32-bit wide PCI component, and operates at 33MHz. Refer to the EB Users Guide for further information.

CLCD

This is a color liquid crystal display controller. ARM PrimeCell PL111 is used in this design. For more information please refer to the PrimeCell documentation.

SYSTEM REGS

This contains a set of APB registers for hardware control of the EB. For a complete list of the functionality of these registers refer to the EB user guide.

SYSTEM CONTROL

This contains a generic set of system control registers. ARM ADK component SP810 is used in this design.

SERIAL BUS I/F

This is a controller for the serial bus to the PISMO and Time of Year clock.

AACI

This is an advanced audio codec interface. ARM PrimeCell PL041 is used in this design. The FIFO is increased in size from the standard to better suit the FPGA operating environment (lower bus frequency). For more information please refer to the PrimeCell documentation. This block instantiates Xilinx block rams.

MCI

This is the multimedia card interface. ARM PrimeCell PL181 is used in this design. For more information please refer to the PrimeCell documentation.

MOUSE

These are keyboard and mouse interfaces. ARM PrimeCell PL050 is used in this design. For more information please refer to the PrimeCell documentation.

CHARACTER LCD

This is a character LCD display interface. It allows the system to communicate with the character LCD display fitted to the baseboard.

UART0-3

These are universal asynchronous receiver-transmitter interfaces (RS-232 serial). ARM PrimeCell PL010 is used in this design. For more information please refer to the PrimeCell documentation.

SSP

This is a synchronous serial port. ARM PrimeCell PL022 is used in this design. For more information please refer to the PrimeCell documentation. This block instantiates Xilinx block rams.

SCI

This is the smart card interface. ARM PrimeCell PL131 is used in this design. For more information please refer to the PrimeCell documentation.

WATCHDOG

This is a watchdog controller. It allows for the generation of an interrupt or reset after a defined time, to prevent against system lockup/failure. ARM ADK component SP805 is used in this design.

TIMERS 0-3

These are timer modules. ARM ADK component SP804 is used in this design.

GPIO 0-2

These are general purpose input/output modules. ARM PrimeCell PL061 is used in this design. For more information please refer to the PrimeCell documentation.

REAL TIME CLOCK

This is a real time clock module. Real time refers to total time from an event, and not actual real world time (measured using the Time Of Year Clock). ARM PrimeCell PL031 is used in this design. For more information please refer to the PrimeCell documentation.

PCI CONTROL REGS

This is the PCI control block. This contains a set of registers to allow configuration of the PCI system (if used). See the programmer's reference section for more information on the functionality of these registers.

3.3 Clock architecture

The clock architecture is carefully designed to minimize the skew (difference) in the clock edge position between different components across the system. The User Guides for all the boards used in this configuration explain the clock options.

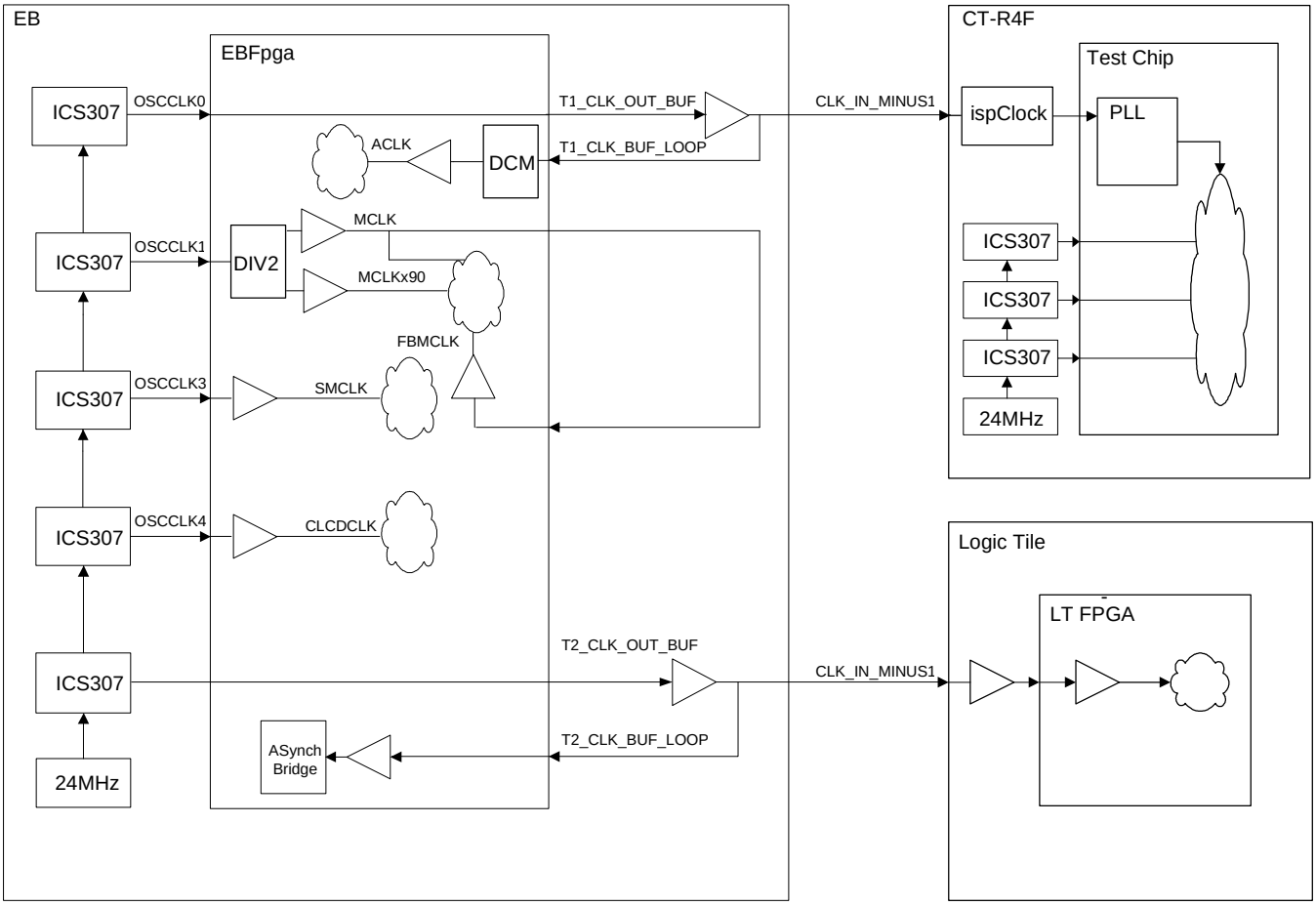


Figure 3-3 Clock architecture

There are 6 clock domains in this design:

3.3.1 EB Internal AXI System Bus and CPU External System Bus

The EB Internal AXI System Bus is clocked by ACLK. ACLK is generated from T1_CLK_BUF_LOOP through a clock buffer from OSCCLK0.

The CPU External System Bus is clocked at the same speed as the EB Internal AXI System by CLK_IN_MINUS1 input clock also generated from OSCCLK0 through a clock buffer.

3.3.2 DDR_CLK

OSCCLK1 is the source for all the DMC clocks: MCLK, MCLK90, and FBMCLK. It is divided by 2 to delivery MCLK and divided by 2 and half cycle deleted to delivery MCLK90 clock.

3.3.3 LT System Bus

The LT External System Bus is clocked by OSCCLK2.

The LT Internal System Bus in this diagram is the same frequency as the LT External System Bus. (LT Internal System Bus = LT External).

3.3.4 EB Internal and External Synchronous Static Memory Bus

The SMCLK is used to drive Synchronous Static Memory Controller and Static Memory bus peripherals. The SMCLK is directly connected to OSCCLK3.

3.3.5 CLCDCLK

CLCDCLK is directly connected to OSCCLK4.

3.3.6 CPU Internal Buses

The CPU Internal AXI System Bus clocks domains includes the R4F CPU domain, the DMA Controller, internal matrix domain, DMC domain and CLCD domain. The default behavior is for the ACLK PLL in the test chip to multiply the Core Tile input clock by 5 to drive R4F CPU clock domain. The DMA Controller and internal matrix domain is by default clocked by half of frequency of R4F CPU clock domain. The MCLK PLL in the test chip is used only to minimize the skew for DMC clock domain.

3.3.7 Default operating frequencies

Clock	Use	Default Frequency
OSCCLK0	Tile site 1 R4F external AXI bus clock and ACLK	38MHz
OSCCLK1	DMC (DDR clock divided by 2)	110MHz
OSCCLK2	Tile site 2 (Logic Tile)	25MHz
OSCCLK3	SMC clock	33MHz
OSCCLK4	CLCD	25MHz
CT_R4F_OSC0	R4F CPU clock (multiply by 5 by default) and TC DMA, internal matrix (multiply by 2.5 by default)	50MHz
CT_R4F_OSC1	CT-R4F SDRAM clock (multiply by 2 by ispClock chip)	67MHz
CT_R4F_OSC2	CT-R4F CLCD clock	25MHz

Table 3-1 Default operating frequencies

3.3.8 EB Clock Configuration Switch

A switch SW8.5 is used to select EB dynamic memory clock and static memory bus clock frequency to make it easy to set the clock for most customers.

