

DOCUMENT TYPE: TECHNICAL NOTE

TITLE: HK/CONF BOARD FOR THE TEST EQUIPMENT OF
THE MINICALORIMETER PROTOTYPE

DOCUMENT Ref. No.: AGILE-ITE-TN-009 **N° OF PAGES:** i-iv, 30

ISSUE No.: 02 **DATE:** December 2002
IASF section of Bologna Report 355/02

PREPARED BY: A. BULGARELLI, M. PREST, E. VALLAZZA, E. CELESTI,
F. GIANOTTI, C. LABANTI, M. TRIFOGLIO

CHECKED BY: M. TRIFOGLIO

SUBSYSTEM MANAGER: M. TRIFOGLIO

APPROVED BY:

SUBSYSTEM LEADER: G. DI COCCO **DATE:**

PROJECT LEADER: M. TAVANI **DATE:**

PAYLOAD MANAGER: A. ZAMBRA **DATE:**

PAPM: A. BERNABEO **DATE:**

CONFIGURATION: C. MANGILI **DATE:**

[illegible]

CHANGE RECORD

Issue	Date	Page	Description of Changes	Release
01	18.09.02	All	First issue of the document	1
02	19.12.02		New version due to the new version of the Altera VHDL code after the test, as reported in the test report	

Saved date: 11 March 2003
Printed date: 20 September 2019

SUMMARY

DISTRIBUTION LIST.....	iii
CHANGE RECORD	iv

INTRODUCTION	1
--------------------	---

REFERENCE DOCUMENTS	2
---------------------------	---

LIST OF ACRONYMS.....	3
-----------------------	---

1. BOARD ARCHITECTURE OVERVIEW	4
--------------------------------------	---

1.1 SYSTEM OVERVIEW	6
---------------------------	---

2. FUNCTIONALITY	7
------------------------	---

3. DATA FORMAT AND SIGNAL INPUT/OUTPUT.....	8
---	---

3.1 VME SIGNAL INTERFACE	8
--------------------------------	---

3.2 BOARD INPUT SIGNAL.....	8
-----------------------------	---

3.3 BOARD OUTPUT SIGNAL.....	8
------------------------------	---

3.4 READING OPERATIONS	9
------------------------------	---

3.5 WRITING OPERATIONS.....	9
-----------------------------	---

3.6 SERIAL BUS SIGNAL FORMAT	10
------------------------------------	----

3.6.1 ADDRESS FORMAT FROM HC TO BOARD	10
---	----

3.6.2 ADDRESS FORMAT FROM BOARD TO DFE	10
--	----

3.6.3 DATA FORMAT	10
-------------------------	----

3.7 BASIC OPERATIONS	10
----------------------------	----

3.7.1 RESET MODE	10
------------------------	----

3.7.2 MCAL_RESET	11
------------------------	----

3.7.3 WRITE MODE.....	11
-----------------------	----

3.7.4 HK_PULSE READING MODE	12
-----------------------------------	----

3.7.5 READ MODE	14
-----------------------	----

3.7.6 READ LAST VALUE MODE.....	15
---------------------------------	----

3.8 BOARD UTILIZATION.....	16
----------------------------	----

3.9 CONNECTORS PIN FUNCTION.....	17
----------------------------------	----

3.10 ELECTRICAL SCHEMAS OF THE BOARD.....	18
---	----

4. BOARD DETAILED DESIGN	19
--------------------------------	----

4.1 MAIN I/O SIGNAL OF THE IC4 ALTERA PLD	19
---	----

4.2 MAIN INTERNAL SIGNAL OF THE IC4 ALTERA PLD	19
--	----

4.3 ADDRESS BOARD, ACTION AND MODE SELECTION	19
--	----

4.4 READ AND WRITE FROM VME TO BOARD	21
--	----

4.5 REAL AND TEST DATA	24
------------------------------	----

4.5.1 THE TEST DATA GENERATION.....	24
-------------------------------------	----

4.6 WRITING DATA INTO MEMORY	25
------------------------------------	----

4.7 ACTIONS.....	26
------------------	----

4.7.1 RESET OF THE DFE	26
------------------------------	----

4.7.2	ENABLE WRITE CYCLE	26
4.8	HOUSEKEEPING MANAGEMENT.....	27
5.	SOFTWARE	29
5.1	OPERATIONS	29
5.1.1	READING HOUSEKEEPING.....	29
5.2	INFN SOFTWARE	29
5.2.1	RESET BOARD.....	29
5.3	CNR/IASF SECTION OF BOLOGNA SOFTWARE	29
APPENDIX A	30
5.4	IC4.....	30

INDEX OF FIGURES

Figure 1: Board architecture overview.....	5
Figura 2: Reading operations	9
Figure 3: Writing operation.....	9
Figura 4: reset board operation.....	11
Figure 5: write operation	13
Figure 6: read operation	15
Figure 7: read last value	16
Figure 8: hardware layout of the board	18
Figura 9: from bitin to bitout.....	23
Figura 10: housekeeping reading operation	29
Figura 11: IC4	30

INDEX OF TABLES

Table 1: timing of reading operation signals.....	9
Table 2: timing of writing operation signals	9

INTRODUCTION

This HK/CONF board is a VME board designed to send and receive on a serial bus the configuration data for the

REFERENCE DOCUMENTS

1. 'Test equipment for MCAL FEE in BURST mode', T. V. Frøysland, 12-2001.
2. 'Agile MCAL_TE – MCAL front end electrical I/F', L. Nicolini, 02-2002.
3. 'Design report of the Proto MCAL Test Equipment for the Agile Minicalorimeter SEM model', M. Trifoglio et al., AGILE-ITE-RE-001, Tesre report 336/02, 02-2002.
4. E. Celesti et al., "*BURST board for the test equipment of the Minicalorimeter prototype*", AGILE-ITE-TN-007 issue 01, 02-2002
5. A Bulgarelli et al., "*GRID board for the test equipment of the Minicalorimeter prototype*", AGILE-ITE-TN-009 issue 01, 09-2002
6. DESIGN REPORT OF THE PROTO MCAL TEST EQUIPMENT FOR THE AGILE MINICALORIMETER SEM MODEL, A.Bulgarelli et al., AGILE-ITE-RE-001, February 2002, Te.S.R.E. Report 336/02

