1 ROM BIST

This chapter describes the ROM BIST used by ***Platform XXX - Hardware Revision Major.minor***

1.1 Features

- Configurable start and end address (20 bits)
- Configurable signature initialization
- Key protected start up
- Check with or without ECC
- Configurable ECC position (every 1, 2, 4 or 8 words)
- Automatic error signaling
- Resynchronized start up

1.2 Block diagram

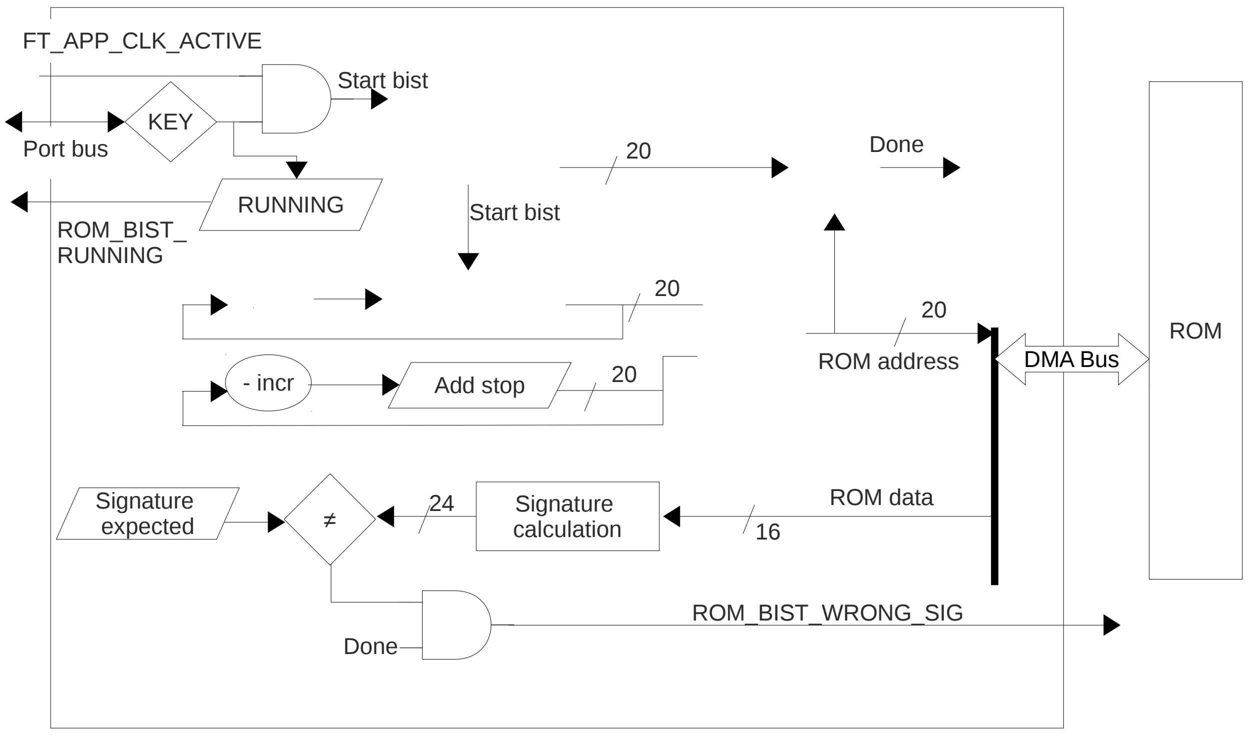

Figure 1: ROM BIST block diagram

On Figure 1, a simplified block diagram of the ROM BIST.

A BIST can start only when the block received a correct key and the application clock is present (FT_APP_CLK_ACTIVE=1).

It is composed of 2 registers to define the ROM address, the ROM address is alternatively the start address register and the stop address register. Find more details in chapter 1.5.2

A comparison is done between the initial start address and the ROM address accessed to determine when the BIST is finished.

All the data received from the ROM are send to the signature calculation block. At the end of the BIST, a comparison between the signature calculated and the signature expected is done, and a pulse on ROM_BIST_WRONG_SIG is generated if the comparison is wrong.

