RCUBE

S P E C I F I C A T I O N

Laboratoire MASI CAO-VLSI UPMC PARIS VI

This document contains preliminary information for the RCUBE router.

February 1997

Version 1.7

Author: Vincent REIBALDI
Laboratoire MASI / Institut Blaise Pascal
Université Pierre et Marie Curie (Paris 6)
Couloir 55-65 2ème étage Bureau 211
4, place Jussieu, 75252 Paris Cedex 05
Tel: (33) 1 44 27 30 43
Fax: (33) 1 44 27 62 86
e-mail: vincent.reibaldi@masi.ibp.fr

Contents

List of Figures

List of Tables

1.INTRODUCTION

RCUBE (Rapid Reconfigurable Router) is a low latency router (150 ns without contention) for transputing and networking applications. Designed with an internal clock of 80 MHz, it includes 8 bidirectional high speed links, 8 independent programmable routing tables (one routing table per input link), and one 8x8 non blocking crossbar switch which enables the routing of a packet from any input link to any output link. The architecture is fully parallel and the transfer of a packet between one pair of links does not affect the data rate or the latency for another packet passing through another pair of links.

It implements a wormhole flow control technique, prefix and interval routing schemes, and provides efficient supports for adaptivity.

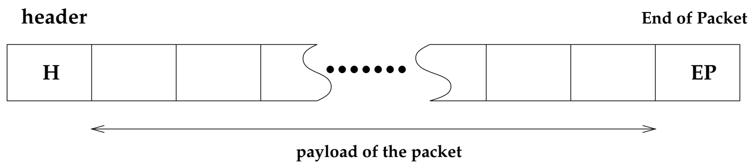
Based on CMOS high speed links developped at BULL, it delivers a global throughput of 1 Gigabit/s per link per direction, achieving a peak global throughput of 640 Mbyte/s.

RCUBE can route packets of any length. A packet is composed of a header which is the destination address in the network, followed by a sequence of characters which compose the payload of the packet and ended by an end packet character called EP or an exceptional end of packet character called EEP (fig.1.1). The payload of the packet can be composed of data characters or any control character different from EP or EEP. A character is a 9-bit word whose most significant bit (9th in big endian coding) is used to differentiate control characters ('1') from data characters ('0').

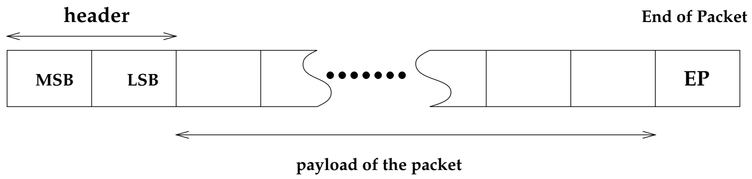
RCUBE provides the ability to specify for each link the two following functionalities :

- handling 1 or 2 character headers.
- enabling a header deletion mechanism.

The technology targeted is a 3.3V, 0.5 μm process. The IC can be packaged either in a 480 pads PBGA (Plastic Ball Grid Array) or a 208 pins Quad Flat Pack.



one byte header



two bytes header

Figure 1.1: Packet Format

1.1 Wormhole routing

In the wormhole routing, the routing decision is taken as soon as the packet header has been received. If a possible output link is free, the header is transmitted and the rest of the packet is sent directly from the input to the output without being stored. If all the possible outputs are unavailable the packet waits in the input fifo until a possible output link is free.

The packet header, in passing through a network of RCUBEs creates a temporary circuit through which the packet flows. As the end of the packet is pulled through, this virtual circuit disappears.

A packet can pass through several RCUBEs at the same time and the head of a packet can reach its destination before the end has been transmitted by the source. The latency is thus reduced.

1.2 Interval labelling

The routing scheme used by RCUBE is called interval labelling. Each device in a network is labelled with a number which is used as the destination address in a packet header (programmable length : 1 or 2 characters). Each of the 8 input links includes routing tables whereby each output link is assigned an interval of labels. When a packet arrives on an input link the routing decision is made by comparing the header to each bound of the intervals. When the header falls within an interval, the associated output is chosen. The intervals are and must be contiguous and non overlapping so that each header can only belongs to one interval.

1.3 Adaptivity

RCUBE can be programmed so that, for a given interval, several possible output links can be assigned to an input link. Therefore, if the default output link is busy, RCUBE routes the packet via an alternative output link unless all the possible outputs are busy.

Adaptivity is especially usefull for :

- bandwith increase (links can be grouped to form a wide logical link with high capacity).
- avoiding packet stall because of link failure.
- avoiding hot spots by providing auxiliary routes.

1.4 Header deletion mode

Each output link of RCUBE can be configured to implement a header discarding mechanism called header deletion mode. When an output link is configured in that mode, it discards the entire header whatever its length before transmitting the packet. Then, the following character(s) become(s) the new header.
This capability overcomes the addressing space limitation due to the finite header length. For example, a first header is used to route a packet across a network towards a sub-network. After the first header has been discarded, the second header gives the device address in the sub-network. It is an easy way to route packets in a hierarchical composition of networks and makes possible the labelling of huge networks.

1.5 Configuration

The routing tables are initialised through a 8-bits wide scanpath directly controlled by external pins. This scanpath contains 258 8-bits words (see section 5).

2.BLOCK DESCRIPTION

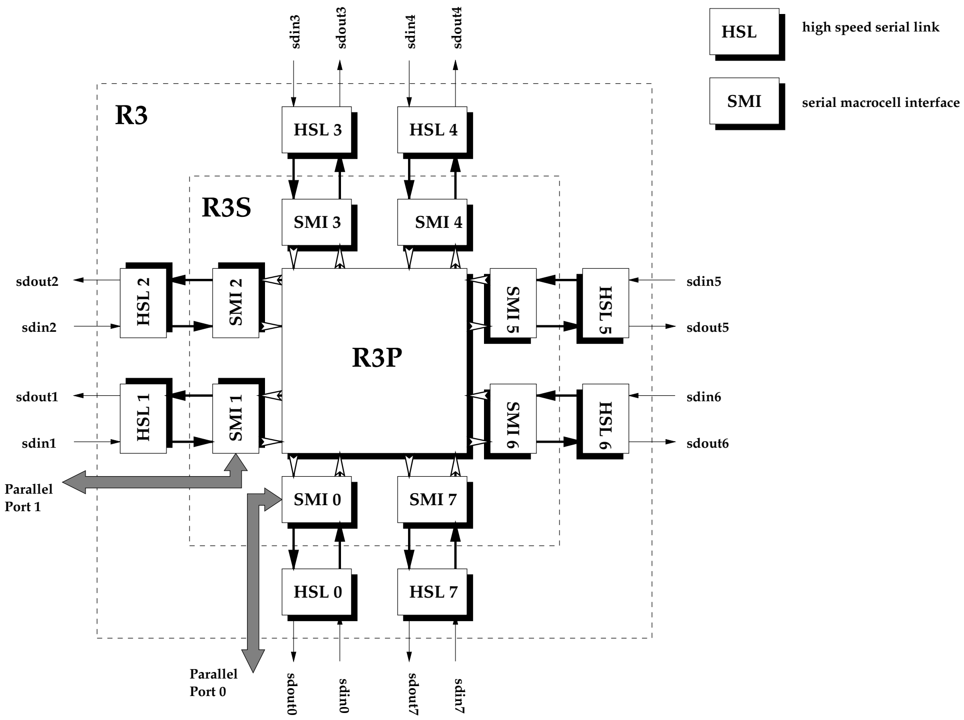


Figure 2.1: Block description

Figure 2.1 describes the 3 main components of RCUBE.

- HSL : the BULL High Speed Serial Link includes the serializer / deserializer blocks and an impedance adaptator ZC.
- SMI : the Serial Macrocell Interface contains the serial link boot protocol and the character encoding/decoding.
- R3P : This is the router core with parallel ports (one character wide).

3.PIN DESCRIPTION

FUNCTIONAL PINS			
Pin name	**I/O**	**Active level**	**Brief Description**
ckin	I	-	Externally provided clock.
nreset	I	low	Chip reset.
nreset_link[7:0]	I	low	Serial link reset.
zcreset	I	low	Impedance adaptator ZC reset.
sdin0	I	-	Link 0 serial data in.
sdin1	I	-	Link 1 serial data in.
sdin2	I	-	Link 2 serial data in.
sdin3	I	-	Link 3 serial data in.
sdin4	I	-	Link 4 serial data in.
sdin5	I	-	Link 5 serial data in.
sdin6	I	-	Link 6 serial data in.
sdin7	I	-	Link 7 serial data in.
sdout0	O	-	Link 0 serial data out.
sdout1	O	-	Link 1 serial data out.
sdout2	O	-	Link 2 serial data out.
sdout3	O	-	Link 3 serial data out.
sdout4	O	-	Link 4 serial data out.
sdout5	O	-	Link 5 serial data out.
sdout6	O	-	Link 6 serial data out.
sdout7	O	-	Link 7 serial data out.
recal0	I	low	Link0 deserializer reset.
recal1	I	low	Link1 deserializer reset.
recal2	I	low	Link2 deserializer reset.
recal3	I	low	Link3 deserializer reset.
recal4	I	low	Link4 deserializer reset.
recal5	I	low	Link5 deserializer reset.
recal6	I	low	Link6 deserializer reset.
CONFIGURATION PINS			
clockconfig	I	-	Configuration strobe.
scaninconfig [7:0]	I	-	Input of the 8-bits parallel configuration port.
scanoutconfig [7:0]	O	-	Output of the 8-bits parallel configuration port.

PARALLEL PORTS : port 0 and port 1			
Pin name	**I/O**	**Active level**	**Brief Description**
datastrobe0	I	-	write strobe of the input asynchronous fifo.
datain0[8:0]	I	-	incoming character on parallel port 0.
fccreceived0	I	high	adds a flit to the emission credit of the port 0.
sendfcc0	O	high	Port 0 can receive a flit.
senddata0	O	high	Data sending flag.
dataout0[8:0]	O	-	outgoing character from parallel port 0.
muxps0	I	high	selects the parallel input on link 0.
datastrobe1	I	-	write strobe of the input asynchronous fifo.
datain1[8:0]	I	-	incoming character on parallel port 1.
fccreceived1	I	high	adds a flit to the emission credit of the port 1.
sendfcc1	O	high	Port 1 can receive a flit.
senddata1	O	high	Data sending flag.
dataout1[8:0]	O	-	outgoing character from parallel port 1.
muxps1	I	high	selects the parallel input on link 1.

TEST PINS			
Pin name	**I/O**	**Active level**	**Brief Description**
test	I	high	All the functional registers of R3P are controlable and observable through a scanpath. The clock used in test mode is the functional clock ckin.
scanin	I	-	R3P Scanpath input.
scanout	O	-	R3P Scanpath output.
test_smi	I	high	The functional registers of the SMIs clocked by the clock ckin are controlable and observable through a scanpath. The clock used in test mode is the functional clock ckin.
scanin_smi	I	-	SMIs Scanpath input.
scanout_smi	O	-	SMIs Scanpath output.
test_hsl	I	low	enables the scanning of each register of the HSLs through a test scanpath.
scanin_hsl	I	-	input of the test scanpath of the HSLs.
scanout_hsl	O	-	output of the test scanpath of the HSLs.
zctest	I	low	enables the scanning of each register of the ZC block through a dedicated test scanpath.
zcscin	I	-	input of the ZC test scanpath.
zcscout	O	-	output of the ZC test scanpath.
loop_par[7:0]	I	high	enables a direct connection between the output and the input of the SMIs before entering the HSLs.
loop_ser[7:0]	I	low	enables an internal loop back between the serial output and the serial input of each HSL.
ckm1,ckm2	I	-	HSLs, ZC block test clocks.
zin,zou	I	-	must be externally connected via a 100 ohm resistor.

TOTAL: 138 pins

3.1 Logical buffers specifications

Buffer type	max input V	max V for logical 0	min V for logical 1	max output V
input	5v / 3.3v	0.8v	2v	-
output	-	-	-	3.3v

WARNING: the maximum voltage applied on the clock input ckin MUST be 3.3V and CAN'T support 5V level.
Each input buffer (except the ckin input) has a pull-up of about 10 KOhm. Therefore, care must be especially taken for input pins that are active at '1'. This is particularly critical for the pin **Nreset** which must be at '0' during power on. If a pull-down resistor is used, its value must insure a maximal voltage of 0.8 V.

3.2 Functional pins

The informations given hereafter specify for each signal name, its direction IN (for input) or OUT (for output) as well as the high level voltage that can be applied on input buffer or delivered by output buffer.

ckin . . . . **3.3v** **IN**

RCUBE uses an internal 80 MHz clock externally provided by the board.

nreset . . . . **3.3v / 5v** **IN**

When low, this signal synchronously resets on the falling edge of **ckin**, the serial links and all the R3S functional registers except the configuration registers. The signal **nreset** must be active at least for 10 **ckin** clock cycles to insure the minimal reset period of both the HSL links and the ZC block.

nreset_link(7:0) . . . . . . **3.3v / 5v** **IN**

When this signal remains at low level for at least 100 ns, it asynchronously forces the serial link into an uncalibrated state. When the signal raises, the serial link enters a boot phase (unless **nreset** is active while the clock **ckin** is running).

zcreset . . . . **3.3v / 5v** **IN**

When this signal is forced at low level for at least 100 ns, it asynchronously initializes the ZC registers contents. It must be active whenever **nreset** is active.

sdin0...7 . . . . **IN**

Link 0...7 input serial data.

sdout0...7 . . . . **OUT**

Link 0...7 output serial data.

recal0...7 . . . . **3.3v / 5v** **IN**

When low, this signal reinitializes the deserializer part of the serial link7..0. These signal are used only with optical fiber link. If copper links are used, these signal must always be forced at a high level.

linkstatus(7:0) . . . . **3.3v** **OUT**

When high, this signal indicates that the HS link is calibrated.

zin,zou . . . . **IN**

These pins must be connected through a 100 ohm resistor.

3.3 Configuration pins

clockconfig . . . . **3.3v / 5v** **IN**

This strobe is used during the initialisation phase to load the configuration registers.

scaninconfig(7:0) . . . . . **3.3v / 5v** **IN**

Input of the parallel 8-bits scanpath used to load the configuration registers.

scanoutconfig(7:0) . . . . **3.3v** **OUT**

Output of the parallel 8-bits configuration scanpath. This output may be observed to check the content of the configuration registers.

3.4 Parallel ports

For test consideration, two ports are multiplexed : the high speed serial links 0 and 1 can be replaced by parallel ports thanks to a multiplexor. This enables the access to R3P without using the serial link.

muxps0, muxps1 . . . . **3.3v / 5v** **IN**

When this signal is high, the parallel access of port 0 (resp. 1) is used. The choice of using port 0 (resp. port 1) as a parallel port or a serial port must be made while the **nreset** signal is active.

datastrobe0, datastrobe1 . **3.3v / 5v** **IN**

This signal commands on its falling edge the writing of the incoming data in the input fifo of port 0 (resp. 1).

datain0(8:0), datain1(8:0) . **3.3v / 5v** **IN**

9-bit incoming character on parallel port 0 (resp. 1).

fccreceived0, fccreceived1 **3.3v / 5v** **IN**

This signal is used to control the characters flow. When high, the output part of the port 0 (resp. 1) gets an emission credit of 32 characters.

sendfcc0, sendfcc1 . . . . **3.3v** **OUT**

When the input port 0 (resp. 1) has consumed(routed) a flit (32 characters), it commands its output port to raise the signal **sendfcc** for one **ckin** clock cycle. When the transmitter detects that signal active, it sends another flit to the router.

senddata0, senddata1 . . . . **3.3v** **OUT**

This flag goes high for one **ckin** clock cycle whenever a data character is available on the output port 0 (resp. 1).

dataout0(8:0), dataout1(8:0) **3.3v** **OUT**

9-bit outgoing character from parallel port 0 (resp. 1).

3.5 Test pins

test . . . . **3.3v / 5v** **IN**

When high, all the functional registers of R3P, but not the configuration registers, are chained in a scanpath.

scanin . . . . **3.3v / 5v** **IN**

Input of the test scanpath of R3P.

scanout . . . . **3.3v** **OUT**

Output of the test scanpath of R3P.

test_smi . . . . **3.3v / 5v** **IN**

When high, the registers of the SMIs clocked by the clock **ckin** are chained in a scanpath.

scanin_smi . . . . **3.3v / 5v** **IN**

Input of the test scanpath of the SMIs.

scanout_smi . . . . . . **3.3v** **OUT**

Output of the test scanpath of the SMIs.

test_hsl . . . . **3.3v / 5v** **IN**

When low, this signal allows the scanning of each register of the HSLs.

scanin_hsl . . . . **3.3v / 5v** **IN**

Input of the test scanpath of the HSls.

scanout_hsl . . . . . . **3.3v** **OUT**

Ouput of the test scanpath of the HSLs.

zctest . . . . **3.3v / 5v** **IN**

When low, this signal enables the scanning of each register of the ZC block through a dedicated test scanpath.

zcscin . . . . **3.3v / 5v** **IN**

Input of the test scanpath of the ZC block.

zcscout . . . . **3.3v** **OUT**

Output of the test scanpath of the ZC block.

ckm1, ckm2 . . . . . . **3.3v / 5v** **IN**

The HSLs and ZC scanpathes use a biphase clocking scheme (different from the main clock) (fig. 3.6). These clocks can be software drived and don't need to be free running.

loop_par(7:0) . . . . . **3.3v / 5v** **IN**

For test consideration, each SMI implements a parallel loop back mechanism (fig. 3.1). When high, the signal **loop_par** enables a connection between the parallel output of a port and its parallel output. This signal must be set while **nreset** is active.

loop_ser(7:0) . . . . . **3.3v / 5v** **IN**

Each HSL implements a serial loop back mechanism that enables an internal direct connection between its serial output and its serial input (fig.3.1). When low, the signal **loop_ser[7:0]** activates this internal serial loop back in the HSL 7..0. This signal must be set while **nreset** is active.

