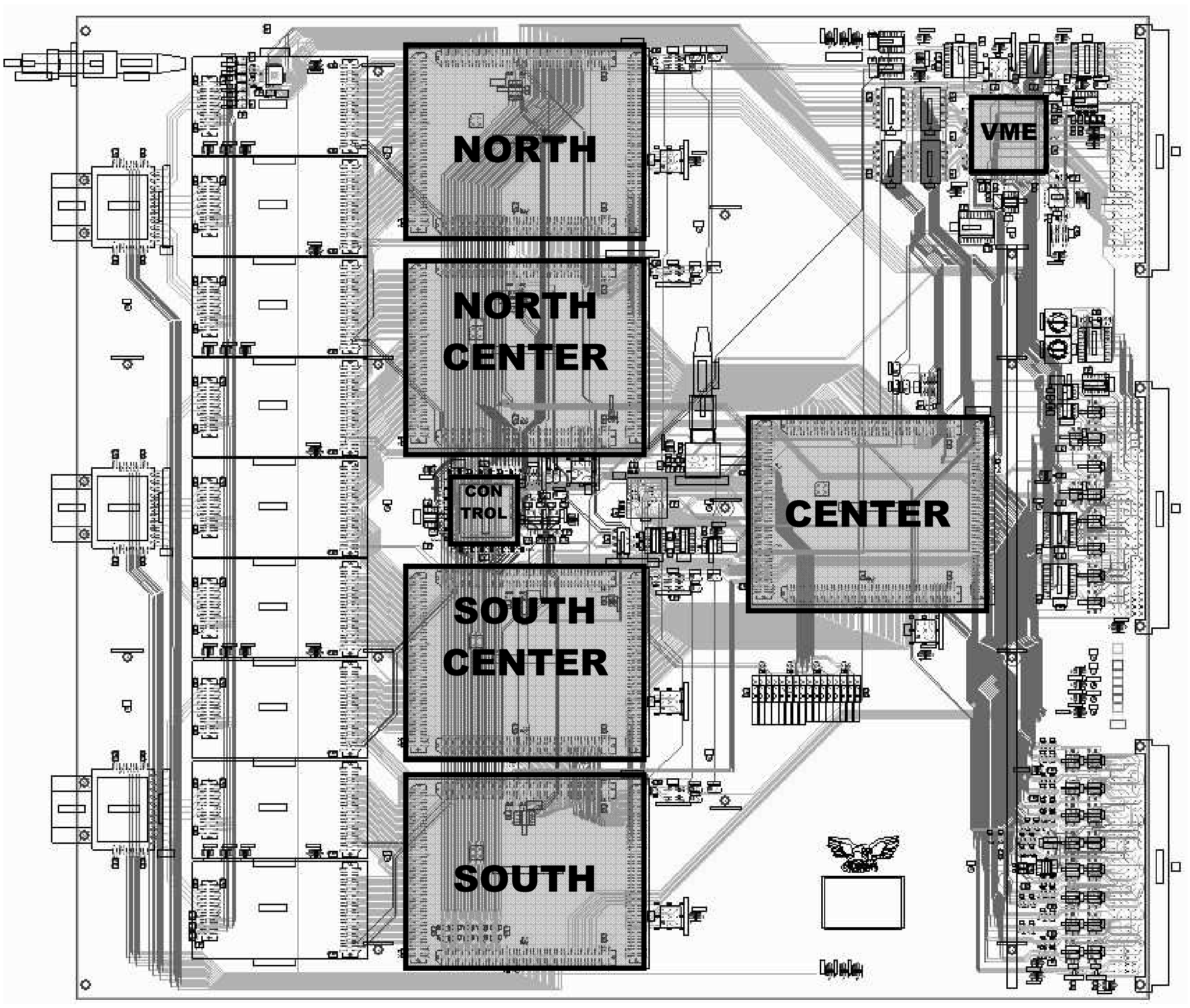
General Description:



The main parts of the PixelFed are the front FPGAs called **“ NORTH “, “ NORTH-CENTER ”, “ SOUTH-CENTER ”, “ SOUTH ”** housing the logic functionality for the decoding state machines, with FIFO I plus FIFO II buffers and the **“ CENTER “** FPGA containing the final FIFO III and the S-Link connection.

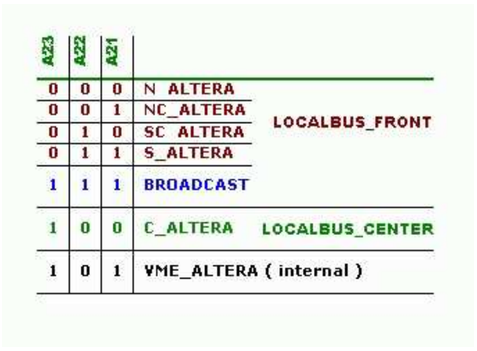
An additional **“ CONTROL “** CPLD for signal distribution and a **“ VME “** FPGA for system connection complete the module.

PREFACE
We tried to design as simple and compact as possible to increase system reliability and to facilitate maintenance work.

“ VME “ FPGA LocalBus Connection:

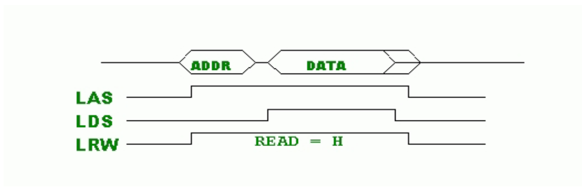
Two interfaces for supporting the front FPGAs and the center FPGA are foreseen. Each with 32 Address/Data lines and four Control lines, all Clock synchronized.

Address spaces:



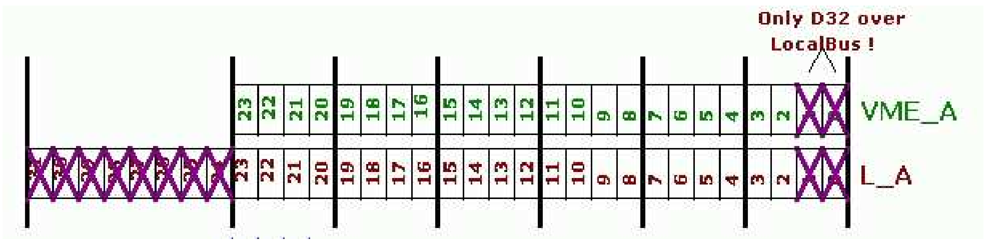
A23	A22	A21		
0	0	0	N_ALTERA	LOCALBUS_FRONT
0	0	1	NC_ALTERA	
0	1	0	SC_ALTERA	
0	1	1	S_ALTERA	
1	1	1	BROADCAST	
1	0	0	C_ALTERA	LOCALBUS_CENTER
1	0	1	VME_ALTERA (internal)	

LocalBusTiming:



The fourth Control line **LRES:** 100ns clock synchronized pulse

VME – LocalBus address matching:



Only A32 / D32 addressing is supported in either Single- or Block- Transfer mode !

“ VME “ FPGA TTCrx Reset:

```
RES_TTCrx        (FEDBASE+0xa00038)
```

A write cycle to this address triggers an external MonoFlop which generates a 50us reset pulse for the TTCrx chip. Since the VME Altera needs the TTCrx Clock it is necessary that the next VME access has to have a minimum delay of 100us.
In practice, the TTCrx clock may take longer to recover. A 1ms minimum wait is recommended after a TTCrx reset.

“ VME “ FPGA JTAG Busses:

It is possible to program all ALTERA EEPROMs (of course except the one of the VME ALTERA) on the board over the VME Interface.

This is done with two independent JTAG Buses.

JTAG chain 1 :	**“ NORTH “**
	“ NORTH-CENTER “
	“ SOUTH-CENTER “
	“ SOUTH “
JTAG chain 2 :	**“ CENTER “**
	“ CONTROL “

The VME Addresses are as follows:

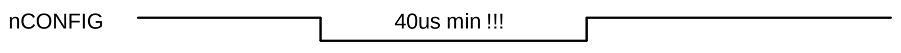
```
JTAG1_WRITE    (FEDBASE+0xa00018)            JTAG2_WRITE    (FEDBASE+0xa0001c)
               D[1]    TMS                                  D[1]    TMS
               D[2]    TDI                                  D[2]    TDI

JTAG_1or2_READ (FEDBASE+0xa00020) read with TDI=0 and TMS=0
               (FEDBASE+0xa00024) read with TDI=0 and TMS=1
               (FEDBASE+0xa00028) read with TDI=1 and TMS=0
               (FEDBASE+0xa0002c) read with TDI=1 and TMS=1

               Data read:        D[7]    TDO
```

Jam Byte Code files are used with an adapted “JAM BYTE CODE PLAYER” from ALTERA.

Attention: The original ALTERA Software Version 2.1 doesn’t support EPC8 EEPROMs so please use Version 2.2

Reprogramming an ALTERA EEPROM doesn’t immediately change the FPGAs behaviour. You have to invoke a configuration cycle, this is done either by POWER off/on or with the VME registered command:

```
nCONFIG        (FEDBASE+0xa00018)
                D[0] = 1   nCONFIG = Low ( due to external inverter )
               D[0] = 0   nCONFIG = High
```



On V5 modules we have two separate configuration lines, one for the **front** chips (served by ***D[0]***) and one for the **center** chips (served by ***D[1]***).

The reason for this modification was to ensure that varying configuration times have no influence on the module startup behavior.