SW8.5 – sets the DMC clock frequency by changing OSCCLK1 and SMC clock frequency by changing OSCCLK3.

SW8.5	Clock Setting
OFF	DMC (DDR clock is 55MHz) 110MHz OSCCLK3 (SMC clock) 33MHz
ON	DMC (DDR clock is 50) 100MHz OSCCLK3 (SMC clock) 25MHz

Table 3-2 Clock selection on EB

This switch is only sampled at the initial power up, and it is still possible to set any clock frequency in software using the EB SYS_OSCRESETx registers.

3.4 CT-R4F reset structure

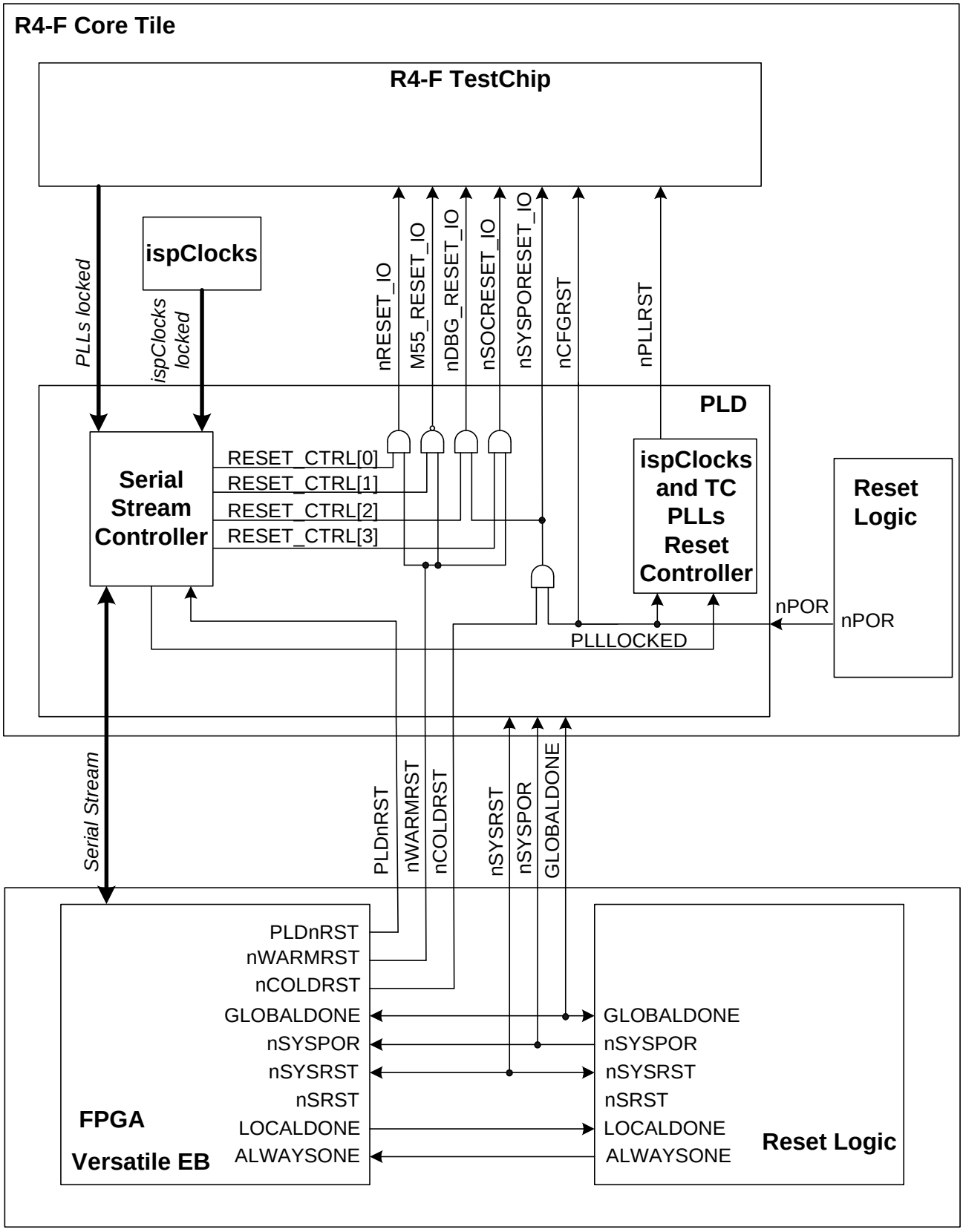


Figure 3-4 CT-R4F reset routing

Name	Direction to/from Reset Logic	Note
LOCALDONE	Output	Goes high when the EB FPGA has configured
GLOBAL_DONE	Input open drain	Goes high when all FPGAs have configured and the serial PLD interfaces have configured the TestChip PLL and Tile Mux/Switch
nSYSPOR	Output	Goes high approximately 7µs after GLOBAL_DONE is high
nSYSRST	Output	Goes high approximately 20µs after nSYSPOR goes high or goes low after nSRST goes low and returns high approximately 20us after nSRST goes high
PLDnRST	Output	Resets and starts the PLD serial data transfer
nSRST	Input open drain	Generates an nSYSRST request
nWARMRST	Output	Connected to nSYSRST in EB FPGA
nCOLDRST	Output	Goes high approximately 2.4µs after nSYSPOR and 9.4us after GLOBAL_DONE is high.