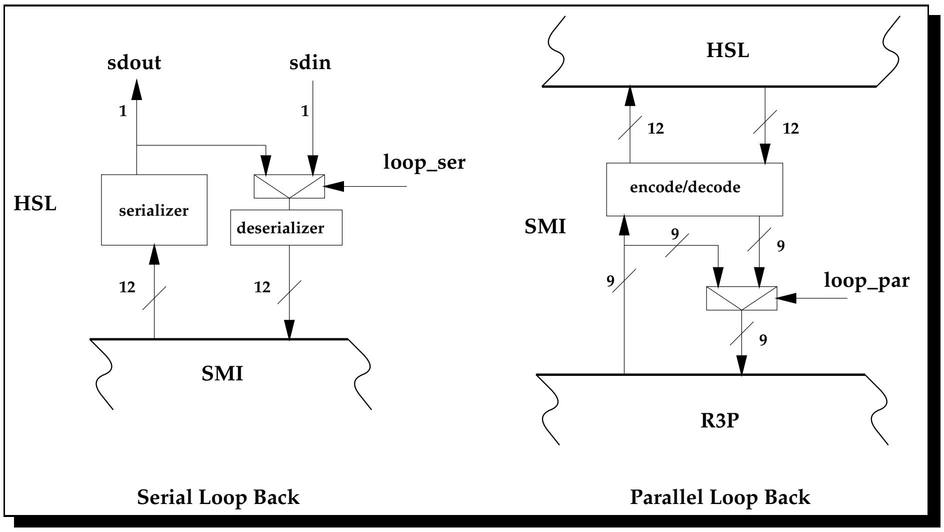


Figure 3.1: Loop back modes

3.6 Timing

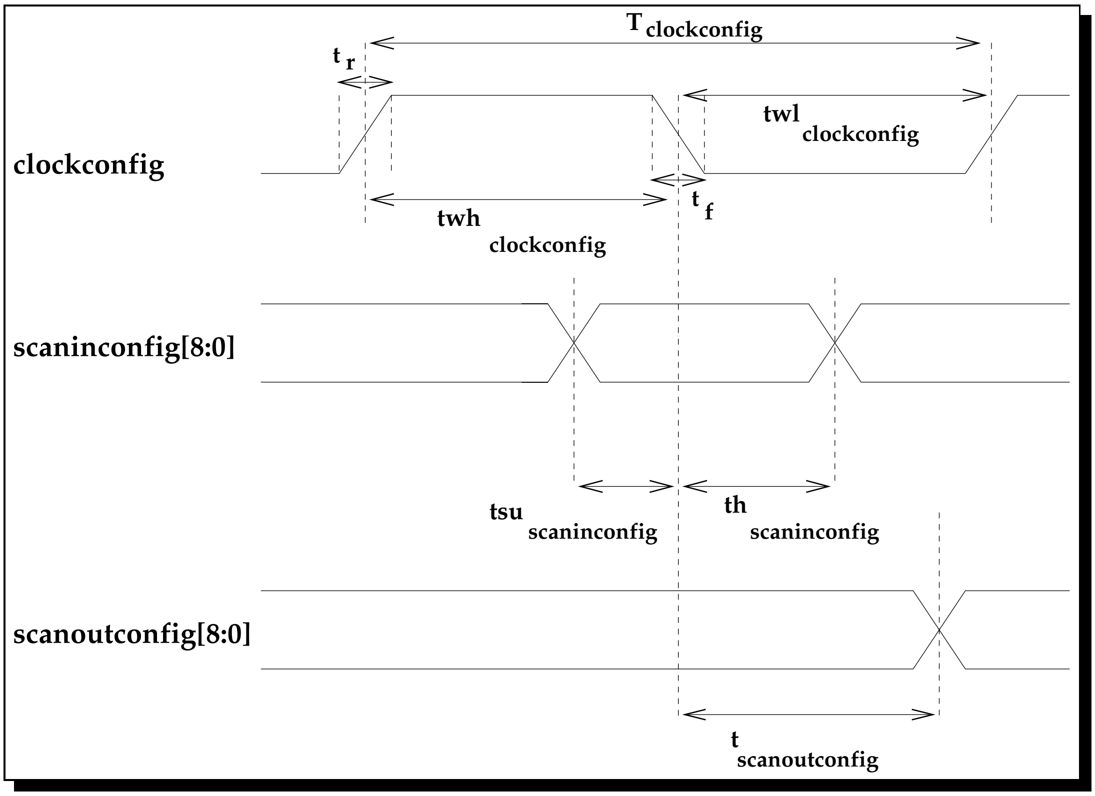


Figure 3.2: Clockconfig timing

symbol	parameters	min	max
$t_{clockconfig}$	clockconfig period	12 ns	-
$twh_{clockconfig}$	clockconfig high width	5 ns	-
$twl_{clockconfig}$	clockconfig low width	5 ns	-
$tr_{clockconfig}$	clockconfig edge rise time	1 ns	-
$tf_{clockconfig}$	clockconfig edge fall time	1 ns	-
$tsu_{scaninconfig}$	set-up time of scaninconfig[8:0]	1 ns	-
$th_{scaninconfig}$	hold time of scaninconfig[8:0]	3 ns	-
$t_{scanoutconfig}$	clockconfig falling edge to scanoutconfig	-	5 ns

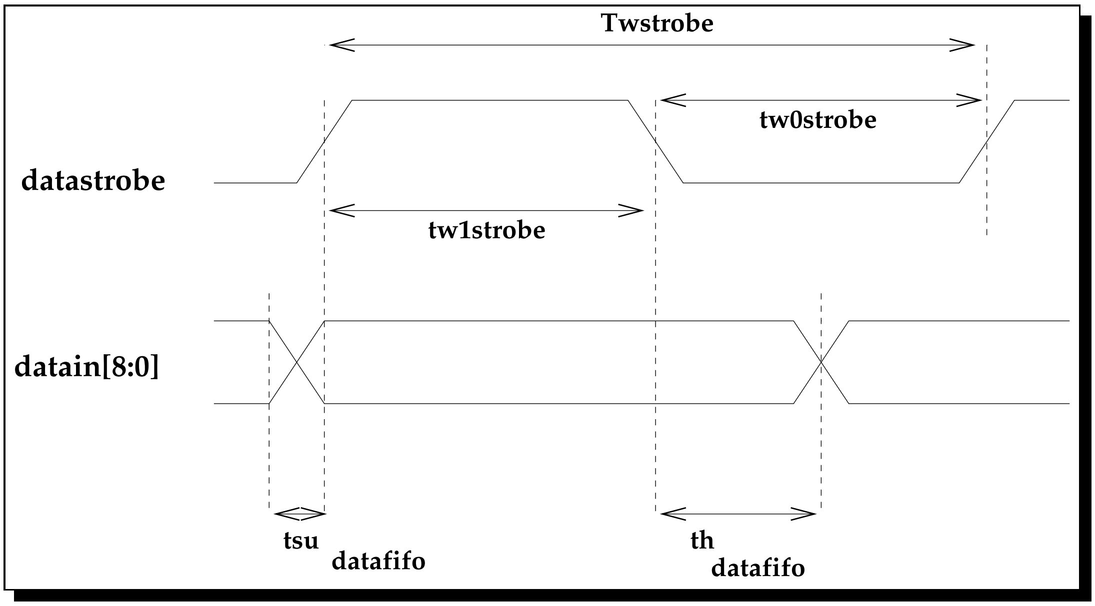


Figure 3.3: Fifo write strobe

symbol	parameters	min	max
tw_{strobe}	datastrobe period	12 ns	
$tw1_{strobe}$	width datastrobe high	5 ns	
$tw0_{strobe}$	width datasrobe low	5 ns	
$tsu_{datafifo}$	set-up time of datain[8:0]	0 ns	
$th_{datafifo}$	hold time of datain[8:0]	3 ns	

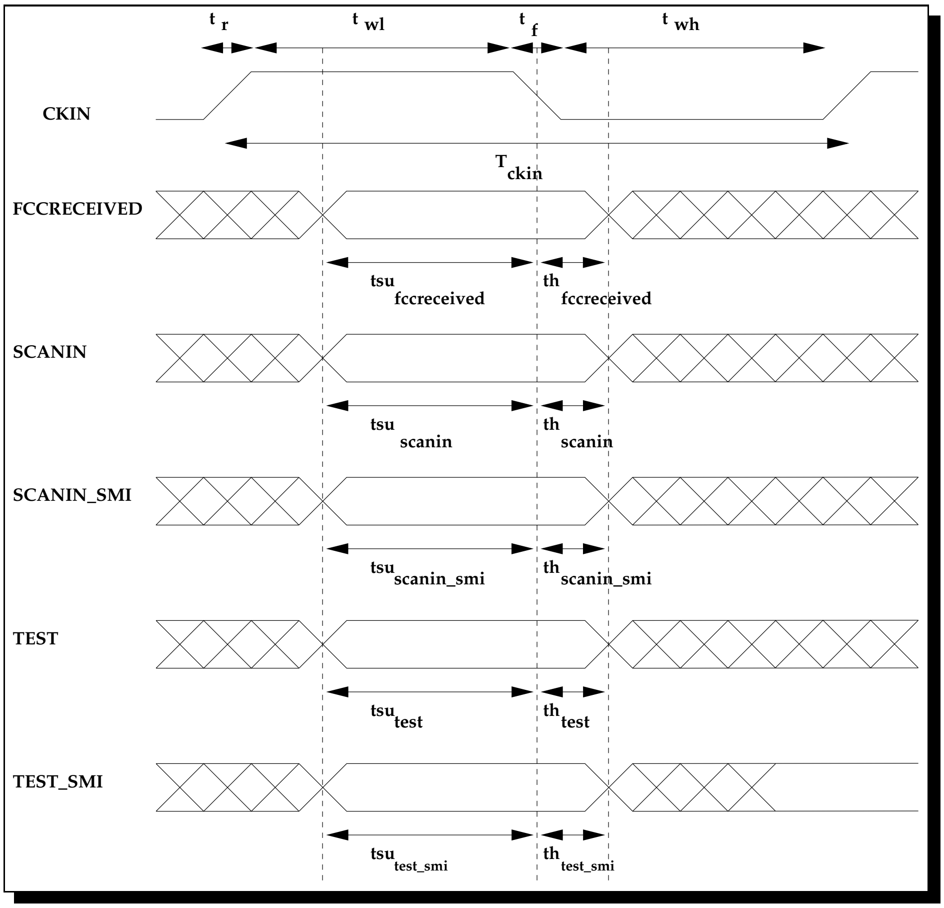


Figure 3.4: Input signals timing

symbol	parameters	min	max
t_{ckin}	ckin period	12 ns	18 ns
twh_{ckin}	ckin high width	5 ns	8 ns
twl_{ckin}	ckin low width	5 ns	8 ns
tr_{ckin}	ckin edge rise time	1 ns	3 ns
tf_{ckin}	ckin edge fall time	1 ns	3 ns
$tsu_{fccreceived}$	set-up time of fccreceived0,1	6 ns	
$th_{fccreceived}$	hold time of fccreceived0,1	0 ns	
tsu_{test}	set-up time of test	25 ns	
th_{test}	hold time of test	0 ns	
tsu_{scanin}	set-up time of scanin	5 ns	
th_{scanin}	hold time of scanin	1 ns	
tsu_{test_smi}	set-up time of test_smi	10 ns	
th_{test_smi}	hold time of test_smi	0 ns	
tsu_{scanin_smi}	set-up time of scanin_smi	5 ns	
th_{scanin_smi}	hold time of scanin_smi	1 ns	

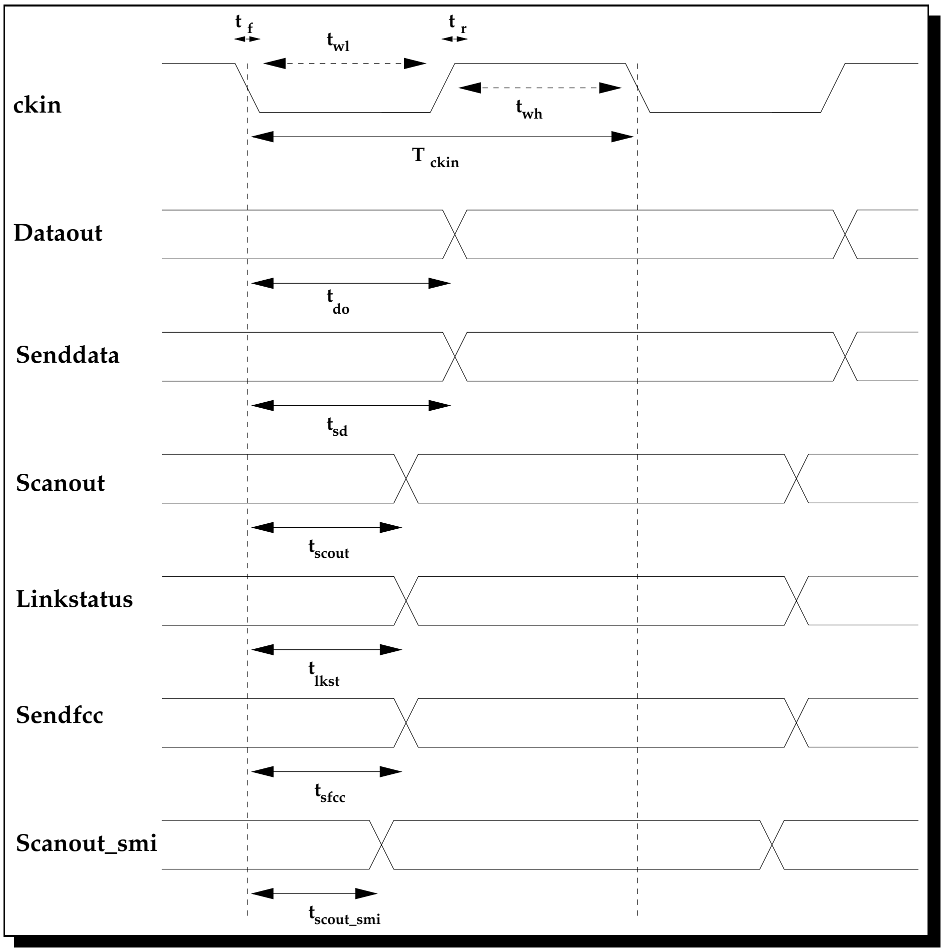


Figure 3.5: Output signals timing

symbol	parameters	min	max
t_{ckin}	ckin period	12 ns	18 ns
twh_{ckin}	ckin high width	5 ns	8 ns
twl_{ckin}	ckin low width	5 ns	8 ns
tr_{ckin}	ckin edge rise time	1 ns	3 ns
tf_{ckin}	ckin edge fall time	1 ns	3 ns
t_{do}	ckin falling edge to dataout0,1		8 ns
t_{sd}	ckin falling edge to senddata0,1		8 ns
t_{scout}	ckin falling edge to scanout		6 ns
t_{lkst}	ckin falling edge to linkstatus[7:0]		5.5 ns
t_{sfcc}	ckin falling edge to sendfcc0,1		6 ns
t_{scout_smi}	ckin falling edge to scanout_smi		5 ns

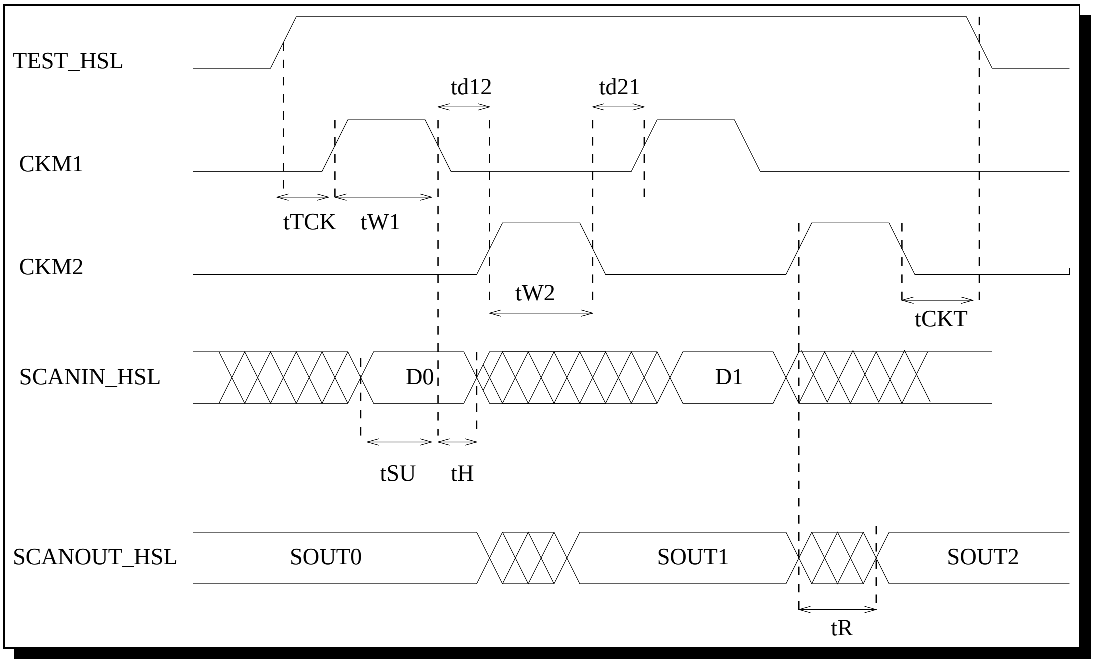


Figure 3.6: HSL test timing

symbol	parameters	min	max
tTCK	test_hsl stable to ckm1	100 ns	-
tW1	width ckm1 high	10 ns	-
td12	ckm1 falling edge to ckm2 rising edge	10 ns	-
tW2	width ckm2 high	10 ns	-
td21	ckm2 falling edge to ckm1 rising edge	10 ns	-
tSU	data setup time(scanin to ckm1 falling edge)	10 ns	-
tH	data hold time(ckm1 falling edge to scanin change)	10 ns	-
tR	ckm2 rising edge to scanout	-	10 ns
tCKT	ckm2 to test_hsl change	100	-

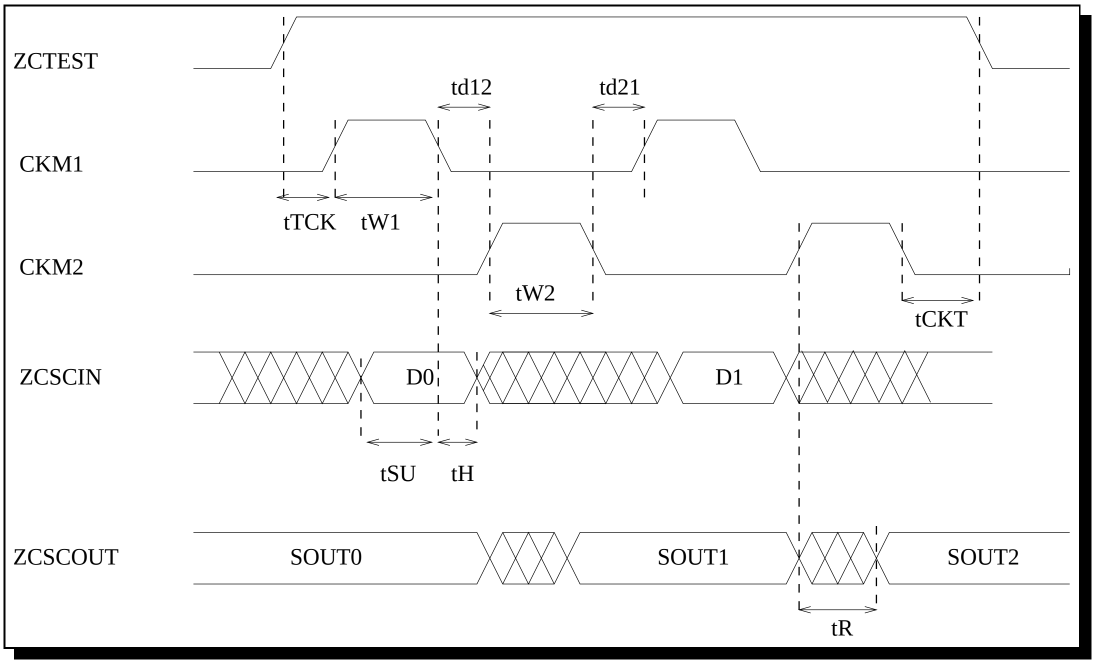


Figure 3.7: ZC block test timing

symbol	parameters	min	max
tTCK	zctest stable to ckm1	100 ns	-
tW1	width ckm1 high	10 ns	-
td12	ckm1 falling edge to ckm2 rising edge	10 ns	-
tW2	width ckm2 high	10 ns	-
td21	ckm2 falling edge to ckm1 rising edge	10 ns	-
tSU	data setup time(zcscin to ckm1 falling edge)	10 ns	-
tH	data hold time(ckm1 falling edge to zcscin change)	10 ns	-
tR	ckm2 rising edge to zcscout	-	10 ns
tCKT	ckm2 to zctest change	100	-

4.POWER ON

4.1 Multiple driver prevention

Tri-state buses are used inside the crossbar. The tri-state gates commands come from registers. At power on, the contents of these crossbar command registers are set randomly. This can lead to short-circuits because of multiple drivers on buses. To prevent this, the commands of these tri-state gates are disabled when the signal **nreset** is active.