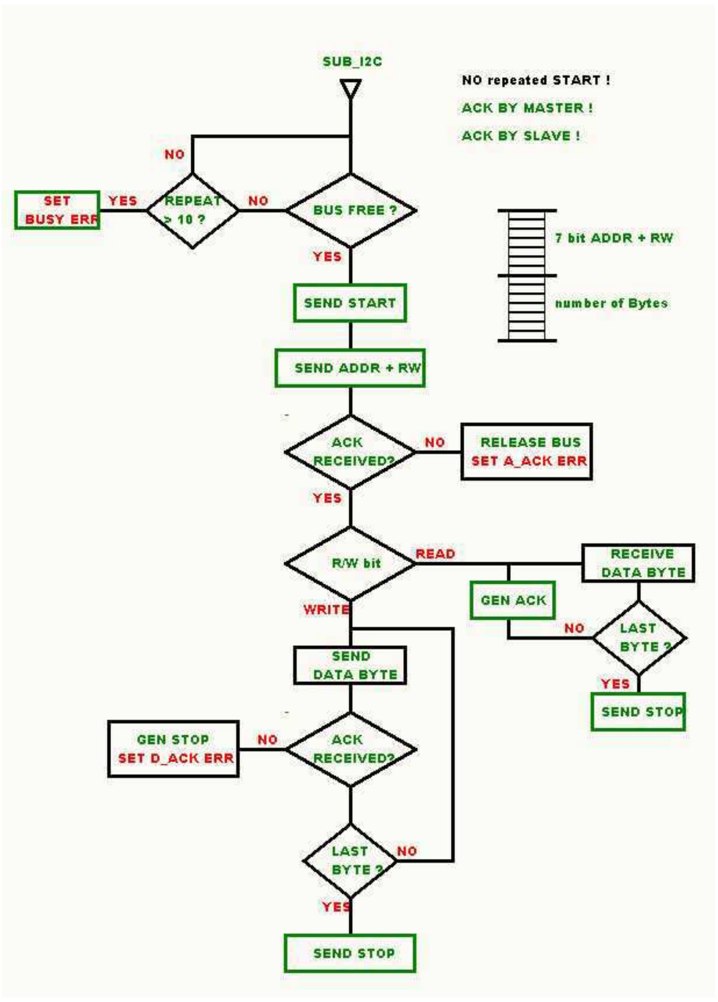
“ VME “ FPGA I2C MASTER :

The simple I2C Master implemented as a VHDL State machine is capable of writing and reading up to four Bytes. “Bus Busy” and “I2C Acknowledge” Errors are reported.
Repeated Start is not supported !

In the PXLFED it provides an I2C interface to the TTCrx Chip.

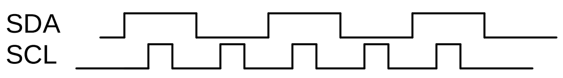
VME Addresses:

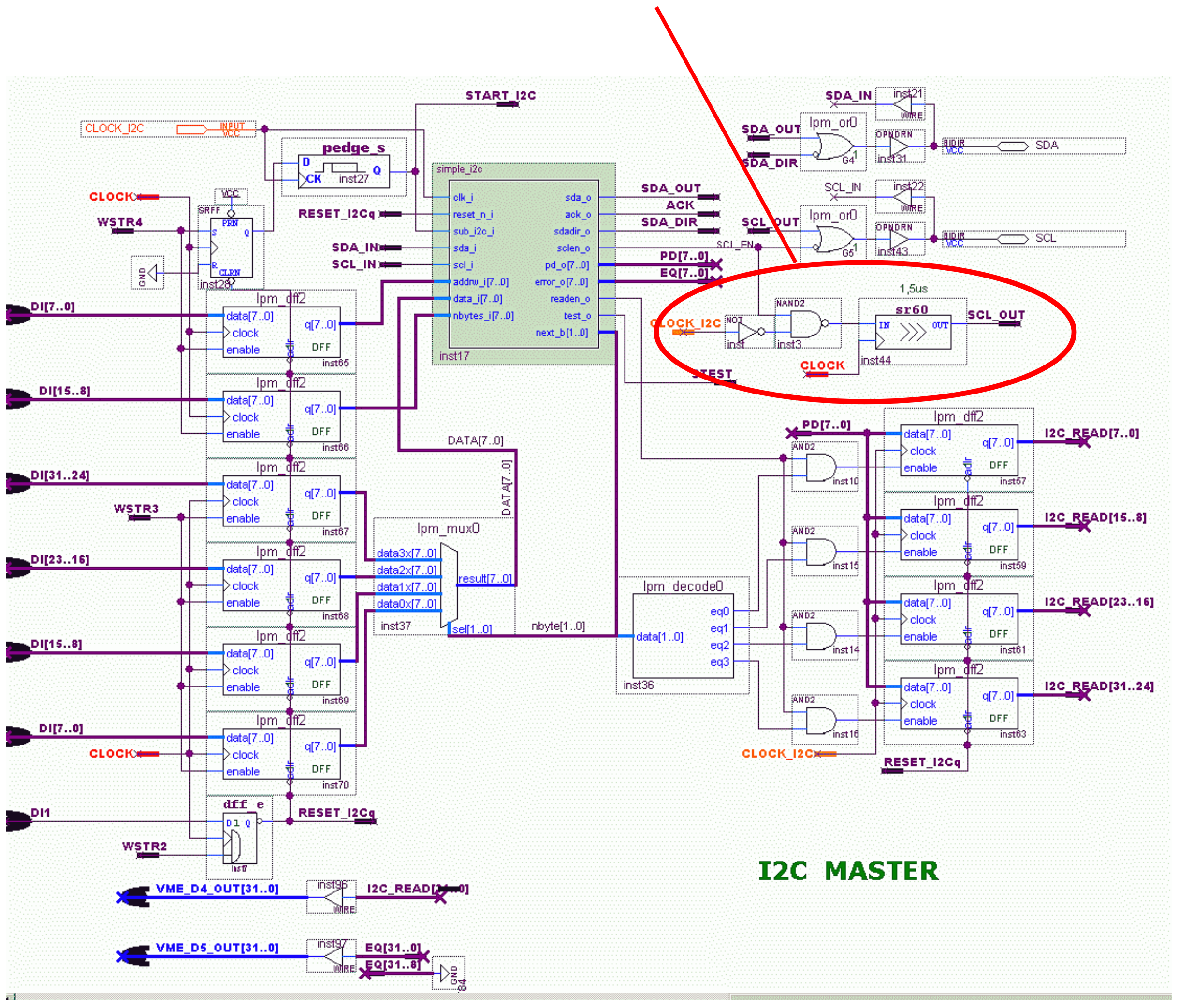
I2C_RES	(FEDBASE+0xa00008)
Reset for the I2C State Machine	
I2C_LOAD	(FEDBASE+0xa0000c)
Payload :	***D[7..0] Byte 1***
	D[15..8] Byte2
	D[23..16] Byte3
	D[31..24] Byte4
I2C_ADDR_RW	(FEDBASE+0xa00010)
Address :	***D[0] 1 = R 0 = W***
	D[7..1] I2C Addr.
	D[9..8] Number of Bytes
I2C_RD_DATA	(FEDBASE+0xa00010)
Read :	***D[7..0] Byte 1***
	D[15..8] Byte2
	D[23..16] Byte3
	D[31..24] Byte4
I2C_RD_STAT	(FEDBASE+0xa00014)
Error Register:	***Bit 0 = 1 Bus Busy Error***
	Bit 1 = 1 AddrAck Error
	Bit 2 = 1 WByteAck Error
	Bit 3 = 1 LastByteAck Error



Wait cycles have to be included before Status Register can be read because of the slow I2C Clock.
(I2C SCL is adjusted to 40Mhz / 256 = 156.25kHz)

Schematic Details:



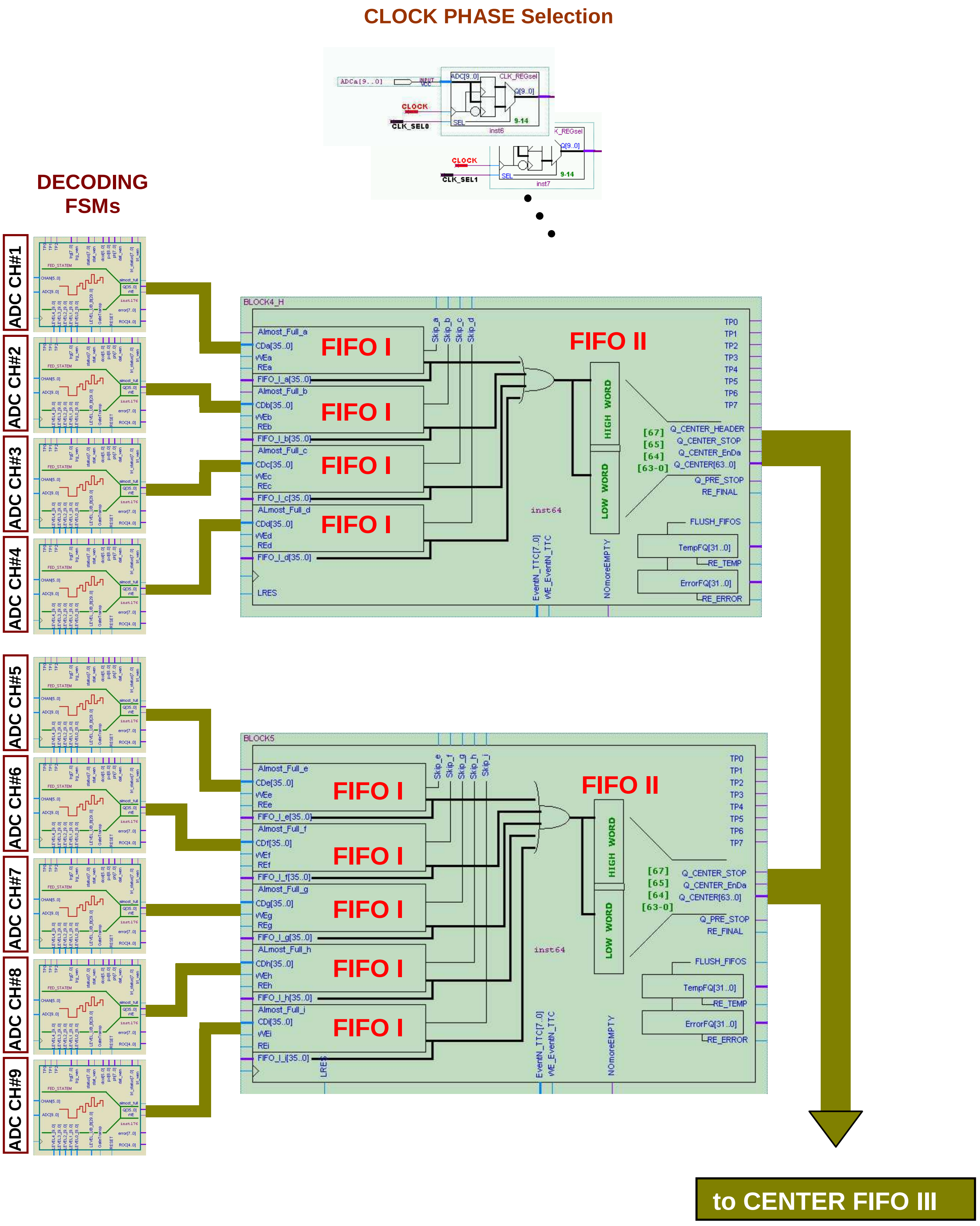


STATE MACHINE DETAIL:

```
p_get_SDA: process (clk_i, reset_n_i)
begin

if (reset_n_i = '0') then  s_sda_response <= '1';
elsif clk_i'EVENT AND clk_i = '0' then
  if ( s_sdadir= '0' AND sda_i = '0') then s_sda_response <= '0';
  else s_sda_response <= '1';
  end if;
end if;
ack_o <= s_sda_response;

end process p_get_SDA;
```

SDA is monitored on negative edge to have time enough for acknowledge decision.