Table 3-3 CT-R4F Reset Signals

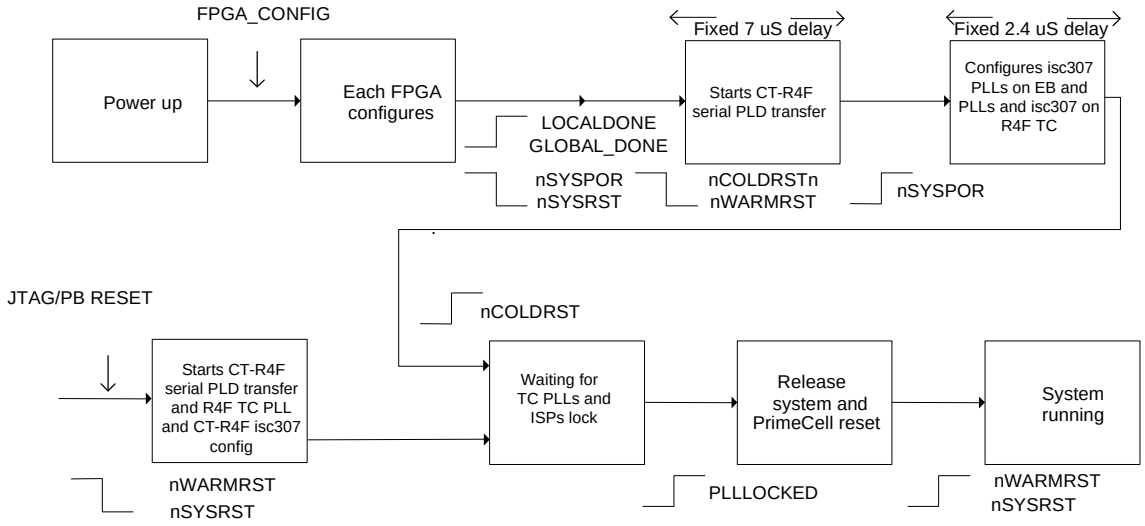


Figure 3-5 Reset sequence

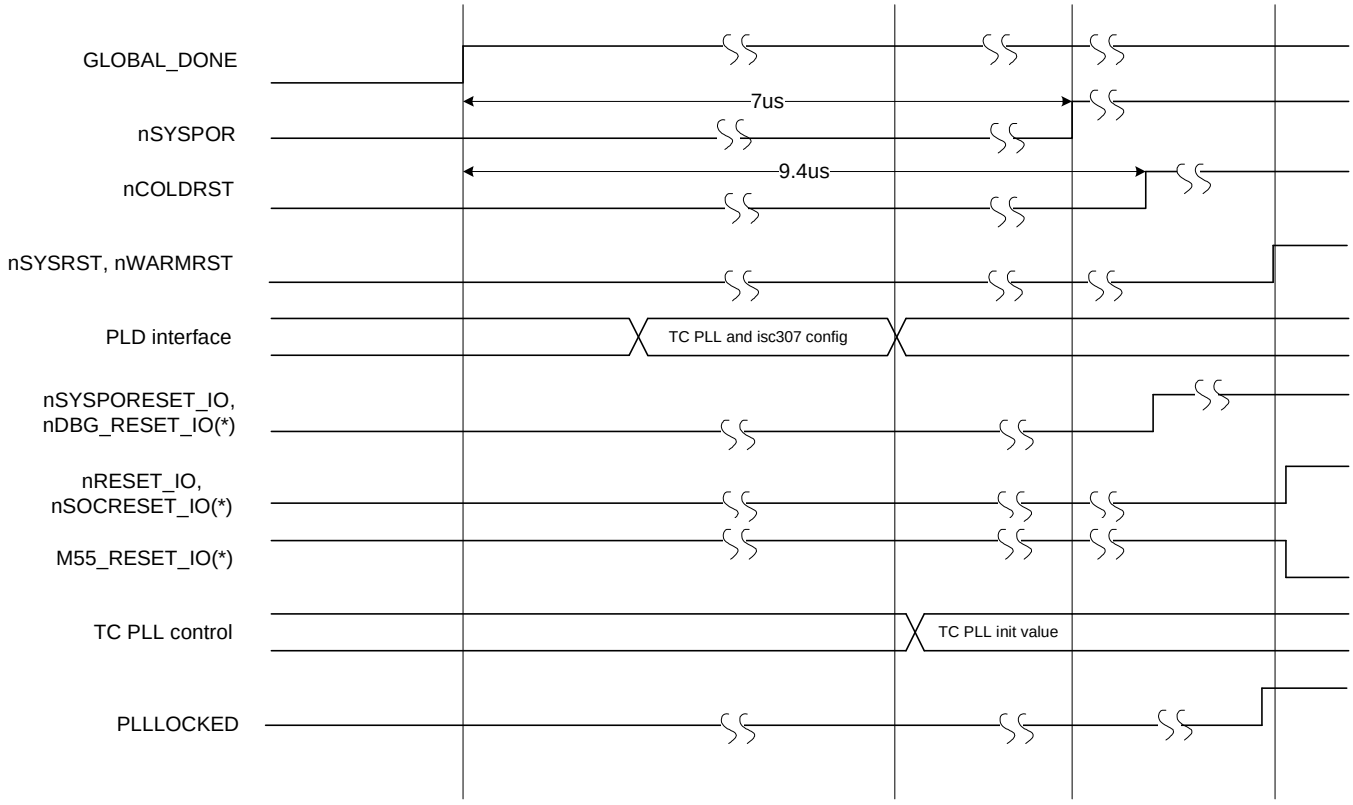
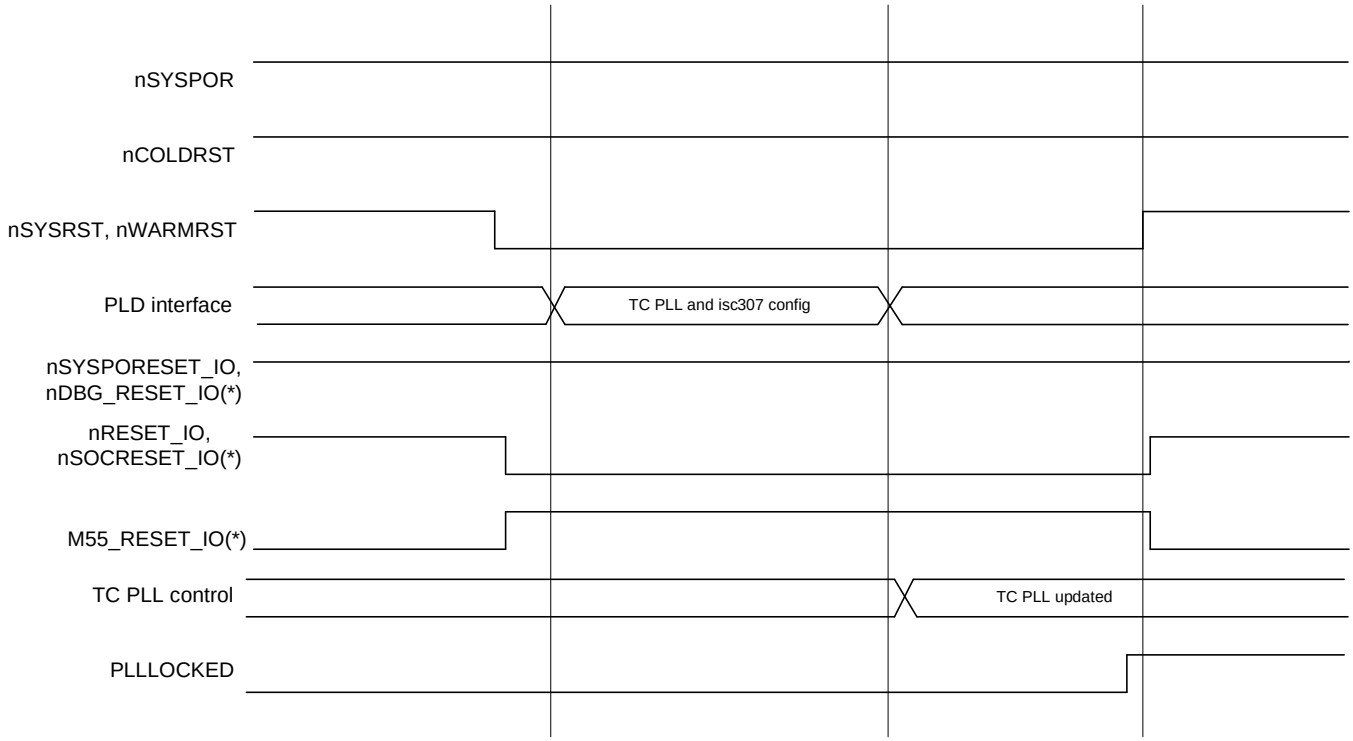


Figure 3-6 Power-on reset timing



* RESET_CTRL[3:0] = 4'b1111

Figure 3-7 Reset timing

3.5 Interrupt architecture

The interrupt scheme makes use of multiple Interrupt Controllers to facilitate the connection of an IRQ and FIQ to both tile sites, which would allow the connection of a second Core Tile on the EB tile site 2. Both tile sites can generate sources of interrupts and receive interrupts generated by the interrupt controllers.

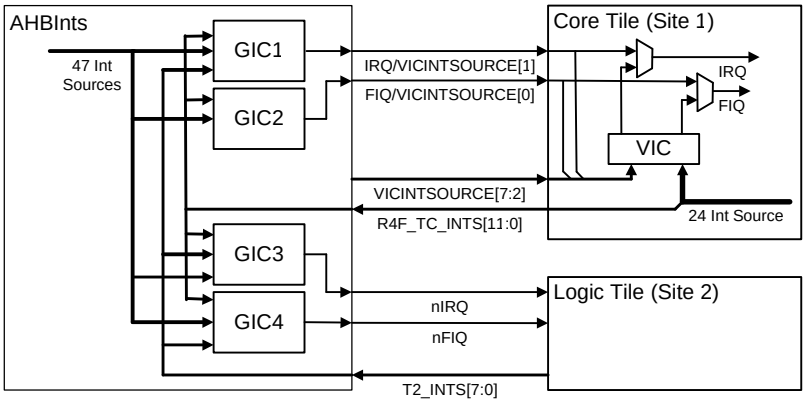


Figure 3-8 Interrupt architecture

A GIC (Generic Interrupt Controller) is chosen for this design, as it accepts a large number of interrupt sources without cascading interrupt controllers.

The all of interrupts are common to all four GICs.

A VIC (Vectored Interrupt Controller) implemented in R4F TC is by default bypassed and R4F processor IRQ and FIQ inputs are driven directly by EB GICs.

For a mapping of the 47 interrupt sources onto the 64-bit GIC interrupt input refer to the Emulation Baseboard User Guide. The memory mapping of the GICs is also shown in the User Guide.

4 Hardware Description

4.1 EB Top Level (EBFpga.v)

The top level of the design is of particular importance for a number of reasons. This level defined the clock structure, tri-state buffers and pull-up and pull-down resistors.

4.2 EB AXI Subsystem (EBFpgaCTR4F.v)

This level connects all the components together and ties off static pins. This includes all the major blocks as shown in **Figure 3-1 System interconnect**.

4.3 EB AXI bus multiplexing (R4F_demux.v, T2_demux.v)

This level contains multiplexing and demultiplexing modules and defines the mapping from the HDRX, HDRY and HDRZ busses from the tile site into their functional allocations. The R4F_demux.v contains multiplexing and serial stream controller for CT-R4F and the T2_demux.v contains multiplexing for Tile 2.

By using a 2:1 multiplexer and registers scheme as shown below on **Figure 4-1 AXI multiplexing scheme** it is possible to reduce the pin count for the AXI bus into a realistic size for implementation on the only Tile X header.