WARNING : When the chip is powered, the signal **nreset** must be active at low.

4.2 Power rising time

There's no constraint on the power rising time.

5.INITIALIZATION

5.1 Configuration phase

Prior to any use of RCUBE, an initialization phase is necessary to load the configuration registers. This phase must be done before the serial link calibration phase. Hence, signal **nreset** must be active during the configuration phase.

The following informations must be loaded :

- the bounds tables,
- the selection tables,
- the priority tables,
- the header length table,
- the header deletion table.

RCUBE includes :

- for each link :
  - 8 16-bit bounds registers (since 2 byte headers are allowed). Each 16-bit bounds register is physically divided in 2 8-bit registers,
  - 8 8-bit selection registers,
  - 8 8-bit priority registers
- one 8-bit headerdeletion register
- one 8-bit headerlength register.

5.2 Bound Registers

In the interval labelling scheme, the addressing space is divided in intervals defined by an upper and lower limits called bounds. Each bound is coded on 2 bytes called mbound (Most Significant Byte) and lbound (Less Significant Byte) since we allow 2 byte headers. The user must load these bounds registers according to the topology of the network. In an addressing space coded on 16 bits, we obtain 65536 different destinations for a packet. Since there are 8 bounds registers, we divide this space into 8 intervals. An example is given on table 5.1. A header whose value stands in interval [53699,...,65535] ∪ [0,...,501[falls in interval 7. If it belongs to [501,...,2570[, it stands in interval 0 and so on. In case of 1 byte headers the value of the lbound is unused.

interval number	mbound[7:0]	lbound[7:0]	#dec
0	00000001	11110101	501
1	00001010	00001010	2570
2	00010100	11000011	5315
3	00111111	10000011	16259
4	01101001	10001000	27016
5	10001111	01000000	36672
6	11001000	00100101	51237
7	11010001	11000011	53699

Table 5.1: Exemple of interval bound table

5.3 Selection Registers

A list of possible output links is associated to each interval. Hence, the user can choose for each input link which output links can be taken, according to the header value.

interval number	output configuration register value[7:0]	out 7	out 6	out 5	out 4	out 3	out 2	out 1	out 0
0	11001011	y	y	n	n	y	n	y	y
1	00100100	y	y	n	y	y	y	n	n
2	11010110	y	y	n	y	n	y	y	n
3	00011111	n	n	n	y	y	y	y	y
4	11011011	y	y	n	y	y	n	y	y
5	00000001	n	n	n	n	n	n	n	y
6	11111101	y	y	y	y	y	y	n	y
7	01010111	n	y	n	y	n	y	y	y

Table 5.2: Exemple of outputs table

NOTE: If a selection register contains the value full zero, any packet whose header belongs to the corresponding interval is discarded.

5.4 Priority Registers

This register allows the user to define priorities between several ports associated to the same interval.
This table contains the value that can be used to define the default output. This output has the highest priority.
The x means that the bit can take any value. The value of the x only defines the priority of the alternative outputs.
Additional informations on the coding scheme are given in annexe 1.

priority register value[7:0]	default output
x00x0xxx	7
x10x0xxx	6
xx100xxx	5
xx110xxx	4
xxxx100x	3
xxxx110x	2
xxxx1x10	1
xxxx1x11	0

Table 5.3: Priority table

5.5 Header Deletion Register

This register allows the user to define which output links implement the header discarding mechanism (noted hd in the following table).

header deletion register value[7:0]	out 7	out 6	out 5	out 4	out 3	out 2	out 1	out 0
01100100		hd	hd			hd		
10010010	hd			hd			hd	
00010100				hd		hd		

Table 5.4: Exemple of header deletion register

5.6 Header Length Register

This register allows the user to define whether input links treats 1 byte headers or 2 byte headers.

5.7 Configuration port

All these registers are called configuration registers and are loaded during the initialization phase via an 8-bits parallel scanpath. The input of this configu-

header length register value[7:0]	input 7	input 6	input 5	input 4	input 3	input 2	input 1	input 0
00110110	1	1	2	2	1	2	2	1
00111110	1	1	2	2	2	2	2	1
11100010	2	2	2	1	1	1	2	1

Table 5.5: Exemple of header length register

ration port is scaninconfig[7:0] and the data are written on the falling edge of the clockconfig strobe.
The configuration registers are chained in the following order :

scaninconfig[7:0] ->

link0
(mbound0,lbound0,mbound1,lbound1,mbound2,lbound2,mbound3,lbound3,
mbound4,lbound4,mbound5,lbound5,mbound6,lbound6,mbound7,lbound7)
->
link2
(mbound0,lbound0,mbound1,lbound1,mbound2,lbound2,mbound3,lbound3,
mbound4,lbound4,mbound5,lbound5,mbound6,lbound6,mbound7,lbound7)
->
link4
(mbound0,lbound0,mbound1,lbound1,mbound2,lbound2,mbound3,lbound3,
mbound4,lbound4,mbound5,lbound5,mbound6,lbound6,mbound7,lbound7)
->
link6
(mbound0,lbound0,mbound1,lbound1,mbound2,lbound2,mbound3,lbound3,
mbound4,lbound4,mbound5,lbound5,mbound6,lbound6,mbound7,lbound7)
->
link6
(output7,priority7,output6,priority6,output5,priority5,output4,priority4,
output3,priority3,output2,priority2,output1,priority1,output0,priority0)
->
link4
(output7,priority7,output6,priority6,output5,priority5,output4,priority4,
output3,priority3,output2,priority2,output1,priority1,output0,priority0)
->
link2
(output7,priority7,output6,priority6,output5,priority5,output4,priority4,
output3,priority3,output2,priority2,output1,priority1,output0,priority0)
->
link0
(output7,priority7,output6,priority6,output5,priority5,output4,priority4,
output3,priority3,output2,priority2,output1,priority1,output0,priority0)
->
headerdeletion,headerlength ->
link1
(mbound0,lbound0,mbound1,lbound1,mbound2,lbound2,mbound3,lbound3,
mbound4,lbound4,mbound5,lbound5,mbound6,lbound6,mbound7,lbound7)
->
link3
(mbound0,lbound0,mbound1,lbound1,mbound2,lbound2,mbound3,lbound3,
mbound4,lbound4,mbound5,lbound5,mbound6,lbound6,mbound7,lbound7)
->

link5 (mbound0,lbound0,mbound1,lbound1,mbound2,lbound2,mbound3,lbound3,
mbound4,lbound4,mbound5,lbound5,mbound6,lbound6,mbound7,lbound7)
->
link7 (mbound0,lbound0,mbound1,lbound1,mbound2,lbound2,mbound3,lbound3,
mbound4,lbound4,mbound5,lbound5,mbound6,lbound6,mbound7,lbound7)
->
link7 (output7,priority7,output6,priority6,output5,priority5,output4,priority4,
output3,priority3,output2,priority2,output1,priority1,output0,priority0)
->
link5 (output7,priority7,output6,priority6,output5,priority5,output4,priority4,
output3,priority3,output2,priority2,output1,priority1,output0,priority0)
->
link3 (output7,priority7,output6,priority6,output5,priority5,output4,priority4,
output3,priority3,output2,priority2,output1,priority1,output0,priority0)
->
link1 (output7,priority7,output6,priority6,output5,priority5,output4,priority4,
output3,priority3,output2,priority2,output1,priority1,output0,priority0)
->
scanoutconfig[7:0]

6.TEST

6.1 Functional test

6.1.1 Internal parallel loop back

In order to ease as much as possible the functional test of RCUBE, the test of R3P can be done without using the HSLs. Data can be sent to R3P through the parallel port 0 or 1. These data can flow through every link if the routing table are correctly programmed and the parallel loop back mechanism is activated on the SMI. The figure 6.1 shows an exemple of the test method.
The parallel loop back mechanism is activated in seven SMIs (macro 1,2,...,7). Data are sent into R3P by the parallel input of link 0. The routing tables are loaded so that the data follow the path running through SMI1,SMI2,...,SMI7 to be finally available on the parallel output of port 0.
For clarity, the parallel port 1 is not represented on the figure. Two parallel ports are required because when parallel port 0 is used, SMI0 and HSL0 are not tested.

6.1.2 Internal serial loop back

In this next step, a serial loop back mechanism is activated in each HSL in order to connect internally its serial output with its serial input. In a similar way, data are sent into R3P by the parallel port 0, the routing tables are loaded so that the data follow the path running through SMI1/HSL1, SMI2/HSL2, ..., SMI7/HSL7 to be finally available on the parallel output of port 0.

6.1.3 External serial loop back

In this last step, the serial output of each HSL is externally connected to its serial input. Then, data are sent into R3P by the parallel port 0, the routing tables are loaded so that the data follow the path running through HSL1, HSL2, ..., HSL7 to be finally available on the parallel output of port 0.

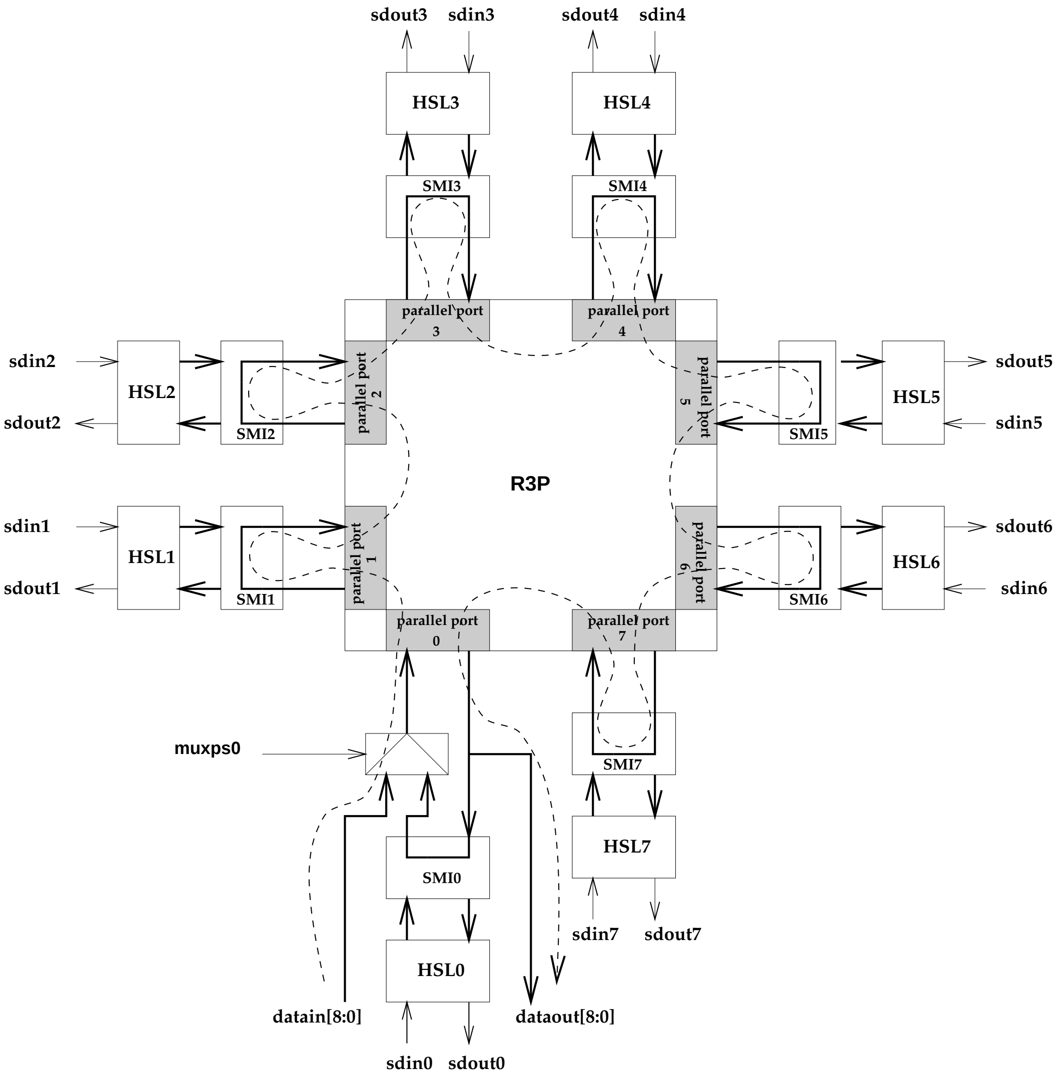


Figure 6.1: Functional test scenario

6.2 Structural test

RCUBE has 4 test scanpaths :

- The scanpath of R3P.
- The scanpath that chains all the SMIs.
- The scanpath that chains all the HSLs.
- The scanpath of the ZC block.

6.2.1 R3P scanpath description

This scanpath contains 1128 bits. It does not include the configurations registers. When the scanpath content is shifted, the crossbar commands that come from registers are disabled.

1	Tbreg0_Cmp0_Bequ.T	26	Tbreg2_Cmp0_Bequ.T
2	Tbreg0_Cmp0_Rescomp.T	27	Tbreg2_Cmp0_Rescomp.T
3	Tbreg0_Cmp1_Bequ.T	28	Tbreg2_Cmp1_Bequ.T
4	Tbreg0_Cmp1_Rescomp.T	29	Tbreg2_Cmp1_Rescomp.T
5	Tbreg0_Cmp2_Bequ.T	30	Tbreg2_Cmp2_Bequ.T
6	Tbreg0_Cmp2_Rescomp.T	31	Tbreg2_Cmp2_Rescomp.T
7	Tbreg0_Cmp3_Bequ.T	32	Tbreg2_Cmp3_Bequ.T
8	Tbreg0_Cmp3_Rescomp.T	33	Tbreg2_Cmp3_Rescomp.T
9	Tbreg0_Cmp4_Bequ.T	34	Tbreg2_Cmp4_Bequ.T
10	Tbreg0_Cmp4_Rescomp.T	35	Tbreg2_Cmp4_Rescomp.T
11	Tbreg0_Cmp5_Bequ.T	36	Tbreg2_Cmp5_Bequ.T
12	Tbreg0_Cmp5_Rescomp.T	37	Tbreg2_Cmp5_Rescomp.T
13	Tbreg0_Cmp6_Bequ.T	38	Tbreg2_Cmp6_Bequ.T
14	Tbreg0_Cmp6_Rescomp.T	39	Tbreg2_Cmp6_Rescomp.T
15	Tbreg0_Cmp7_Bequ.T	40	Tbreg2_Cmp7_Bequ.T
16	Tbreg0_Cmp7_Rescomp.T	41	Tbreg2_Cmp7_Rescomp.T
17	Atlog0_Arbitreqemit_Outreq_0.T	42	Atlog2_Arbitreqemit_Outreq_0.T
18	Atlog0_Arbitreqemit_Outreq_1.T	43	Atlog2_Arbitreqemit_Outreq_1.T
19	Atlog0_Arbitreqemit_Outreq_2.T	44	Atlog2_Arbitreqemit_Outreq_2.T
20	Atlog0_Arbitreqemit_Outreq_3.T	45	Atlog2_Arbitreqemit_Outreq_3.T
21	Atlog0_Arbitreqemit_Outreq_4.T	46	Atlog2_Arbitreqemit_Outreq_4.T
22	Atlog0_Arbitreqemit_Outreq_5.T	47	Atlog2_Arbitreqemit_Outreq_5.T
23	Atlog0_Arbitreqemit_Outreq_6.T	48	Atlog2_Arbitreqemit_Outreq_6.T
24	Atlog0_Arbitreqemit_Outreq_7.T	49	Atlog2_Arbitreqemit_Outreq_7.T
25	Atlog0_Glueat_Globerr.T	50	Atlog2_Glueat_Globerr.T

51	Tbreg4_Cmp0_Bequ.T	92	Atlog6_Arbitreqemit_Outreq_0.T
52	Tbreg4_Cmp0_Rescomp.T	93	Atlog6_Arbitreqemit_Outreq_1.T
53	Tbreg4_Cmp1_Bequ.T	94	Atlog6_Arbitreqemit_Outreq_2.T
54	Tbreg4_Cmp1_Rescomp.T	95	Atlog6_Arbitreqemit_Outreq_3.T
55	Tbreg4_Cmp2_Bequ.T	96	Atlog6_Arbitreqemit_Outreq_4.T
56	Tbreg4_Cmp2_Rescomp.T	97	Atlog6_Arbitreqemit_Outreq_5.T
57	Tbreg4_Cmp3_Bequ.T	98	Atlog6_Arbitreqemit_Outreq_6.T
58	Tbreg4_Cmp3_Rescomp.T	99	Atlog6_Arbitreqemit_Outreq_7.T
59	Tbreg4_Cmp4_Bequ.T	100	Atlog6_Glueat_Globerr.T
60	Tbreg4_Cmp4_Rescomp.T	101	Oam0246_Oamlogic_Oa1_Outbas0.F
61	Tbreg4_Cmp5_Bequ.T	102	Oam0246_Oamlogic_Oa1_Outbas1.F
62	Tbreg4_Cmp5_Rescomp.T	103	Oam0246_Oamlogic_Oa1_Outbas2.F
63	Tbreg4_Cmp6_Bequ.T	104	Oam0246_Oamlogic_Oa1_Outbas3.F
64	Tbreg4_Cmp6_Rescomp.T	105	Oam0246_Oamlogic_Oa1_Outbas4.F
65	Tbreg4_Cmp7_Bequ.T	106	Oam0246_Oamlogic_Oa1_Outbas5.F
66	Tbreg4_Cmp7_Rescomp.T	107	Oam0246_Oamlogic_Oa1_Outbas6.F
67	Atlog4_Arbitreqemit_Outreq_0.T	108	Oam0246_Oamlogic_Oa1_Outbas7.F
68	Atlog4_Arbitreqemit_Outreq_1.T	109	Oam0246_Oamlogic_Oa1_Ocounter0.T
69	Atlog4_Arbitreqemit_Outreq_2.T	110	Oam0246_Oamlogic_Oa1_Ocounter1.T
70	Atlog4_Arbitreqemit_Outreq_3.T	111	Oam0246_Oamlogic_Oa1_Ocounter2.T
71	Atlog4_Arbitreqemit_Outreq_4.T	112	Oam0246_Oamlogic_Oa1_Outbusy.F
72	Atlog4_Arbitreqemit_Outreq_5.T	113	Oam0246_Oamlogic_Oa2_Outbas0.F
73	Atlog4_Arbitreqemit_Outreq_6.T	114	Oam0246_Oamlogic_Oa2_Outbas1.F
74	Atlog4_Arbitreqemit_Outreq_7.T	115	Oam0246_Oamlogic_Oa2_Outbas2.F
75	Atlog4_Glueat_Globerr.T	116	Oam0246_Oamlogic_Oa2_Outbas3.F
76	Tbreg6_Cmp0_Bequ.T	117	Oam0246_Oamlogic_Oa2_Outbas4.F
77	Tbreg6_Cmp0_Rescomp.T	118	Oam0246_Oamlogic_Oa2_Outbas5.F
78	Tbreg6_Cmp1_Bequ.T	119	Oam0246_Oamlogic_Oa2_Outbas6.F
79	Tbreg6_Cmp1_Rescomp.T	120	Oam0246_Oamlogic_Oa2_Outbas7.F
80	Tbreg6_Cmp2_Bequ.T	121	Oam0246_Oamlogic_Oa2_Ocounter0.T
81	Tbreg6_Cmp2_Rescomp.T	122	Oam0246_Oamlogic_Oa2_Ocounter1.T
82	Tbreg6_Cmp3_Bequ.T	123	Oam0246_Oamlogic_Oa2_Ocounter2.T
83	Tbreg6_Cmp3_Rescomp.T	124	Oam0246_Oamlogic_Oa2_Outbusy.F
84	Tbreg6_Cmp4_Bequ.T	125	Oam0246_Oamlogic_Oa3_Outbas0.F
85	Tbreg6_Cmp4_Rescomp.T	126	Oam0246_Oamlogic_Oa3_Outbas1.F
86	Tbreg6_Cmp5_Bequ.T	127	Oam0246_Oamlogic_Oa3_Outbas2.F
87	Tbreg6_Cmp5_Rescomp.T	128	Oam0246_Oamlogic_Oa3_Outbas3.F
88	Tbreg6_Cmp6_Bequ.T	129	Oam0246_Oamlogic_Oa3_Outbas4.F
89	Tbreg6_Cmp6_Rescomp.T	130	Oam0246_Oamlogic_Oa3_Outbas5.F
90	Tbreg6_Cmp7_Bequ.T	131	Oam0246_Oamlogic_Oa3_Outbas6.F
91	Tbreg6_Cmp7_Rescomp.T	132	Oam0246_Oamlogic_Oa3_Outbas7.F