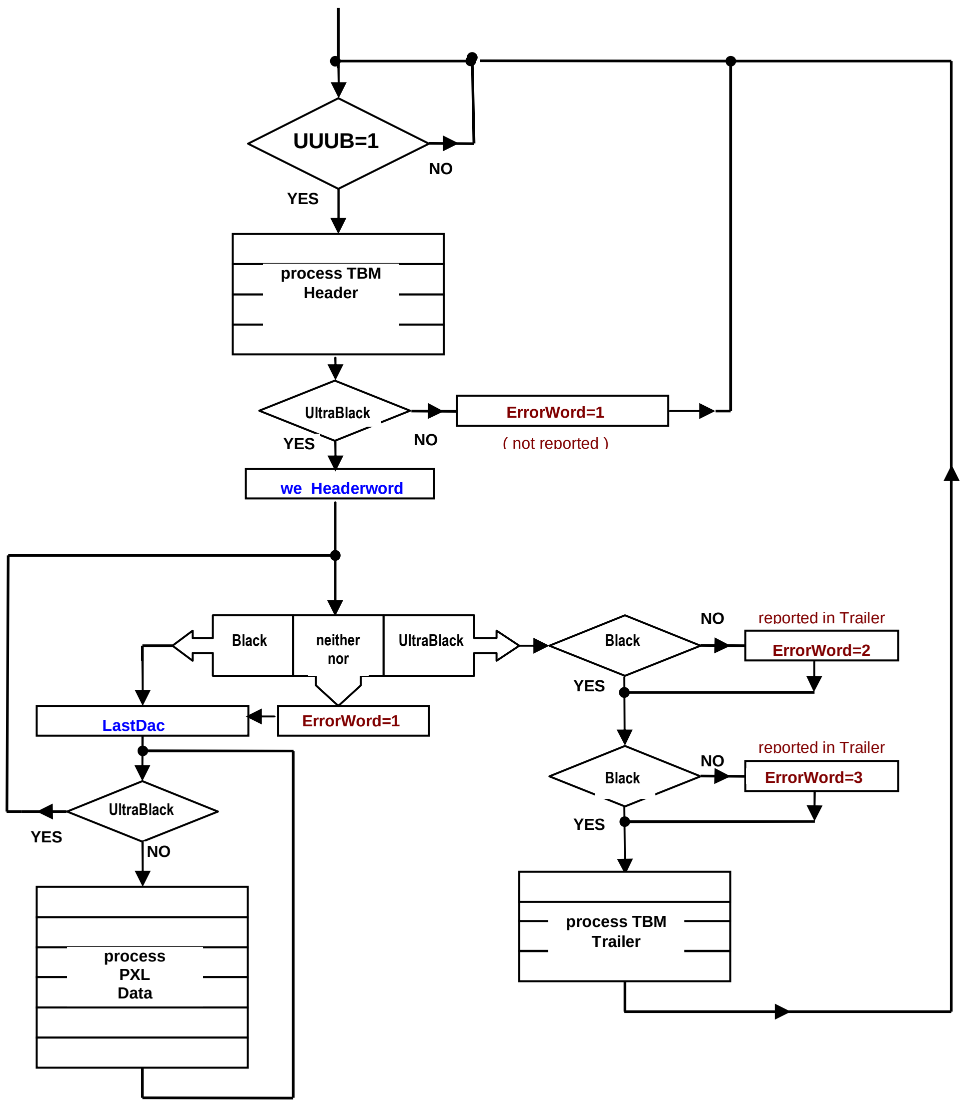
“ FRONT “ FPGAs Data Path Block Diagram :



“ FRONT “ FPGAs Decoding FSM :

The encoding of the ADC Data is done by a State Machine written in VHDL. Necessary multipliers are implemented as ALTERA library blocks to ensure proper synthesize results.

STATE DIAGRAM:



Decoding FSM output data format “normal mode”:

```
BitPos: 35 34 33 32|31 30 29 28 27 26|25 24 23 22 21|20 19 18 17 16 15 14 13 12 11 10 09 08|07 06 05 04 03 02 01 00|

Header:  1  0  0  0| < -CHANNEL #-- >| 1  1  1  1  1| 0---------------------------------- 0|< --TBM HEADER TRG#-- >|
            0x8                         0x1f (31)
         HeaderID                    Header Marker
```

```
BitPos: 35 34 33 32|31 30 29 28 27 26|25 24 23 22 21|20 19 18 17 16 15 14 13 12 11 10 09 08|07 06 05 04 03 02 01 00|

LastDac: 1  1  0  0| < -CHANNEL #-- >| < --ROC #-- >| 0---------------------------------- 0|< ----- LAST DAC ---- >|
            0xc
         LastDacID
```

```
BitPos: 35 34 33 32|31 30 29 28 27 26|25 24 23 22 21|20 19 18 17 16|15 14 13 12 11 10 09 08|07 06 05 04 03 02 01 00|

Data:    0  0  0  1| < -CHANNEL #-- >| < --ROC #-- >| < --DCOL -- >|< ------- Pxl ------- >|< ----PulseHeight---- >|
            0x1
          DataID
```

```
BitPos: 35 34 33 32|31 30 29 28 27 26|25 24 23 22 21|20 19 18 17 16 15 14 13 12 11 10 09 08|07 06 05 04 03 02 01 00|

Trailer: 0  1  0  0| < -CHANNEL #-- >| 1  1  1  1  0| 0 --------------------- 0 e2 e1 e0  0|< --TBM Trailerword-- >|
            0x4                         0x1c (30)                            e2        => invalid number of ROCs
         TrailerID                   Trailer Marker                             e1 e0  => FSM ErrorByte
```

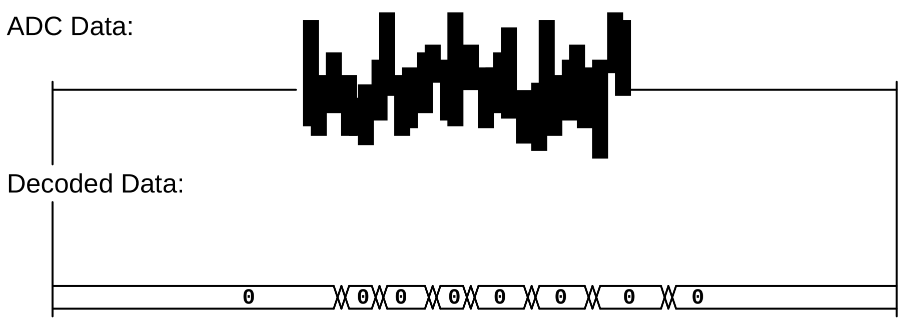
An Error-Trailer is generated if there are more than 192 Pixels / Channel :

```
BitPos: 35 34 33 32|31 30 29 28 27 26|25 24 23 22 21|20 19 18 17 16 15 14 13 12 11 10 09 08|07 06 05 04 03 02 01 00|

ErrTrl:  0  1  0  0| < -CHANNEL #-- >| 1  1  1  1  0| 0 --------------------- 0 e2 e1 e0  1| 0 ------------------ 0|
            0x4                         0x1e (30)
         TrailerID                   Trailer Marker
```

When the Decoding Machine detects an “Almost Full” from FIFO I then only “HEADER” , “LASTDAC” and “TRAILER” words are produced.

Decoding FSM output data format "transparent mode":

In "transparent mode" a block of 512 ADC Data , starting with a trigger Signal from the Center chip, is stored in FIFO I



When Pixel decoding is finished, the result is stored in Bits 21..0 , otherwise these Bits are 0!

```
BitPos: 35 34 33 32 |31 30 29 28 27 26 25 24 23 22|21| 20 19 18 17 16| 15 14 13 12 11 10 09 08 |07 06 05 04 03 02 01 00|

Transp:  0  0  0  1 | < -----ADC[9..0]---------- >|ROC#| < --DCOL -- >| < ------- Pxl ------- > |< ----PulseHeight---- >|
                0x1                                 LSB
              DataID
```

The 513th Data in transparent mode is marked with 0xff in Bits 7..0 !

RESET possibilities:

Each Local Bus has its own Reset line called LRES and CLRES.

LRES (FEDBASE+0xa00000) ***D[31] = 1*** **CLRES** (FEDBASE+0xa00004) ***D[31] = 1***

Write Cycles to this addresses with D[31]=1 generate 100ns reset pulses common to the Front Alteras (LRES) or the Center Altera (CLRES).

LRES resets the PixelStateMachine, clears all pipeline registers and flushes FIFO I, FIFO II, ErrorFIFO, and TemperatureFIFO.

CLRES flushes the HEADER FIFO and FIFO III.

One can selectively reset the Phase Locked Loop (PLL) on the front AlteraFPGA's with the following
LRES (FEDBASE+0x1c8000) ***D[29] = 1***

This LocalBus Address is used for special reset possibilities:

BROADCAST or CHP_ADDR + 0x1c8000 D[31..0]

Previously only D[31] was used in the FrontAlteras for flushing all SpyFIFOs and in the NORTH Chip for resetting the Test DAC RAMCounter.

Pixel B-channel Commands:

BRCST[5..0] = 0x2 (= reset EventCounter)
BRCST[5..0] = 0x14 (= FullReset)

FullReset is distributed to the Front ALTERAs and has the same effect as sending LRES and CLRES !