LIST OF ACRONYMS

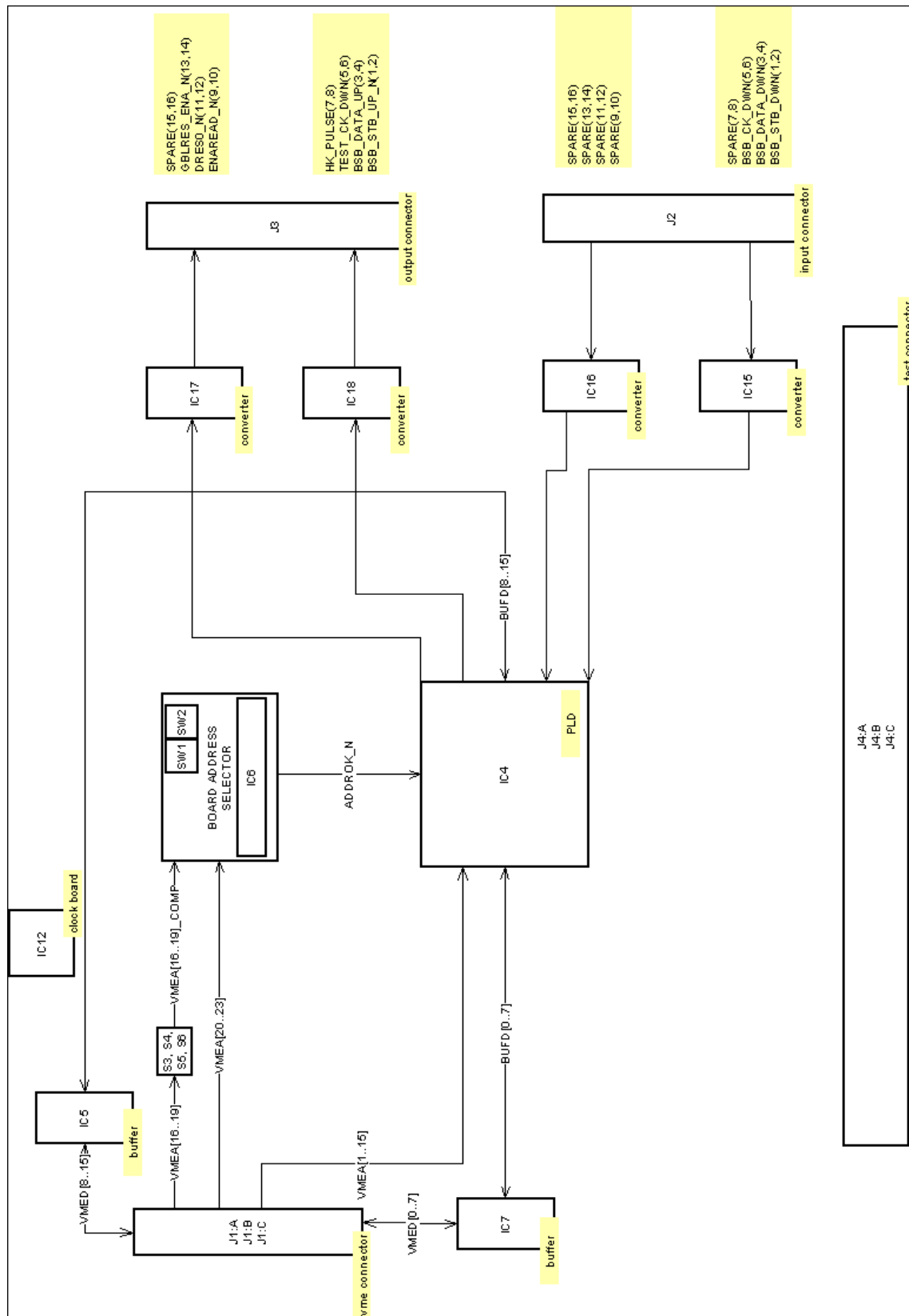
DFE	Digital Front End
FEE	Front End Electronic
HK	Housekeeping
MCAL	Minicalorimeter
PLD	Programmable Logical Device
PD	Photodiode
VHDL	Very High Speed Integrated Circuit (VHSIC) Hardware Description Language

1. BOARD ARCHITECTURE OVERVIEW

The main components of the HK/CONF board are the following:

- 1) an Altera PLD (IC4 component) that contains the all logic of the board;
- 2) a vme connector (J1:A, J1:B and J1:C components) for the communication and connection with VME bus;
- 3) a J3 output connector, with two components (IC17 and IC18) for the conversion from TTL to LVDS signals;
- 4) a J2 input connector, with two components (IC15 and IC16) for the conversion from TTL to LVDS signals;
- 5) a J4 test connector;
- 6) a board address selector (mainly, components IC6 and two switch for the board address selection, SW1 and SW2);
- 7) a buffer from VME bus and Altera PLD, compound by IC5 and IC7 components;
- 8) an internal clock generated from IC12 component.

The RS422/LVDS interface is provided. The data are physically transmitted and received on the serial bus by means of RS422 or LVDS drivers/receivers. Since they have the same pinout, by replacing the ICs one of the two standards can be chosen.



1.1 SYSTEM OVERVIEW

Below is shown an image of the HK/CONF board.

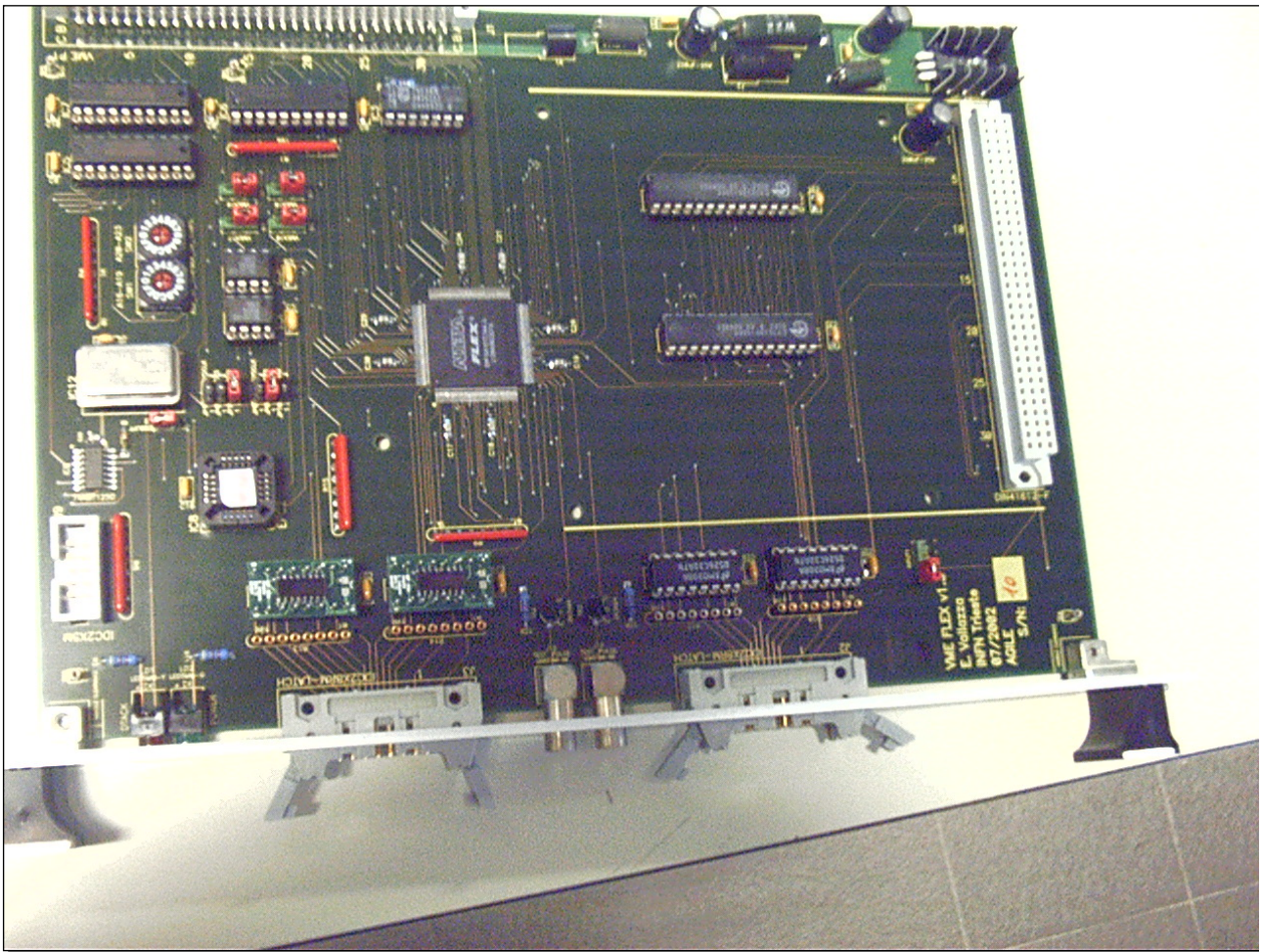


Figure 2: the board.

2. FUNCTIONALITY

The main functions of the board are:

- 1) sends command to MCAL DFE. In particular:
 - writes a DFE register. For do this the right operations are: write an address and write a value
 - reads a DFE register. For do this the right operations are: write an address and read a value
- 2) read the digital HKs from DFE. It is necessary to perform reading operations of DFE registers.

3. DATA FORMAT AND SIGNAL INPUT/OUTPUT

3.1 VME SIGNAL INTERFACE

The HK/CONF board has an I/O interface with the VME bus. The signals of this interface are the following:

- VME ADDR[1..15] = vmea[1..15] indicates the internal address of the board;
- VME ADDR[16..23] for the board selection over the VME bus;
- VME DATA = bufd[0..15]. These are the bidirectional buffered VME data signals.

The current address board is **0x88**.

3.2 BOARD INPUT SIGNAL

The board has the following input signal

- BSB_STB_DWN: Active low window that defines the bits read from MCAL.
- BSB_DATA_DWN: It's the data line for reading operations from MCAL DFE. 16 bits of data.
- BSB_CK_DWN: Clock coming from MCAL DFE.

See paragraph 3.4 for the reading operations. Signal timing diagram details are given in [2].