133	Oam0246_Oamlogic_Oa3_Ocounter0.T	174	Tbreg3_Cmp0_Bequ.T
134	Oam0246_Oamlogic_Oa3_Ocounter1.T	175	Tbreg3_Cmp0_Rescomp.T
135	Oam0246_Oamlogic_Oa3_Ocounter2.T	176	Tbreg3_Cmp1_Bequ.T
136	Oam0246_Oamlogic_Oa3_Outbusy.F	177	Tbreg3_Cmp1_Rescomp.T
137	Oam0246_Oamlogic_Oa4_Outbas0.F	178	Tbreg3_Cmp2_Bequ.T
138	Oam0246_Oamlogic_Oa4_Outbas1.F	179	Tbreg3_Cmp2_Rescomp.T
139	Oam0246_Oamlogic_Oa4_Outbas2.F	180	Tbreg3_Cmp3_Bequ.T
140	Oam0246_Oamlogic_Oa4_Outbas3.F	181	Tbreg3_Cmp3_Rescomp.T
141	Oam0246_Oamlogic_Oa4_Outbas4.F	182	Tbreg3_Cmp4_Bequ.T
142	Oam0246_Oamlogic_Oa4_Outbas5.F	183	Tbreg3_Cmp4_Rescomp.T
143	Oam0246_Oamlogic_Oa4_Outbas6.F	184	Tbreg3_Cmp5_Bequ.T
144	Oam0246_Oamlogic_Oa4_Outbas7.F	185	Tbreg3_Cmp5_Rescomp.T
145	Oam0246_Oamlogic_Oa4_Ocounter0.T	186	Tbreg3_Cmp6_Bequ.T
146	Oam0246_Oamlogic_Oa4_Ocounter1.T	187	Tbreg3_Cmp6_Rescomp.T
147	Oam0246_Oamlogic_Oa4_Ocounter2.T	188	Tbreg3_Cmp7_Bequ.T
148	Oam0246_Oamlogic_Oa4_Outbusy.F	189	Tbreg3_Cmp7_Rescomp.T
149	Tbreg1_Cmp0_Bequ.T	190	Atlog3_Arbitreqemit_Outreq_0.T
150	Tbreg1_Cmp0_Rescomp.T	191	Atlog3_Arbitreqemit_Outreq_1.T
151	Tbreg1_Cmp1_Bequ.T	192	Atlog3_Arbitreqemit_Outreq_2.T
152	Tbreg1_Cmp1_Rescomp.T	193	Atlog3_Arbitreqemit_Outreq_3.T
153	Tbreg1_Cmp2_Bequ.T	194	Atlog3_Arbitreqemit_Outreq_4.T
154	Tbreg1_Cmp2_Rescomp.T	195	Atlog3_Arbitreqemit_Outreq_5.T
155	Tbreg1_Cmp3_Bequ.T	196	Atlog3_Arbitreqemit_Outreq_6.T
156	Tbreg1_Cmp3_Rescomp.T	197	Atlog3_Arbitreqemit_Outreq_7.T
157	Tbreg1_Cmp4_Bequ.T	198	Atlog3_Glueat_Globerr.T
158	Tbreg1_Cmp4_Rescomp.T	199	Tbreg5_Cmp0_Bequ.T
159	Tbreg1_Cmp5_Bequ.T	200	Tbreg5_Cmp0_Rescomp.T
160	Tbreg1_Cmp5_Rescomp.T	201	Tbreg5_Cmp1_Bequ.T
161	Tbreg1_Cmp6_Bequ.T	202	Tbreg5_Cmp1_Rescomp.T
162	Tbreg1_Cmp6_Rescomp.T	203	Tbreg5_Cmp2_Bequ.T
163	Tbreg1_Cmp7_Bequ.T	204	Tbreg5_Cmp2_Rescomp.T
164	Tbreg1_Cmp7_Rescomp.T	205	Tbreg5_Cmp3_Bequ.T
165	Atlog1_Arbitreqemit_Outreq_0.T	206	Tbreg5_Cmp3_Rescomp.T
166	Atlog1_Arbitreqemit_Outreq_1.T	207	Tbreg5_Cmp4_Bequ.T
167	Atlog1_Arbitreqemit_Outreq_2.T	208	Tbreg5_Cmp4_Rescomp.T
168	Atlog1_Arbitreqemit_Outreq_3.T	209	Tbreg5_Cmp5_Bequ.T
169	Atlog1_Arbitreqemit_Outreq_4.T	210	Tbreg5_Cmp5_Rescomp.T
170	Atlog1_Arbitreqemit_Outreq_5.T	211	Tbreg5_Cmp6_Bequ.T
171	Atlog1_Arbitreqemit_Outreq_6.T	212	Tbreg5_Cmp6_Rescomp.T
172	Atlog1_Arbitreqemit_Outreq_7.T	213	Tbreg5_Cmp7_Bequ.T
173	Atlog1_Glueat_Globerr.T	214	Tbreg5_Cmp7_Rescomp.

215	Atlog5_Arbitreqemit_Outreq_0.T	256	Oam1357_Oamlogic_Oa1_Outbas7.F
216	Atlog5_Arbitreqemit_Outreq_1.T	257	Oam1357_Oamlogic_Oa1_Ocounter0.T
217	Atlog5_Arbitreqemit_Outreq_2.T	258	Oam1357_Oamlogic_Oa1_Ocounter1.T
218	Atlog5_Arbitreqemit_Outreq_3.T	259	Oam1357_Oamlogic_Oa1_Ocounter2.T
219	Atlog5_Arbitreqemit_Outreq_4.T	260	Oam1357_Oamlogic_Oa1_Outbusy.F
220	Atlog5_Arbitreqemit_Outreq_5.T	261	Oam1357_Oamlogic_Oa2_Outbas0.F
221	Atlog5_Arbitreqemit_Outreq_6.T	262	Oam1357_Oamlogic_Oa2_Outbas1.F
222	Atlog5_Arbitreqemit_Outreq_7.T	263	Oam1357_Oamlogic_Oa2_Outbas2.F
223	Atlog5_Glueat_Globerr.T	264	Oam1357_Oamlogic_Oa2_Outbas3.F
224	Tbreg7_Cmp0_Bequ.T	265	Oam1357_Oamlogic_Oa2_Outbas4.F
225	Tbreg7_Cmp0_Rescomp.T	266	Oam1357_Oamlogic_Oa2_Outbas5.F
226	Tbreg7_Cmp1_Bequ.T	267	Oam1357_Oamlogic_Oa2_Outbas6.F
227	Tbreg7_Cmp1_Rescomp.T	268	Oam1357_Oamlogic_Oa2_Outbas7.F
228	Tbreg7_Cmp2_Bequ.T	269	Oam1357_Oamlogic_Oa2_Ocounter0.T
229	Tbreg7_Cmp2_Rescomp.T	270	Oam1357_Oamlogic_Oa2_Ocounter1.T
230	Tbreg7_Cmp3_Bequ.T	271	Oam1357_Oamlogic_Oa2_Ocounter2.T
231	Tbreg7_Cmp3_Rescomp.T	272	Oam1357_Oamlogic_Oa2_Outbusy.F
232	Tbreg7_Cmp4_Bequ.T	273	Oam1357_Oamlogic_Oa3_Outbas0.F
233	Tbreg7_Cmp4_Rescomp.T	274	Oam1357_Oamlogic_Oa3_Outbas1.F
234	Tbreg7_Cmp5_Bequ.T	275	Oam1357_Oamlogic_Oa3_Outbas2.F
235	Tbreg7_Cmp5_Rescomp.T	276	Oam1357_Oamlogic_Oa3_Outbas3.F
236	Tbreg7_Cmp6_Bequ.T	277	Oam1357_Oamlogic_Oa3_Outbas4.F
237	Tbreg7_Cmp6_Rescomp.T	278	Oam1357_Oamlogic_Oa3_Outbas5.F
238	Tbreg7_Cmp7_Bequ.T	279	Oam1357_Oamlogic_Oa3_Outbas6.F
239	Tbreg7_Cmp7_Rescomp.T	280	Oam1357_Oamlogic_Oa3_Outbas7.F
240	Atlog7_Arbitreqemit_Outreq_0.T	281	Oam1357_Oamlogic_Oa3_Ocounter0.T
241	Atlog7_Arbitreqemit_Outreq_1.T	282	Oam1357_Oamlogic_Oa3_Ocounter1.T
242	Atlog7_Arbitreqemit_Outreq_2.T	283	Oam1357_Oamlogic_Oa3_Ocounter2.T
243	Atlog7_Arbitreqemit_Outreq_3.T	284	Oam1357_Oamlogic_Oa3_Outbusy.F
244	Atlog7_Arbitreqemit_Outreq_4.T	285	Oam1357_Oamlogic_Oa4_Outbas0.F
245	Atlog7_Arbitreqemit_Outreq_5.T	286	Oam1357_Oamlogic_Oa4_Outbas1.F
246	Atlog7_Arbitreqemit_Outreq_6.T	287	Oam1357_Oamlogic_Oa4_Outbas2.F
247	Atlog7_Arbitreqemit_Outreq_7.T	288	Oam1357_Oamlogic_Oa4_Outbas3.F
248	Atlog7_Glueat_Globerr.T	289	Oam1357_Oamlogic_Oa4_Outbas4.F
249	Oam1357_Oamlogic_Oa1_Outbas0.F	290	Oam1357_Oamlogic_Oa4_Outbas5.F
250	Oam1357_Oamlogic_Oa1_Outbas1.F	291	Oam1357_Oamlogic_Oa4_Outbas6.F
251	Oam1357_Oamlogic_Oa1_Outbas2.F	292	Oam1357_Oamlogic_Oa4_Outbas7.F
252	Oam1357_Oamlogic_Oa1_Outbas3.F	293	Oam1357_Oamlogic_Oa4_Ocounter0.T
253	Oam1357_Oamlogic_Oa1_Outbas4.F	294	Oam1357_Oamlogic_Oa4_Ocounter1.T
254	Oam1357_Oamlogic_Oa1_Outbas5.F	295	Oam1357_Oamlogic_Oa4_Ocounter2.T
255	Oam1357_Oamlogic_Oa1_Outbas6.F	296	Oam1357_Oamlogic_Oa4_Outbusy.F

297	Lc1_Lclogic_Output_Credit_0.T	338	Lc1_Lclogic_Output_Mot2_5.T
298	Lc1_Lclogic_Output_Credit_1.T	339	Lc1_Lclogic_Output_Mot3_5.T
299	Lc1_Lclogic_Output_Credit_2.T	340	Lc1_Lclogic_Output_Mot4_5.T
300	Lc1_Lclogic_Output_Credit_3.T	341	Lc1_Lclogic_Output_Mot5_5.T
301	Lc1_Lclogic_Output_Credit_4.T	342	Lc1_Lclogic_Output_Mot1_6.T
302	Lc1_Lclogic_Output_Cred0.T	343	Lc1_Lclogic_Output_Mot2_6.T
303	Lc1_Lclogic_Output_Cred1.T	344	Lc1_Lclogic_Output_Mot3_6.T
304	Lc1_Lclogic_Output_Cred2.T	345	Lc1_Lclogic_Output_Mot4_6.T
305	Lc1_Lclogic_Output_Ptr_0.F	346	Lc1_Lclogic_Output_Mot5_6.T
306	Lc1_Lclogic_Output_Ptr_1.F	347	Lc1_Lclogic_Output_Mot1_7.T
307	Lc1_Lclogic_Output_Ptr_2.F	348	Lc1_Lclogic_Output_Mot2_7.T
308	Lc1_Lclogic_Output_Ptr_3.F	349	Lc1_Lclogic_Output_Mot3_7.T
309	Lc1_Lclogic_Output_Ptr_4.F	350	Lc1_Lclogic_Output_Mot4_7.T
310	Lc1_Lclogic_Output_Ptr_5.F	351	Lc1_Lclogic_Output_Mot5_7.T
311	Lc1_Lclogic_Output_Ptr_6.F	352	Lc1_Lclogic_Output_Mot1_8.T
312	Lc1_Lclogic_Output_Mot1_0.T	353	Lc1_Lclogic_Output_Mot2_8.T
313	Lc1_Lclogic_Output_Mot2_0.T	354	Lc1_Lclogic_Output_Mot3_8.T
314	Lc1_Lclogic_Output_Mot3_0.T	355	Lc1_Lclogic_Output_Mot4_8.T
315	Lc1_Lclogic_Output_Mot4_0.T	356	Lc1_Lclogic_Output_Mot5_8.T
316	Lc1_Lclogic_Output_Mot5_0.T	357	Lc1_Lclogic_Glue_Erronalink.T
317	Lc1_Lclogic_Output_Mot1_1.T	358	Lc1_Lclogic_Glue_Outffull.T
318	Lc1_Lclogic_Output_Mot2_1.T	359	Lc1_Lclogic_Glue_Fifofull.T
319	Lc1_Lclogic_Output_Mot3_1.T	360	Lc1_Lclogic_Glue_Tr0.T
320	Lc1_Lclogic_Output_Mot4_1.T	361	Lc1_Lclogic_Glue_Tr1.T
321	Lc1_Lclogic_Output_Mot5_1.T	362	Lc1_Lclogic_Glue_Tr2.T
322	Lc1_Lclogic_Output_Mot1_2.T	363	Lc1_Lclogic_Glue_Tr3.T
323	Lc1_Lclogic_Output_Mot2_2.T	364	Lc1_Lclogic_Glue_Tr4.T
324	Lc1_Lclogic_Output_Mot3_2.T	365	Lc1_Lclogic_Glue_Tr5.T
325	Lc1_Lclogic_Output_Mot4_2.T	366	Lc1_Lclogic_Glue_Tr6.T
326	Lc1_Lclogic_Output_Mot5_2.T	367	Lc1_Lclogic_Glue_Tr7.T
327	Lc1_Lclogic_Output_Mot1_3.T	368	Lc1_Lclogic_Glue_Tr8.T
328	Lc1_Lclogic_Output_Mot2_3.T	369	Lc1_Lclogic_Glue_Sigcmptini_0.T
329	Lc1_Lclogic_Output_Mot3_3.T	370	Lc1_Lclogic_Glue_Sigcmptini_1.T
330	Lc1_Lclogic_Output_Mot4_3.T	371	Lc1_Lclogic_Glue_Sigcmptini_2.T
331	Lc1_Lclogic_Output_Mot5_3.T	372	Lc1_Lclogic_Glue_Carry2.T
332	Lc1_Lclogic_Output_Mot1_4.T	373	Lc1_Lclogic_Glue_Motslus_Reg_0.T
333	Lc1_Lclogic_Output_Mot2_4.T	374	Lc1_Lclogic_Glue_Motslus_Reg_1.T
334	Lc1_Lclogic_Output_Mot3_4.T	375	Lc1_Lclogic_Glue_Motslus_Reg_2.T
335	Lc1_Lclogic_Output_Mot4_4.T	376	Lc1_Lclogic_Glue_Motslus_Reg_3.T
336	Lc1_Lclogic_Output_Mot5_4.T	377	Lc1_Lclogic_Glue_Motslus_Reg_4.T
337	Lc1_Lclogic_Output_Mot1_5.T	378	Lc1_Lclogic_Glue_Carry1.T

379	Lc1_Lclogic_Auto_Regs_0.T	420	Lc3_Lclogic_Output_Mot1_3.T
380	Lc1_Lclogic_Auto_Regs_1.T	421	Lc3_Lclogic_Output_Mot2_3.T
381	Lc1_Lclogic_Auto_Regs_2.T	422	Lc3_Lclogic_Output_Mot3_3.T
382	Lc1_Lclogic_Auto_Regs_3.T	423	Lc3_Lclogic_Output_Mot4_3.T
383	Lc1_Lclogic_Auto_Regs_4.T	424	Lc3_Lclogic_Output_Mot5_3.T
384	Lc1_Lclogic_Auto_Regs_5.T	425	Lc3_Lclogic_Output_Mot1_4.T
385	Lc1_Lclogic_Auto_Regs_6.T	426	Lc3_Lclogic_Output_Mot2_4.T
386	Lc1_Lclogic_Auto_Regs_7.T	427	Lc3_Lclogic_Output_Mot3_4.T
387	Lc1_Lclogic_Auto_Regs_8.T	428	Lc3_Lclogic_Output_Mot4_4.T
388	Lc1_Lclogic_Auto_Regs_9.T	429	Lc3_Lclogic_Output_Mot5_4.T
389	Lc1_Lclogic_Auto_Regs_10.T	430	Lc3_Lclogic_Output_Mot1_5.T
390	Lc3_Lclogic_Output_Credit_0.T	431	Lc3_Lclogic_Output_Mot2_5.T
391	Lc3_Lclogic_Output_Credit_1.T	432	Lc3_Lclogic_Output_Mot3_5.T
392	Lc3_Lclogic_Output_Credit_2.T	433	Lc3_Lclogic_Output_Mot4_5.T
393	Lc3_Lclogic_Output_Credit_3.T	434	Lc3_Lclogic_Output_Mot5_5.T
394	Lc3_Lclogic_Output_Credit_4.T	435	Lc3_Lclogic_Output_Mot1_6.T
395	Lc3_Lclogic_Output_Cred0.T	436	Lc3_Lclogic_Output_Mot2_6.T
396	Lc3_Lclogic_Output_Cred1.T	437	Lc3_Lclogic_Output_Mot3_6.T
397	Lc3_Lclogic_Output_Cred2.T	438	Lc3_Lclogic_Output_Mot4_6.T
398	Lc3_Lclogic_Output_Ptr_0.F	439	Lc3_Lclogic_Output_Mot5_6.T
399	Lc3_Lclogic_Output_Ptr_1.F	440	Lc3_Lclogic_Output_Mot1_7.T
400	Lc3_Lclogic_Output_Ptr_2.F	441	Lc3_Lclogic_Output_Mot2_7.T
401	Lc3_Lclogic_Output_Ptr_3.F	442	Lc3_Lclogic_Output_Mot3_7.T
402	Lc3_Lclogic_Output_Ptr_4.F	443	Lc3_Lclogic_Output_Mot4_7.T
403	Lc3_Lclogic_Output_Ptr_5.F	444	Lc3_Lclogic_Output_Mot5_7.T
404	Lc3_Lclogic_Output_Ptr_6.F	445	Lc3_Lclogic_Output_Mot1_8.T
405	Lc3_Lclogic_Output_Mot1_0.T	446	Lc3_Lclogic_Output_Mot2_8.T
406	Lc3_Lclogic_Output_Mot2_0.T	447	Lc3_Lclogic_Output_Mot3_8.T
407	Lc3_Lclogic_Output_Mot3_0.T	448	Lc3_Lclogic_Output_Mot4_8.T
408	Lc3_Lclogic_Output_Mot4_0.T	449	Lc3_Lclogic_Output_Mot5_8.T
409	Lc3_Lclogic_Output_Mot5_0.T	450	Lc3_Lclogic_Glue_Erronalink.T
410	Lc3_Lclogic_Output_Mot1_1.T	451	Lc3_Lclogic_Glue_Outffull.T
411	Lc3_Lclogic_Output_Mot2_1.T	452	Lc3_Lclogic_Glue_Fifofull.T
412	Lc3_Lclogic_Output_Mot3_1.T	453	Lc3_Lclogic_Glue_Tr0.T
413	Lc3_Lclogic_Output_Mot4_1.T	454	Lc3_Lclogic_Glue_Tr1.T
414	Lc3_Lclogic_Output_Mot5_1.T	455	Lc3_Lclogic_Glue_Tr2.T
415	Lc3_Lclogic_Output_Mot1_2.T	456	Lc3_Lclogic_Glue_Tr3.T
416	Lc3_Lclogic_Output_Mot2_2.T	457	Lc3_Lclogic_Glue_Tr4.T
417	Lc3_Lclogic_Output_Mot3_2.T	458	Lc3_Lclogic_Glue_Tr5.T
418	Lc3_Lclogic_Output_Mot4_2.T	458	Lc3_Lclogic_Glue_Tr6.T
419	Lc3_Lclogic_Output_Mot5_2.T	460	Lc3_Lclogic_Glue_Tr7.T