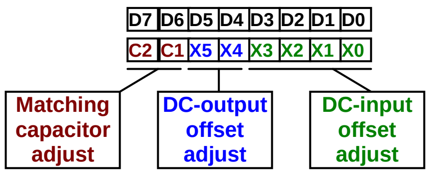
OPTO-RECEIVER parameters:

The parameters for the opto receivers can be set as follows:

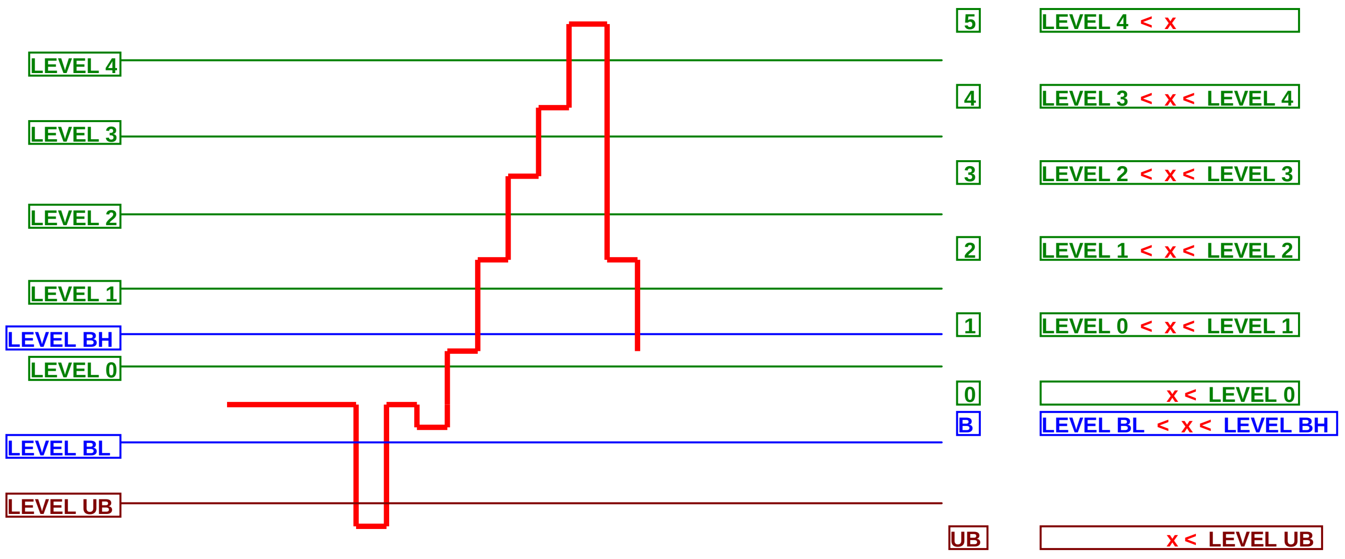
LocalBus Addr:

```
LAD_S + 0x180000      OPTOPAR RECEIVER1 D[7..0]     Channels  1 to 12
LAD_S + 0x188000      OPTOPAR RECEIVER2 D[7..0]     Channels 13 to 24
LAD_S + 0x190000      OPTOPAR RECEIVER3 D[7..0]     Channels 25 to 36
```

Data word :



Setting thresholds for the 6 discrete levels :



One have to set threshold levels for every ROC on all channels.

Address space:

```
UB_B_THRESH :    A[20..17]=Channel_Addr:        A[16..14]=2     common for all ROCs !
L012_THRESH :    A[20..17]=Channel_Addr:        A[16..14]=4     A[6..2]=ROC Address
UB_B_THRESH :    A[20..17]=Channel_Addr:        A[16..14]=5     A[6..2]=ROC Address
```

Data words:

```
BitPos: 31 30|29 28 27 26 25 24 23 22 21 20|19 18 17 16 15 14 13 12 11 10|09 08 07 06 05 04 03 02 01 00|

UB_B_THRESH  | < -----    BH    ---------- > | < -------    BL    ------- > | < ------    UB    -------- >|


BitPos: 31 30|29 28 27 26 25 24 23 22 21 20|19 18 17 16 15 14 13 12 11 10|09 08 07 06 05 04 03 02 01 00|

L012         | < -----LEVEL 2 ---------- > | < ------- LEVEL 1 ------- > | < ------  LEVEL 0 -------- >|

BitPos: 31 30 29 28 27 26 25 24 23 22 21 20|19 18 17 16 15 14 13 12 11 10|09 08 07 06 05 04 03 02 01 00|

L34                                        | < ------- LEVEL 4 ------- > | < ------  LEVEL 3 -------- >|
```

ROC numbering starts with one! ROC address zero is intended to be used for TBM HEADER levels and the address (Number of ROCs used + 1) can be used for TBM TRAILER levels since the internal ROC counter counts with each UB sequence and the maximum number of Rocs used is 24 (5bit ROC address range).

Individual channel offsets:

There are three 12 Channel DACs (AD8802) for channel offset control. The serial interface for these DACs is located in the NORTH Chip.

Address:
LAD_N + 0x190000

DataWord:

D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
CS3	CS2	CS1	A3	A2	A1	A0	D7	D6	D5	D4	D3	D2	D1	D0

CHIP select

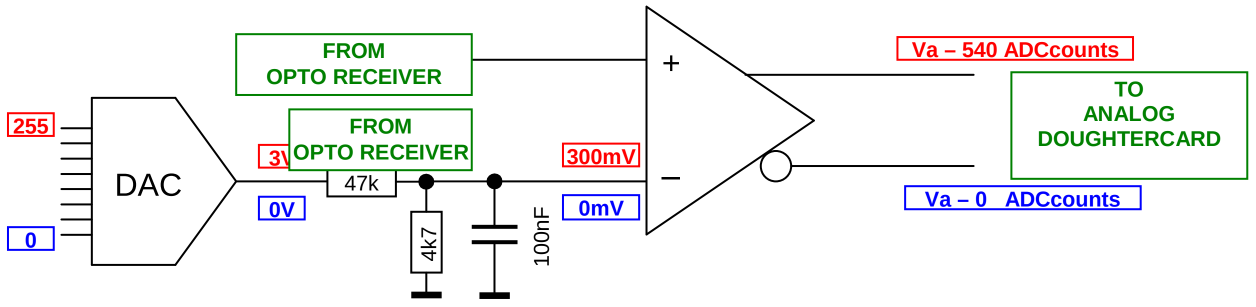
```
CS1: CH# 01..12
CS2: CH# 13..24
CS3: CH# 25..36
```

CHANNEL address

Dataword

Settling time due to external decoupling capacitor aprox. 1ms !

Analog Levels:



Individual channel ADC gain factor:

Each individual bit controls one ADC chip = 2 ADC channels ! I.e. Bit 0 for the North FPGA controls ADC channels 1 and 2, The ADC conversion can be changed from 1Vpp = 1024 counts (bit=0) to 2 Vpp = 1024 counts (bit=1).
Also don't forget to change the offset DAC: 0 ADC counts for 1Vpp(2Vpp) = -0.5V (-1.0V)!

Address:
LAD_N(NC,SC,S) + 0x1b8000

DataWord:

	D5	D4	D3	D2	D1	D0
NORTH	CH#12 CH#11	CH#10 CH#09	CH#08 CH#07	CH#06 CH#05	CH#04 CH#03	CH#02 CH#01
NORTH Cntr			CH#20 CH#19	CH#18 CH#17	CH#16 CH#15	CH#14 CH#13
SOUTH Cntr			CH#28 CH#27	CH#26 CH#25	CH#24 CH#23	CH#22 CH#21
SOUTH			CH#36 CH#35	CH#34 CH#33	CH#32 CH#31	CH#30 CH#29

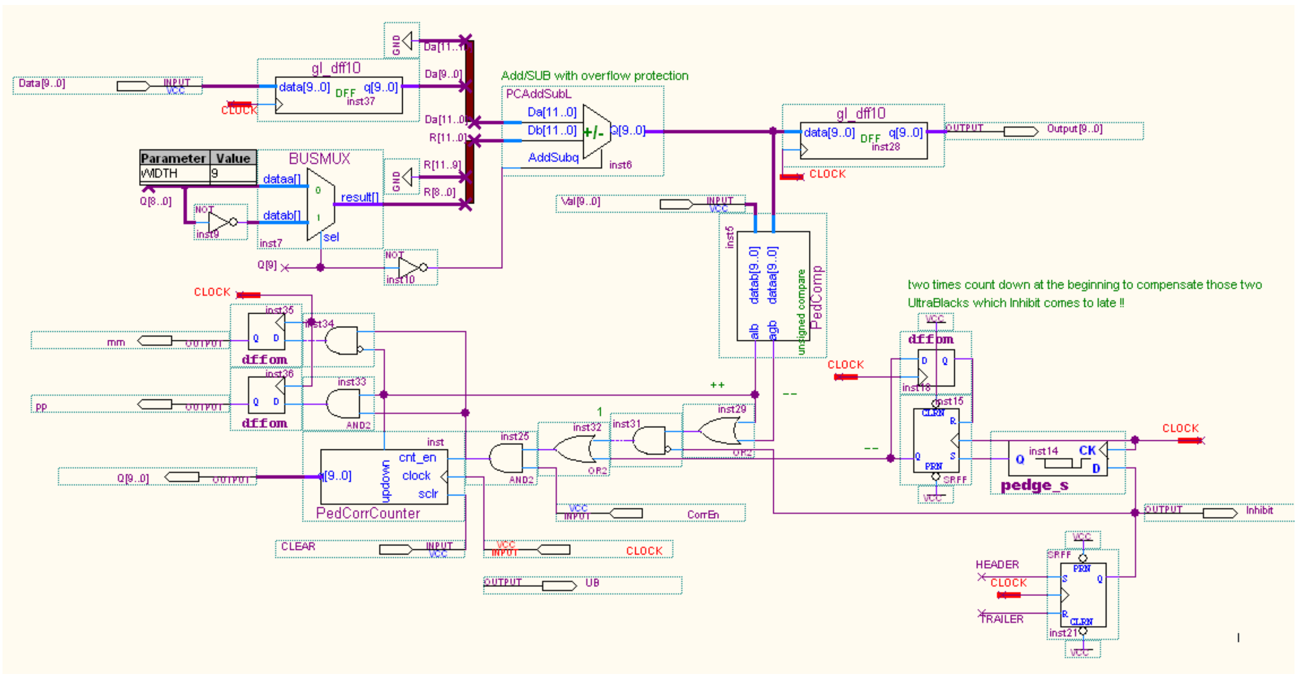
Automatic baseline correction:

If this feature is enabled the FED uses the time between TBM_TRAILER and TBM_HEADER to correct the “black level” to a programmable value by digitally adding or subtracting.