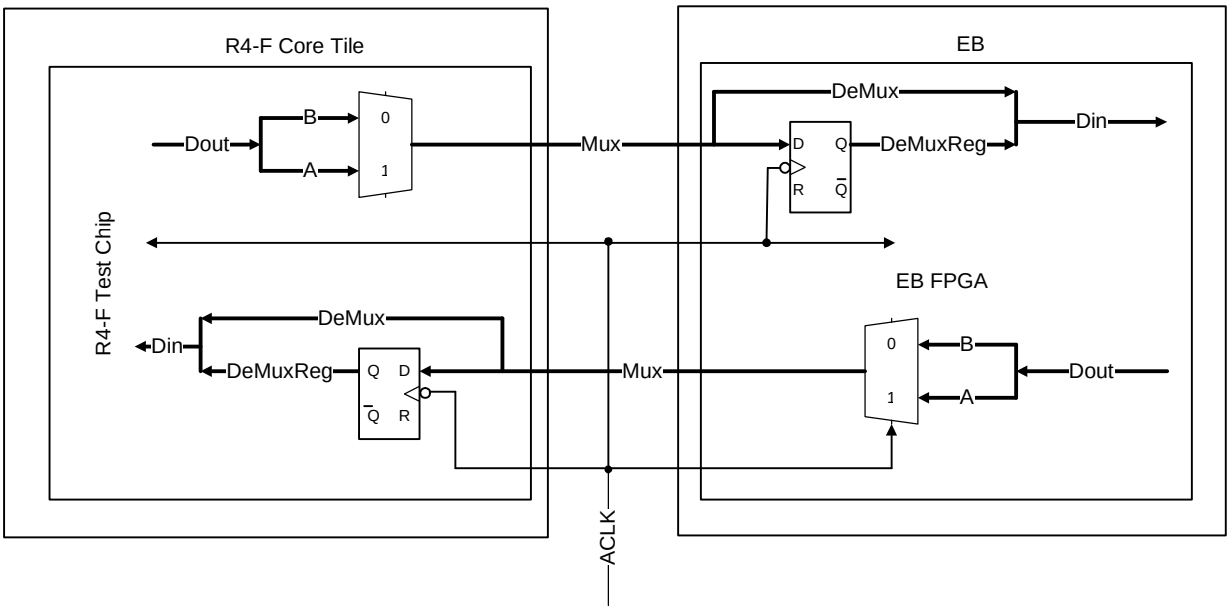


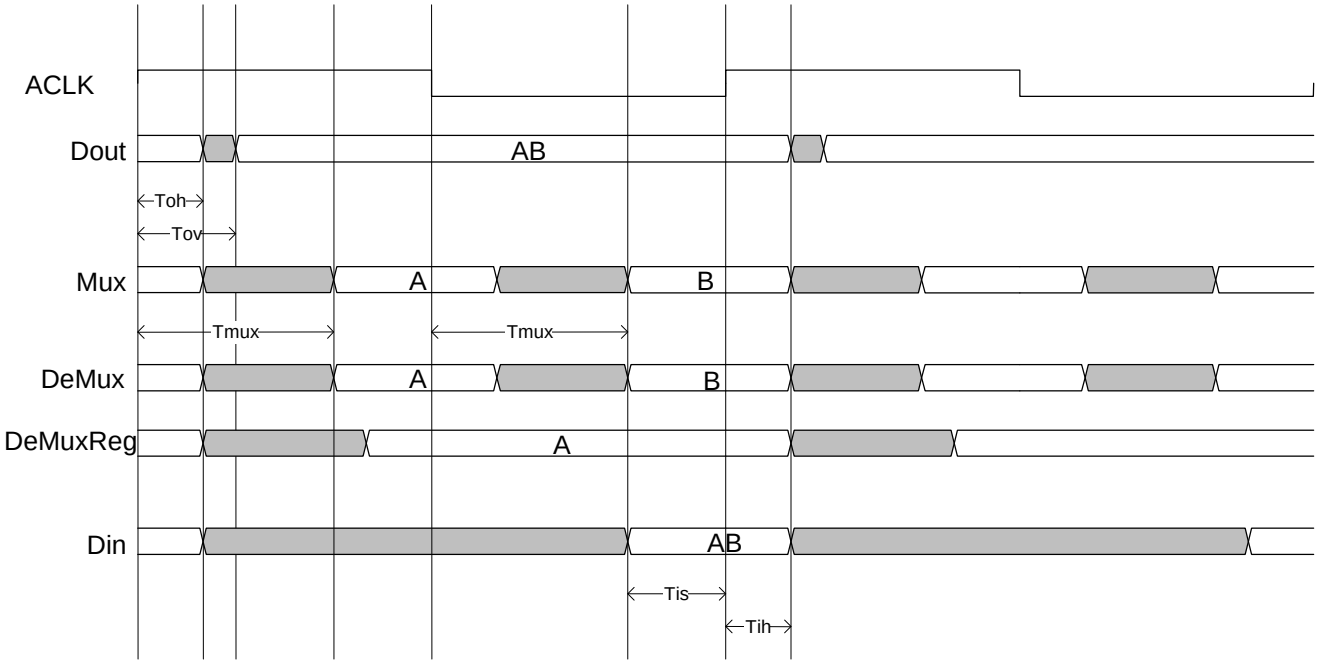
Figure 4-1 AXI multiplexing scheme

The output data is multiplexed on the level of ACLK, to generate the multiplexed bus (X and Y). The de-multiplexing is performed by registering the data (A) generated on the high level of CLK when CLK goes low (DeMuxReg). The data (B), generated on the low side of CLK is passed straight. This design assumes data is always generated and captured on the rising edge of CLK.

The READY signals on AXI can not be multiplexed in this way due to their timing requirements and must be passed directly between devices.

The CT-R4F has implemented the multiplexing and de-multiplexing logic within the R4F TC. The EB baseboard has implemented the multiplexing and de-multiplexing logic within the FPGA using DDR registers.

The following timing diagram (see **Figure 4-2 AXI timing requirements**) shows the data flow from FPGA to TC through the design with expected delays.



Timing requirements ;

Toh	min = 1ns	(output hold)
Tov	max = 1.3ns	(output valid)
Tis	max = 4ns	(Input setup)
Tih	max = 1ns	(input hold)
Tmux	max = 3ns	(multiplexing and board delay)

The ACLK is the clock driven to the CT-R4F TC.All I/O timing must be with respect to this clock .

Figure 4-2 AXI timing requirements

4.4 CT-R4F and AXI (AMBA 3) pin allocation differences

A number of signals from the R4F test chip differ from the generic Tile header pin allocation. These signals will be connected to the Z headers along with serial PLD control signals as show on **Table 4-1 Z bus signals**.

Z Bus	HDRZ	CT-R4F	Dir	Function
200	112	FIQ/VICINTSOURCE[0]	I	Testchip External fast interrupt input – asynchronous / Testchip GIC External interrupt input
201	110	IRQ/VICINTSOURCE[1]	I	Testchip External interrupt input – asynchronous / Testchip GIC External interrupt input
202	108	VICINTSOURCE[2]	I	Testchip GIC External interrupt input
203	106	VICINTSOURCE[3]	I	Testchip GIC External interrupt input
204	104	VICINTSOURCE[4]	I	Testchip GIC External interrupt input
205	102	VICINTSOURCE[5]	I	Testchip GIC External interrupt input
206	100	VICINTSOURCE[6]	I	Testchip GIC External interrupt input
207	98	VICINTSOURCE[7]	I	Testchip GIC External interrupt input
208	96	CT_nFIQExt	O	Testchip GIC External fast interrupt output – asynchronous.
209	94	CT_nIRQExt	O	Testchip GIC External interrupt output – asynchronous.
210	92	Reserved	O	Reserved
211	90	Reserved	O	Reserved
212	88	CT_CLCDIRQ	O	Testchip CLCD interrupt
213	86	CT_DMAC_IRQ[0]	O	Testchip DMAC interrupt
214	84	CT_DMAC_IRQ[1]	O	Testchip DMAC interrupt
215	82	CT_DMAC_IRQ[2]	O	Testchip DMAC interrupt
216	80	nSYSPOR	I	Power-on reset to clock and reset control logic – asynchronous
217	78	CT_COMMRX	I	Additional testchip login input. Passed through PLD.
218	76	CT_COMMTX	I	Additional testchip login input. Passed through PLD.
219	74	CT_DMAC_IRQ[3]	I	Testchip DMAC interrupt
220	72	Reserved	O	Reserved
221	70	Reserved	O	Reserved
222	68	Reserved	O	Reserved
223	66	Spare[0]	I/O	Reserved
224	64	Spare[1]	I/O	Reserved
225	62	Spare[2]	I/O	Reserved
226	60	Spare[3]	I/O	Reserved
227	58	PLDD1	I	Serial PLD interface data input
228	56	PLDD0	O	Serial PLD interface data output
229	54	PLDRESETn	I	Serial PLD interface reset
230	52	PLDCLK	I	Serial PLD interface clock
231	50	nSYSRST	I	Asynchronous RESET input. Active LOW.

Table 4-1 Z bus signals

4.5 Header HDRX and HDRY AXI pin allocation

The four AXI buses (T1X, T1Y, T2X, T2Y) connect to the HDRX and HDRY Tile headers. The pin connections are the same for all four buses. T1X is shown as an example on **Table 4-2 Header HDRX and HDRY AXI pin allocation**.