461	Lc3_Lclogic_Glue_Tr8.T	502	Lc5_Lclogic_Output_Mot5_0.T
462	Lc3_Lclogic_Glue_Sigcmptini_0.T	503	Lc5_Lclogic_Output_Mot1_1.T
463	Lc3_Lclogic_Glue_Sigcmptini_1.T	504	Lc5_Lclogic_Output_Mot2_1.T
464	Lc3_Lclogic_Glue_Sigcmptini_2.T	505	Lc5_Lclogic_Output_Mot3_1.T
465	Lc3_Lclogic_Glue_Carry2.T	506	Lc5_Lclogic_Output_Mot4_1.T
466	Lc3_Lclogic_Glue_Motslus_Reg_0.T	507	Lc5_Lclogic_Output_Mot5_1.T
467	Lc3_Lclogic_Glue_Motslus_Reg_1.T	508	Lc5_Lclogic_Output_Mot1_2.T
468	Lc3_Lclogic_Glue_Motslus_Reg_2.T	509	Lc5_Lclogic_Output_Mot2_2.T
469	Lc3_Lclogic_Glue_Motslus_Reg_3.T	510	Lc5_Lclogic_Output_Mot3_2.T
470	Lc3_Lclogic_Glue_Motslus_Reg_4.T	511	Lc5_Lclogic_Output_Mot4_2.T
471	Lc3_Lclogic_Glue_Carry1.T	512	Lc5_Lclogic_Output_Mot5_2.T
472	Lc3_Lclogic_Auto_Regs_0.T	513	Lc5_Lclogic_Output_Mot1_3.T
473	Lc3_Lclogic_Auto_Regs_1.T	514	Lc5_Lclogic_Output_Mot2_3.T
474	Lc3_Lclogic_Auto_Regs_2.T	515	Lc5_Lclogic_Output_Mot3_3.T
475	Lc3_Lclogic_Auto_Regs_3.T	516	Lc5_Lclogic_Output_Mot4_3.T
476	Lc3_Lclogic_Auto_Regs_4.T	517	Lc5_Lclogic_Output_Mot5_3.T
477	Lc3_Lclogic_Auto_Regs_5.T	518	Lc5_Lclogic_Output_Mot1_4.T
478	Lc3_Lclogic_Auto_Regs_6.T	519	Lc5_Lclogic_Output_Mot2_4.T
479	Lc3_Lclogic_Auto_Regs_7.T	520	Lc5_Lclogic_Output_Mot3_4.T
480	Lc3_Lclogic_Auto_Regs_8.T	521	Lc5_Lclogic_Output_Mot4_4.T
481	Lc3_Lclogic_Auto_Regs_9.T	522	Lc5_Lclogic_Output_Mot5_4.T
482	Lc3_Lclogic_Auto_Regs_10.T	523	Lc5_Lclogic_Output_Mot1_5.T
483	Lc5_Lclogic_Output_Credit_0.T	524	Lc5_Lclogic_Output_Mot2_5.T
484	Lc5_Lclogic_Output_Credit_1.T	525	Lc5_Lclogic_Output_Mot3_5.T
485	Lc5_Lclogic_Output_Credit_2.T	526	Lc5_Lclogic_Output_Mot4_5.T
486	Lc5_Lclogic_Output_Credit_3.T	527	Lc5_Lclogic_Output_Mot5_5.T
487	Lc5_Lclogic_Output_Credit_4.T	528	Lc5_Lclogic_Output_Mot1_6.T
488	Lc5_Lclogic_Output_Cred0.T	529	Lc5_Lclogic_Output_Mot2_6.T
489	Lc5_Lclogic_Output_Cred1.T	530	Lc5_Lclogic_Output_Mot3_6.T
490	Lc5_Lclogic_Output_Cred2.T	531	Lc5_Lclogic_Output_Mot4_6.T
491	Lc5_Lclogic_Output_Ptr_0.F	532	Lc5_Lclogic_Output_Mot5_6.T
492	Lc5_Lclogic_Output_Ptr_1.F	533	Lc5_Lclogic_Output_Mot1_7.T
493	Lc5_Lclogic_Output_Ptr_2.F	534	Lc5_Lclogic_Output_Mot2_7.T
494	Lc5_Lclogic_Output_Ptr_3.F	535	Lc5_Lclogic_Output_Mot3_7.T
495	Lc5_Lclogic_Output_Ptr_4.F	536	Lc5_Lclogic_Output_Mot4_7.T
496	Lc5_Lclogic_Output_Ptr_5.F	537	Lc5_Lclogic_Output_Mot5_7.T
497	Lc5_Lclogic_Output_Ptr_6.F	538	Lc5_Lclogic_Output_Mot1_8.T
498	Lc5_Lclogic_Output_Mot1_0.T	539	Lc5_Lclogic_Output_Mot2_8.T
499	Lc5_Lclogic_Output_Mot2_0.T	540	Lc5_Lclogic_Output_Mot3_8.T
500	Lc5_Lclogic_Output_Mot3_0.T	541	Lc5_Lclogic_Output_Mot4_8.T
501	Lc5_Lclogic_Output_Mot4_0.T	542	Lc5_Lclogic_Output_Mot5_8.T

543	Lc5_Lclogic_Glue_Erronalink.T	584	Lc7_Lclogic_Output_Ptr_0.F
544	Lc5_Lclogic_Glue_Outffull.T	585	Lc7_Lclogic_Output_Ptr_1.F
545	Lc5_Lclogic_Glue_Fifofull.T	586	Lc7_Lclogic_Output_Ptr_2.F
546	Lc5_Lclogic_Glue_Tr0.T	587	Lc7_Lclogic_Output_Ptr_3.F
547	Lc5_Lclogic_Glue_Tr1.T	588	Lc7_Lclogic_Output_Ptr_4.F
548	Lc5_Lclogic_Glue_Tr2.T	589	Lc7_Lclogic_Output_Ptr_5.F
549	Lc5_Lclogic_Glue_Tr3.T	590	Lc7_Lclogic_Output_Ptr_6.F
550	Lc5_Lclogic_Glue_Tr4.T	591	Lc7_Lclogic_Output_Mot1_0.T
551	Lc5_Lclogic_Glue_Tr5.T	592	Lc7_Lclogic_Output_Mot2_0.T
552	Lc5_Lclogic_Glue_Tr6.T	593	Lc7_Lclogic_Output_Mot3_0.T
553	Lc5_Lclogic_Glue_Tr7.T	594	Lc7_Lclogic_Output_Mot4_0.T
554	Lc5_Lclogic_Glue_Tr8.T	595	Lc7_Lclogic_Output_Mot5_0.T
555	Lc5_Lclogic_Glue_Sigcmptini_0.T	596	Lc7_Lclogic_Output_Mot1_1.T
556	Lc5_Lclogic_Glue_Sigcmptini_1.T	597	Lc7_Lclogic_Output_Mot2_1.T
557	Lc5_Lclogic_Glue_Sigcmptini_2.T	598	Lc7_Lclogic_Output_Mot3_1.T
558	Lc5_Lclogic_Glue_Carry2.T	599	Lc7_Lclogic_Output_Mot4_1.T
559	Lc5_Lclogic_Glue_Motslus_Reg_0.T	600	Lc7_Lclogic_Output_Mot5_1.T
560	Lc5_Lclogic_Glue_Motslus_Reg_1.T	601	Lc7_Lclogic_Output_Mot1_2.T
561	Lc5_Lclogic_Glue_Motslus_Reg_2.T	602	Lc7_Lclogic_Output_Mot2_2.T
562	Lc5_Lclogic_Glue_Motslus_Reg_3.T	603	Lc7_Lclogic_Output_Mot3_2.T
563	Lc5_Lclogic_Glue_Motslus_Reg_4.T	604	Lc7_Lclogic_Output_Mot4_2.T
564	Lc5_Lclogic_Glue_Carry1.T	605	Lc7_Lclogic_Output_Mot5_2.T
565	Lc5_Lclogic_Auto_Regs_0.T	606	Lc7_Lclogic_Output_Mot1_3.T
566	Lc5_Lclogic_Auto_Regs_1.T	607	Lc7_Lclogic_Output_Mot2_3.T
567	Lc5_Lclogic_Auto_Regs_2.T	608	Lc7_Lclogic_Output_Mot3_3.T
568	Lc5_Lclogic_Auto_Regs_3.T	609	Lc7_Lclogic_Output_Mot4_3.T
569	Lc5_Lclogic_Auto_Regs_4.T	610	Lc7_Lclogic_Output_Mot5_3.T
570	Lc5_Lclogic_Auto_Regs_5.T	611	Lc7_Lclogic_Output_Mot1_4.T
571	Lc5_Lclogic_Auto_Regs_6.T	612	Lc7_Lclogic_Output_Mot2_4.T
572	Lc5_Lclogic_Auto_Regs_7.T	613	Lc7_Lclogic_Output_Mot3_4.T
573	Lc5_Lclogic_Auto_Regs_8.T	614	Lc7_Lclogic_Output_Mot4_4.T
574	Lc5_Lclogic_Auto_Regs_9.T	615	Lc7_Lclogic_Output_Mot5_4.T
575	Lc5_Lclogic_Auto_Regs_10.T	616	Lc7_Lclogic_Output_Mot1_5.T
576	Lc7_Lclogic_Output_Credit_0.T	617	Lc7_Lclogic_Output_Mot2_5.T
577	Lc7_Lclogic_Output_Credit_1.T	618	Lc7_Lclogic_Output_Mot3_5.T
578	Lc7_Lclogic_Output_Credit_2.T	619	Lc7_Lclogic_Output_Mot4_5.T
579	Lc7_Lclogic_Output_Credit_3.T	620	Lc7_Lclogic_Output_Mot5_5.T
580	Lc7_Lclogic_Output_Credit_4.T	621	Lc7_Lclogic_Output_Mot1_6.T
581	Lc7_Lclogic_Output_Cred0.T	622	Lc7_Lclogic_Output_Mot2_6.T
582	Lc7_Lclogic_Output_Cred1.T	623	Lc7_Lclogic_Output_Mot3_6.T
583	Lc7_Lclogic_Output_Cred2.T	624	Lc7_Lclogic_Output_Mot4_6.T

625	Lc7_Lclogic_Output_Mot5_6.T	666	Lc7_Lclogic_Auto_Regs_8.T
626	Lc7_Lclogic_Output_Mot1_7.T	667	Lc7_Lclogic_Auto_Regs_9.T
627	Lc7_Lclogic_Output_Mot2_7.T	668	Lc7_Lclogic_Auto_Regs_10.T
628	Lc7_Lclogic_Output_Mot3_7.T	669	Lc6_Lclogic_Output_Credit_0.T
629	Lc7_Lclogic_Output_Mot4_7.T	670	Lc6_Lclogic_Output_Credit_1.T
630	Lc7_Lclogic_Output_Mot5_7.T	671	Lc6_Lclogic_Output_Credit_2.T
631	Lc7_Lclogic_Output_Mot1_8.T	672	Lc6_Lclogic_Output_Credit_3.T
632	Lc7_Lclogic_Output_Mot2_8.T	673	Lc6_Lclogic_Output_Credit_4.T
633	Lc7_Lclogic_Output_Mot3_8.T	674	Lc6_Lclogic_Output_Cred0.T
634	Lc7_Lclogic_Output_Mot4_8.T	675	Lc6_Lclogic_Output_Cred1.T
635	Lc7_Lclogic_Output_Mot5_8.T	676	Lc6_Lclogic_Output_Cred2.T
636	Lc7_Lclogic_Glue_Erronalink.T	677	Lc6_Lclogic_Output_Ptr_0.F
637	Lc7_Lclogic_Glue_Outffull.T	678	Lc6_Lclogic_Output_Ptr_1.F
638	Lc7_Lclogic_Glue_Fifofull.T	679	Lc6_Lclogic_Output_Ptr_2.F
639	Lc7_Lclogic_Glue_Tr0.T	680	Lc6_Lclogic_Output_Ptr_3.F
640	Lc7_Lclogic_Glue_Tr1.T	681	Lc6_Lclogic_Output_Ptr_4.F
641	Lc7_Lclogic_Glue_Tr2.T	682	Lc6_Lclogic_Output_Ptr_5.F
642	Lc7_Lclogic_Glue_Tr3.T	683	Lc6_Lclogic_Output_Ptr_6.F
643	Lc7_Lclogic_Glue_Tr4.T	684	Lc6_Lclogic_Output_Mot1_0.T
644	Lc7_Lclogic_Glue_Tr5.T	685	Lc6_Lclogic_Output_Mot2_0.T
645	Lc7_Lclogic_Glue_Tr6.T	686	Lc6_Lclogic_Output_Mot3_0.T
646	Lc7_Lclogic_Glue_Tr7.T	687	Lc6_Lclogic_Output_Mot4_0.T
647	Lc7_Lclogic_Glue_Tr8.T	688	Lc6_Lclogic_Output_Mot5_0.T
648	Lc7_Lclogic_Glue_Sigcmptini_0.T	689	Lc6_Lclogic_Output_Mot1_1.T
649	Lc7_Lclogic_Glue_Sigcmptini_1.T	690	Lc6_Lclogic_Output_Mot2_1.T
650	Lc7_Lclogic_Glue_Sigcmptini_2.T	691	Lc6_Lclogic_Output_Mot3_1.T
651	Lc7_Lclogic_Glue_Carry2.T	692	Lc6_Lclogic_Output_Mot4_1.T
652	Lc7_Lclogic_Glue_Motslus_Reg_0.T	693	Lc6_Lclogic_Output_Mot5_1.T
653	Lc7_Lclogic_Glue_Motslus_Reg_1.T	694	Lc6_Lclogic_Output_Mot1_2.T
654	Lc7_Lclogic_Glue_Motslus_Reg_2.T	695	Lc6_Lclogic_Output_Mot2_2.T
655	Lc7_Lclogic_Glue_Motslus_Reg_3.T	696	Lc6_Lclogic_Output_Mot3_2.T
656	Lc7_Lclogic_Glue_Motslus_Reg_4.T	697	Lc6_Lclogic_Output_Mot4_2.T
657	Lc7_Lclogic_Glue_Carry1.T	698	Lc6_Lclogic_Output_Mot5_2.T
659	Lc7_Lclogic_Auto_Regs_0.T	699	Lc6_Lclogic_Output_Mot1_3.T
659	Lc7_Lclogic_Auto_Regs_1.T	700	Lc6_Lclogic_Output_Mot2_3.T
660	Lc7_Lclogic_Auto_Regs_2.T	701	Lc6_Lclogic_Output_Mot3_3.T
661	Lc7_Lclogic_Auto_Regs_3.T	702	Lc6_Lclogic_Output_Mot4_3.T
662	Lc7_Lclogic_Auto_Regs_4.T	703	Lc6_Lclogic_Output_Mot5_3.T
663	Lc7_Lclogic_Auto_Regs_5.T	704	Lc6_Lclogic_Output_Mot1_4.T
664	Lc7_Lclogic_Auto_Regs_6.T	705	Lc6_Lclogic_Output_Mot2_4.T
665	Lc7_Lclogic_Auto_Regs_7.T	706	Lc6_Lclogic_Output_Mot3_4.T

707	Lc6_Lclogic_Output_Mot4_4.T	748	Lc6_Lclogic_Glue_Motslus_Reg_3.T
708	Lc6_Lclogic_Output_Mot5_4.T	749	Lc6_Lclogic_Glue_Motslus_Reg_4.T
709	Lc6_Lclogic_Output_Mot1_5.T	750	Lc6_Lclogic_Glue_Carry1.T
710	Lc6_Lclogic_Output_Mot2_5.T	751	Lc6_Lclogic_Auto_Regs_0.T
711	Lc6_Lclogic_Output_Mot3_5.T	752	Lc6_Lclogic_Auto_Regs_1.T
712	Lc6_Lclogic_Output_Mot4_5.T	753	Lc6_Lclogic_Auto_Regs_2.T
713	Lc6_Lclogic_Output_Mot5_5.T	754	Lc6_Lclogic_Auto_Regs_3.T
714	Lc6_Lclogic_Output_Mot1_6.T	755	Lc6_Lclogic_Auto_Regs_4.T
715	Lc6_Lclogic_Output_Mot2_6.T	756	Lc6_Lclogic_Auto_Regs_5.T
716	Lc6_Lclogic_Output_Mot3_6.T	757	Lc6_Lclogic_Auto_Regs_6.T
717	Lc6_Lclogic_Output_Mot4_6.T	758	Lc6_Lclogic_Auto_Regs_7.T
718	Lc6_Lclogic_Output_Mot5_6.T	759	Lc6_Lclogic_Auto_Regs_8.T
719	Lc6_Lclogic_Output_Mot1_7.T	760	Lc6_Lclogic_Auto_Regs_9.T
720	Lc6_Lclogic_Output_Mot2_7.T	761	Lc6_Lclogic_Auto_Regs_10.T
721	Lc6_Lclogic_Output_Mot3_7.T	762	Lc4_Lclogic_Output_Credit_0.T
722	Lc6_Lclogic_Output_Mot4_7.T	763	Lc4_Lclogic_Output_Credit_1.T
723	Lc6_Lclogic_Output_Mot5_7.T	764	Lc4_Lclogic_Output_Credit_2.T
724	Lc6_Lclogic_Output_Mot1_8.T	765	Lc4_Lclogic_Output_Credit_3.T
725	Lc6_Lclogic_Output_Mot2_8.T	766	Lc4_Lclogic_Output_Credit_4.T
726	Lc6_Lclogic_Output_Mot3_8.T	767	Lc4_Lclogic_Output_Cred0.T
727	Lc6_Lclogic_Output_Mot4_8.T	768	Lc4_Lclogic_Output_Cred1.T
728	Lc6_Lclogic_Output_Mot5_8.T	769	Lc4_Lclogic_Output_Cred2.T
729	Lc6_Lclogic_Glue_Erronalink.T	770	Lc4_Lclogic_Output_Ptr_0.F
730	Lc6_Lclogic_Glue_Outffull.T	771	Lc4_Lclogic_Output_Ptr_1.F
731	Lc6_Lclogic_Glue_Fifofull.T	772	Lc4_Lclogic_Output_Ptr_2.F
732	Lc6_Lclogic_Glue_Tr0.T	773	Lc4_Lclogic_Output_Ptr_3.F
733	Lc6_Lclogic_Glue_Tr1.T	774	Lc4_Lclogic_Output_Ptr_4.F
734	Lc6_Lclogic_Glue_Tr2.T	775	Lc4_Lclogic_Output_Ptr_5.F
735	Lc6_Lclogic_Glue_Tr3.T	776	Lc4_Lclogic_Output_Ptr_6.F
736	Lc6_Lclogic_Glue_Tr4.T	777	Lc4_Lclogic_Output_Mot1_0.T
737	Lc6_Lclogic_Glue_Tr5.T	778	Lc4_Lclogic_Output_Mot2_0.T
738	Lc6_Lclogic_Glue_Tr6.T	779	Lc4_Lclogic_Output_Mot3_0.T
739	Lc6_Lclogic_Glue_Tr7.T	780	Lc4_Lclogic_Output_Mot4_0.T
740	Lc6_Lclogic_Glue_Tr8.T	781	Lc4_Lclogic_Output_Mot5_0.T
741	Lc6_Lclogic_Glue_Sigcmptini_0.T	782	Lc4_Lclogic_Output_Mot1_1.T
742	Lc6_Lclogic_Glue_Sigcmptini_1.T	783	Lc4_Lclogic_Output_Mot2_1.T
743	Lc6_Lclogic_Glue_Sigcmptini_2.T	784	Lc4_Lclogic_Output_Mot3_1.T
744	Lc6_Lclogic_Glue_Carry2.T	785	Lc4_Lclogic_Output_Mot4_1.T
745	Lc6_Lclogic_Glue_Motslus_Reg_0.T	786	Lc4_Lclogic_Output_Mot5_1.T
746	Lc6_Lclogic_Glue_Motslus_Reg_1.T	787	Lc4_Lclogic_Output_Mot1_2.T
747	Lc6_Lclogic_Glue_Motslus_Reg_2.T	788	Lc4_Lclogic_Output_Mot2_2.T