We suggest the following procedure:

.) Disable the auto correction
.) Add offset to get black Level approximately correct
.) Determine UB and B levels in transparent mode (necessary to recognise header and trailer !)
.) Program desired black level and enable auto correction
.) Perform additional transparent run to determine the other levels

Schematic detail:

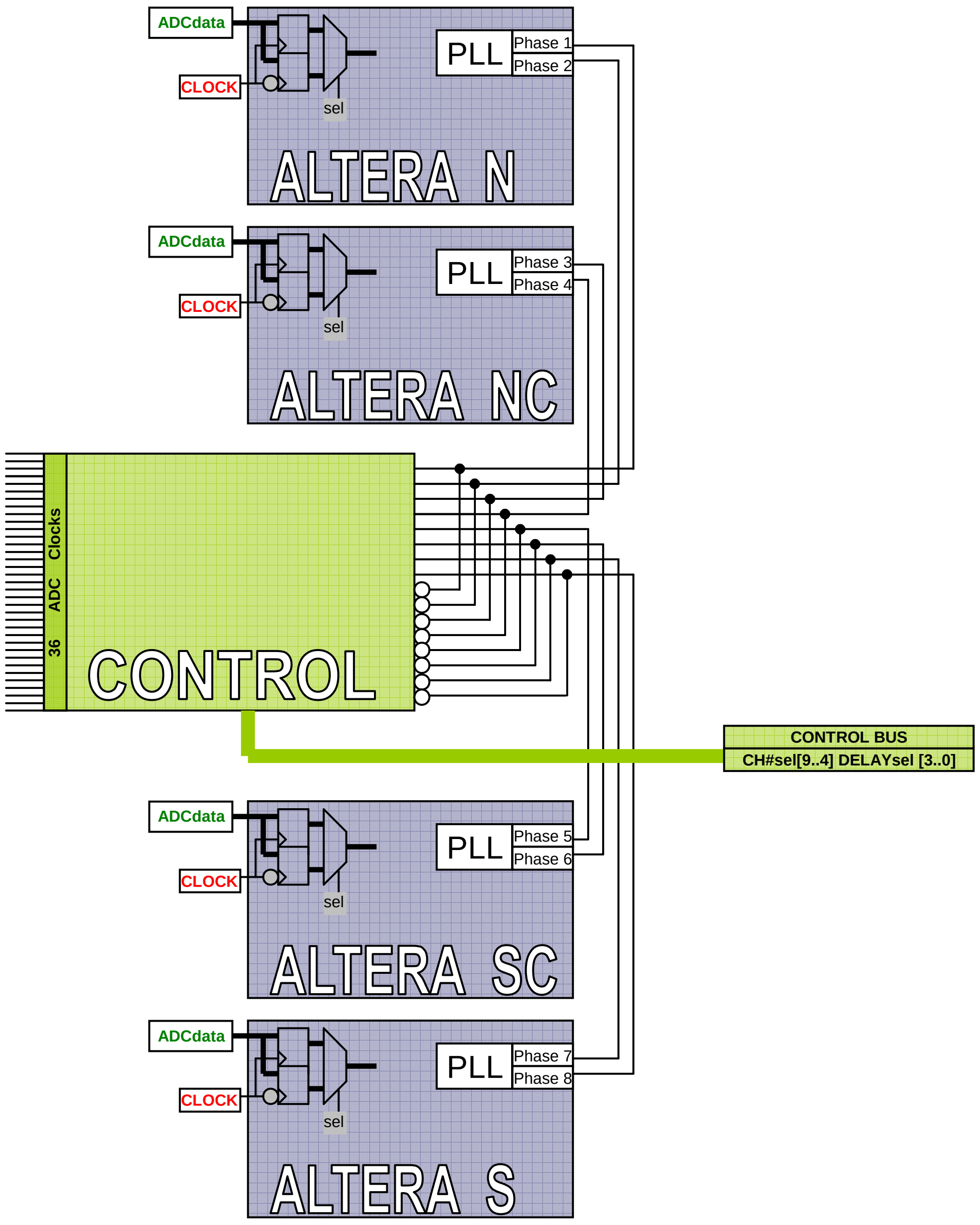

gl_dff10
Add/SUB with overflow protection
PCAddSubL
AddSubq
BUSMUX
PedCorrCounter
PedComp
two times count down at the beginning to compensate those two UltraBlacks which Inhibit comes to late !!
pedge_s
CLOCK
CLEAR
HEADER
TRAILER
Inhibit
Output[9..0]
Data[9..0]
UB
CorrEn

Addresses:
write:
```
CHP_ADDR+0x1d0000      D[24..16] enableAutoCorr CH#[9..1]  D[9..0] Common value for channels 1- 9
```
read:
```
CHP_ADDR+0x1d0000      D[29..0] Correction Value CH# 3 2 1  ( each 10 bit signed  )
CHP_ADDR+0x1d8000      D[29..0] Correction Value CH# 6 5 4
CHP_ADDR+0x1e0000      D[29..0] Correction Value CH# 9 8 7
```

CLOCK PHASE ADJUSTEMENT:

It is possible to adjust the ADC clock in 16 steps of about 1.5ns
Since the ADC clock is shifted relative to the ALTERA system clock it is necessary to implement a negative edge clocked register for some delay positions to ensure proper setup and hold times for the ADC data path.

Clock phase scheme:



Clock phase setting is done as follows:

First: One have to set the clock phase for the selected channel.

Address:
`LAD_C + 0x198000`

DataWord:

D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
E	X	A5	A4	A3	A2	A1	A0	D3	D2	D1	D0

Enable Bit
to distinguish between delay settings and other data in the CONTROL chip

CHANNEL number

Delay value

Second: Don’t forget to select the appropriate clock edge for the first register in the ADC data path !

Address:
`LAD_CHIPnr + 0x1b0000`

DataWord:

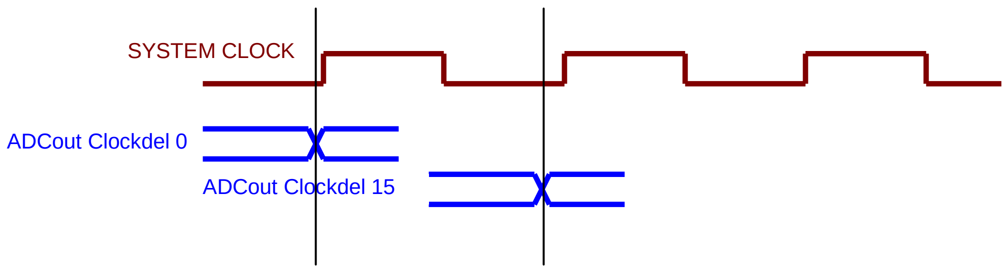
D8	D7	D6	D5	D4	D3	D2	D1	D0
S9	S8	S7	S6	S5	S4	S3	S2	S1

ChannelClockPhaseSelect
Bit set => use neg. clock phase

To save address space there is only one 9bit wide register for all input channels of a chip (N, NC, SC, S), therefore it is necessary to restore phase information from the other channels.

The window for positive and negative clock phase depends on the phase relationship between system clock and delayed clocks.

For version 3 modules :



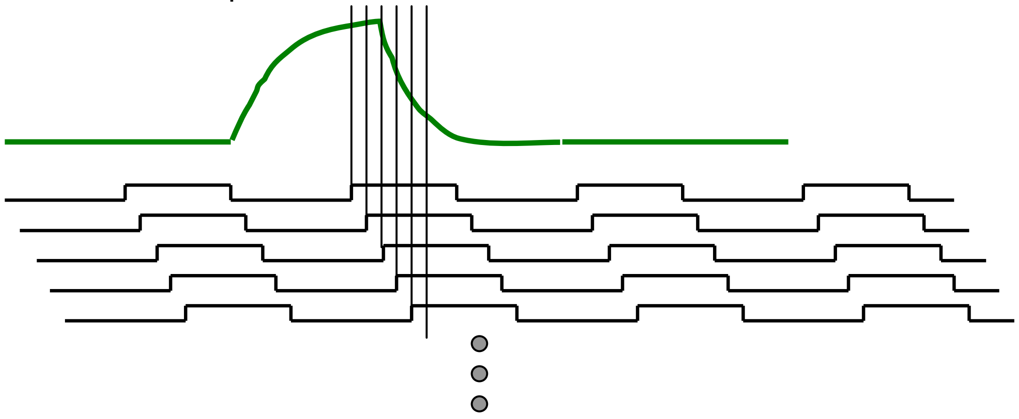
this gives the window:

Del 0	Del 1	Del 2	Del 3	Del 4	Del 5	Del 6	Del 7	Del 8	Del 9	Del10	Del11	Del12	Del13	Del14	Del15
N	N	N	N	N	N	P	P	P	P	P	P	P	N	N	N