3.3 BOARD OUTPUT SIGNAL

The most important output signals are:

- BSB_STB_UP_N Active low window that defines the bits sent to MCAL DFE
- BSB_DATA_UP Data line
 - 12 address bits and 16 bit of data for writing operations
 - 12 address bits for reading
- TEST_CK_DWN
- HK_PULSE 1us every 16 sec. Used for digital and analog housekeepings.
- MCAL_RESET

Signal timing diagram details are given in [2].

3.4 READING OPERATIONS

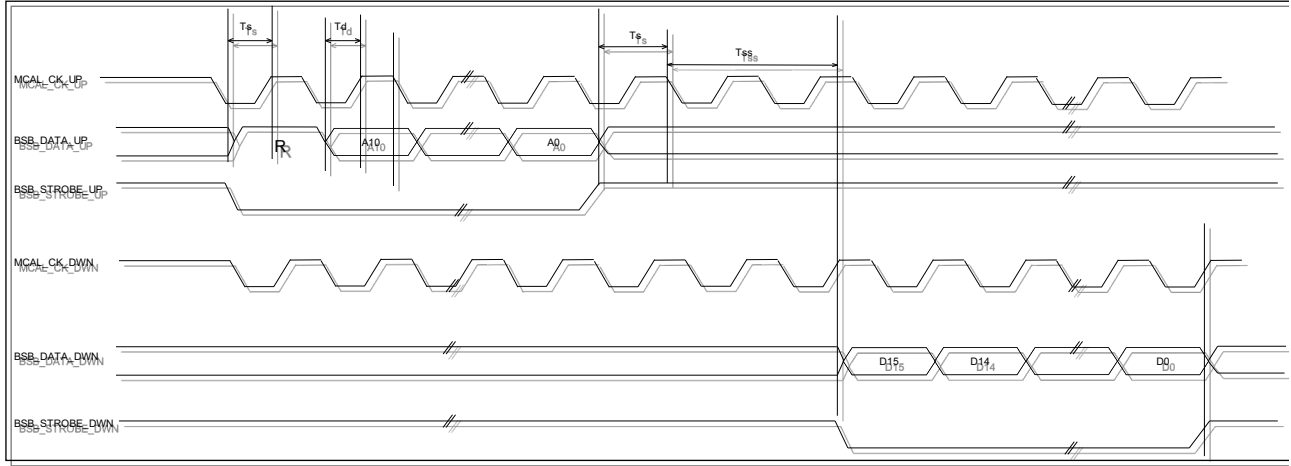


Figura 3: Reading operations

Name	Formula	Min (ns)	Max (ns)	Description
Td	[30,100]	30	100	Data setup time
Ts	[30,100]	30	100	Strobe setup time
Tclk	200	200	200	clock
Tss	[0,(16*Tclk)]	0	3200	Address to data

Table 1: timing of reading operation signals

3.5 WRITING OPERATIONS

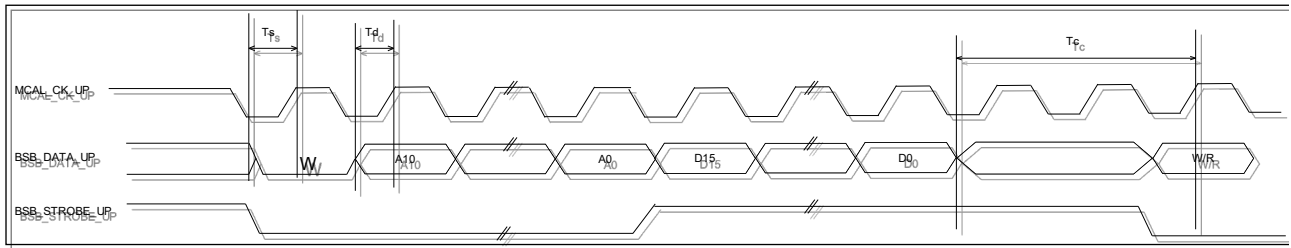


Figure 4: Writing operation

Name	Formula	Min (ns)	Max (ns)	Description
Td	[30,100]	30	100	Data setup time
Ts	[30,100]	30	100	Strobe setup time
Tclk	200	200	200	clock
Tc	[2*Tclk]	400		Cycle to cycle

Table 2: timing of writing operation signals

3.6 SERIAL BUS SIGNAL FORMAT

3.6.1 ADDRESS FORMAT FROM HC TO BOARD

For the communication between the HC and the board, 15 bits are used:

- The bits A11-A1 (11 bits) must contain the address of a register of the DFE;
- A12 bit is used for the configuration of the board. If it is high the board works in real mode (data sent and received to/from DFE); if it is low the board works in test mode (data routed internally);
- A15-A13 (3 bits) are used for selecting an operational mode. See paragraph 3.7 for a description.

MSB														LSB	
A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
ACT3	ACT2	ACT1	CONF	ADD11	ADD10	ADD9	ADD8	ADD7	ADD6	ADD5	ADD4	ADD3	ADD2	ADD1	0

3.6.2 ADDRESS FORMAT FROM BOARD TO DFE

This is the format of an address sent from board to DFE.

MSB										LSB	
A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0	
ADD11	ADD10	ADD9	ADD8	ADD7	ADD6	ADD5	ADD4	ADD3	ADD2	ADD1	

3.6.3 DATA FORMAT

This is the format of the data exchanged between HC, board and DFE.

MSB															
D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2		
DATA15	DATA14	DATA13	DATA12	DATA11	DATA10	DATA9	DATA8	DATA7	DATA6	DATA5	DATA4	DATA3	DATA2		

3.7 BASIC OPERATIONS

The basic operations are addressed with the A15-A13 bit pattern:

- 1) "010" reset mode
- 2) "001" MCAL_RESET signal
- 3) "011" write operation (address and data) and verify write operation
- 4) "101" HK_PULSE reading mode
- 5) "110" read operation: write address and verify write operation
- 6) "100" read operation: read the data

It is important to verify if the board is BUSY (see the next sections) before each read and write operation.

3.7.1 RESET MODE

If desired, the board can be reset by writing any data on a address with 0x2 on the three bits A15-A13. All the other address bits are ignored, but an even address must be used (otherwise a bus error occur). See 4.7.1 for a detailed description.

VME A15-A13	A11-A1	VME R/W action	Action	Possible address, value
0x02	Any	W	Reset	0x4000, any

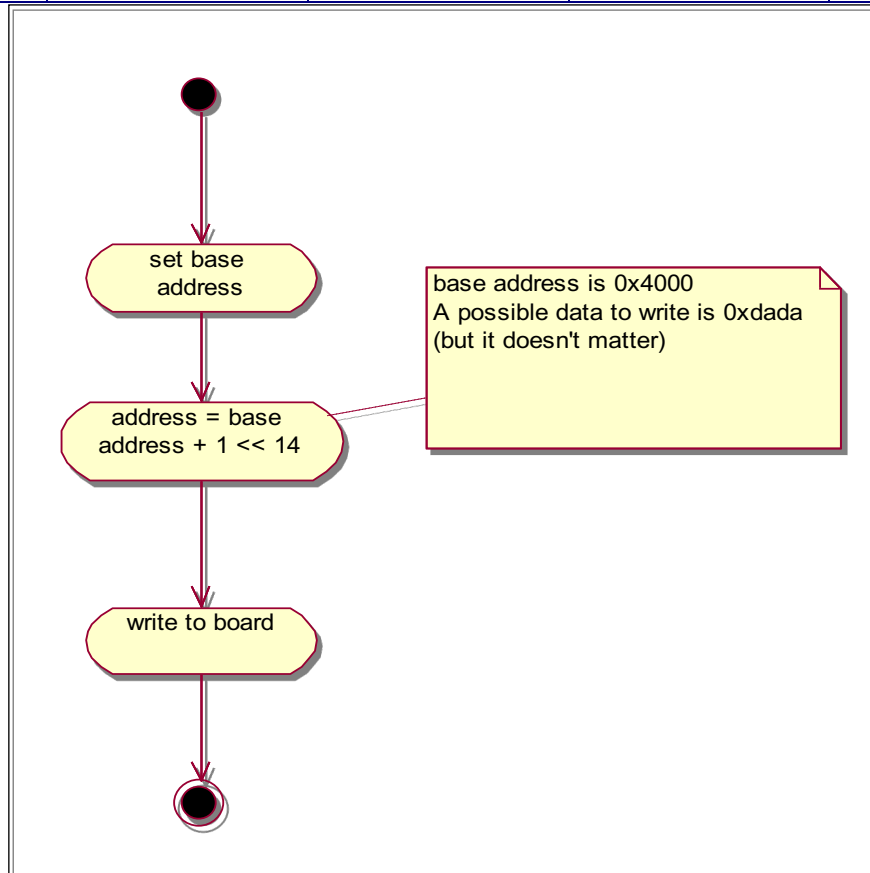


Figura 5: reset board operation

3.7.2 MCAL_RESET

VME A15-A13	A11-A1	VME R/W action	Action	Possible address, value
0x01	Any	W	Generate MCAL_RESET signal	0x2000, any