789	Lc4_Lclogic_Output_Mot3_2.T	830	Lc4_Lclogic_Glue_Tr5.T
790	Lc4_Lclogic_Output_Mot4_2.T	831	Lc4_Lclogic_Glue_Tr6.T
791	Lc4_Lclogic_Output_Mot5_2.T	832	Lc4_Lclogic_Glue_Tr7.T
792	Lc4_Lclogic_Output_Mot1_3.T	833	Lc4_Lclogic_Glue_Tr8.T
793	Lc4_Lclogic_Output_Mot2_3.T	834	Lc4_Lclogic_Glue_Sigcmptini_0.T
794	Lc4_Lclogic_Output_Mot3_3.T	835	Lc4_Lclogic_Glue_Sigcmptini_1.T
795	Lc4_Lclogic_Output_Mot4_3.T	836	Lc4_Lclogic_Glue_Sigcmptini_2.T
796	Lc4_Lclogic_Output_Mot5_3.T	837	Lc4_Lclogic_Glue_Carry2.T
797	Lc4_Lclogic_Output_Mot1_4.T	838	Lc4_Lclogic_Glue_Motslus_Reg_0.T
798	Lc4_Lclogic_Output_Mot2_4.T	839	Lc4_Lclogic_Glue_Motslus_Reg_1.T
799	Lc4_Lclogic_Output_Mot3_4.T	840	Lc4_Lclogic_Glue_Motslus_Reg_2.T
800	Lc4_Lclogic_Output_Mot4_4.T	841	Lc4_Lclogic_Glue_Motslus_Reg_3.T
801	Lc4_Lclogic_Output_Mot5_4.T	842	Lc4_Lclogic_Glue_Motslus_Reg_4.T
802	Lc4_Lclogic_Output_Mot1_5.T	843	Lc4_Lclogic_Glue_Carry1.T
803	Lc4_Lclogic_Output_Mot2_5.T	844	Lc4_Lclogic_Auto_Regs_0.T
804	Lc4_Lclogic_Output_Mot3_5.T	845	Lc4_Lclogic_Auto_Regs_1.T
805	Lc4_Lclogic_Output_Mot4_5.T	846	Lc4_Lclogic_Auto_Regs_2.T
806	Lc4_Lclogic_Output_Mot5_5.T	847	Lc4_Lclogic_Auto_Regs_3.T
807	Lc4_Lclogic_Output_Mot1_6.T	848	Lc4_Lclogic_Auto_Regs_4.T
808	Lc4_Lclogic_Output_Mot2_6.T	849	Lc4_Lclogic_Auto_Regs_5.T
809	Lc4_Lclogic_Output_Mot3_6.T	850	Lc4_Lclogic_Auto_Regs_6.T
810	Lc4_Lclogic_Output_Mot4_6.T	851	Lc4_Lclogic_Auto_Regs_7.T
811	Lc4_Lclogic_Output_Mot5_6.T	852	Lc4_Lclogic_Auto_Regs_8.T
812	Lc4_Lclogic_Output_Mot1_7.T	853	Lc4_Lclogic_Auto_Regs_9.T
813	Lc4_Lclogic_Output_Mot2_7.T	854	Lc4_Lclogic_Auto_Regs_10.T
814	Lc4_Lclogic_Output_Mot3_7.T	855	Lc2_Lclogic_Output_Credit_0.T
815	Lc4_Lclogic_Output_Mot4_7.T	856	Lc2_Lclogic_Output_Credit_1.T
816	Lc4_Lclogic_Output_Mot5_7.T	857	Lc2_Lclogic_Output_Credit_2.T
817	Lc4_Lclogic_Output_Mot1_8.T	858	Lc2_Lclogic_Output_Credit_3.T
818	Lc4_Lclogic_Output_Mot2_8.T	859	Lc2_Lclogic_Output_Credit_4.T
819	Lc4_Lclogic_Output_Mot3_8.T	860	Lc2_Lclogic_Output_Cred0.T
820	Lc4_Lclogic_Output_Mot4_8.T	861	Lc2_Lclogic_Output_Cred1.T
821	Lc4_Lclogic_Output_Mot5_8.T	862	Lc2_Lclogic_Output_Cred2.T
822	Lc4_Lclogic_Glue_Erronalink.T	863	Lc2_Lclogic_Output_Ptr_0.F
823	Lc4_Lclogic_Glue_Outffull.T	864	Lc2_Lclogic_Output_Ptr_1.F
824	Lc4_Lclogic_Glue_Fifofull.T	865	Lc2_Lclogic_Output_Ptr_2.F
825	Lc4_Lclogic_Glue_Tr0.T	866	Lc2_Lclogic_Output_Ptr_3.F
826	Lc4_Lclogic_Glue_Tr1.T	867	Lc2_Lclogic_Output_Ptr_4.F
827	Lc4_Lclogic_Glue_Tr2.T	868	Lc2_Lclogic_Output_Ptr_5.F
828	Lc4_Lclogic_Glue_Tr3.T	869	Lc2_Lclogic_Output_Ptr_6.F
829	Lc4_Lclogic_Glue_Tr4.T	870	Lc2_Lclogic_Output_Mot1_0.T

871	Lc2_Lclogic_Output_Mot2_0.T	912	Lc2_Lclogic_Output_Mot3_8.T
872	Lc2_Lclogic_Output_Mot3_0.T	913	Lc2_Lclogic_Output_Mot4_8.T
873	Lc2_Lclogic_Output_Mot4_0.T	914	Lc2_Lclogic_Output_Mot5_8.T
874	Lc2_Lclogic_Output_Mot5_0.T	915	Lc2_Lclogic_Glue_Erronalink.T
875	Lc2_Lclogic_Output_Mot1_1.T	916	Lc2_Lclogic_Glue_Outffull.T
876	Lc2_Lclogic_Output_Mot2_1.T	917	Lc2_Lclogic_Glue_Fifofull.T
877	Lc2_Lclogic_Output_Mot3_1.T	918	Lc2_Lclogic_Glue_Tr0.T
878	Lc2_Lclogic_Output_Mot4_1.T	919	Lc2_Lclogic_Glue_Tr1.T
879	Lc2_Lclogic_Output_Mot5_1.T	920	Lc2_Lclogic_Glue_Tr2.T
880	Lc2_Lclogic_Output_Mot1_2.T	921	Lc2_Lclogic_Glue_Tr3.T
881	Lc2_Lclogic_Output_Mot2_2.T	922	Lc2_Lclogic_Glue_Tr4.T
882	Lc2_Lclogic_Output_Mot3_2.T	923	Lc2_Lclogic_Glue_Tr5.T
883	Lc2_Lclogic_Output_Mot4_2.T	924	Lc2_Lclogic_Glue_Tr6.T
884	Lc2_Lclogic_Output_Mot5_2.T	925	Lc2_Lclogic_Glue_Tr7.T
885	Lc2_Lclogic_Output_Mot1_3.T	926	Lc2_Lclogic_Glue_Tr8.T
886	Lc2_Lclogic_Output_Mot2_3.T	927	Lc2_Lclogic_Glue_Sigcmptini_0.T
887	Lc2_Lclogic_Output_Mot3_3.T	928	Lc2_Lclogic_Glue_Sigcmptini_1.T
888	Lc2_Lclogic_Output_Mot4_3.T	929	Lc2_Lclogic_Glue_Sigcmptini_2.T
889	Lc2_Lclogic_Output_Mot5_3.T	930	Lc2_Lclogic_Glue_Carry2.T
890	Lc2_Lclogic_Output_Mot1_4.T	931	Lc2_Lclogic_Glue_Motslus_Reg_0.T
891	Lc2_Lclogic_Output_Mot2_4.T	932	Lc2_Lclogic_Glue_Motslus_Reg_1.T
892	Lc2_Lclogic_Output_Mot3_4.T	933	Lc2_Lclogic_Glue_Motslus_Reg_2.T
893	Lc2_Lclogic_Output_Mot4_4.T	934	Lc2_Lclogic_Glue_Motslus_Reg_3.T
894	Lc2_Lclogic_Output_Mot5_4.T	935	Lc2_Lclogic_Glue_Motslus_Reg_4.T
895	Lc2_Lclogic_Output_Mot1_5.T	936	Lc2_Lclogic_Glue_Carry1.T
896	Lc2_Lclogic_Output_Mot2_5.T	937	Lc2_Lclogic_Auto_Regs_0.T
897	Lc2_Lclogic_Output_Mot3_5.T	938	Lc2_Lclogic_Auto_Regs_1.T
898	Lc2_Lclogic_Output_Mot4_5.T	939	Lc2_Lclogic_Auto_Regs_2.T
899	Lc2_Lclogic_Output_Mot5_5.T	940	Lc2_Lclogic_Auto_Regs_3.T
900	Lc2_Lclogic_Output_Mot1_6.T	941	Lc2_Lclogic_Auto_Regs_4.T
901	Lc2_Lclogic_Output_Mot2_6.T	942	Lc2_Lclogic_Auto_Regs_5.T
902	Lc2_Lclogic_Output_Mot3_6.T	943	Lc2_Lclogic_Auto_Regs_6.T
903	Lc2_Lclogic_Output_Mot4_6.T	944	Lc2_Lclogic_Auto_Regs_7.T
904	Lc2_Lclogic_Output_Mot5_6.T	945	Lc2_Lclogic_Auto_Regs_8.T
905	Lc2_Lclogic_Output_Mot1_7.T	946	Lc2_Lclogic_Auto_Regs_9.T
906	Lc2_Lclogic_Output_Mot2_7.T	947	Lc2_Lclogic_Auto_Regs_10.T
907	Lc2_Lclogic_Output_Mot3_7.T	948	Lc0_Lclogic_Output_Credit_0.T
908	Lc2_Lclogic_Output_Mot4_7.T	949	Lc0_Lclogic_Output_Credit_1.T
909	Lc2_Lclogic_Output_Mot5_7.T	950	Lc0_Lclogic_Output_Credit_2.T
910	Lc2_Lclogic_Output_Mot1_8.T	951	Lc0_Lclogic_Output_Credit_3.T
911	Lc2_Lclogic_Output_Mot2_8.T	952	Lc0_Lclogic_Output_Credit_4.T

953	Lc0_Lclogic_Output_Cred0.T	997	Lc0_Lclogic_Output_Mot5_6.T
954	Lc0_Lclogic_Output_Cred1.T	998	Lc0_Lclogic_Output_Mot1_7.T
955	Lc0_Lclogic_Output_Cred2.T	999	Lc0_Lclogic_Output_Mot2_7.T
956	Lc0_Lclogic_Output_Ptr_0.F	1000	Lc0_Lclogic_Output_Mot3_7.T
957	Lc0_Lclogic_Output_Ptr_1.F	1001	Lc0_Lclogic_Output_Mot4_7.T
958	Lc0_Lclogic_Output_Ptr_2.F	1002	Lc0_Lclogic_Output_Mot5_7.T
959	Lc0_Lclogic_Output_Ptr_3.F	1003	Lc0_Lclogic_Output_Mot1_8.T
960	Lc0_Lclogic_Output_Ptr_4.F	1004	Lc0_Lclogic_Output_Mot2_8.T
961	Lc0_Lclogic_Output_Ptr_5.F	1005	Lc0_Lclogic_Output_Mot3_8.T
962	Lc0_Lclogic_Output_Ptr_6.F	1006	Lc0_Lclogic_Output_Mot4_8.T
963	Lc0_Lclogic_Output_Mot1_0.T	1007	Lc0_Lclogic_Output_Mot5_8.T
964	Lc0_Lclogic_Output_Mot2_0.T	1008	Lc0_Lclogic_Glue_Erronalink.T
965	Lc0_Lclogic_Output_Mot3_0.T	1009	Lc0_Lclogic_Glue_Outffull.T
966	Lc0_Lclogic_Output_Mot4_0.T	1010	Lc0_Lclogic_Glue_Fifofull.T
967	Lc0_Lclogic_Output_Mot5_0.T	1011	Lc0_Lclogic_Glue_Tr0.T
968	Lc0_Lclogic_Output_Mot1_1.T	1012	Lc0_Lclogic_Glue_Tr1.T
969	Lc0_Lclogic_Output_Mot2_1.T	1013	Lc0_Lclogic_Glue_Tr2.T
970	Lc0_Lclogic_Output_Mot3_1.T	1014	Lc0_Lclogic_Glue_Tr3.T
971	Lc0_Lclogic_Output_Mot4_1.T	1015	Lc0_Lclogic_Glue_Tr4.T
972	Lc0_Lclogic_Output_Mot5_1.T	1016	Lc0_Lclogic_Glue_Tr5.T
973	Lc0_Lclogic_Output_Mot1_2.T	1017	Lc0_Lclogic_Glue_Tr6.T
974	Lc0_Lclogic_Output_Mot2_2.T	1018	Lc0_Lclogic_Glue_Tr7.T
975	Lc0_Lclogic_Output_Mot3_2.T	1019	Lc0_Lclogic_Glue_Tr8.T
976	Lc0_Lclogic_Output_Mot4_2.T	1020	Lc0_Lclogic_Glue_Sigcmptini_0.T
977	Lc0_Lclogic_Output_Mot5_2.T	1021	Lc0_Lclogic_Glue_Sigcmptini_1.T
978	Lc0_Lclogic_Output_Mot1_3.T	1022	Lc0_Lclogic_Glue_Sigcmptini_2.T
979	Lc0_Lclogic_Output_Mot2_3.T	1023	Lc0_Lclogic_Glue_Carry2.T
980	Lc0_Lclogic_Output_Mot3_3.T	1024	Lc0_Lclogic_Glue_Motslus_Reg_0.T
981	Lc0_Lclogic_Output_Mot4_3.T	1025	Lc0_Lclogic_Glue_Motslus_Reg_1.T
982	Lc0_Lclogic_Output_Mot5_3.T	1026	Lc0_Lclogic_Glue_Motslus_Reg_2.T
983	Lc0_Lclogic_Output_Mot1_4.T	1027	Lc0_Lclogic_Glue_Motslus_Reg_3.T
984	Lc0_Lclogic_Output_Mot2_4.T	1028	Lc0_Lclogic_Glue_Motslus_Reg_4.T
985	Lc0_Lclogic_Output_Mot3_4.T	1029	Lc0_Lclogic_Glue_Carry1.T
986	Lc0_Lclogic_Output_Mot4_4.T	1030	Lc0_Lclogic_Auto_Regs_0.T
987	Lc0_Lclogic_Output_Mot5_4.T	1031	Lc0_Lclogic_Auto_Regs_1.T
988	Lc0_Lclogic_Output_Mot1_5.T	1032	Lc0_Lclogic_Auto_Regs_2.T
989	Lc0_Lclogic_Output_Mot2_5.T	1033	Lc0_Lclogic_Auto_Regs_3.T
990	Lc0_Lclogic_Output_Mot3_5.T	1034	Lc0_Lclogic_Auto_Regs_4.T
991	Lc0_Lclogic_Output_Mot4_5.T	1035	Lc0_Lclogic_Auto_Regs_5.T
992	Lc0_Lclogic_Output_Mot5_5.T	1036	Lc0_Lclogic_Auto_Regs_6.T
993	Lc0_Lclogic_Output_Mot1_6.T	1037	Lc0_Lclogic_Auto_Regs_7.T
994	Lc0_Lclogic_Output_Mot2_6.T	1038	Lc0_Lclogic_Auto_Regs_8.T
995	Lc0_Lclogic_Output_Mot3_6.T	1039	Lc0_Lclogic_Auto_Regs_9.T
996	Lc0_Lclogic_Output_Mot4_6.T	1040	Lc0_Lclogic_Auto_Regs_10.T

1041	Fifo0_Rdout_0.T	1085	Fifo7_Rdout_0.T
1042	Fifo0_Rdout_1.T	1086	Fifo7_Rdout_1.T
1043	Fifo0_Rdout_2.T	1087	Fifo7_Rdout_2.T
1044	Fifo0_Rdout_3.T	1088	Fifo7_Rdout_3.T
1045	Fifo0_Rdout_4.T	1089	Fifo7_Rdout_4.T
1046	Fifo0_Rdout_5.T	1090	Fifo7_Rdout_5.T
1047	Fifo0_Rdout_6.T	1091	Fifo7_Rdout_6.T
1048	Fifo0_Rdout_7.T	1092	Fifo7_Rdout_7.T
1049	Fifo0_Rdout_8.T	1093	Fifo7_Rdout_8.T
1050	Fifo0_Rdout_9.T	1094	Fifo7_Rdout_9.T
1051	Fifo0_Rempty.T	1095	Fifo7_Rempty.T
1052	Fifo2_Rdout_0.T	1096	Fifo5_Rdout_0.T
1053	Fifo2_Rdout_1.T	1097	Fifo5_Rdout_1.T
1054	Fifo2_Rdout_2.T	1098	Fifo5_Rdout_2.T
1055	Fifo2_Rdout_3.T	1099	Fifo5_Rdout_3.T
1056	Fifo2_Rdout_4.T	1100	Fifo5_Rdout_4.T
1057	Fifo2_Rdout_5.T	1101	Fifo5_Rdout_5.T
1058	Fifo2_Rdout_6.T	1102	Fifo5_Rdout_6.T
1059	Fifo2_Rdout_7.T	1103	Fifo5_Rdout_7.T
1060	Fifo2_Rdout_8.T	1104	Fifo5_Rdout_8.T
1061	Fifo2_Rdout_9.T	1105	Fifo5_Rdout_9.T
1062	Fifo2_Rempty.T	1106	Fifo5_Rempty.T
1063	Fifo4_Rdout_0.T	1107	Fifo3_Rdout_0.T
1064	Fifo4_Rdout_1.T	1108	Fifo3_Rdout_1.T
1065	Fifo4_Rdout_2.T	1109	Fifo3_Rdout_2.T
1066	Fifo4_Rdout_3.T	1110	Fifo3_Rdout_3.T
1067	Fifo4_Rdout_4.T	1111	Fifo3_Rdout_4.T
1068	Fifo4_Rdout_5.T	1112	Fifo3_Rdout_5.T
1069	Fifo4_Rdout_6.T	1113	Fifo3_Rdout_6.T
1070	Fifo4_Rdout_7.T	1114	Fifo3_Rdout_7.T
1071	Fifo4_Rdout_8.T	1115	Fifo3_Rdout_8.T
1072	Fifo4_Rdout_9.T	1116	Fifo3_Rdout_9.T
1073	Fifo4_Rempty.T	1117	Fifo3_Rempty.T
1074	Fifo6_Rdout_0.T	1118	Fifo1_Rdout_0.T
1075	Fifo6_Rdout_1.T	1119	Fifo1_Rdout_1.T
1076	Fifo6_Rdout_2.T	1120	Fifo1_Rdout_2.T
1077	Fifo6_Rdout_3.T	1121	Fifo1_Rdout_3.T
1078	Fifo6_Rdout_4.T	1122	Fifo1_Rdout_4.T
1079	Fifo6_Rdout_5.T	1123	Fifo1_Rdout_5.T
1080	Fifo6_Rdout_6.T	1124	Fifo1_Rdout_6.T
1081	Fifo6_Rdout_7.T	1125	Fifo1_Rdout_7.T
1082	Fifo6_Rdout_8.T	1126	Fifo1_Rdout_8.T
1083	Fifo6_Rdout_9.T	1127	Fifo1_Rdout_9.T
1084	Fifo6_Rempty.T	1128	Fifo1_Rempty.T

6.2.2 SMIs scanpath description

This scanpath contains 368 bits. The SMIs are chained in the following order : scanin_smi -> SMI0,SMI2,SMI4,SMI6,SMI7,SMI5,SMI3,SMI1 -> scanout_smi. Bit 1 is written first.