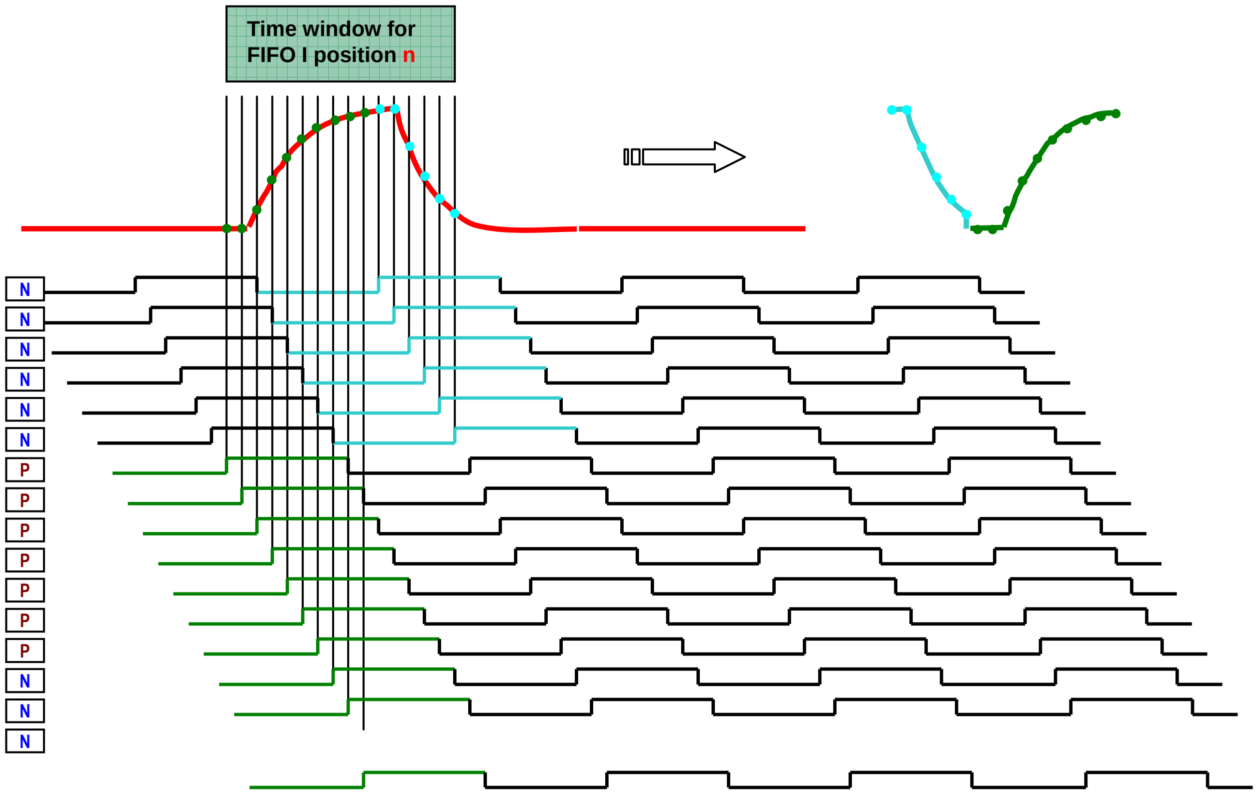
CLOCK PHASE

TEST DAC PULSE SCAN:

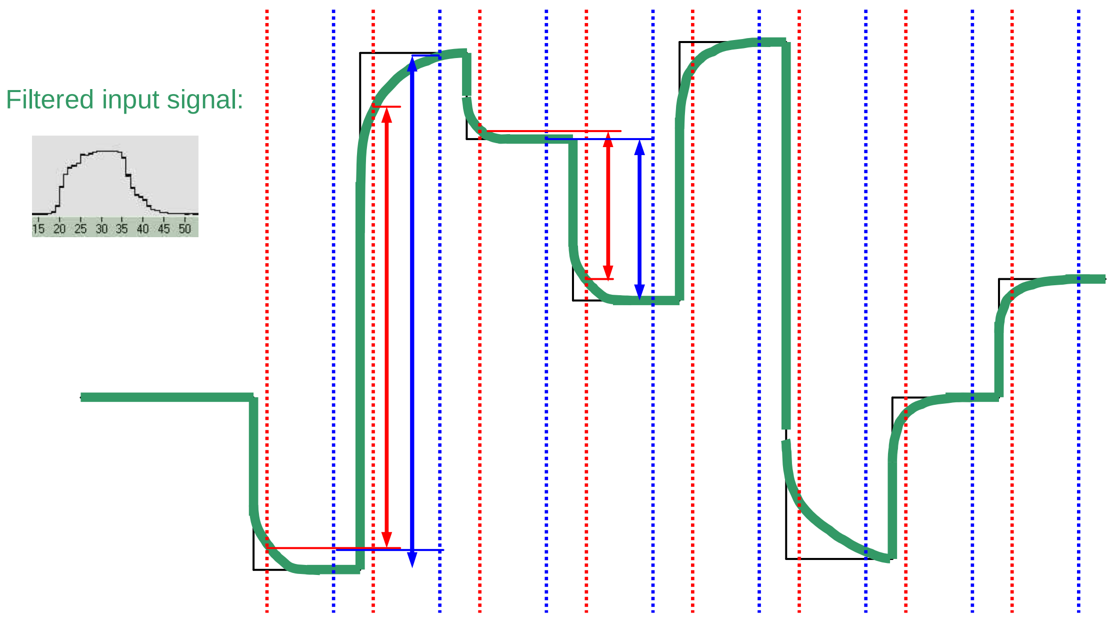
To understand the basic mechanism of clock phase adjustment, please have a look on following example.
We send sixteen times a 25ns pulse over our TEST DAC system and scan it each time with a different clock phase.



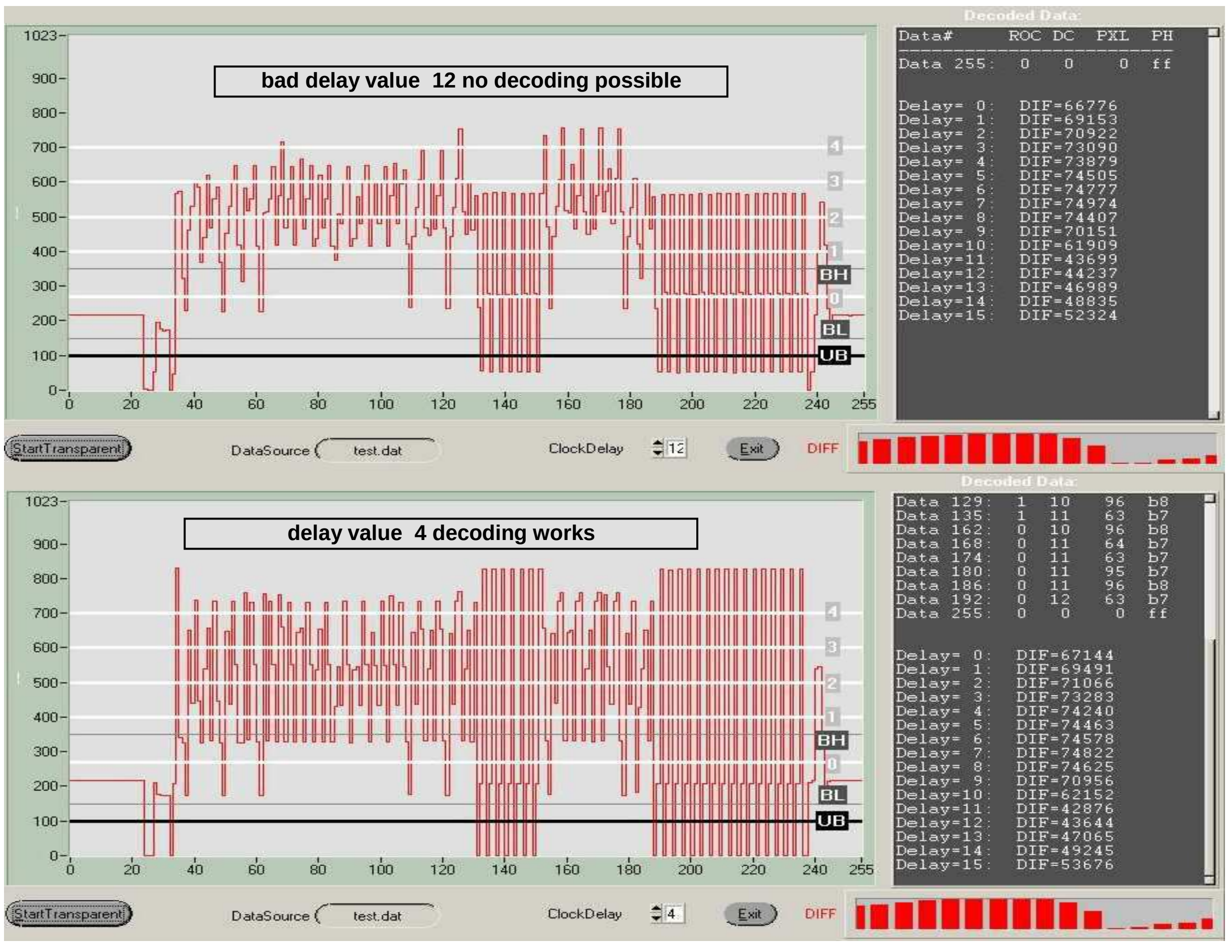
When you have found the position **n** of the pulse in FIFO I it is necessary to realign all sixteen samples to get the correct pulse shape.



An idea how to find the best sampling point:



When we shift the sampling clock and build the difference of two consecutive samples, we will get a minimum when we sample at the time when the data is changing.
In the figure above one can see that the red differences are always smaller than the blue ones.