3.7.3 WRITE MODE

The 16 bits of data can be written to a given address (11 bits) with the following sequence:

1. send a reset to clear the state of the board;

2. write the 16 data bits to a VME address with 0x3 in the bits A15-A13 and the configuration address in the bits A11-A1. Set the bit A12 in the desired working mode;
3. check that all the data have been sent reading any address with 0x3 in the bits A15-A13 and checking the bit D15. A value of 1 means that the board is busy while a value of 0 means that the cycle is finished. The other 15 bits should be equal to 0x0055.

Step 1: write the address and data to the board:

VME A15-A13	A11-A1	VME R/W action	Action	Possible address, value
0x03	config address	W	Write	0x6000 + config address, data value

Step 2: verify that the writing operation is terminated correctly

VME A15-A13	A11-A1	VME R/W action	Action	Possible address, value
0x03	any	R	Verify write	0x6000 + any, 0x0055

In the Figure 6 is shown an UML activity diagram of the write operation.

3.7.4 HK_PULSE READING MODE

VME A15-A13	A11-A1	VME R/W action	Action	Possible address, value
0x05	Any	R	Read hk_bit and hk_ena	0x????, value

The hk_bit shows that the HK_PULSE signal is generated. This bit switch when the HK_PULSE is generated. An hk_pulse is generated every 16 sec. The hk_ena bit indicates that the generation of the HK_PULSE (SYNCH16) is enabled.

$Hk_bit = value \gg 15 \& 0x1$

$Hk_ena = value \gg 14 \& 0x1$

In every case the SYNC16 (HK_PULSE) is generated on the LEMO connector, even tough the $hk_ena = 0$.

To set the hk_ena it is necessary to perform the following operations

0 disable

1 enable

Write at the base address the value above.

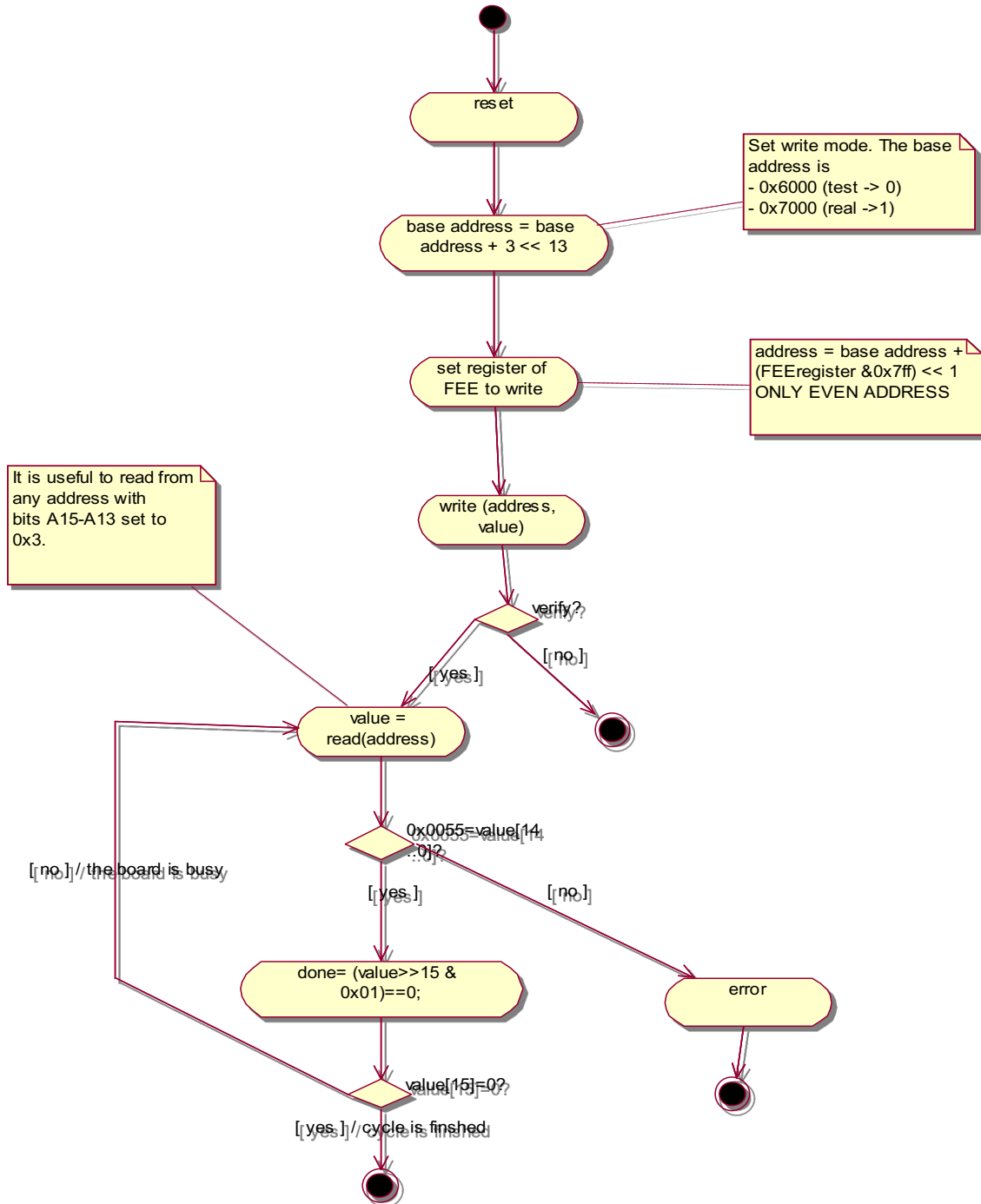


Figure 6: write operation

3.7.5 READ MODE

The 16 bits of data can be read from a given address (11 bits) with the following sequence:

1. send a reset to clear the state of the board
2. write to a VME address with 0x7 in the bits A15-A13 and the address in the bits A11-A1
3. check that all the data have been received reading any address with 0x7 in the bits A15-A13 and checking the bit D15. A value of 1 means that the board is busy while a value of 0 means that the cycle is finished. The other 15 bits should be equal to 0x5500;
4. read the 16 data bits from any address with 0x4 in the bits A15-A13.

Step 1: write the address for reading operation

VME A15-A13	A12	A11-A1	VME R/W action	Action	Possible address, value
0x07	0 (test)	config address	W	Write address	0xE000 + config address, any
0x07	1 (real)	config address	W	Write address	0xF000 + config address, any

Step 2: verify that the writing operation is terminated correctly

VME A15-A13	A12	A11-A1	VME R/W action	Action	Possible address, value
0x07	0 (test)	any	R	Verify write	0xE000 + any, 0x5500
0x07	1 (real)	any	R	Verify write	0xF000 + any, 0x5500

Step 3: read the value coming from DFE

VME A15-A13	A12	A11-A1	VME R/W action	Action	Possible address, value
0x04	0 (test)	any	R	Read value	0x8000 + any, DFE value
0x04	1 (real)	any	R	Read value	0x9000 + any, DFE value

In the Figure 7 is showed an UML activity diagram of the read operation.