X/Y Bus	HDRX pin	HDRY pin	signal	X/Y Bus	HDRX pin	HDRY pin	signal
0	180	179	WDATA0/32	48	84	83	AWADDR7/23
1	178	177	WDATA1/33	49	82	81	AWADDR8/24
2	176	175	WDATA2/34	50	80	79	AWADDR9/25
3	174	173	WDATA3/35	51	78	77	AWADDR10/26
4	172	171	WDATA4/36	52	76	75	AWADDR11/27
5	170	169	WDATA5/37	53	74	73	AWADDR12/28
6	168	167	WDATA6/38	54	72	71	AWADDR13/29
7	166	165	WDATA7/39	55	70	69	AWADDR14/30
8	164	163	WDATA8/40	56	68	67	AWADDR15/31
9	162	161	WDATA9/41	57	66	65	AWID0/2
10	160	159	WDATA10/42	58	64	63	AWID1/3
11	158	157	WDATA11/43	59	62	61	AWLEN0/2
12	156	155	WDATA12/44	60	60	59	AWLEN1/3
13	154	153	WDATA13/45	61	58	57	AWSIZE0/1
14	152	151	WDATA14/46	62	56	55	AWID4/AWPROT2
15	150	149	WDATA15/47	63	54	53	RESERVED
16	148	147	WDATA16/48	64	52	51	AWPROT0/1
17	146	145	WDATA17/49	65	50	49	AWBURST0/1
18	144	143	WDATA18/50	66	48	47	AWLOCK0/1
19	142	141	WDATA19/51	67	46	45	AWCACHE0/2
20	140	139	WDATA20/52	68	44	43	AWCACHE1/3
21	138	137	WDATA21/53	69	42	41	AWVALID/AWID5
22	136	135	WDATA22/54	70	40	39	AWREADY
23	134	133	WDATA23/55	71	38	37	BID0/2
24	132	131	WDATA24/56	72	36	35	BID1/3
25	130	129	WDATA25/57	73	34	33	BID4/BID5
26	128	127	WDATA26/58	74	32	31	BRESP0/1
27	126	125	WDATA27/59	75	30	29	BVALID
28	124	123	WDATA28/60	76	28	27	BREADY
29	122	121	WDATA29/61	77	26	25	ARADDR0/16
30	120	119	WDATA30/62	78	24	23	ARADDR1/17
31	118	117	WDATA31/63	79	22	21	ARADDR2/18
32	116	115	WID0/2	80	20	19	ARADDR3/19
33	114	113	WID1/3	81	18	17	ARADDR4/20
34	112	111	WSTRB0/4	82	16	15	ARADDR5/21
35	110	109	WSTRB1/5	83	14	13	ARADDR6/22
36	108	107	WSTRB2/6	84	12	11	ARADDR7/23
37	106	105	WSTRB3/7	85	10	9	ARADDR8/24
38	104	103	WLAST/WID4	86	8	7	ARADDR9/25
39	102	101	WVALID/WID5	87	6	5	ARADDR10/26
40	100	99	WREADY	88	4	3	ARADDR11/27
41	98	97	AWADDR0/16	89	2	1	ARADDR12/28
42	96	95	AWADDR1/17	90	1	2	ARADDR13/29
43	94	93	AWADDR2/18	91	3	4	ARADDR14/30
44	92	91	AWADDR3/19	92	5	6	ARADDR15/31
45	90	89	AWADDR4/20	93	7	8	ARID0/2
46	88	87	AWADDR5/21	94	9	10	ARID1/3
47	86	85	AWADDR6/22	95	11	12	ARLEN0/2

X/Y Bus	HDRX pin	HDRY pin	Signal	X/Y Bus	HDRX pin	HDRY pin	Signal
96	13	14	ARLEN1/3	120	61	62	RDATA14/46
97	15	16	ARSIZE0/1	121	63	64	RDATA15/47
98	17	18	ARID4/ARPROT2	122	65	66	RDATA16/48
99	19	20	ARPROT0/1	123	67	68	RDATA17/49
100	21	22	ARBURST0/1	124	69	70	RDATA18/50
101	23	24	ARLOCK0/1	125	71	72	RDATA19/51
102	25	26	ARCACHE0/2	126	73	74	RDATA20/52
103	27	28	ARCACHE1/3	127	75	76	RDATA21/53
104	29	30	ARVALID/ARID5	128	77	78	RDATA22/54
105	31	32	ARREADY	129	79	80	RDATA23/55
106	33	34	RDATA0/32	130	81	82	RDATA24/56
107	35	36	RDATA1/33	131	83	84	RDATA25/57
108	37	38	RDATA2/34	132	85	86	RDATA26/58
109	39	40	RDATA3/35	133	87	88	RDATA27/59
110	41	42	RDATA4/36	134	89	90	RDATA28/60
111	43	44	RDATA5/37	135	91	92	RDATA29/61
112	45	46	RDATA6/38	136	93	94	RDATA30/62
113	47	48	RDATA7/39	137	95	96	RDATA31/63
114	49	50	RDATA8/40	138	97	98	RID0/2
115	51	52	RDATA9/41	139	99	100	RID1/3
116	53	54	RDATA10/42	140	101	102	RRESP0/1
117	55	56	RDATA11/43	141	103	104	RLAST/RID4
118	57	58	RDATA12/44	142	105	106	RVALID/RID5
119	59	60	RDATA13/45	143	107	108	RREADY

Table 4-2 Header HDRX and HDRY AXI pin allocation

4.6 CT-R4F configuration PLD

The CT-R4F Tile contains a configuration PLD (see **Figure 4-3 CT-R4F configuration PLD**) which defines:

- R4F testchip initialization signals
- R4F testchip resets
- Profile and status of the phase control logic
- ADC control.

A serial data stream from the FPGA on the EB below is used to configure the PLD settings after power up.

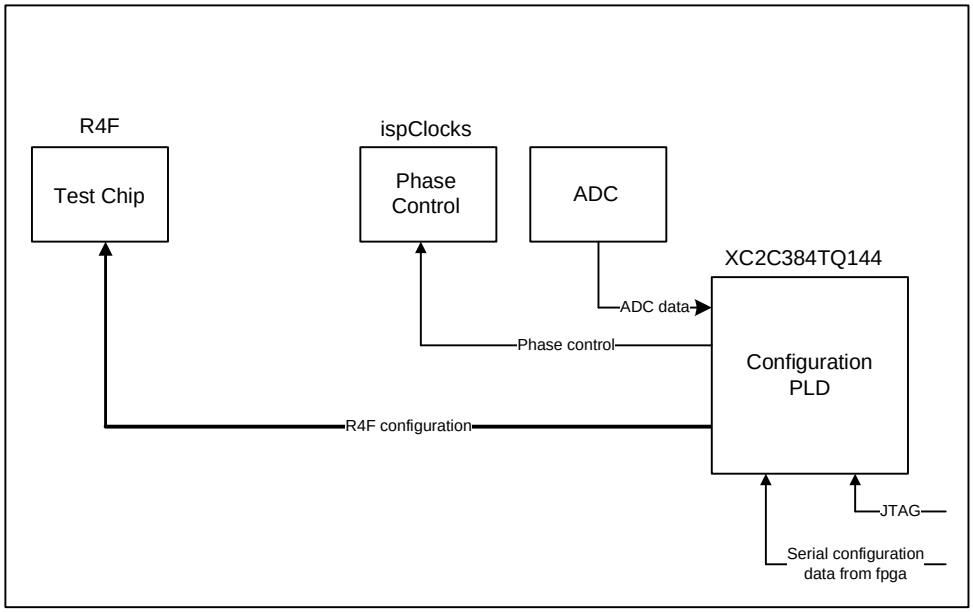


Figure 4-3 CT-R4F configuration PLD

4.7 CT-R4F configuration PLD serial interface

Serial control of the configuration PLD is achieved through a three wire interface (see **Figure 4-4 CT-R4F configuration PLD serial interface**). Reset signal PLDRESETn is also used to ensure the PLD logic is in the defined state before the data stream is loaded.

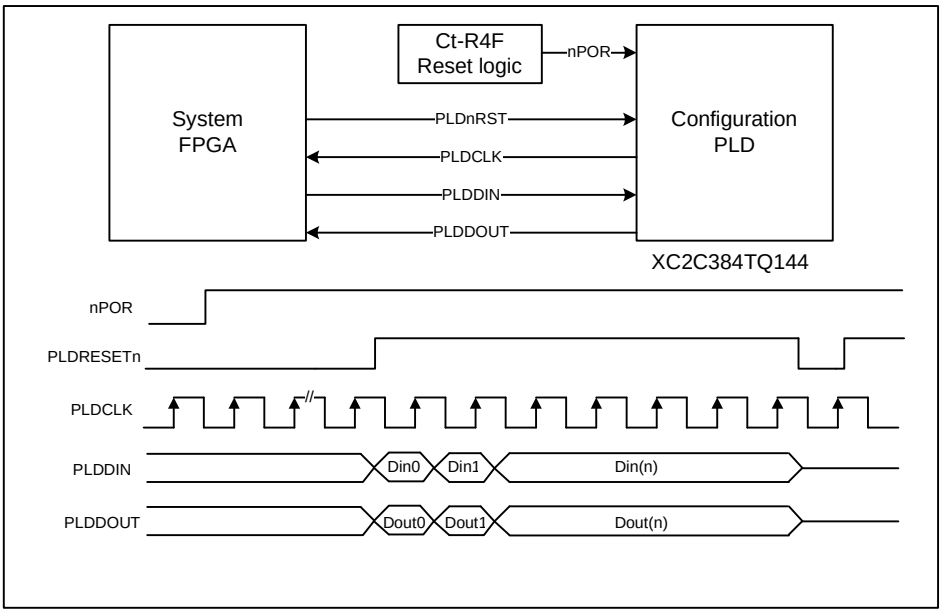


Figure 4-4 CT-R4F configuration PLD serial interface

All data bits are clocked into the PLD on the rising edge of PLDCLK.