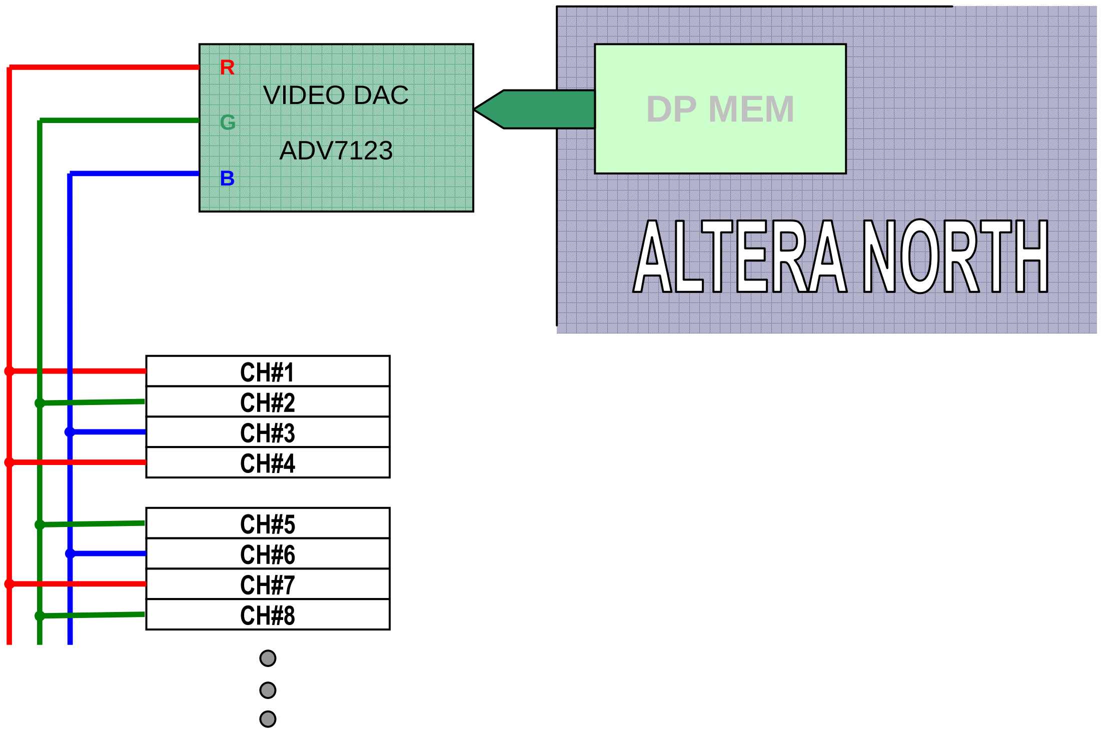


```
Data#        ROC  DC   PXL   PH
-------------------------------
Data 255:     0    0     0   ff


Delay= 0:   DIF=66776
Delay= 1:   DIF=69153
Delay= 2:   DIF=70922
Delay= 3:   DIF=73090
Delay= 4:   DIF=73879
Delay= 5:   DIF=74505
Delay= 6:   DIF=74777
Delay= 7:   DIF=74974
Delay= 8:   DIF=74407
Delay= 9:   DIF=70151
Delay=10:   DIF=61909
Delay=11:   DIF=43699
Delay=12:   DIF=44237
Delay=13:   DIF=46989
Delay=14:   DIF=48835
Delay=15:   DIF=52324
```

```
Data 129:   1  10    96  b8
Data 135:   1  11    63  b7
Data 162:   0  10    96  b8
Data 168:   0  11    64  b7
Data 174:   0  11    63  b7
Data 180:   0  11    95  b7
Data 186:   0  11    96  b8
Data 192:   0  12    63  b7
Data 255:   0   0     0  ff


Delay= 0:   DIF=67144
Delay= 1:   DIF=69491
Delay= 2:   DIF=71066
Delay= 3:   DIF=73283
Delay= 4:   DIF=74240
Delay= 5:   DIF=74463
Delay= 6:   DIF=74578
Delay= 7:   DIF=74822
Delay= 8:   DIF=74625
Delay= 9:   DIF=70956
Delay=10:   DIF=62152
Delay=11:   DIF=42876
Delay=12:   DIF=43644
Delay=13:   DIF=47065
Delay=14:   DIF=49245
Delay=15:   DIF=53676
```

TestDAC Loading:

The built in Test system is based on the following scheme and is used for the generation of “Pixel look a like Data” to test the entire module.



Every third channel gets the same test pattern !

Address:
LAD_N + 0x180000

DataWord:

D31	D29	D28	D27	D26	D25	D24	D23	D22	D21	D20	D19	D18	D17	D16	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

Dataword B **Dataword G** **Dataword R**

Test pattern should start at the beginning of the transparent gate, since we wait now only 64 clocks after L1A for the first header word, and should be a maximum of 256 DataWords long.

The gate can be shortened by setting bit[31] = 1 at the end of the test pattern!

NORTH_V3 LocalBus Addressmap:

		A20	A19	A18	A17	A16	A15	A14	
CHP_ADDR + 0x20000	Channel#1	0	0	0	1	0	0	0	Number of estimated ROCs D[4..0]
CHP_ADDR + 0x28000	Channel#1	0	0	0	1	0	1	0	UltraBlack Threshold D[29..0]
CHP_ADDR + 0x30000	Channel#1	0	0	0	1	1	0	0	Threshold_012 D[29..0]
CHP_ADDR + 0x34000	Channel#1	0	0	0	1	1	0	1	Threshold_34 D[19..0]
CHP_ADDR + 0x38000	Channel#1	0	0	0	1	1	1	0	READ FIFO1 D[31..0]
CHP_ADDR + 0x40000	Channel#2	0	0	1	0	0	0	0	Number of estimated ROCs D[4..0]
CHP_ADDR + 0x48000	Channel#2	0	0	1	0	0	1	0	UltraBlack Threshold D[29..0]
CHP_ADDR + 0x50000	Channel#2	0	0	1	0	1	0	0	Threshold_012 D[29..0]
CHP_ADDR + 0x54000	Channel#2	0	0	1	0	1	0	1	Threshold_34 D[19..0]
CHP_ADDR + 0x58000	Channel#2	0	0	1	0	1	1	0	READ FIFO1 D[31..0]
CHP_ADDR + 0x60000	Channel#3	0	0	1	1	0	0	0	Number of estimated ROCs D[4..0]
CHP_ADDR + 0x68000	Channel#3	0	0	1	1	0	1	0	UltraBlack Threshold D[29..0]
CHP_ADDR + 0x70000	Channel#3	0	0	1	1	1	0	0	Threshold_012 D[29..0]
CHP_ADDR + 0x74000	Channel#3	0	0	1	1	1	0	1	Threshold_34 D[19..0]
CHP_ADDR + 0x78000	Channel#3	0	0	1	1	1	1	0	READ FIFO1 D[31..0]
CHP_ADDR + 0x80000	Channel#4	0	1	0	0	0	0	0	Number of estimated ROCs D[4..0]
CHP_ADDR + 0x88000	Channel#4	0	1	0	0	0	1	0	UltraBlack Threshold D[29..0]
CHP_ADDR + 0x90000	Channel#4	0	1	0	0	1	0	0	Threshold_012 D[29..0]
CHP_ADDR + 0x94000	Channel#4	0	1	0	0	1	0	1	Threshold_34 D[19..0]
CHP_ADDR + 0x98000	Channel#4	0	1	0	0	1	1	0	READ FIFO1 D[31..0]
CHP_ADDR + 0xa0000	Channel#5	0	1	0	1	0	0	0	Number of estimated ROCs D[4..0]
CHP_ADDR + 0xa8000	Channel#5	0	1	0	1	0	1	0	UltraBlack Threshold D[29..0]
CHP_ADDR + 0xb0000	Channel#5	0	1	0	1	1	0	0	Threshold_012 D[29..0]
CHP_ADDR + 0xb4000	Channel#5	0	1	0	1	1	0	1	Threshold_34 D[19..0]
CHP_ADDR + 0xb8000	Channel#5	0	1	0	1	1	1	0	READ FIFO1 D[31..0]
CHP_ADDR + 0xc0000	Channel#6	0	1	1	0	0	0	0	Number of estimated ROCs D[4..0]
CHP_ADDR + 0xc8000	Channel#6	0	1	1	0	0	1	0	UltraBlack Threshold D[29..0]
CHP_ADDR + 0xd0000	Channel#6	0	1	1	0	1	0	0	Threshold_012 D[29..0]
CHP_ADDR + 0xd4000	Channel#6	0	1	1	0	1	0	1	Threshold_34 D[19..0]
CHP_ADDR + 0xd8000	Channel#6	0	1	1	0	1	1	0	READ FIFO1 D[31..0]
CHP_ADDR + 0xe0000	Channel#7	0	1	1	1	0	0	0	Number of estimated ROCs D[4..0]
CHP_ADDR + 0xe8000	Channel#7	0	1	1	1	0	1	0	UltraBlack Threshold D[29..0]
CHP_ADDR + 0xf0000	Channel#7	0	1	1	1	1	0	0	Threshold_012 D[29..0]
CHP_ADDR + 0xf4000	Channel#7	0	1	1	1	1	0	1	Threshold_34 D[19..0]
CHP_ADDR + 0xf8000	Channel#7	0	1	1	1	1	1	0	READ FIFO1 D[31..0]
CHP_ADDR + 0x100000	Channel#8	1	0	0	0	0	0	0	Number of estimated ROCs D[4..0]
CHP_ADDR + 0x108000	Channel#8	1	0	0	0	0	1	0	UltraBlack Threshold D[29..0]
CHP_ADDR + 0x110000	Channel#8	1	0	0	0	1	0	0	Threshold_012 D[29..0]
CHP_ADDR + 0x114000	Channel#8	1	0	0	0	1	0	1	Threshold_34 D[19..0]
CHP_ADDR + 0x118000	Channel#8	1	0	0	0	1	1	0	READ FIFO1 D[31..0]
CHP_ADDR + 0x120000	Channel#9	1	0	0	1	0	0	0	Number of estimated ROCs D[4..0]
CHP_ADDR + 0x128000	Channel#9	1	0	0	1	0	1	0	UltraBlack Threshold D[29..0]
CHP_ADDR + 0x130000	Channel#9	1	0	0	1	1	0	0	Threshold_012 D[29..0]
CHP_ADDR + 0x134000	Channel#9	1	0	0	1	1	0	1	Threshold_34 D[19..0]
CHP_ADDR + 0x138000	Channel#9	1	0	0	1	1	1	0	READ FIFO1 D[31..0]
CHP_ADDR + 0x140000		1	0	1	0	0	0		READ_ERROR_FIFOup D[31..0]
CHP_ADDR + 0x148000		1	0	1	0	0	1		READ TEMP_FIFOup D[31..0]
CHP_ADDR + 0x150000		1	0	1	0	1	0		READ SPY-FIFO2up D[31..0]
CHP_ADDR + 0x158000		1	0	1	0	1	1		READ SPY-I-up D[31..0]
CHP_ADDR + 0x160000		1	0	1	1	0	0		READ_ERROR_FIFOdown D[31..0]
CHP_ADDR + 0x168000		1	0	1	1	0	1		READ TEMP_FIFOdown D[31..0]
CHP_ADDR + 0x170000		1	0	1	1	1	0		READ SPY-FIFO2down D[31..0]
CHP_ADDR + 0x178000		1	0	1	1	1	1		READ SPY-I-down D[31..0]
CHP_ADDR + 0x180000		1	1	0	0	0	0		Write DAC_MEM D[29..0]
CHP_ADDR + 0x188000		1	1	0	0	0	1		Write Ovfl Val. D[7..0]+192 not impl.
CHP_ADDR + 0x190000		1	1	0	0	1	0		W OffsetDAC D15,D14,D13,D12,D[11..0]
CHP_ADDR + 0x190000		1	1	0	0	1	1		R StateM Errors[D26..0]
CHP_ADDR + 0x1a0000		1	1	0	1	0	0		Write/Read ControlReg D[31..0]
CHP_ADDR + 0x1a8000		1	1	0	1	0	1		Write/Read TestReg D[31..0]
CHP_ADDR + 0x1a0000		1	1	0	1	1	0		Write CLK_Reg D[8..0]
CHP_ADDR + 0x1a8000		1	1	0	1	1	1		Write GAIN_Reg D[3..0]
BROADCAST or CHP_ADDR + 0x1c0000		1	1	1	0	0	0		Write ModeReg D[31..0]
BROADCAST or CHP_ADDR + 0x1c8000		1	1	1	0	0	1		Write ResetPuls D[31..0]
CHP_ADDR + 0x1d0000		1	1	1	0	1	0		AutoBaselEn[D24..16] ValueD [9..0]
CHP_ADDR + 0x1d0000		1	1	1	0	1	0		Read CorrVal ch# 3 2 1 [D29..0]
CHP_ADDR + 0x1d8000		1	1	1	0	1	1		Read CorrVal ch# 6 5 4 [D29..0]
CHP_ADDR + 0x1e0000		1	1	1	1	0	0		Read CorrVal ch# 9 8 7 [D29..0]
CHP_ADDR + 0x1f0000		1	1	1	1	1	0		ReadFirmwareVersion
CHP_ADDR + 0x1e800		1	1	1	1	0	1		Feature Register