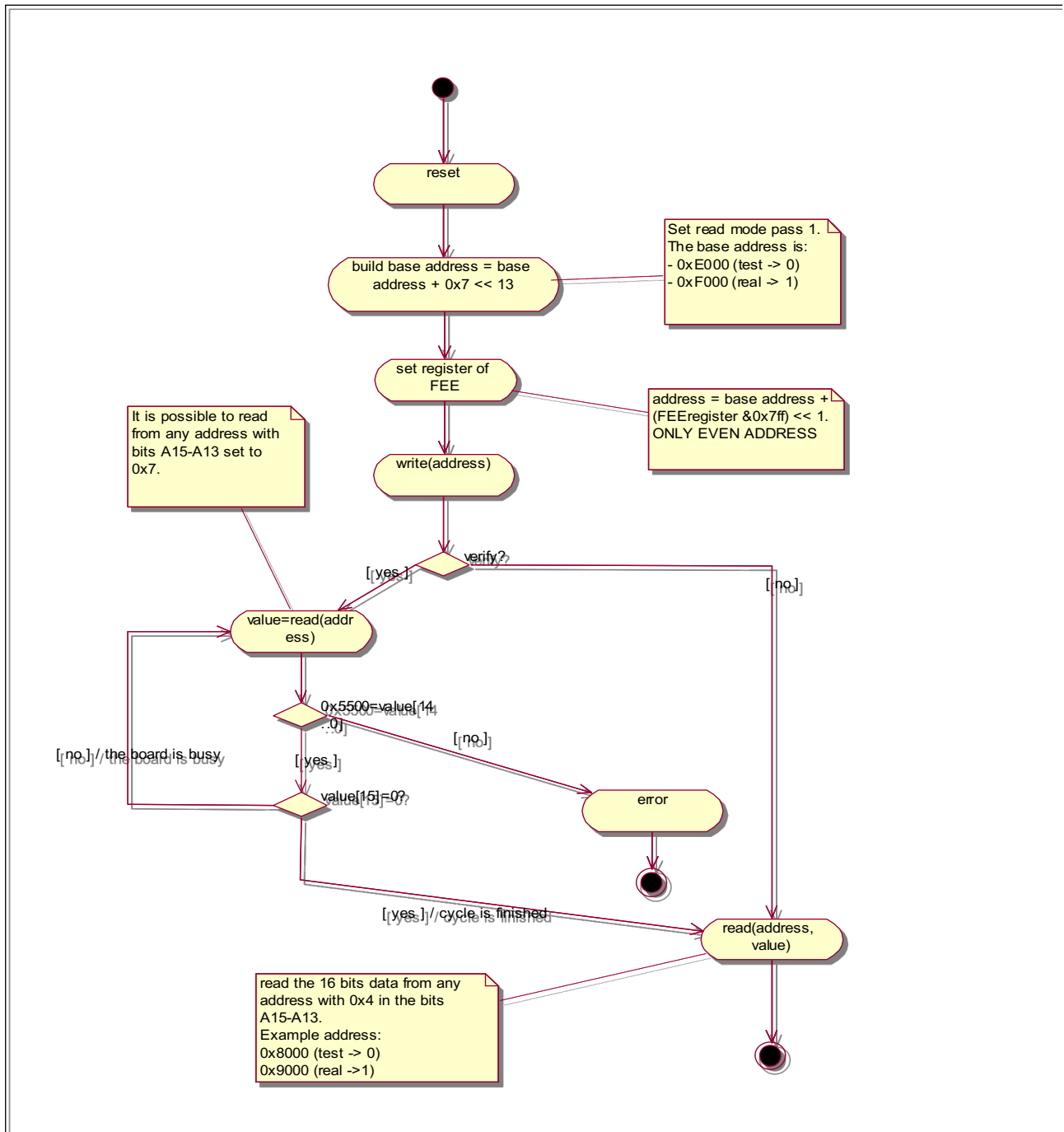


Figure 7: read operation

3.7.6 READ LAST VALUE MODE

It is possible to read the last value sent to data bus. For do this it is only necessary to read from an address with A15-A13 setted to '000'.

VME A15-	A11-A1	VME R/W	Action	Possible address, value
----------	--------	---------	--------	-------------------------

A13	action			
0x00	any	R	Read value	0x0000 + any, last value

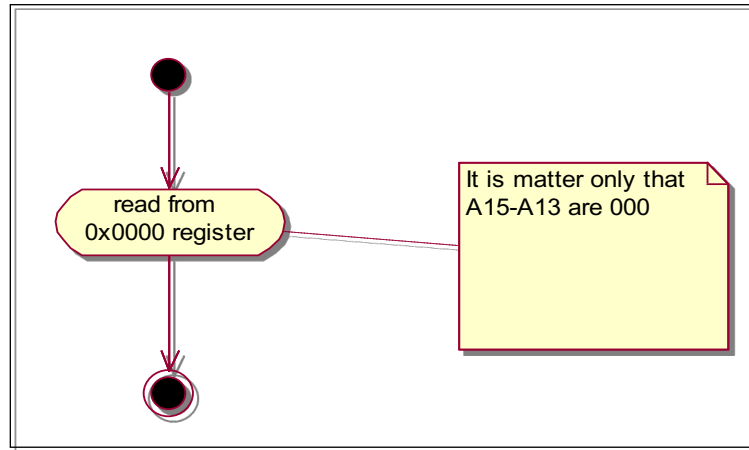


Figura 8: read last value

3.8 BOARD UTILIZATION

The board can be used for the following operations:

- 1) **configure DFE:** for this purpose the basic operations used are a write operation (that write data into a DFE address).
- 2) **digital HK acquisition:** it is necessary to read the value of the digital HK from predefined DFE address.

3.9 CONNECTORS PIN FUNCTION

MCAL HK/CONF Board exchanges signals with the MCAL-DFE through the 2 connectors listed below:

Connector J2 (16 PIN) IDC2X8 – INPUT (lower)	Pol.	PIN	DIR	Description
BSB_STB_DWN	+	1	IN	Active low window that defines the bits read from MCAL.
	-	2		
BSB_DATA_DWN	+	3	IN	It's the data line for reading operations from MCAL DFE. 16 bits of data.
	-	4		
BSB_CK_DWN	+	5	IN	Clock coming from MCAL DFE.
	-	6		
SPARE	+	7	IN	
	-	8		
SPARE	+	9	IN	
	-	10		
SPARE	+	11	IN	
	-	12		
SPARE	+	13	IN	
	-	14		
SPARE	+	15	IN	
	-	16		

Connector J3 (16 PIN) IDC2X8 – OUTPUT (higher)	Pol.	PIN	DIR	Description
BSB_STB_UP	+	1	OUT	Active low window that defines the bits sent to MCAL DFE
	-	2		
BSB_DATA_UP	+	3	OUT	Data line - 12 address bits and 16 bit of data for writing operations - 12 address bits for reading
	-	4		
TEST_CK_DWN	+	5	OUT	BSB_CK_UP
	-	6		
HK_PULSE	+	7	OUT	1us every 16 sec. Used for digital housekeepings.
	-	8		
CK_DWN1	+	9	OUT	Clock down for re-routing of the clock
	-	10		
CK_DWN2	+	11	OUT	Clock down for re-routing of the clock
	-	12		
CK_DWN3	+	13	OUT	Clock down for re-routing of the clock
	-	14		
MCAL_RESET_N	+	15	OUT	The MCAL_RESET signal

	-	16		
--	---	----	--	--

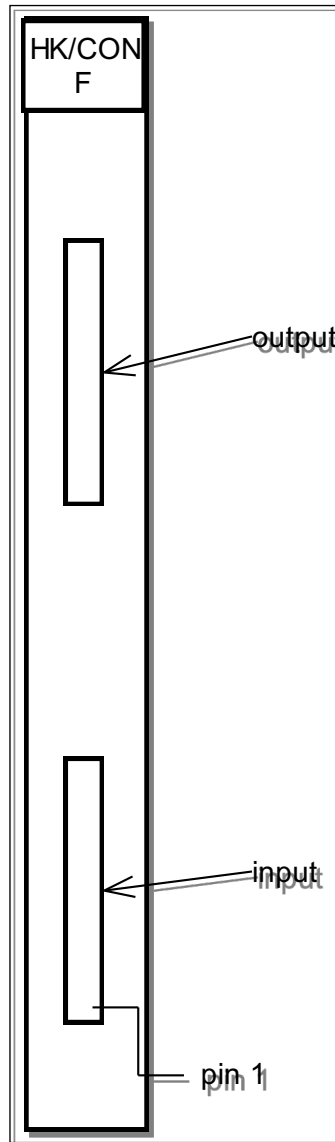


Figure 9: hardware layout of the board

3.10 ELECTRICAL SCHEMAS OF THE BOARD

See Annex.