1	smilogic.mux.reg_11	24	smilogic.automate.regs_7
2	smilogic.mux.reg_10	25	smilogic.automate.regs_8
3	smilogic.mux.reg_9	26	smilogic.automate.regs_9
4	smilogic.mux.reg_8	27	smilogic.huitversdouze.rbo_0
5	smilogic.mux.reg_7	28	smilogic.huitversdouze.rbo_1
6	smilogic.mux.reg_6	29	smilogic.huitversdouze.rbo_2
7	smilogic.mux.reg_5	30	smilogic.huitversdouze.rbo_3
8	smilogic.mux.reg_4	31	smilogic.huitversdouze.rbo_4
9	smilogic.mux.reg_3	32	smilogic.quatremille.carry_l1
10	smilogic.mux.reg_2	33	smilogic.quatremille.reg_1_5
11	smilogic.mux.reg_1	34	smilogic.quatremille.reg_1_4
12	smilogic.mux.reg_0	35	smilogic.quatremille.reg_1_3
13	smilogic.mux.poub	36	smilogic.quatremille.reg_2_2
14	smilogic.paraloop.fcc	37	smilogic.quatremille.reg_1_1
15	smilogic.paraloop.link_etat	38	smilogic.quatremille.reg_1_0
16	smilogic.paraloop.ap_reg	39	smilogic.quatremille.carry_l2
17	smilogic.automate.regs_0	40	smilogic.quatremille.reg_2_5
18	smilogic.automate.regs_1	41	smilogic.quatremille.reg_2_4
19	smilogic.automate.regs_2	42	smilogic.quatremille.reg_2_3
20	smilogic.automate.regs_3	43	smilogic.quatremille.reg_1_2
21	smilogic.automate.regs_4	44	smilogic.quatremille.reg_2_1
22	smilogic.automate.regs_5	45	smilogic.quatremille.reg_2_0
23	smilogic.automate.regs_6	46	smilogic.quatremille.counterstop

Table 6.6: SMI block scanpath description

6.2.3 ZC block scanpath description

Bit 1 is written first.

#bit	name	value after reset	#bit	name	value after reset
1	talr(0)	0	20	comp(2)	0
2	talr(1)	1	21	comp(3)	0
3	talpe(0)	0	22	comp(4)	0
4	talpe(1)	0	23	comp(5)	0
5	talpe(2)	1	24	openbouclr	0
6	talneb(0)	1	25	comn(1)	0
7	talneb(1)	1	26	comn(2)	0
8	talneb(2)	0	27	comn(3)	0
9	nrec(3)	0	28	comn(4)	0
10	nrec(2)	0	29	comn(5)	0
11	nrec(1)	0	30	scdecpe	1
12	nrec(0)	0	31	openbouclem	0
13	pem(2)	0	32	scdecne	0
14	pem(1)	0	33	binn_rec(0)	0
15	pem(0)	0	34	binn_rec(1)	0
16	nem(2)	0	35	binn_rec(2)	0
17	nem(1)	0	36	binn_rec(3)	0
18	nem(0)	0	37	scdecnr	1
19	comp(1)	0	38	testzc	0

Table 6.7: ZC block scanpath description

6.2.4 HSLs scanpath description

The scanpath of each HSL contains 217 bits. The HSLs are chained in the following order :
scanin_hsl -> HSL0,HSL2,HSL4,HSL6,HSL7,HSL5,HSL3,HSL1 -> scanout_hsl.
Bit 1 is written first.

1	olb	41	cmd1(3)	81	cmd7(1)	121	whod(0)	161	del4(3)	201	del10(1)
2	olbd	42	cmd1(4)	82	cmd7(2)	122	whod(1)	162	del4(4)	202	del10(2)
3	olb15	43	cmd1(5)	83	cmd7(3)	123	whod(2)	163	del4(5)	203	del10(3)
4	olb15r	44	dout(2)	84	cmd7(4)	124	whod(3)	164	din(4)	204	del10(4)
5	fincb	45	cmd2(0)	85	cmd7(5)	125	phi(0)	165	del5(0)	205	del10(5)
6	fdecb	46	cmd2(1)	86	dout(8)	126	phi(1)	166	del5(1)	206	din(10)
7	inhib0	47	cmd2(2)	87	cmd8(0)	127	phi(2)	167	del5(2)	207	del11(0)
8	inhib1	48	cmd2(3)	88	cmd8(1)	128	phi(3)	168	del5(3)	208	del11(1)
9	tsdpb	49	cmd2(4)	89	cmd8(2)	129	sdopen	169	del5(4)	209	del11(2)
10	cmd15(0)	50	cmd2(5)	90	cmd8(3)	130	del0(0)	170	del5(5)	210	del11(3)
11	cmd15(1)	51	dout(3)	91	cmd8(4)	131	del0(1)	171	din(5)	211	del11(4)
12	cmd15(2)	52	cmd3(0)	92	cmd8(5)	132	del0(2)	172	del6(0)	212	del11(5)
13	cmd15(3)	53	cmd3(1)	93	dout(9)	133	del0(3)	173	del6(1)	213	din(11)
14	cmd15(4)	54	cmd3(2)	94	cmd9(0)	134	del0(4)	174	del6(2)	214	who(0)
15	cmd15(5)	55	cmd3(3)	95	cmd9(1)	135	del0(5)	175	del6(3)	215	who(1)
16	dout(1)	56	cmd3(4)	96	cmd9(2)	136	din(0)	176	del6(4)	216	who(2)
17	phi15(0)	57	cmd3(5)	97	cmd9(3)	137	del1(0)	177	del6(5)	217	who(3)
18	phi15(1)	58	dout(4)	98	cmd9(4)	138	del1(1)	178	din(6)		
19	phi15(2)	59	cmd4(0)	99	cmd9(5)	139	del1(2)	179	del7(0)		
20	phi15(3)	60	cmd4(1)	100	dout(10)	140	del1(3)	180	del7(1)		
21	dec3(0)	61	cmd4(2)	101	cmd10(0)	141	del1(4)	181	del7(2)		
22	dec3(1)	62	cmd4(3)	102	cmd10(1)	142	del1(5)	182	del7(3)		
23	dec3(2)	63	cmd4(4)	103	cmd10(2)	143	din(1)	183	del7(4)		
24	phi15r(0)	64	cmd4(5)	104	cmd10(3)	144	del2(0)	184	del7(5)		
25	phi15r(1)	65	dout(5)	105	cmd10(4)	145	del2(1)	185	din(7)		
26	phi15r(2)	66	cmd5(0)	106	cmd10(5)	146	del2(2)	186	del8(0)		
27	phi15r(3)	67	cmd5(1)	107	dout(11)	147	del2(3)	187	del8(1)		
28	cmd15r(0)	68	cmd5(2)	108	cmd11(0)	148	del2(4)	188	del8(2)		
29	cmd15r(1)	79	cmd5(3)	109	cmd11(1)	149	del2(5)	189	del8(3)		
30	cmd15r(2)	70	cmd5(4)	110	cmd11(2)	150	din(2)	190	del8(4)		
31	cmd15r(3)	71	cmd5(5)	111	cmd11(3)	151	del3(0)	191	del8(5)		
32	cmd15r(4)	72	dout(6)	112	cmd11(4)	152	del3(1)	192	din(8)		
33	cmd15r(5)	73	cmd6(0)	113	cmd11(5)	153	del3(2)	193	del9(0)		
34	phid(0)	74	cmd6(1)	114	dout(0)	154	del3(3)	194	del9(1)		
35	phid(1)	75	cmd6(2)	115	cmd12(0)	155	del3(4)	195	del9(2)		
36	phid(2)	76	cmd6(3)	116	cmd12(1)	156	del3(5)	196	del9(3)		
37	phid(3)	77	cmd6(4)	117	cmd12(2)	157	din(3)	197	del9(4)		
38	cmd1(0)	78	cmd6(5)	118	cmd12(3)	158	del4(0)	198	del9(5)		
39	cmd1(1)	79	dout(7)	119	cmd12(4)	159	del4(1)	199	din(9)		
40	cmd1(2)	80	cmd7(0)	120	cmd12(5)	160	del4(2)	200	del10(0)		

Table 6.8: HSL macrocell scanpath description

7.FUNCTIONAL DESCRIPTION

7.1 Control characters

RCUBE uses three control characters which are the end of packet character EP, the exceptional end of packet character EEP and the flow control character FCC. The IDLE character is used to keep the serial macrocells calibrated when no data are sent.

name	coding [7:0]
EP	0010 1011
EEP	0010 1101
FCC	1111 1000
IDLE	0000 1111

7.2 Flow control mechanism

on serial link

RCUBE implements a flow control mechanism based on the sending and the receiving of a special control character called FCC.
When the link controller of a given link has consumed 32 characters (a flit) in its input fifo, it commands its output part to send a FCC on the link to inform the source that it can send another flit. If the output part is transmitting data when it receives that commands, it simply inserts the FCC in the flow. Similary, the output part of the link has an emission credit which is increased by 32 whenever the input part receives a FCC.

on parallel port

An example is shown on figure 7.1. A message of more than 64 characters is written into parallel port 0 (datain0[8:0]). The routing tables are loaded in order to route the message through several links configured in serial loop back mode and having it available on the parallel port 0 output.
Two pulses have been applied on **fccreceived0** giving port 0 an emission

credit of two flits (64 characters). When port 0 transmits characters, the signal **sendata0** is high. When 64 characters have been sent, port 0 waits for another emission credit. The remainder of the message is emitted only when another pulse occurs on **fccreceived0**.
Port 0 pulses the signal **sendfcc0**, for one **ckin** clock cycle, each time 32 characters have been routed. This informs the emitter that it can send another flit to port 0.

7.3 Serial link boot phase

7.3.1 Serial macrocell

name	I/O	active level	brief description
CKTX	I	-	transmission clock
DIN[11:0]	I	0xxxxxxxxxx1	parallel data to serialized
SDOUT	O	-	outgoing serialized data
RESET	I	1	macrocell reset
S_OP_EN	I	0	SDOUT forced to 0
CKR	O	-	recovered clock
DOUT[11:0]	O	-	deserialized data
SDIN	I	-	incoming serial data
CAL	O	1	deserializer is calibrated

The calibration phase between two serial macrocells connected through a coax transmission line is divided into 2 steps.

- When the signal RESET is released, the serializer starts calibrating. Since SDOUT is not well defined yet, it is set at 0 by maintaining S_OP_EN at 0. The calibration phase of the serializer lasts for about 2400 CKTX clock cycles. When S_OP_EN is set to 1, SDOUT is validated, the parallel data DIN is serialized.

- The deserializer must received IDLE charactères for at least 3000 CKR clock cycles to be calibrated. When the deserializer is calibrated, the signal CAL gets high.

Synchronization between the two macrocells is maintained as long as the transmission is continued and the parallel data follow the condition din[0]=1 and din[11]=0.

7.3.2 Boot Protocol

The boot protocol of the RCube HSL links is implemented in a finite state machine (one by serial link). The calibration procedure between two serial macrocells, belonging to two RCube A and B, and connected together (fig.7.3) is describe thereafter.

1. Just after **nreset** and **nreset_link[i]**$_i$ $= 0..7$ are released on a Rcube, every HSL link is reseted for 4096 **ckin** clock cycles. Every **sdout0..7** is forced at à 0.

2. Then ,the serializer calibration phase lasts for 4096 **ckin** clock cycles. Every **sdout0..7** is forced at à 0.

3. After that the serializer must be calibrated. The **sdout0..7** outputs are validated and begin emitting IDLE characters in order to calibrate the deserializer they are connected to.

4. Every deserializer receives IDLE characters. After a period that lasts at least 3000 recovered clock cycles, the deserializer is also calibrated.

5. Finally, a protocol based on the exchange of request and acknowledge control characters is used to test the good synchronisation between the two serial macrocell A and B. When this last phase is over, the signal **linkstatus[i]**$_{(i=0..7)}$ goes high indicating that the serial link[i] is working.

When the links are functional, RCUBE automatically sends on every link, two flow control characters FCC (spaced of 8 cycles) to inform the others devices that it is ready to receive two flits.

WARNING : when a device wants to inform RCUBE that it can receive two flits, it must space the sending of the two FCC characters by at least 4 cycles.

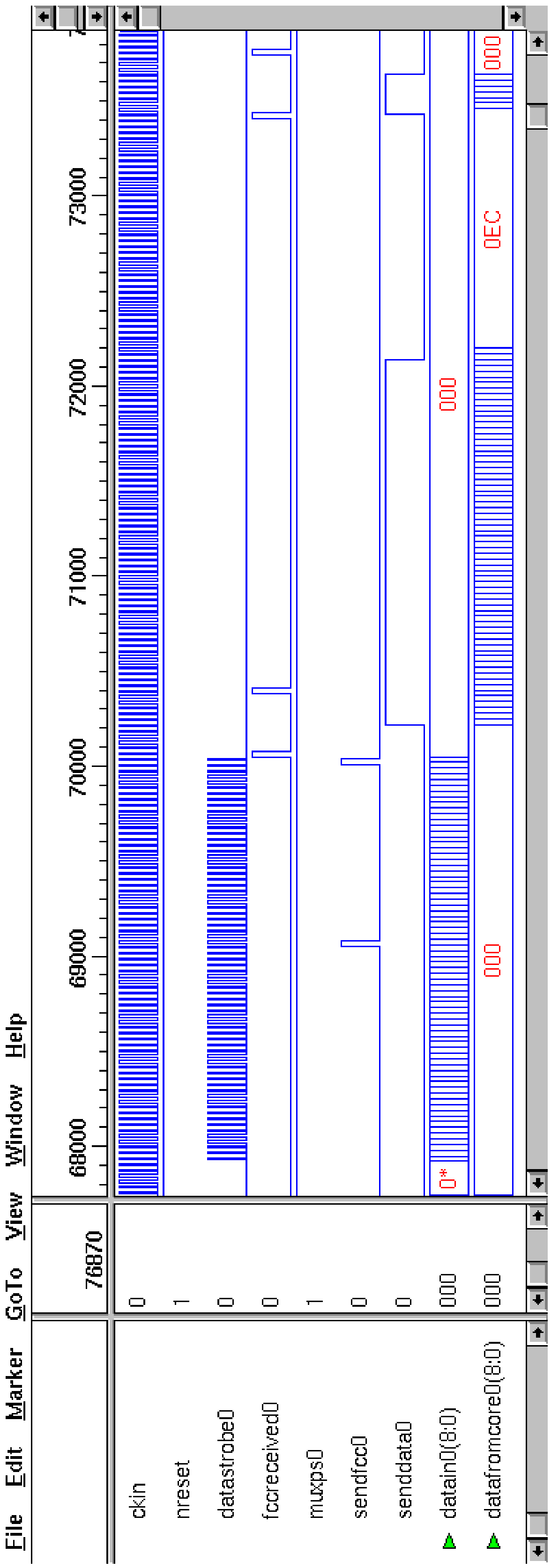


Figure 7.1: Flow Control On link 0

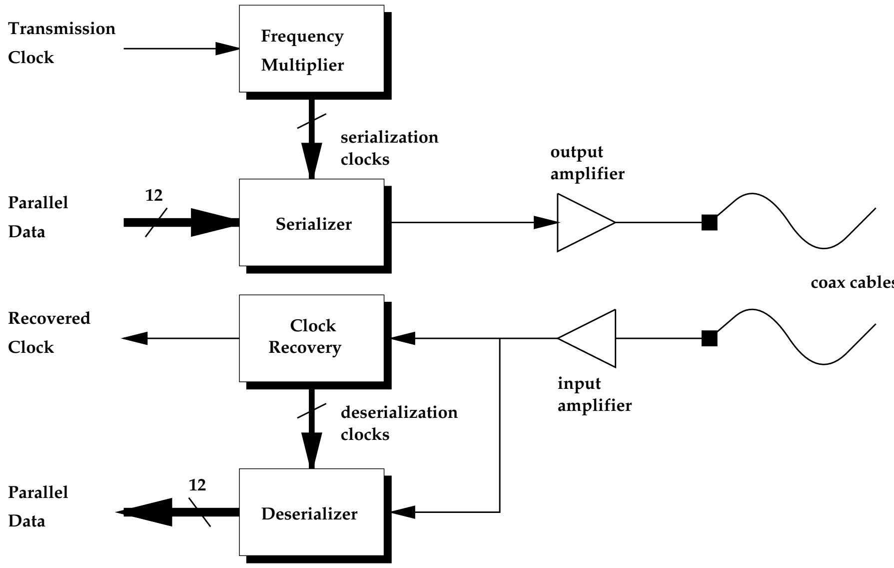


Figure 7.2: Serial Macrocell

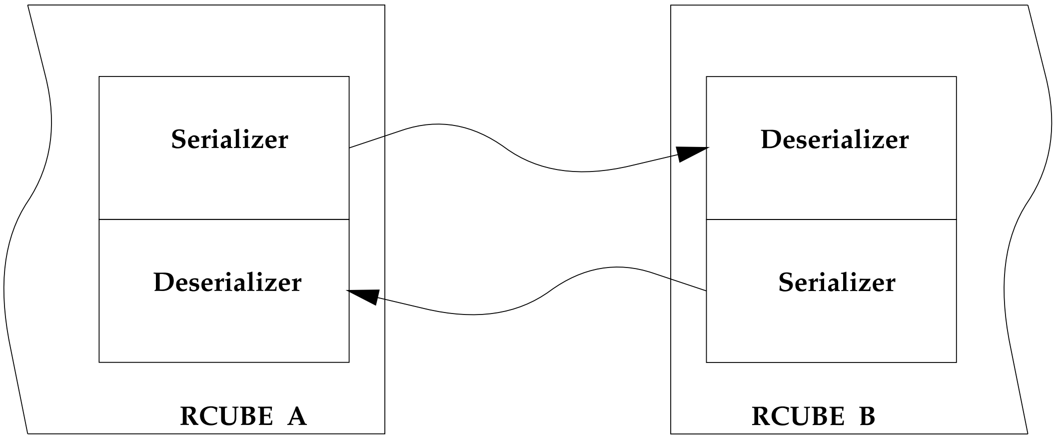


Figure 7.3: Macrocells connected together

8.ERROR DETECTION

RCUBE detects several types of errors :

- parity error on an incoming character.
- loss of calibration on a serial link.
- type error character.