1.3 Hardware description

1.3.1 I/Os

Signal name	Direction	Description
ROM_BIST_WRONG_SIG	Output	Pulse indicating a wrong signature at the end of the BIST
ROM_BIST_RUNNING	Output	Set to 1 when a BIST is in progress

Table 1: ROM BIST I/Os

1.3.2 Parameters

Parameter	Value			Description
	Min	Max	Default	
PIPE_DEPTH	0	-	1	Depth of the pipe

Table 2: ROM BIST parameters

1.4 Ports definition

1.4.1 ADDRESS port

Offset		Reset	Access
0x00000		0x0000	Word, Byte
Field name	**Bit**	**R/W**	**Description**
ADD_START_L	[15:0]	W	Start address of the BIST (16 LSB) Note : Address has to be aligned on a word 32 bits (2 LSB are ignored)
		R	Current value of the start address pointer (16 LSB)

Offset		Reset	Access
0x00002		0x0000	Word, Byte
Field name	**Bit**	**R/W**	**Description**
-	[15:4]	R	Always read 0
		W	No effect
ADD_START_H	[3:0]	W	Start address of the BIST (4 MSB)
		R	Current value of the start address pointer (4 MSB)

Offset		Reset	Access
0x00004		0x0000	Word, Byte
Field name	**Bit**	**R/W**	**Description**
ADD_STOP_L	[15:0]	W	Stop address of the BIST (16 LSB)
		R	Current value of the stop address pointer (16 LSB) Note : LSB is always equal to 1

Offset		Reset	Access
0x00006		0x0000	Word, Byte
Field name	**Bit**	**R/W**	**Description**
-	[15:4]	R W	Always read 0 No effect
ADD_STOP_H	[3:0]	W R	Stop address of the BIST (4 MSB) Current value of the stop address pointer (4MSB)

The BIST is done in a specific interval. The start address is configured through the ports ADD_START_L, and ADD_START_H. The stop address is configured through the ports ADD_STOP_L, and ADD_STOP_H.

Important

- Address start is equal to the first even address aligned on a word 32 bits (2 LSB are always ignored)
- Address stop is equal to the last address accessed
- Memory has to return the ECC value on the last odd address of an entire ROM word.

Example:

For a memory 16 bytes, if the user wants to test all the memory:

Address start = 0x00000 (first address accessed is 0)

Address stop = 0x0000F (last address accessed is 15)

1.4.2 SIG_EXPECTED port

Offset		Reset	Access
0x00008		0x0000	Word, Byte
Field name	**Bit**	**R/W**	**Description**
SIG_EXPECTED_L	[15:0]	RW	Signature expected at the end of the BIST (16 LSB)

Offset		Reset	Access
0x0000A		0x0000	Word, Byte
Field name	**Bit**	**R/W**	**Description**
-	[15:8]	R W	Always read 0 No effect
SIG_EXPECTED_H	[7:0]	RW	Signature expected at the end of the BIST (8 MSB)

SIG_EXPECTED is the signature expected at the end of the BIST. If the signature calculated is different to this ports a pulse on ROM_BIST_WRONG_SIG is generated.

1.4.3 CONFIGURATION port

Offset	Reset	Access
0x0000C	0x0000	Word, Byte

Field name	Bit	R/W	Description
COMPLETED	15	R	1 = BIST completed 0 = BIST in progress
-	[14:12]	R W	Always read 0 No effect
VALID_CLOCK	11	R	1 = Application clock is valid 0 = Application clock is invalid
-	10	R W	Always read 0 No effect
BIST_REQUEST	9	R	1 = BIST requested 0 = No BIST requested
-	[8:7]	R W	Always read 0 No effect
MASK_SIG_ERR	6	RW	1 = Mask the error signal at the end of BIST if the signature expected is different to the signature calculated 0 = Do not mask the error signal at the end of BIST if the signature expected is different to the signature calculated
SINGLE_RAMP	5	RW	1 = Monotonic up BIST (only one pointer from add start to add stop used) 0 = Alternative BIST (two pointers alternating used)
BIST	4	RW	1 = Test ECC + non corrected values 0 = Test corrected values
-	[3:2]	R W	Always read 0 No effect
ECC_POSITION	[1:0]	RW	Indicates all how many words an ECC is present - 00 : Every word - 01 : Every 2 words - 10 : Every 4 words - 11 : Every 8 words

This port is a configuration port.

COMPLETED indicates if a BIST is in progress or not.

VALID_CLOCK indicates if the application clock is valid, if it is not the case a BIST could not start.

MASK_SIG_ERR allows to mask the signal indicating if the signature calculated is different to the signature expected. It can be used if the memory is test in multiple part, to enable the comparison between the signature expected and the signature calculated only on the last BIST.

SINGLE_RAMP selects how many pointers are used during the BIST (see chapter 1.5.3, 1.5.4 and 1.5.5 for more details).