4.8 Serial data register

The serial interface is designed to mimic a memory mapped peripheral in that data sent to PLD has an address and data phase. This solution allows for an optimized logic implementation in the PLD and speeds up access to TC serial stream via the PLD.

The first bit of the input serial data contains the direction of the transaction. The followed 13 bits contain the address and the next 32-bits the data to be written to that address. The PLD serial data out contains Board ID, PLLs lock signals and the data from the selected address.

Bit	Input Serial Signal Name	Output Serial Signal Name
0	Data Dir	BOARD_ID[0]
1	Address[12]	BOARD_ID[1]
2	Address[11]	BOARD_ID[2]
3	Address[10]	BOARD_ID[3]
4	Address[9]	BOARD_ID[4]
5	Address[8]	BOARD_ID[5]
6	Address[7]	BOARD_ID[6]
7	Address[6]	BOARD_ID[7]
8	Address[5]	BOARD_ID[8]
9	Address[4]	BOARD_ID[9]
10	Address[3]	TC PLL0 LOCK
11	Address[2]	TC PLL1 LOCK
12	Address[1]	TC PLL2 LOCK
13	Address[0]	Isp0 LOCKn
14	Data in[31]	isp1 LOCKn
15	Data in[30]	Isp2 LOCKn
16	Data in[29]	Data out[0]
17	Data in[28]	Data out[1]
18	Data in[27]	Data out[2]
19	Data in[26]	Data out[3]
20	Data in[25]	Data out[4]
21	Data in[24]	Data out[5]
22	Data in[23]	Data out[6]
23	Data in[22]	Data out[7]
24	Data in[21]	Data out[8]
25	Data in[20]	Data out[9]
26	Data in[19]	Data out[10]
27	Data in[18]	Data out[11]
28	Data in[17]	Data out[12]
29	Data in[16]	Data out[13]
30	Data in[15]	Data out[14]
31	Data in[14]	Data out[15]
32	Data in[13]	Data out[16]
33	Data in[12]	Data out[17]
34	Data in[11]	Data out[18]
35	Data in[10]	Data out[19]
36	Data in[9]	Data out[20]
37	Data in[8]	Data out[21]
38	Data in[7]	Data out[22]
39	Data in[6]	Data out[23]
40	Data in[5]	Data out[24]
41	Data in[4]	Data out[25]
42	Data in[3]	Data out[26]
43	Data in[2]	Data out[27]
44	Data in[1]	Data out[28]
45	Data in[0]	Data out[29]

46	Not used	Data out[30]
47	Not used	Data out[31]

Table 4-3 CT-R4F serial configuration interface data

Serial stream accessible registers:

Address	Name	Access	Details
0x0000	CT_R4F_TC_CFG0	W	CT-R4F TC config word 0
0x0004	CT_R4F_TC_CFG1	W	CT-R4F TC config word 1
0x0008	CT_R4F_TC_CFG2	W	CT-R4F TC config word 2
0x1000	CT_R4F_PLD_ID	R	CT-R4F PLD ID
0x1004	CT_R4F_OSC0	W	CT-R4F ACLK ICS307 oscillator chip
0x1005	CT_R4F_OSC1	W	CT-R4F SDRAM ICS307 oscillator chip
0x1006	CT_R4F_OSC2	W	CT-R4F CLCD ICS307 oscillator chip
0x1008	CT_R4F_DVI	W	CT-R4F DVI and CLCD control
0x100A	CT_R4F_CTRL	R/W	CT-R4F control
0x1010	CT_R4F_ADC0	R	CT-R4F ADC channel 0 – Cortex R4 current
0x1011	CT_R4F_ADC1	R	CT-R4F ADC channel 1 – *Reserved*
0x1012	CT_R4F_ADC2	R	CT-R4F ADC channel 2 – VSoc current
0x1013	CT_R4F_ADC3	R	CT-R4F ADC channel 3 – Core voltage
0x1014	CT_R4F_ADC4	R	CT-R4F ADC channel 4 – PLL voltage
0x1015	CT_R4F_ADC5	R	CT-R4F ADC channel 5– IO voltage

Table 4-4 CT-R4F serial configuration registers

4.9 CT-R4F power-up configuration

Following EB FPGA initialization a serial data controller reads BOARD_ID and CT_R4F_PLD_ID register from CT-R4F PLD using serial interface to recognise CT-R4F board. The serial data controller then transfers values of the EB FPGA CT_R4F_TC_CFGx and CT_R4F_OSCx registers and CT-R4F PLD configures R4F TC and on board oscillators.

After that the serial data controller constantly reads CT_R4F_ADCx registers to update PLLs lock status and CT_R4F_ADCx registers implemented on the EB FPGA. If any changes of CT_R4F_OSCx registers value are detected the new value is transferred to PLD.

Please refer to the CT-R4F User Guide for more R4F configuration process and timing details.

4.10 CT-R4F status LEDs

The two status LEDs D1, D2 indicate the status of the PLL lock signals and serial interface as follows:

LED	D1	D2
Signal	ispClock and TC PLL locked	Serial stream reset
Status after reset	OFF	ON

Table 4-5 Status LEDs

The status of LED is ON when signal is LOW and OFF when signal is HIGH.

4.11 EB CT-R4F specific registers

Setting of the R4F TC serial configuration registers, CT-R4F oscillators and CT-R4F PLD control registers is possible through the EB system registers.

The CT register base is 0x10000000. All registers must be unlocked by writing 0xA05F to base+0x20 first.

Offset	Name	Access	Description
0x110	CT_R4F_OSC0	R/W	CT-R4F ACLK ICS307 oscillator chip
0x114	CT_R4F_OSC1	R/W	CT-R4F SDRAM ICS307 oscillator chip
0x11C	CT_R4F_OSC2	R/W	CT-R4F CLCD ICS307 oscillator chip
0x120	CT_R4F_OSCRESET0	R/W	CT-R4F ACLK ICS307 oscillator chip after reset
0x124	CT_R4F_OSCRESET1	R/W	CT-R4F SDRAM ICS307 oscillator chip after reset
0x12C	CT_R4F_OSCRESET2	R/W	CT-R4F CLCD ICS307 oscillator chip after reset
0x074	CT_R4F_TC_CFG0	R/W	CT-R4F TC config word 0
0x078	CT_R4F_TC_CFG1	R/W	CT-R4F TC config word 1
0x07C	CT_R4F_TC_CFG2	R/W	CT-R4F TC config word 2
0x0A0	CT_R4F_ADC0	R	CT-R4F ADC channel 0 – Cortex R4 current
0x0A4	CT_R4F_ADC1	R	CT-R4F ADC channel 1 – Reserved
0x0A8	CT_R4F_ADC2	R	CT-R4F ADC channel 2 – VSoc current
0x0AC	CT_R4F_ADC3	R	CT-R4F ADC channel 3 – Core voltage
0x0B0	CT_R4F_ADC4	R	CT-R4F ADC channel 4 – PLL voltage
0x0B4	CT_R4F_ADC5	R	CT-R4F ADC channel 5– IO voltage
0x13C	CT_R4F_DVI	R/W	CT-R4F DVI and CLCD control
0x140	CT_R4F_CTRL	R/W	CT-R4F control

Table 4-6 CT_R4F specific registers

The registers values are passed to/from CT-R4F PLD via Serial Stream Controller implemented on EB FPGA. The CT_R4F_x_ID, CT_R4F_OSCRESETx, CT_R4F_TC_CFGx registers values are passed to/from PLD only during CT initialization. The CT_R4F_OSCx, CT_R4F_DVI and CT_R4F_CTRL registers values are sent to CT after register update. The CT_R4F_ADCx read registers are constantly updated by serial stream controller.

4.11.1 CT_R4F_OSCx and CT_R4F_OSCRESETx registers

The input reference frequency to all oscillators is 24MHz. For more information on the ICS307 oscillator settings please see the ICS307 web site.
The CT_R4F_OSCRESETx registers are maintained through reset and can be used to load the oscillator after reset.

The CT_R4F_OSC0 register at base+0x110 on EB board sets the R4F internal ACLK frequency.

Bit	Dir	Name	Default	Details
8:0	W	VDW[8:0]	0x5C	
15:9	W	RDW[6:0]	0x16	
18:16	W	OD[2:0]	0x3	f = 50MHz
31:19	R	RESERVED[12:0]	0x0	Reserved

Table 4-7 CT_R4F_OSC0 register

The CT_R4F_OSC1 register at base+0x114 on EB board sets the R4F SDRAM frequency.

Bit	Dir	Name	Default	Details
8:0	W	VDW[8:0]	0x7E	
15:9	W	RDW[6:0]	0x16	
18:16	W	OD[2:0]	0x3	f = 67MHz
31:19	R	RESERVED[12:0]	0x0	Reserved

Table 4-8 CT_R4F_OSC1 register

The CT_R4F_OSC2 register at base+0x11C on EB board sets the R4F CLCD frequency.