In this version of RCUBE there is no recovering mechanism in case of loss of a FCC character.

8.1 Parity Error

The parity of an incoming character is checked. When a parity error occurs, the character is written in the asynchronous fifo with a parity flag set to '0'. When the character is read from the input fifo, the packet is cut, the faulty character is replaced by an EEP character and the output link is deallocated becoming available for another request. The link controller reads the content of the input fifo until it find an EP character in order to discard the rest of the incoming packet.

8.2 Calibration loss on serial link

When a loss of calibration occurs on a serial link, no more emission nor reception is possible on that link.

8.2.1 On receiver

If the error occurs when the link is receiving data and sending them through the crossbar, the packet is truncated and an exceptional end of packet character EEP is added at the end of the outgoing packet. Its input fifo is reseted and the link cannot be allocated until a boot phase has been executed. Since the emission stops, the RCube serial link at the other end of the cable loses its calibration and therefore also enters a boot phase.

8.2.2 On transmitter

If the error occurs when the link is transmitting data on the cable, it is deallocated, and the input link to which it was connected through the crossbar, flushes the content of its input fifo until it finds an EP or EEP character.

8.3 Type error

A type error occurs when an unexpected control character is received just after an end packet character in place of a header. The faulty character is replaced by an EEP and the link is deallocated. The link controller reads the input fifo to discard the rest of the packet until it finds an EP or an EEP character.

9.PACKAGING

9.1 PBGA480

pin no	pin name	I/O	pin no	pin name	I/O	pin no	pin name	I/O
1	dataout0[0]	O	37	VDDI		73	nreset_link2	I
2			38			74		
3	VDDE		39	VSSI		75	linkstatus4	O
4			40			76		
5	dataout0[1]	O	41	datain0[4]	I	77	recal4	I
6			42			78		
7	dataout0[2]	O	43	datain0[5]	I	79	VSSE	
8			44			80		
9	dataout0[3]	O	45	datain0[6]	I	81	nreset_link4	I
10			46			82		
11	dataout0[4]	O	47	datain0[7]	I	83	linkstatus6	O
12			48			84		
13	dataout0[5]	O	49			85	recal6	I
14			50			86		
15	dataout0[6]	O	51	datain0[8]	I	87	nreset_link6	I
16			52			88		
17	dataout0[7]	O	53	VDDE		89	test_hsl	I
18			54			90		
19	dataout0[8]	O	55	datastrobe0	I	91	scanin_smi	I
20			56			92		
21	VSSE		57	linkstatus0	O	93	test_smi	I
22			58			94		
23	senddata0	O	59			95	scanin_hsl	I
24			60			96		
25	sendfcc0	O	61	recal0	I	97	VDDI	
26			62			98		
27	fccreceived0	I	63	nreset_link0	I	99	VSSI	
28			64			100		
29	datain0[0]	I	65	linkstatus2	O	101	VDDE	
30			66			102		
31	datain0[1]	I	67	recal2	I	103	VSSE	
32			68			104		
33	datain0[2]	I	69	VSSI		105	zcscin	I
34			70			106		
35	datain0[3]	I	71	VDDI		107	ckm2	I
36			72					

pin no	pin name	I/O	pin no	pin name	I/O	pin no	pin name	I/O
1	dataout0[0]	O	37	VDDI		73	nreset_link2	I
2			38			74		
3	VDDE		39	VSSI		75	linkstatus4	O
4			40			76		
5	dataout0[1]	O	41	datain0[4]	I	77	recal4	I
6			42			78		
7	dataout0[2]	O	43	datain0[5]	I	79	VSSE	
8			44			80		
9	dataout0[3]	O	45	datain0[6]	I	81	nreset_link4	I
10			46			82		
11	dataout0[4]	O	47	datain0[7]	I	83	linkstatus6	O
12			48			84		
13	dataout0[5]	O	49			85	recal6	I
14			50			86		
15	dataout0[6]	O	51	datain0[8]	I	87	nreset_link6	I
16			52			88		
17	dataout0[7]	O	53	VDDE		89	test_hsl	I
18			54			90		
19	dataout0[8]	O	55	datastrobe0	I	91	scanin_smi	I
20			56			92		
21	VSSE		57	linkstatus0	O	93	test_smi	I
22			58			94		
23	senddata0	O	59			95	scanin_hsl	I
24			60			96		
25	sendfcc0	O	61	recal0	I	97	VDDI	
26			62			98		
27	fccreceived0	I	63	nreset_link0	I	99	VSSI	
28			64			100		
29	datain0[0]	I	65	linkstatus2	O	101	VDDE	
30			66			102		
31	datain0[1]	I	67	recal2	I	103	VSSE	
32			68			104		
33	datain0[2]	I	69	VSSI		105	zcscin	I
34			70			106		
35	datain0[3]	I	71	VDDI		107	ckm2	I
36			72					

pin no	pin name	I/O	pin no	pin name	I/O	pin no	pin name	I/O
108	VDD		144	VDD		180	VSS	
109			145			181		
110	sdout0	O	146	sdout6	O	182	sdin5	I
111			147			183		
112	VSS		148	VSS		184	VDD	
113			149			185		
114	VDD		150	VDD		186	VSS	
115			151			187		
116	sdin0	I	152	sdin6	I	188	sdout5	O
117			153			189		
118	VSS		154	VSS		190	VDD	
119			155			191		
120	VDD		156	VSS	O	192	VSS	
121			157			193		
122	sdout2	O	158	VDD		194	sdin3	I
123			159			195		
124	VSS		160	zin	I	196	VDD	
125			161			197		
126	VDD		162	zou	O	198	VSS	
127			163			199		
128	sdin2	I	164	VSS		200	sdout3	O
129			165			201		
130	VSS		166			202	VDD	
131			167			203		
132	VDD		168	VSS		204	VSS	
133			169			205		
134	sdout4	O	170	sdin7	I	206	sdin1	I
135			171			207		
136	VSS		172	VDD		208	VDD	
137			173			209		
138	VDD		174	VSS		210	VSS	
139			175			211		
140	sdin4	I	176	sdout7	O	212	sdout1	O
141			177			213		
142	VSS		178	VDD		214	VDD	
143			179					

pin no	pin name	I/O	pin no	pin name	I/O	pin no	pin name	I/O
215	ckm1	I	251	VDDI		287	datain1[3]	I
216			252			288		
217	zcscout	O	253	VSSI		289	datain1[2]	I
218			254			290		
219	VSSE		255	recal3	I	291	datain1[1]	I
220			256			292		
221	zctest	I	257	linkstatus3	O	293	VSSE	
222			258			294		
223	zcreset	I	259	nreset_link1	I	295	datain1[0]	I
224			260			296		
225	scanout_hsl	O	261	recal1	I	297	fccreceived1	I
226			262			298		
227	scanout_smi	O	263			299	sendfcc1	O
228			264			300		
229	VDDI		265	linkstatus1	O	301	VDDE	
230			266			302		
231	VSSI		267	VDDE		303	senddata1	O
232			268			304		
233	VDDE		269	datastrobe1	I	305	dataout1[8]	O
234			270			306		
235	nreset_link7	I	271	datain1[8]	I	307	dataout1[7]	O
236			272			308		
237	recal7	I	273			309	dataout1[6]	O
238			274			310		
239	linkstatus7	O	275	datain1[7]	I	311	dataout1[5]	O
240			276			312		
241	nreset_link5	I	277	datain1[6]	I	313	dataout1[4]	O
242			278			314		
243	VSSE		279	datain1[5]	I	315	dataout1[3]	O
244			280			316		
245	recal5	I	281	datain1[4]	I	317	dataout1[2]	O
246			282			318		
247	linkstatus5	O	283	VSSI		319	dataout1[1]	O
248			284			320		
249	nreset_link3	I	285	VDDI		321	dataout1[0]	O
250			286					

pin no	pin name	I/O	pin no	pin name	I/O	pin no	pin name	I/O
322	muxps1	I	358	VDDI		394	scaninconfig[4]	I
323			359			395		
324	VDDE		360	VSSI		396	scaninconfig[5]	I
325			361			397		
326	loop_par[1]	I	362	scanoutconfig[6]	O	398	scaninconfig[6]	I
327			363			399		
328	loop_ser[1]	I	364	VDDE		400	scaninconfig[7]	I
329			365			401		
330	loop_par[3]	I	366	scanoutconfig[7]	O	402	scanin	I
331			367			403		
332	loop_ser[3]	I	368	clockconfig	I	404	loop_par[6]	I
333			369			405		
334	loop_par[5]	I	370			406	loop_ser[6]	I
335			371			407		
336	loop_ser[5]	I	372	ckin	I	408	VDDE	
337			373			409		
338	loop_par[7]	I	374	test	I	410	loop_par[4]	I
339			375			411		
340	loop_ser[7]	I	376	nreset	I	412	loop_ser[4]	I
341			377			413		
342	VSSE		378	scaninconfig[0]	I	414	loop_par[2]	I
343			379			415		
344	scanout	O	380			416	loop_ser[2]	I
345			381			417		
346	scanoutconfig[0]	O	382	scaninconfig[1]	I	418	VDDI	
347			383			419		
348	scanoutconfig[1]	O	384	scaninconfig[2]	I	420	VSSI	
349			385			421		
350	scanoutconfig[2]	O	386	scaninconfig[3]	I	422	loop_par[0]	I
351			387			423		
352	scanoutconfig[3]	O	388	VSSE		424	VSSE	
353			389			425		
354	scanoutconfig[4]	O	390	VSSI		426	loop_ser[0]	I
355			391			427		
356	scanoutconfig[5]	O	392	VDDI		428	muxps0	I
357			393					

9.2 QFP208

We assume a cavity-up package.

9.2.1 West Face

pin no	pin name	I/O	pin no	pin name	I/O	pin no	pin name	I/O
208	dataout0[0]	O	190	VDDI		172	recal4	I
207	VDDE		189	VSSI		171	VSSE	
206	dataout0[1]	O	188	datain0[4]	I	170	nreset_link4	I
205	dataout0[2]	O	187	datain0[5]	I	169	linkstatus6	O
204	dataout0[3]	O	186	datain0[6]	I	168	recal6	I
203	dataout0[4]	O	185	datain0[7]	I	167	nreset_link6	I
202	dataout0[5]	O	184	datain0[8]	I	166	test_hsl	I
201	dataout0[6]	O	183	VDDE		165	scanin_smi	I
200	dataout0[7]	O	182	datastrobe0	I	164	test_smi	I
199	dataout0[8]	O	181	linkstatus0	O	163	scanin_hsl	I
198	VSSE		180	recal0	I	162	VDDI	
197	senddata0	O	179	nreset_link0	I	161	VSSI	
196	sendfcc0	O	178	linkstatus2	O	160	VDDE	
195	fccreceived0	I	177	recal2	I	159	VSSE	
194	datain0[0]	I	176	VSSI		158	zcscin	I
193	datain0[1]	I	175	VDDI		157	ckm2	I
192	datain0[2]	I	174	nreset_link2	I			
191	datain0[3]	I	173	linkstatus4	O			

9.2.2 North Face

pin no	pin name	I/O	pin no	pin name	I/O	pin no	pin name	I/O
156	VDD		138	VDD		120	VDD	
155	sdout0	O	137	sdout6	O	119	VSS	
154	VSS		136	VSS		118	sdout5	O
153	VDD		135	VDD		117	VDD	
152	sdin0	I	134	sdin6	I	116	VSS	
151	VSS		133	VSS		115	sdin3	I
150	VDD		132	VDD		114	VDD	
149	sdout2	O	131	zin	I	113	VSS	
148	VSS		130	zou	O	112	sdout3	O
147	VDD		129	VSS		111	VDD	
146	sdin2	I	128	VSS		110	VSS	
145	VSS		127	sdin7	I	109	sdin1	I
144	VDD		126	VDD		108	VDD	
143	sdout4	O	125	VSS		107	VSS	
142	VSS		124	sdout7	O	106	sdout1	O
141	VDD		123	VDD		105	VDD	
140	sdin4	I	122	VSS				
139	VSS		121	sdin5	I			

9.2.3 East Face

pin no	pin name	I/O	pin no	pin name	I/O	pin no	pin name	I/O
104	ckm1	I	86	VDDI		68	datain1[1]	I
103	zcscout	O	85	VSSI		67	VSSE	
102	VSSE		84	recal3	I	66	datain1[0]	I
101	zctest	I	83	linkstatus3	O	65	fccreceived1	I
100	zcreset	I	82	nreset_link1	I	64	sendfcc1	O
99	scanout_hsl	O	81	recal1	I	63	VDDE	
98	scanout_smi	O	80	linkstatus1	O	62	senddata1	O
97	VDDI		79	VDDE		61	dataout1[8]	O
96	VSSI		78	datastrobe1	I	60	dataout1[7]	O
95	VDDE		77	datain1[8]	I	59	dataout1[6]	O
94	nreset_link7	I	76	datain1[7]	I	58	dataout1[5]	O
93	recal7	I	75	datain1[6]	I	57	dataout1[4]	O
92	linkstatus7	O	74	datain1[5]	I	56	dataout1[3]	O
91	nreset_link5	I	73	datain1[4]	I	55	dataout1[2]	O
90	VSSE		72	VSSI		54	dataout1[1]	O
89	recal5	I	71	VDDI		53	dataout1[0]	O
88	linkstatus5	O	70	datain1[3]	I			
87	nreset_link3	I	69	datain1[2]	I			

9.2.4 South Face

pin no	pin name	I/O	pin no	pin name	I/O	pin no	pin name	I/O
52	muxps1	I	34	VDDI		16	scaninconfig[6]	I
51	VDDE		33	VSSI		15	scaninconfig[7]	I
50	loop_par[1]	I	32	scanoutconfig[6]	O	14	scanin	I
49	loop_ser[1]	I	31	VDDE		13	loop_par[6]	I
48	loop_par[3]	I	30	scanoutconfig[7]	O	12	loop_ser[6]	I
47	loop_ser[3]	I	29	clockconfig	I	11	VDDE	
46	loop_par[5]	I	28	ckin	I	10	loop_par[4]	I
45	loop_ser[5]	I	27	test	I	9	loop_ser[4]	I
44	loop_par[7]	I	26	nreset	I	8	loop_par[2]	I
43	loop_ser[7]	I	25	scaninconfig[0]	I	7	loop_ser[2]	I
42	VSSE		24	scaninconfig[1]	I	6	VDDI	
41	scanout	O	23	scaninconfig[2]	I	5	VSSI	
40	scanoutconfig[0]	O	22	scaninconfig[3]	I	4	loop_par[0]	I
39	scanoutconfig[1]	O	21	VSSE		3	VSSE	
38	scanoutconfig[2]	O	20	VSSI		2	loop_ser[0]	I
37	scanoutconfig[3]	O	19	VDDI		1	muxps0	I
36	scanoutconfig[4]	O	18	scaninconfig[4]	I			
35	scanoutconfig[5]	O	17	scaninconfig[5]	I			

Annexe.1

The user can specify a prioritary ordered set of output links for each interval. The priority handling is made by an arbitration tree whose functionality is described below.

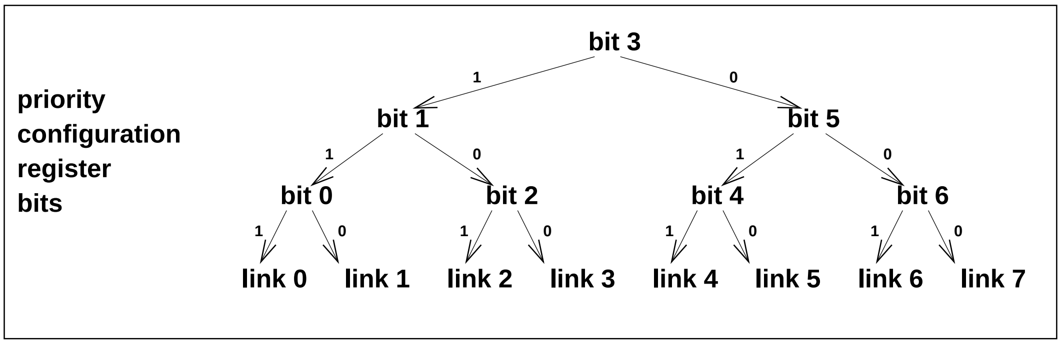


Figure 9.1: Arbitration tree

Only 7 bits of the 8-bits prioritary configuration register are actually used to specify the ordered list of output links.

bit number	bit value	highest priority output selected
0	0	1
0	1	0
1	0	3,2
1	1	0,1
2	0	3
2	1	2
3	0	4,5,6,7
3	1	0,1,2,3
4	0	5
4	1	4
5	0	6,7
5	1	4,5
6	0	7
6	1	6

This arbitration tree does not allow all the 8! possible orders of outputs but only 2^7.

interval number	priority configuration register value	out 7 prio rank	out 6 prio rank	out 5 prio rank	out 4 prio rank	out 3 prio rank	out 2 prio rank	out 1 prio rank	out 0 prio rank
0	x1101100	8	7	5	6	2	1	3	4
1	x0001011	5	6	7	8	3	4	1	2
2	x0000000	1	2	3	4	5	6	7	8
3	x0010000	1	2	4	3	5	6	7	8
4	x1101101	8	7	5	6	2	1	4	3
5	x0001000	5	6	7	8	1	2	3	4
6	x0011000	5	6	8	7	1	2	3	4
7	x1000101	2	1	3	4	6	5	8	7

Table 9.9: Example of priority register value

Application :
If we refer to the previous tables 5.1, 5.2 and figure 9.1, a packet whose header is 2403 will be routed towards output link 7 if it is available. Effectively, 2403 belongs to interval 0 whose possible outputs are 7,6,3,1,0 and the highest priority is 3.

Let's describe how do we get this.

The binary tree is used to order a set of output links associated to an interval. If a priority register associated to the interval contains the value xxxx1000, the ordering of the eventual allowed output links would be: 3 > 2 > 1 > 0 > 4,5,6,7.

This means that if output 3 is allowed (i.e specified in the selection register), it have the highest priority and it will be chosen unless it is busy. If it's the case, the next one is output 2. If this output 2 is allowed, it is taken. And so on.

Let consider a header whose value is 2403 (decimal). If you refer to routing table 5.1, you can see the bounds values (also given in decimal) which divide the address space into 8 intervals. 2403 belongs to the interval [501,2570[i.e interval 0.

Now refer to the selection table 5.2. You can see that the output links associated to interval 0 are : 7,6,3,1,0.

Now we have to determine which is the priority output. Refer to the figure 9.1. The priority register value for interval 0 is x1101100.
Now we will detail how the order is found.

In our example bit 3 is set to 1 so we can say now that:
link 0,1,2,3 > 4,5,6,7 where ">" means higher priority and "," means no ordering yet.

bit 1 is set to 0 and bit 5 is set to 1 so: link 2,3 > 0,1 > 4,5 > 6,7.

Finally bit 0 is set to 0, bit 2 is set to 1, bit 4 is set to 0 and bit 6 is set to 1. These four values lead to the following order: 2 > 3 > 1 > 0 > 5 > 4 > 6 > 7.

Since output 2 is not allowed, the next highest priority allowed output is 3.