- If SINGLE_RAMP = 1, only one pointer, starting at ADD_START and ending at ADD_STOP, is used
- If SINGLE_RAMP = 0, two pointers, one starting at ADD_START and ending at ADD_STOP and one starting at ADD_STOP and ending at ADD_START, are used alternatively.

BIST allows to select how the memory will be tested (see chapter 1.5.1 for more details).

- If BIST is set to 0, the memory is tested with the corrected values. Only the even addresses from ADD_START to ADD_STOP will be accessed.
- If BIST is set to 1, the memory is tested with the non corrected values. All the even addresses and all the odd addresses which return an ECC value from ADD_ START to ADD_STOP will be accessed.

A way to recover the ECC value is to read the odd address at the end of an entire ROM word. The field ECC_POSITION indicates all how many words (16 bytes) an ECC is present.

- For ROM 16 bits, there is an ECC every word
- For ROM 32 bits, there is an ECC every 2 words
- For ROM 64 bits, there is an ECC every 4 words
- For ROM 128 bits, there is an ECC every 8 words

See the example at chapter 1.7 to understand how the addresses are accessed.

1.4.4 START_BIST port

Offset		Reset	Access
0x0000E		0x0000	Word
Field name	**Bit**	**R/W**	**Description**
START_BIST	[15:0]	W	BIST request : KEY = 0x11EB
		R	Always read 0

To start a BIST, the key 0x11EB have to be written into this port. If the application clock is valid and no BIST is in progress, it starts immediately.

1.4.5 SIG_RECEIVED port

Offset		Reset	Access
0x00010		0x0001	Word, Byte
Field name	**Bit**	**R/W**	**Description**
SIG_RECEIVED_L	[15:0]	R	Signature calculated (16 LSB)
		W	Initialize the signature value (16 LSB)

Offset		Reset	Access
0x00012		0x0000	Word, Byte
Field name	**Bit**	**R/W**	**Description**
-	[15:8]	R	Always read 0
		W	No effect
SIG_RECEIVED_H	[7:0]	R	Signature calculated (8 MSB)
		W	Initialize the signature value (8 MSB)

The user have the possibility to initialize the signature before a BIST by writing the initialized value into this ports.

Important

– Before each new BIST, the signature have to be initialized. The commonly used value is 0x000001, but it could be an other one.

The signature received is accessible by reading this port.

1.5 Functional description

1.5.1 BIST principles

The principle of the memory BIST consists in reading all the words of a memory in a specific interval and generating a unique number, called signature. If the signature calculated is equal to the signature expected (given via a port) then the content of the memory is correct, else it is corrupted.

Some memories have an error code correction (ECC), which return for an address not the content of the memory but the content corrected.

It exists 2 philosophies to test a memory, the first one consists in testing the value corrected, it is not important to know if the value has been corrected or not, the most important thing is to have a correct result for a specific address.

The other philosophy consists in reading the real value stored in the memory (data + ECC) to be sure that no bit are incorrect (stuck at 0, stuck at 1,...)

1.5.2 Launch a BIST

To launch a BIST, the user have to:

1. Configure the start address in two times through the ports ADD_START_L and ADD_START_H
2. Configure the stop address in two times through the ports ADD_STOP_L and ADD_STOP_H
3. Configure the mode of the BIST, how many pointer used and ECC_POSITION through the configuration port.
4. Configure the signature expected through the port SIG_EXPECTED_L and SIG_EXPECTED_H
5. Configure the signature initialization through the port SIG_RECEIVED _L and SIG_RECEIVED _H if it is a new BIST
6. Write the key 0x11EB in the port START_BIST

If the application clock is already present and no BIST is running, the BIST starts immediately.

The key is used to prevent an unintentional BIST starting.

Figure 2: BIST with a good signature

Figure 3: BIST with a bad signature

When the key 0x11EB is written into the port START_BIST, the signal BIST_REQUEST is set to 1 and the signal COMPLETED is set to 0.

When the application clock is valid (FT_APP_CLK_ACTIVE = 1), the BIST sequence “really” starts. The memory is accessed through the wishbone bus (ROM_BIST_CYCLE = 1).

When the last data is received (DONE = 1), the BIST is finished, ROM_BIST_CYCLE and BIST_REQUEST are set to 0. The signal COMPLETED is set to 1 and if the signature calculated is different to the signature expected a pulse on ROM_BIST_WRONG_SIG is generated else it stays to 0.

1.5.3 BIST with field BIST equal to 0 and SINGLE_RAMP equal to 0

A BIST with the field BIST of the configuration port set to 0 consists in accessing all the corrected data of the memory and ignoring the ECC bits. So only even addresses are accessed in this mode.