4. BOARD DETAILED DESIGN

4.1 MAIN I/O SIGNAL OF THE IC4 ALTERA PLD

The address of the board is the following:

```
vmea : in std_logic_vector (15 downto 1);
```

Data in input are:

```
bsb_data_dwn : in      std_logic;  
bsb_stb_dwn  : in      std_logic;  
bsb_ck_dwn   : in      std_logic;
```

Data in output are:

```
-- Test clock is an output  
test_ck_dwn : buffer std_logic;  
-- PDHU data out  
bsb_data_up : buffer std_logic;  
-- PDHU strobe out  
bsb_stb_up_n : buffer std_u logic;  
-- Global reset  
gblres_ena_n : buffer std_logic;  
-- enaread_n: to enable readout  
enaread_n : buffer std_logic;  
-- hkpluse: 1us every 16 sec  
hk_pulse : buffer std_logic;  
-- enaread_n: to enable readout  
enaread_n : buffer std_logic;  
-- Reset clocks  
dres0_n : buffer std_logic;
```

There are bidirectional signals:

```
-- bufd: buffered VME data  
bufd : inout std_logic_vector (15 downto 0);
```

4.2 MAIN INTERNAL SIGNAL OF THE IC4 ALTERA PLD

```
-- temporary signals used to communicate with VME bus  
signal bitout : std_logic_vector(15 downto 0);  
signal bitin  : std_logic_vector(15 downto 0);
```

4.3 ADDRESS BOARD, ACTION AND MODE SELECTION

The command of the board is addressed within the value of the address sent to the board from HC. As specified in the paragraph 3.1, the address vmea[23..16] is used for the board selection and the vmea[15..1] is used for the command of the board. In particular:

- vmea[15..13] is used for the selection of the action. All the possible action are:
 - Reset of the DFE: vmea[15..13] = "010"
 - Write cycle: vmea[15..13] = "011"
 - Read cycle: compound of two pass:
 - Write address to read: vmea[15..13] = "111"
 - Read data: vmea[15..13] = "100"
- vmea[12]: choose between data and test
 - 1 --> real;
 - 0 --> test.

```
action_select          : buffer std_logic_vector (2 downto 0);

-- flip-flop to latch the address lines
-- only when the board is addressed
process (delvme_cycle, bdok_n)
begin
    if rising_edge(delvme_cycle) then
        if bdok_n = '0' then
            action_select(2 downto 0) <= vmea(15 downto 13);
            ext_int_n          <= vmea(12);
            lvme_add           <= vmea(11 downto 1);
        end if;
    end if;
end process;

-- Write cycle is enabled when A14-A13 is 3
with action_select(1 downto 0) select
    cycle_w_ena <= '1' when "11",
    '0'          when others;

-- Global reset is enabled when A14-A13 is 2
with action_select(1 downto 0) select
    gblres_ena_n <= '0' when "10",
    '1'          when others;

-- MCAL reset is enabled when A14-A13 is 1
with action_select(1 downto 0) select
    mcalres_ena_n <= '0' when "01",
    '1'          when others;

-- Mask bit write when A14-A13 is 0
with action_select(1 downto 0) select
    maskbit_ena_n <= '0' when "00",

-- generate bsb strobe
with action_select(2) select
    bsb_stb_up_n <= stb_w_n when '0',
    stb_r_n      when '1',
    '1'          when others;
```

```
-- Global reset is enabled when A15-A14 is 0
with action_select select
    bitout(15 downto 0) <= vme_data_out(15 downto 0) when "000",
    (busy_out_tmp(0 downto 0) & "1010101000000000") when "111",
    (busy_out_tmp(0 downto 0) & "000000001010101") when "011",
    (hk_bit & hk_ena & "00000010101010") when "101",
    tmpdat when others;
```

4.4 READ AND WRITE FROM VME TO BOARD

```
-- buffdir : input buffer direction (high = vme to board; low = board to vme)
buffdir <= write_conf;
-- write_conf/read_conf : vme set in write or in read
-- write_n: vme write signal
write_conf <= not write_n;
```

-- Enable the bidir bufd signals in read mode

bufd is the I/O signals that contains the data. The value of bufd is copied into the bitin register:

```
-- Enable the bidir bufd signals in write mode
with buffdir select
    bitin <= bufd when '1',
    "0000000000000000" when others;
```

After this, the bitin register is automatically passed to data_out. If buffdir is low, bitin is setted to 0 (no value read from vme because the direction is from board to vme).

data_out <= bitin;

After this, the value contained into data_out are copied into vme_data_out and shreg.

```
-- latch vme data
process (clkwrite0_n)
begin
    if rising_edge(clkwrite0_n) then
        cycle_w <= cycle_w_ena;
        vme_data_out <= data_out;
    end if;
end process;

-- shift register out
process (busy, cksyn1_n)
begin
    if busy = '0' then
        shreg(27) <= action_select(2);
        shreg(26 downto 16) <= lvme_add(11 downto 1);
        shreg(15 downto 0) <= data_out(15 downto 0);
    elsif rising_edge(cksyn1_n) then
        shreg(27 downto 1) <= shreg(26 downto 0);
        shreg(0) <= '0';
    end if;
end process;
```

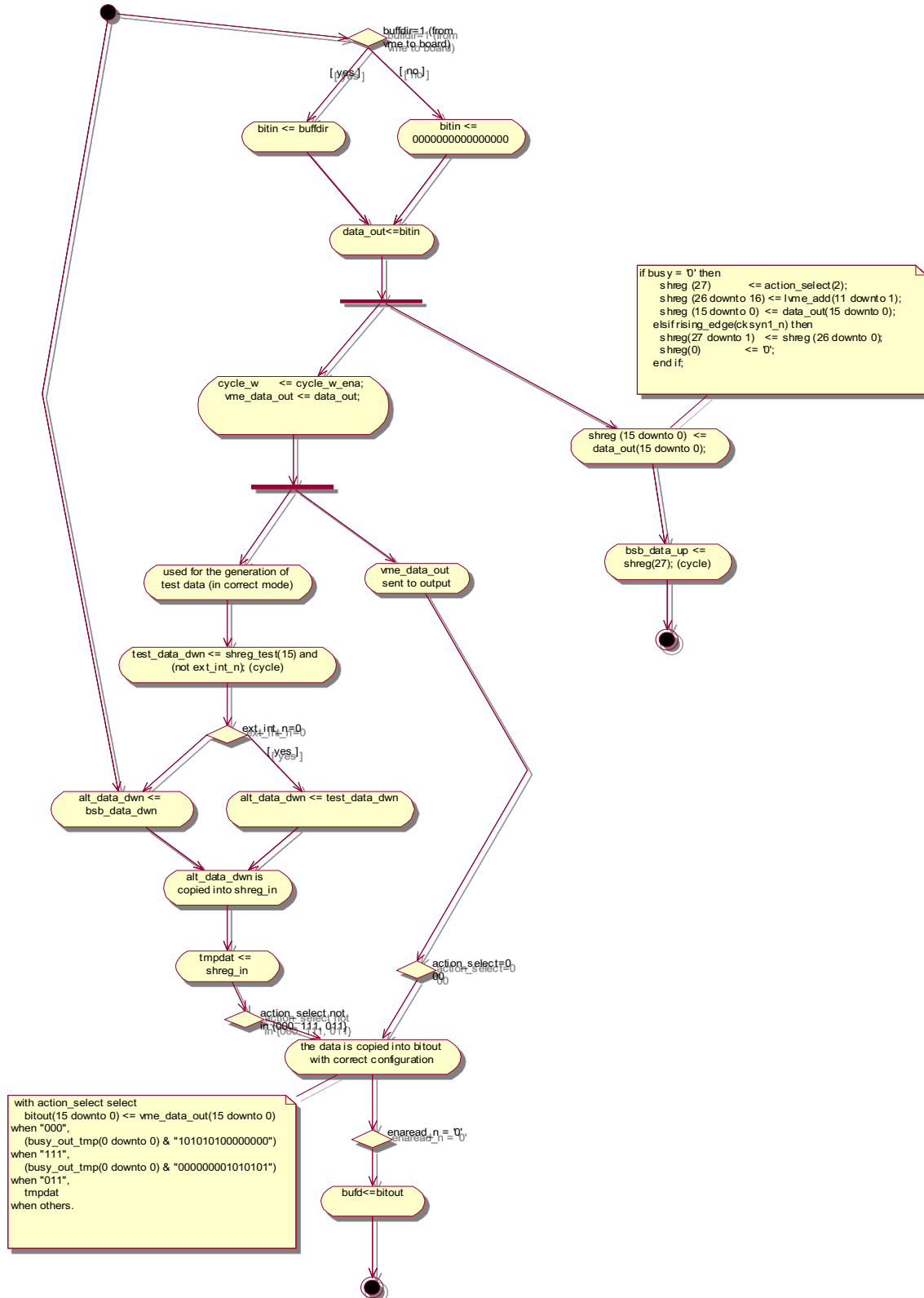
The vme_data_out register is copied into shreg_test (useful in test mode):

```
process (test_ck_dwn, test_stb_dwn)
begin
  if rising_edge(test_ck_dwn) then
    if test_stb_dwn = '1' then
      shreg_test <= vme_data_out;
    else
      shreg_test(15 downto 1) <= shreg_test (14 downto 0);
      shreg_test(0) <= '0';
    end if;
  end if;
end process;
```