Bit	Dir	Name	Default	Details
8:0	W	VDW[8:0]	0x75	
15:9	W	RDW[6:0]	0x16	
18:16	W	OD[2:0]	0x0	f = 25MHz
31:19	R	RESERVED[12:0]	0x0	Reserved

Table 4-9 CT_R4F_OSC2 register

The CT_R4F_OSCRESET0 register at base+0x120 on EB board sets the R4F internal ACLK frequency after reset.

Bit	Dir	Name	Default	Details
8:0	W	VDW[8:0]	0x5C	
15:9	W	RDW[6:0]	0x16	
18:16	W	OD[2:0]	0x3	f = 50MHz
31:19	R	RESERVED[12:0]	0x0	Reserved

Table 4-10 CT_R4F_OSCRESET0 register

The CT_R4F_OSCRESET1 register at base+0x124 on EB board sets the R4F SDRAM frequency after reset.

Bit	Dir	Name	Default	Details
8:0	W	VDW[8:0]	0x7E	
15:9	W	RDW[6:0]	0x16	
18:16	W	OD[2:0]	0x3	f = 67MHz
31:19	R	RESERVED[12:0]	0x0	Reserved

Table 4-11 CT_R4F_OSCRESET1 register

The CT_R4F_OSCRESET2 register at base+0x12C on EB board sets the R4F CLCD frequency after reset.

Bit	Dir	Name	Default	Details
8:0	W	VDW[8:0]	0x75	
15:9	W	RDW[6:0]	0x16	
18:16	W	OD[2:0]	0x0	f = 25MHz
31:19	R	RESERVED[12:0]	0x0	Reserved

Table 4-12 CT_R4F_OSCRESET2 register

4.11.2 CT_R4F_TC_CFGx registers

The CT_R4F_TC_CFGx registers are 32bits wide and contain the values for three R4F TC configuration words.

The CT_R4F_TC_CFG0 register at base+0x74 on EB board sets the R4F initial PLLs settings.

Bit	Dir	Name	Default	Details
3:0	W	M_pll0	0x5	R4F ACLK PLL set up signals
6:4	W	N_pll0	0x1	R4F ACLK PLL set up signals
7	W	BYPASS_pll0	0x0	R4F ACLK PLL set up signals
8	W	ENABLE_pll0	0x1	R4F ACLK PLL set up signals
9	W	DESKEW_pll0	0x0	R4F ACLK PLL set up signals
10	W	RANGE_pll0	0x1	R4F ACLK PLL set up signals
11	W	MACRO_Bypass_pll0	0x0	R4F ACLK PLL set up signals
15:12	W	M_pll1	0x1	R4F MCLK PLL set up signals
18:16	W	N_pll1	0x1	R4F MCLK PLL set up signals
19	W	BYPASS_pll1	0x1	R4F MCLK PLL set up signals
20	W	ENABLE_pll1	0x0	R4F MCLK PLL set up signals
21	W	DESKEW_pll1	0x1	R4F MCLK PLL set up signals
22	W	RANGE_pll1	0x0	R4F MCLK PLL set up signals
23	W	MACRO_Bypass_pll1	0x1	R4F MCLK PLL set up signals
27:24	W	M_pll2	0x1	R4F extACLK PLL set up signals
30:28	W	N_pll2	0x1	R4F extACLK PLL set up signals
31	W	BYPASS_pll2	0x0	R4F extACLK PLL set up signals

Table 4-13 CT_R4F_TC_CFG0 register

The CT_R4F_TC_CFG1 register at base+0x78 on EB board sets the R4F initial settings.

Bit	Dir	Name	Default	Details
0	R	ENABLE_pll2	0x1	R4F extACLK PLL set up signals
1	W	DESKEW_pll2	0x1	R4F extACLK PLL set up signals
2	W	RANGE_pll2	0x0	R4F extACLK PLL set up signals
3	W	MACRO_Bypass_pll2	0x0	R4F extACLK PLL set up signals
5:4	W	Clock mode	0x1	CPU:ACLK ratio
6	W	cfg_CFGEE	0x0	
7	W	cfg_CFGIE	0x0	
8	W	cfg_CPUHALTn	0x1	
9	W	cfg_ENTCM1IF	0x1	
10	W	apb_ETMDBGEN	0x1	
11	W	apb_ETMNIDEN	0x1	
12	W	cfg_INITRAMA	0x0	Enable ATCM boot
13	W	cfg_INITRAMB	0x0	Enable BTCM boot
14	W	cfg_LOCZRAMA	0x1	
15	W	apb_NIDEN	0x1	
16	W	cfg_SLBTCMSB	0x0	
17	W	cfg_TEINIT	0x0	
18	W	cfg_VINITHI	0x0	
19	W	apb_CFGNMFI	0x0	
20	W	apb_DBGNOCLKSTOP	0x0	The processor will not stop the clocks when entering WFI state
21	W	apb_EDBGRQ	0x1	
25:22	W	apb_CFGATCMSZ	0x07	Soft TCM size config
29:26	W	apb_CFGBTCMSZ	0x07	Soft TCM size
30	W	apb_DBGEN	0x1	
31	W	apb_CTINIDEN	0x1	

Table 4-14 CT_R4F_TC_CFG1 register

The CT_R4F_TC_CFG2 register at base+0x7C on EB board sets the R4F initial settings.

Bit	Dir	Name	Default	Details
0	W	apb_CTIDBGEN	0x1	
1	W	*Reserved*	0x0	
26:2	R	*Reserved*	24'b0	
27	W	CPU nFIQExt source select	0x0	Select source for R4F CPU nFIQExt for VIC bypass.
28	W	CPU nIRQExt source select	0x0	Select source for CPU nIRQExt for VIC bypass.
31:29	W	PL301 REMAP[2:0]	0x0	R4F PL301 Remap selects

Table 4-15 CT_R4F_TC_CFG2 register

4.11.3 CT_R4F_ADCx registers

The CT_R4F_ADCx registers are 12 bits wide and all have the same following format.

The CT_R4F_ADC0 register at base+0xA0 board reads the Cortex R4 current.

Bit	Dir	Name	Default	Details
11:0	R	ADC value	0xXXX	Cortex R4 current (2 bits/mA)
31:12	R	*Reserved*	0x0	

Table 4-16 CT_R4F_ADC0 register

The CT_R4F_ADC1 register at base+0xA4 board.

Bit	Dir	Name	Default	Details
11:0	R	ADC value	0xXXX	*Reserved*
31:12	R	*Reserved*	0x0	

Table 4-17 CT_R4F_ADC1 register

The CT_R4F_ADC2 register at base+0xA8 board reads the VSoc current.

Bit	Dir	Name	Default	Details
11:0	R	ADC value	0xXXX	AXI domain current (5bits/mA)
31:12	R	*Reserved*	0x0	

Table 4-18 CT_R4F_ADC2 register

The CT_R4F_ADC3 register at base+0xAC board reads the Core voltage.

Bit	Dir	Name	Default	Details
11:0	R	ADC value	0xXXX	Core voltage (2bits/mV)
31:12	R	*Reserved*	0x0	

Table 4-19 CT_R4F_ADC3 register

The CT_R4F_ADC4 register at base+0xB0 board reads the PLL voltage.

Bit	Dir	Name	Default	Details
11:0	R	ADC value	0xXXX	PLL voltage (bit/mV)
31:12	R	*Reserved*	0x0	

Table 4-20 CT_R4F_ADC4 register

The CT_R4F_ADC5 register at base+0xB4 board reads the IO voltage.

Bit	Dir	Name	Default	Details
11:0	R	ADC value	0xXXX	IO voltage (bit/mV)
31:12	R	*Reserved*	0x0	

Table 4-21 CT_R4F_ADC5 register

4.11.4 CT_R4F_DVI register

The CT_R4F_DVI register at base+0x13C controls DVI interface on the CT-R4F board.

Bit	Dir	Name	Default	Details
0	W	BBDVI_nEN	0x0	enables the DVI output from the CT-R4F to the baseboard
3:1	W	DVI_CTRL[3:1]	0x0	should be set to 0x0 for normal operation
30:12	R	Reserved	0x0	
31	R	CLPOWER	0xX	CLCD power enable

Table 4-22 CT_R4F_DVI register

4.11.5 CT_R4F_CTRL register

The CT_R4F_CTRL register at base+0x140 controls board level functions on the CT-R4F board.

Bit	Dir	Name	Default	Details
0	W	JTAG_SRC	0x0	Controls R4F JTAG source.
7:1	R	Reserved	0x0	
8	W	RESET_CTRL[0]	0x1	Controls the individual reset for R4F core
9	W	RESET_CTRL[1]	0x1	Reserved
10	W	RESET_CTRL[2]	0x1	Controls the individual reset for R4F debug logic
11	W	RESET_CTRL[3]	0x1	Controls the individual reset for R4F Soc logic
15:12	R	Reserved	0x0	
16	R	CLPOWER	0xX	CLCD power enable
23:17	R	Reserved	0x0	
24	R	DMACIRQABORT	0xX	set HIGH by R4F TC DMAC when an abort occurs
31:25	R	Reserved	0x0	

Table 4-23 CT_R4F_CTRL register

5 Programmer's Model

5.1 CT-R4F boot up operation overview

This section is intended as a summary of the boot operation, please refer to the R4F Test Chip TRM for more information on the use of CP15 registers and exact operation.