Because field SINGLE_RAMP is set to 0 and to detect more faults, the memory is not read in a linear way. Two pointers are used to define the memory address accessed:

- A first pointer is initialized with the address start value and will be incremented by two until it is equal to address stop.
- A second pointer is initialized with the address stop value and will be decremented by two until it is equal to address start

Sequencing of the BIST:

1) Access the address indicating by the first pointer
2) Access the address indicating by the second pointer
3) Increment/Decrement the pointers by 2
4) Repeat step 1 to 3 until second pointer is equal to address start.

Example:

Address start = 0x4 Address stop = 0xA

Step	**First pointer value**	**Second pointer value**	**Address accessed**
Initialization	0x4	0xA	X
1	**0x4**	0xA	0x4
2	0x4	**0xA**	0xA
3	**0x6**	0x8	0x6
4	0x6	**0x8**	0x8
5	**0x8**	0x6	0x8
6	0x8	**0x6**	0x6
7	**0xA**	0x4	0xA
8	0xA	**0x4**	0x4

Notes:

Each address is accessed 2 times.

1.5.4 BIST with field BIST equal to 1 and SINGLE_RAMP equal to 0

A BIST with the field BIST of the configuration port set to 1 consists in accessing all the non-corrected data of the memory and all the ECC bits. Depends on the ROM size, ECC can be present every word (16 bits), 2 words, 4 words are 8 words, field ECC_POSITION indicates this information.

This BIST is similar of the BIST presented in chapter 1.5.3, there are 2 pointers (because SINGLE_RAMP = 0) but instead of always incrementing/decrementing them by 2 there incremented/decremented by 1 if an ECC is present on the next address or if the current address returns an ECC value (current address is mandatory odd), else by 2.

Sequencing of the BIST:

1) Access the address indicating by the first pointer
2) Access the address indicating by the second pointer
3) Increment/Decrement pointers
4) Repeat step 1 to 3 until second pointer is equal to address start.

Example 1:

Address start = 0x4 Address stop = 0x7 ECC present every word

Step	First pointer value	Second pointer value	Address accessed
Initialization	0x4	0x7	X
1	**0x4**	0x7	0x4
2	0x4 (+1, ECC in 0x5)	**0x7** (-1, to have even address)	0x7
3	**0x5**	0x6	0x5
4	0x5	**0x6** (-1, ECC in 0x5)	0x6
5	**0x6**	0x5	0x6
6	0x6 (+1, ECC in 0x7)	**0x5** (-1, to have even address)	0x5
7	**0x7**	0x4	0x7
8	0x7	**0x4**	0x4

Example 2:

Address start = 0x4 Address stop = 0xB ECC present every 2 words

Note :

ECC are present every 2 words so ROM return an ECC value at addresses 0x7 and 0xB

Step	First pointer value	Second pointer value	Address accessed
Initialization	0x4	0xB	X
1	**0x4**	0xB	0x4
2	0x4 (+2, no ECC in 0x5)	**0xB** (-1, to have even address)	0xB
3	**0x6**	0xA	0x6
4	0x6 (+1, ECC in 0x7)	**0xA** (-2, no ECC in 0x9)	0xA
5	**0x7**	0x8	0x7
6	0x7 (+1, to have even address)	**0x8** (-1, ECC in 0x7)	0x8
7	**0x8**	0x7	0x8
8	0x8 (+2, no ECC in 0x9)	**0x7** (-1, to have even address)	0x7
9	**0xA**	0x6	0xA
10	0xA (+1, ECC in 0xB)	**0x6** (-2, no ECC in 0x5)	0x6
11	**0xB**	0x4	0xB
12	0xB	**0x4**	0x4

1.5.5 BIST with field SINGLE_RAMP equal to 1

When the field SINGLE_RAMP is set to 1, the second pointer is not used and the ROM is accessed in a linear way.

- If the field BIST is set to 0, ROM returns corrected data and all even addresses from ADD_START to ADD_STOP are accessed.
- If the field BIST is set to 1, ROM returns not corrected data on even addresses and ECC value on odd addresses. All even addresses and all odd addresses containing an ECC from ADD_START to ADD_STOP are accessed.