The vme_data_out is even copied into bitout register

```
-- Global reset is enabled when A15-A14 is 0
with action_select select
  bitout(15 downto 0) <= vme_data_out(15 downto 0) when "000",
  (busy_out_tmp(0 downto 0) & "1010101000000000") when "111",
  (busy_out_tmp(0 downto 0) & "000000001010101") when "011",
  tmpdat
```

In the following picture is showed the data flow into board.



4.5 REAL AND TEST DATA

The main signal are ext_int_n

```
-- ext_int_n : choose between data and test
ext_int_n      : buffer std_logic;
```

As shown in the above paragraph, for the selection of the mode it is necessary to set the vmea[12] bit with the correct value.

```
-- select clock between the test and external clock
-- according to the ext_int_n bit (1=ext, 0=int)
```

```
with ext_int_n select
    alt_clk_dwn <= test_ck_dwn when '0',
    bsb_ck_dwn  when others;
```

```
with ext_int_n select
    alt_stb_dwn <= test_stb_dwn when '0',
    bsb_stb_dwn  when others;
```

```
with ext_int_n select
    alt_data_dwn <= test_data_dwn when '0',
    bsb_data_dwn  when others;
```

The bsb_ck_dwn, bsb_stb_dwn and bsb_data_dwn are coming from DFE.

4.5.1 THE TEST DATA GENERATION

```
test_data_dwn <= shreg_test(15) and (not ext_int_n);
```

```
process (test_ck_dwn, test_stb_dwn)
begin
    if rising_edge(test_ck_dwn) then
        if test_stb_dwn = '1' then
            shreg_test <= vme_data_out;
        else
            shreg_test(15 downto 1) <= shreg_test (14 downto 0);
            shreg_test(0) <= '0';
        end if;
    end if;
end process;
```

It is important to understand the generation of the vme_data_out signal:

```
signal data_out      : std_logic_vector(15 downto 0);
signal vme_data_out  : std_logic_vector(15 downto 0);
```

```
-- latch vme data
process (clkwrite0_n)
begin
    if rising_edge(clkwrite0_n) then
```

```
        cycle_w      <= cycle_w_ena;
        vme_data_out <= data_out;
    end if;
end process;

data_out <= bitin;

-- Enable the bidir bufd signals in write mode
with buffdir select
    bitin <= bufd      when '1',
    "0000000000000000" when others;
```

The data test are read from vme bus.

4.6 WRITING DATA INTO MEMORY

When the data is written into alt_data_dwn, the following step is

```
- shift register in
process (alt_clk_dwn, alt_stb_dwn, gblres_n)
begin
    if gblres_n = '0' then
        shreg_in (15 downto 0) <= "0000000000000000";
    elsif rising_edge(alt_clk_dwn) then
        if alt_stb_dwn = '0' then
            shreg_in(15 downto 1) <= shreg_in (14 downto 0);
            shreg_in(0)          <= alt_data_dwn;
        end if;
    end if;
end if;
end process;

signal shreg_in          : std_logic_vector (15 downto 0);
debug_bit <= shreg_in(0);

-- on the last shift pulse, data are latched (tmpdat)
process (clk_shin, gblres_n)
begin
    if gblres_n = '0' then
        tmpdat <= "1100101011001010";
    elsif falling_edge(clk_shin) then
        tmpdat <= shreg_in;
    end if;
end process;

-- Global reset is enabled when A15-A14 is 0
with action_select select
    bitout(15 downto 0) <= vme_data_out(15 downto 0) when "000",
    (busy_out_tmp(0 downto 0) & "1010101000000000") when "111",
    (busy_out_tmp(0 downto 0) & "0000000010101010") when "011",
    (hk_bit & hk_ena & "0000001010101010")         when "101",
    tmpdat                                              when others;
```

4.7 ACTIONS

4.7.1 RESET OF THE DFE

```
-- Global reset is enabled when A14-A13 is 2
with action_select(1 downto 0) select
    gblres_ena_n <= '0' when "10",
    '1'           when others;
```

4.7.2 ENABLE WRITE CYCLE

The correct configuration of the address sent to board enable the write cycle to DFE:

```
-- Write cycle is enabled when A14-A13 is 3
with action_select(1 downto 0) select
    cycle_w_ena <= '1' when "11",
    '0'         when others;
```

```
-- latch vme data
process (clkwrite0_n)
begin
    if rising_edge(clkwrite0_n) then
        cycle_w      <= cycle_w_ena;
        vme_data_out <= data_out;
    end if;
end process;
```

It is important to generate a strobe that last 28 clock cycles (28 coming from 1 bit for operation type, 11 bits of address and 16 bits of data, see 3.5).

```
-- counter to generate the stop clock signal after 28 clocks
process (cksyn1, cycle_w)
    variable cnt : integer range 0 to 63;
begin
    if cycle_w = '0' then
        cnt := 0;
    elsif rising_edge(cksyn1) then
        cnt := cnt+1;
    end if;
    qtmp <= cnt;
end process;
```

```
stopclk_w_n <= '0' when qtmp = 48 else '1';
```

4.7.2.1 BSB_STB_UP

```
-- generate bsb strobe
with action_select(2) select
    bsb_stb_up_n <= stb_w_n when '0',
    stb_r_n      when '1',
    '1'          when others;
```

```
process (busy, stop_w_n)
begin
```

```
if stop_w_n = '0' then
    stb_w_n <= '1';
elsif busy'event and busy = '1' then
    stb_w_n <= '0';
end if;
end process;
```

4.7.2.2 BSB_DATA_UP

The 28 bits data sent to DFE are generated in the following mode:

```
-- shift register out
process (busy, cksyn1_n)
begin
    if busy = '0' then
        shreg (27)          <= action_select(2);
        shreg (26 downto 16) <= lvme_add(11 downto 1);
        shreg (15 downto 0)  <= data_out(15 downto 0);
    elsif rising_edge(cksyn1_n) then
        shreg(27 downto 1)  <= shreg (26 downto 0);
        shreg(0)            <= '0';
    end if;
end process;

bsb_data_up <= shreg(27);
clk_shin    <= alt_clk_dwn and (not alt_stb_dwn);
```

4.8 HOUSEKEEPING MANAGEMENT

```
-- generate 1MHz clock
process (flex_ck2)
    variable cnt_hk : integer range 0 to 7;
begin
    if rising_edge(flex_ck2) then
        if cnt_hk = 4 then
            cnt_hk := 0;
        else
            cnt_hk := cnt_hk+1;
        end if;
    end if;
    qtmp_hk <= cnt_hk;
end process;

xhk_n      <= '0' when qtmp_hk = 0 else '1';

process (flex_ck2)
begin
    if falling_edge(flex_ck2) then
        ck1m <= xhk_n;
    end if;
end process;

-- generate the pulse clock (1us (here is 15us))
process (ck1m)
    variable cnt_ck1m : integer range 0 to 16700000;
begin
```

```
if rising_edge(ck1m) then
  if cnt_ck1m = 16000000 then
    cnt_ck1m := 0;
  else
    cnt_ck1m := cnt_ck1m+1;
  end if;
end if;
qtmp_ck1m <= cnt_ck1m;
end process;

xck1m      <= '1' when qtmp_ck1m = 0 else '0';

process (ck1m)
begin
  if falling_edge(ck1m) then
    hk_pulse <= xck1m;
  end if;
end process;
```