5.2 R4F TC Memory Map

The CT-R4F on EB example design is in REMAP_MODE = 0.

Please refer to the CT-R4F User Guide for more R4F Test Chip memory map information.

5.3 EB Memory Map

The CT-R4F on EB example design provides a software interface with the majority of features required for a system. Refer to the EB user guide for more information

	Memory range		**Bus type**	**Memory region size**
Peripheral	**Lower limit**	**Upper limit**		
Dynamic Memory	0x00000000	0x0FFFFFFF	AHB/AXI	256M
System Registers	0x10000000	0x10000FFF	APB	4K
System Controller (SP810)	0x10001000	0x10001FFF	APB	4K
I2C control	0x10002000	0x10002FFF	APB	4K
Reserved	0x10003000	0x10003FFF	APB	4K
AACI	0x10004000	0x10004FFF	APB	4K
MCI0	0x10005000	0x10005FFF	APB	4K
KMI0	0x10006000	0x10006FFF	APB	4K
KMI1	0x10007000	0x10007FFF	APB	4K
Character LCD	0x10008000	0x10008FFF	APB	4K
UART0	0x10009000	0x10009FFF	APB	4K
UART1	0x1000A000	0x1000AFFF	APB	4K
UART2	0x1000B000	0x1000BFFF	APB	4K
UART3	0x1000C000	0x1000CFFF	APB	4K
SSP0	0x1000D000	0x1000DFFF	APB	4K
SCI0	0x1000E000	0x1000EFFF	APB	4K
Reserved	*0x1000F000*	*0x1000FFFF*	*APB*	*4K*
Watchdog	0x10010000	0x10010FFF	APB	4K
Timer 0&1	0x10011000	0x10011FFF	APB	4K
Timer 2&3	0x10012000	0x10012FFF	APB	4K
GPIO 0	0x10013000	0x10013FFF	APB	4K
GPIO 1	0x10014000	0x10014FFF	APB	4K
GPIO 2 (Misc onboard I/O)	0x10015000	0x10015FFF	APB	4K
Reserved	*0x10016000*	*0x10016FFF*	*APB*	*4K*
RTC	0x10017000	0x10017FFF	APB	4K
DMC configuration	0x10018000	0x10018FFF	APB	4K
PCI configuration	0x10019000	0x10019FFF	AHB	4K
Reserved for future use	*0x1001A000*	*0x1001FFFF*	*APB*	*28K (4K * 6)*
CLCD configuration	0x10020000	0x1002FFFF	AHB	64K (Note CLCD can only access Dynamic memory)
DMAC configuration	0x10030000	0x1003FFFF	AHB	64K

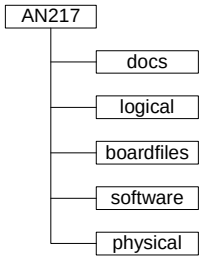
	Memory range		Bus type	Memory region size
Peripheral	**Lower limit**	**Upper limit**		
GIC1 (nFIQ IC) Tile Site 1	0x10040000	0x1004FFFF	AHB	64K
GIC2 (nIRQ IC) Tile Site 1	0x10050000	0x1005FFFF	AHB	64K
GIC3 (nFIQ IC) Tile Site 2	0x10060000	0x1006FFFF	AHB	64K
GIC4 (nIRQ IC) Tile Site 2	0x10070000	0x1007FFFF	AHB	64K
SMC configuration	0x10080000	0x1008FFFF	AHB	64K
Reserved for future use	*0x10090000*	*0x100EFFFF*	*AHB*	*448K (64K * 7)*
DAP ROM table	0x100F0000	0x100FFFFF	APB	64K
Reserved	*0x10100000*	*0x17FFFFFF*	*N/A*	*112M*
Reserved	*0x18000000*	*0x1FFFFFFF*	*N/A*	*128M*
Reserved	*0x20000000*	*0x3FFFFFFF*	*N/A*	*512M*
Static Memory Controller	0x40000000	0x5FFFFFFF	*AHB/AXI*	512M
PCI interface	0x60000000	0x6FFFFFFF	*AHB/AXI*	256M
Dynamic Memory (alias)	0x70000000	0x7FFFFFFF	AHB/AXI	256M
Logic Tile Site 2	0x80000000	0xCFFFFFFF	AHB	2G
Reserved	*0xE0000000*	*0xE0003FFF*	*APB*	*16KB*
CT-R4F PL340	0xE0004000	0xE0004FFF	APB	4KB
CT-R4F DMA	0xE0005000	0xE0005FFF	APB	4KB
CT-R4F PL301	0xE0006000	0xE0006FFF	APB	4KB
CT-R4F Config	0xE0007000	0xE0007FFF	APB	4KB
Reserved	*0xE0008000*	*0xE0008FFF*	*APB*	*4KB*
Reserved	*0xE0009000*	*0xE0009FFF*	*APB*	*4KB*
CT-R4F CLCD	0xE000A000	0xE000DFFF	AHB	16KB
CT-R4F VIC	0xE000E000	0xE0011FFF	AHB	16KB
Reserved	*0xE0012000*	*0xE001FFFF*	*N/A*	*56KB*
CT-R4F APB Debug	0xE0020000	0xE0029FFF	APB	10KB
Reserved	*0xE002A000*	*0xE0FAFFFF*	*N/A*	*208MB*
CT-R4F I cache	0xE0FB0000	0xE0FB3FFF	AXI	16KB
Reserved	*0xE0FB4000*	*0xE0FBFFFF*	*N/A*	*48KB*
CT-R4F D cache	0xE0FC0000	0xE0FC3FFF	AXI	16KB
Reserved	*0xE0FC4000*	*0xE0FCFFFF*	*N/A*	*48KB*
CT-R4F A TCM	0xE0FD0000	0xE0FDFFFF	AXI	64KB
CT-R4F B0 TCM	0xE0FE0000	0xE0FEFFFF	AXI	64KB
CT-R4F B1 TCM	0xE0FF0000	0xE0FFFFFF	AXI	64KB
CT-R4F AXIRAM	0xE1000000	0xE109FFFF	AXI	640KB
CT-R4F SDRAM	0xE10A0000	0xFFFFFFFF	AXI	1022MB

Table 5-1 Memory map

6 RTL

All of the APB RTL for this design is provided as verilog except for MMCI and GIC. AXI components are supplied as netlists. Example files are provided to allow building the system with Synplicity, Synplify Pro and Xilinx ISE tools.

6.1 Directory structure



The application note has directories. These are:

- docs : Related documents including this document.
- logical : All the verilog RTL required for the design.
- boardfiles : The files required to program the design into ARM development boards.
- physical : Synthesis and place and route (P&R) scripts and builds for target board.
- software : ARM code to run on the AN217 system

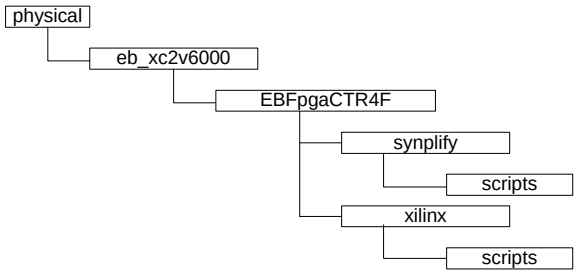
6.2 logical

The logical directory contains all the verilog required to build the system. The physical directory contains pre-synthesised components. The function of each block is shown earlier in **3.2.1 EB Module functionality**.

Each PrimeCell or other large IP block has its own directory (for example dmac_pl081).

The top level for this system is in EBFpga.

6.3 physical



The physical directory contains the scripts for the tools used in the build process.

6.4 Building the App Note using Microsoft Windows or Unix

To rebuild the FPGA image for the EB with CT-R4F you need to change into the EBFpgaCTR4F directory and run the make.scr (for UNIX) or make.bat (for windows) script. This will synthesis and Place & Route the design, linking in the required .NGO files at P&R. Read the readme.txt file in the directory for further build options.

make.bat	(default Synthesis and Place & Route)
make.bat all	(Synthesis and Place & Route)
make.bat synth	(Synthesis only)
make.bat par	(build -> map -> par -> bitgen)

To build the EBFpgaCTR4F image

```
cd EBFpgaCTR4F
make.bat all
```

To synthesis the EBFpgaCTR4F

```
cd EBFpgaCTR4F
make.bat synth
```

Note:
You need to ensure that the Synplify and Xilinx tools executable directories are in the path environment variable for DOS and UNIX.
The Synplify tools do not automatically add the path and the user is required to enter it manually. The Xilinx tools give you the option at installation time.

6.5 Board file selection

To use the pre-built bit file, use one of the following board files.

an217_eb_140c_xc2v6000_ctr4f_le….brd

an217_eb_140c_xc2v6000_ctr4f_pci_le….brd

To use a customer version, use one of the following board files.

an217_eb_140c_xc2v6000_ctr4f_customer_rebuild.brd