Example 1:

Address start = 0x4 Address stop = 0xF BIST = 0

Step	Address accessed
1	0x4
2	0x6
3	0x8
4	0xA
5	0xC
6	0xE

Example 2:

Address start = 0x4 Address stop = 0xF BIST = 1 ECC present every 2 words

Step	Address accessed
1	0x4
2	0x6
3	0x7
4	0x8
5	0xA
6	0xB
7	0xC
8	0xE
9	0xF

1.5.6 BIST Duration

The duration of the BIST depends on 7 parameters:

- Length of the memory = Length = Address stop – Address start + 1
- Memory wait states = W (e.g W = 0 → Memory access in one clock period)
- Memory pipeline = P (e.g P = 0 → Memory is not pipelined)
- Mode of BIST = B (e.g B = 0 → BIST with no ECC reading, B = 1 → BIST with ECC reading)
- Number of Pointer = Po (e.g Po = 1 → 1 pointer used, Po = 2 → 2 pointers used)
- ECC positions = E (e.g E = 1 → an ECC every words, E = 2 → an ECC every 2 words,...)
- Clock period = T_{clk}

Note:

- Additional 2 clock periods are needed to take into account the BIST request and the BIST ending

1.6 Signature Calculation

The polynomial used to calculate the 24 bits signature is: $x^{24} + x^{23} + x^{22} + x^{17}$. The polynomial is a primitive polynomial, it means that for a memory only filled with 0x0000, a signature will be repeated all 65536 values. A polynomial 24 bits has been chosen because there are 20 bits of address so 2^{20} combinations and an error can occur on 1 bit of a byte, so finally there are 2^{20}*8 combinations which can be coded on 23 bits. For commodities a 24 bits polynomial has been chosen.

This polynomial combined with the memory contents have a very high probability to be unique.

A hardware description of the signature calculation is described in the Figure 4.

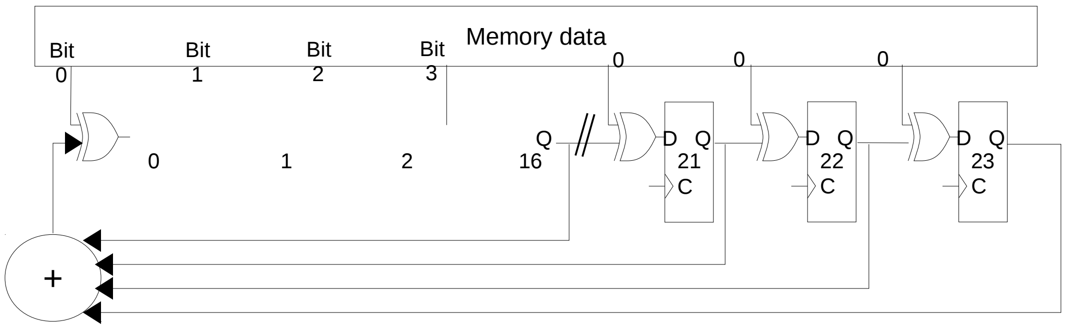

Figure 4: Signature calculation

The value at the initialization should be 0x000001, but the user can define another value by writing into SIGNATURE_RECEIVED.

1.7 Example

In this example, the memory is a 32 bits memory described below:

Address	0x00007	0x00006	0x00004	0x00003	0x00002	0x00000
Memory value	-	0x8002	0x8000	-	0x8002	0x8000
ECC value	0x0001	-	-	0x0003	-	-

Example: BIST of the entire memory, with 2 pointers and ECC verification

Sequencing:

1. Write 0x0000 into ADD_START_L
2. Write 0x0000 into ADD_START_H
3. Write 0x0007 into ADD_STOP_L
4. Write 0x0000 into ADD_STOP_H
5. Write 0x0011 into the configuration port (SINGLE_RAMP = 0, BIST = 1, ECC_POSITION = 01)
6. Write 0x94C9 into SIG_EXPECTED_L
7. Write 0x00B6 into SIG_EXPECTED_H
8. Write 0x0001 into SIG_RECEIVED_L
9. Write 0x0000 into SIG_RECEIVED_H
10. Write 0x11EB into the port ROMBIST_START_BIST

If the application clock is present and if the bus is not busy the BIST starts.

Step	**Address accessed**	**Memory data**	**Signature calculated**
Initialization	0xXXXXX	0xXXXX	0x000001
1	0x00000	0x8000	0x008002
2	0x00007	0x0001	0x010005
3	0x00002	0x8002	0x028009
4	0x00006	0x8006	0x058014
5	0x00003	0x0003	0x0B002A
6	0x00004	0x8004	0x168051
7	0x00004	0x8004	0x2D80A6
8(end of phase 1)	0x00003	0x0003	0x5B014F
9	0x00006	0x8006	0xB68298
10	0x00002	0x8002	0x6D8532
11	0x00007	0x0001	0xDB0A64
12	0x00000	0x0000	0xB694C9