5. SOFTWARE

5.1 OPERATIONS

5.1.1 READING HOUSEKEEPING

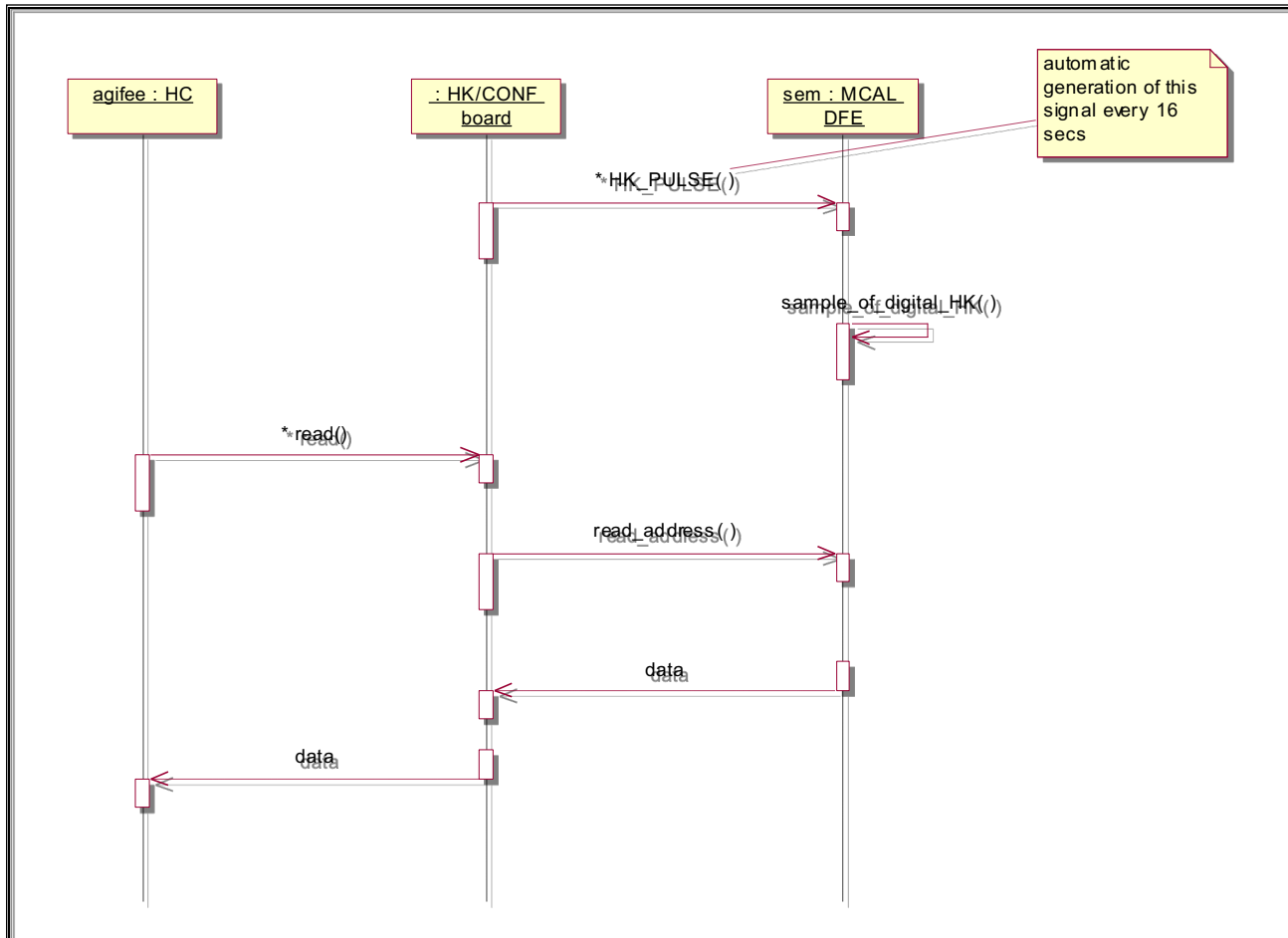


Figura 11: housekeeping reading operation

5.2 INFN SOFTWARE

5.2.1 RESET BOARD

5.3 CNR/IASF SECTION OF BOLOGNA SOFTWARE

APPENDIX A

5.4 IC4

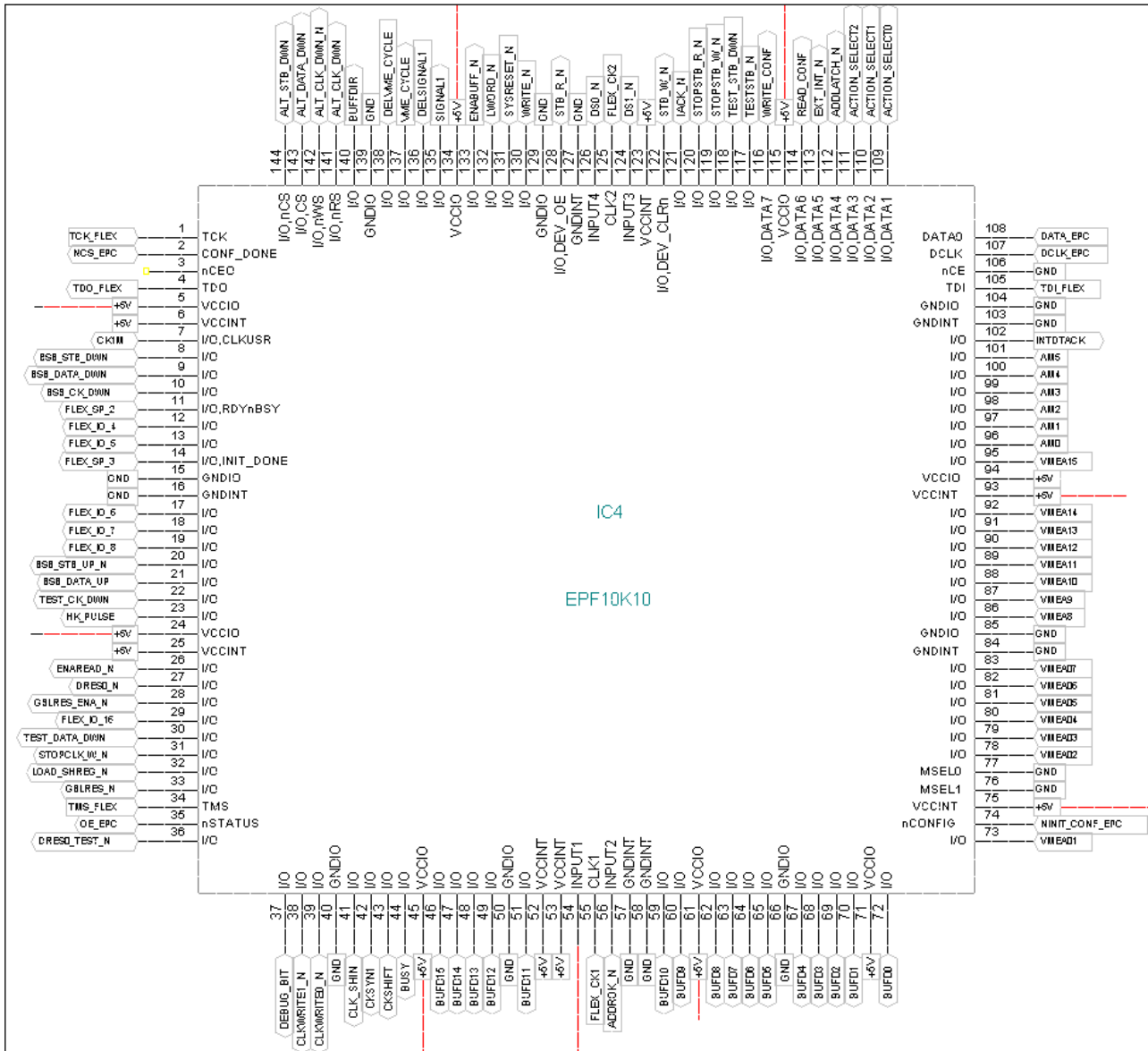


Figura 12: IC4