NORTH_CENTERV2 , SOUTH_CENTERV2 LocalBus Addressmap:

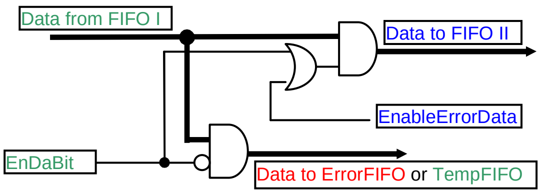
		A20	A19	A18	A17	A16	A15	A14	
CHP_ADDR + 0x20000	Channel#1	0	0	0	1	0	0	0	Number of estimated ROCs D[4..0]
CHP_ADDR + 0x28000	Channel#1	0	0	0	1	0	1	0	UltraBlack Threshold D[29..0]
CHP_ADDR + 0x30000	Channel#1	0	0	0	1	1	0	0	Threshold_012 D[29..0]
CHP_ADDR + 0x34000	Channel#1	0	0	0	1	1	0	1	Threshold_34 D[19..0]
CHP_ADDR + 0x38000	Channel#1	0	0	0	1	1	1	0	READ FIFO1 D[31..0]
CHP_ADDR + 0x40000	Channel#2	0	0	1	0	0	0	0	Number of estimated ROCs D[4..0]
CHP_ADDR + 0x48000	Channel#2	0	0	1	0	0	1	0	UltraBlack Threshold D[29..0]
CHP_ADDR + 0x50000	Channel#2	0	0	1	0	1	0	0	Threshold_012 D[29..0]
CHP_ADDR + 0x54000	Channel#2	0	0	1	0	1	0	1	Threshold_34 D[19..0]
CHP_ADDR + 0x58000	Channel#2	0	0	1	0	1	1	0	READ FIFO1 D[31..0]
CHP_ADDR + 0x60000	Channel#3	0	0	1	1	0	0	0	Number of estimated ROCs D[4..0]
CHP_ADDR + 0x68000	Channel#3	0	0	1	1	0	1	0	UltraBlack Threshold D[29..0]
CHP_ADDR + 0x70000	Channel#3	0	0	1	1	1	0	0	Threshold_012 D[29..0]
CHP_ADDR + 0x74000	Channel#3	0	0	1	1	1	0	1	Threshold_34 D[19..0]
CHP_ADDR + 0x78000	Channel#3	0	0	1	1	1	1	0	READ FIFO1 D[31..0]
CHP_ADDR + 0x80000	Channel#4	0	1	0	0	0	0	0	Number of estimated ROCs D[4..0]
CHP_ADDR + 0x88000	Channel#4	0	1	0	0	0	1	0	UltraBlack Threshold D[29..0]
CHP_ADDR + 0x90000	Channel#4	0	1	0	0	1	0	0	Threshold_012 D[29..0]
CHP_ADDR + 0x94000	Channel#4	0	1	0	0	1	0	1	Threshold_34 D[19..0]
CHP_ADDR + 0x98000	Channel#4	0	1	0	0	1	1	0	READ FIFO1 D[31..0]
CHP_ADDR + 0xa0000	Channel#5	0	1	0	1	0	0	0	Number of estimated ROCs D[4..0]
CHP_ADDR + 0xa8000	Channel#5	0	1	0	1	0	1	0	UltraBlack Threshold D[29..0]
CHP_ADDR + 0xb0000	Channel#5	0	1	0	1	1	0	0	Threshold_012 D[29..0]
CHP_ADDR + 0xb4000	Channel#5	0	1	0	1	1	0	1	Threshold_34 D[19..0]
CHP_ADDR + 0xb8000	Channel#5	0	1	0	1	1	1	0	READ FIFO1 D[31..0]
CHP_ADDR + 0xc0000	Channel#6	0	1	1	0	0	0	0	Number of estimated ROCs D[4..0]
CHP_ADDR + 0xc8000	Channel#6	0	1	1	0	0	1	0	UltraBlack Threshold D[29..0]
CHP_ADDR + 0xd0000	Channel#6	0	1	1	0	1	0	0	Threshold_012 D[29..0]
CHP_ADDR + 0xd4000	Channel#6	0	1	1	0	1	0	1	Threshold_34 D[19..0]
CHP_ADDR + 0xd8000	Channel#6	0	1	1	0	1	1	0	READ FIFO1 D[31..0]
CHP_ADDR + 0xe0000	Channel#7	0	1	1	1	0	0	0	Number of estimated ROCs D[4..0]
CHP_ADDR + 0xe8000	Channel#7	0	1	1	1	0	1	0	UltraBlack Threshold D[29..0]
CHP_ADDR + 0xf0000	Channel#7	0	1	1	1	1	0	0	Threshold_012 D[29..0]
CHP_ADDR + 0xf4000	Channel#7	0	1	1	1	1	0	1	Threshold_34 D[19..0]
CHP_ADDR + 0xf8000	Channel#7	0	1	1	1	1	1	0	READ FIFO1 D[31..0]
CHP_ADDR + 0x100000	Channel#8	1	0	0	0	0	0	0	Number of estimated ROCs D[4..0]
CHP_ADDR + 0x108000	Channel#8	1	0	0	0	0	1	0	UltraBlack Threshold D[29..0]
CHP_ADDR + 0x110000	Channel#8	1	0	0	0	1	0	0	Threshold_012 D[29..0]
CHP_ADDR + 0x114000	Channel#8	1	0	0	0	1	0	1	Threshold_34 D[19..0]
CHP_ADDR + 0x118000	Channel#8	1	0	0	0	1	1	0	READ FIFO1 D[31..0]
CHP_ADDR + 0x120000	Channel#9	1	0	0	1	0	0	0	Number of estimated ROCs D[4..0]
CHP_ADDR + 0x128000	Channel#9	1	0	0	1	0	1	0	UltraBlack Threshold D[29..0]
CHP_ADDR + 0x130000	Channel#9	1	0	0	1	1	0	0	Threshold_012 D[29..0]
CHP_ADDR + 0x134000	Channel#9	1	0	0	1	1	0	1	Threshold_34 D[19..0]
CHP_ADDR + 0x138000	Channel#9	1	0	0	1	1	1	0	READ FIFO1 D[31..0]
CHP_ADDR + 0x140000		1	0	1	0	0	0		READ_ERROR_FIFOup D[31..0]
CHP_ADDR + 0x148000		1	0	1	0	0	1		READ TEMP_FIFOup D[31..0]
CHP_ADDR + 0x150000		1	0	1	0	1	0		READ SPY-FIFO2up D[31..0]
CHP_ADDR + 0x158000		1	0	1	0	1	1		
CHP_ADDR + 0x160000		1	0	1	1	0	0		READ_ERROR_FIFOdown D[31..0]
CHP_ADDR + 0x168000		1	0	1	1	0	1		READ TEMP_FIFOdown D[31..0]
CHP_ADDR + 0x170000		1	0	1	1	1	0		READ SPY-FIFO2down D[31..0]
CHP_ADDR + 0x178000		1	0	1	1	1	1		
CHP_ADDR + 0x180000		1	1	0	0	0	0		
CHP_ADDR + 0x188000		1	1	0	0	0	1		
CHP_ADDR + 0x190000		1	1	0	0	1	0		
CHP_ADDR + 0x1a0000		1	1	0	1	0	0		Write/Read ControlReg D[31..0]
CHP_ADDR + 0x1a8000		1	1	0	1	0	1		Write/Read TestReg D[31..0]
CHP_ADDR + 0x1a0000		1	1	0	1	1	0		Write CLK_Reg D[8..0]
CHP_ADDR + 0x1a8000		1	1	0	1	1	1		Write GAIN_Reg D[3..0]
BROADCAST or CHP_ADDR + 0x1c0000		1	1	1	0	0	0		Write ModeReg D[31..0]
BROADCAST or CHP_ADDR + 0x1c8000		1	1	1	0	0	1		Write ResetPuls D[31..0]
CHP_ADDR + 0x1d0000		1	1	1	0	1	0		AutoBaselEn[D24..16] ValueD [9..0]
CHP_ADDR + 0x1d0000		1	1	1	0	1	0		Read CorrVal ch# 3 2 1 [D29..0]
CHP_ADDR + 0x1d8000		1	1	1	0	1	1		Read CorrVal ch# 6 5 4 [D29..0]
CHP_ADDR + 0x1e0000		1	1	1	1	0	0		Read CorrVal ch# 9 8 7 [D29..0]
CHP_ADDR + 0x1f0000		1	1	1	1	1	0		ReadFirmwareVersion
CHP_ADDR + 0x1e800		1	1	1	1	0	1		Feature Register

SOUTH_V2 LocalBus Addressmap:

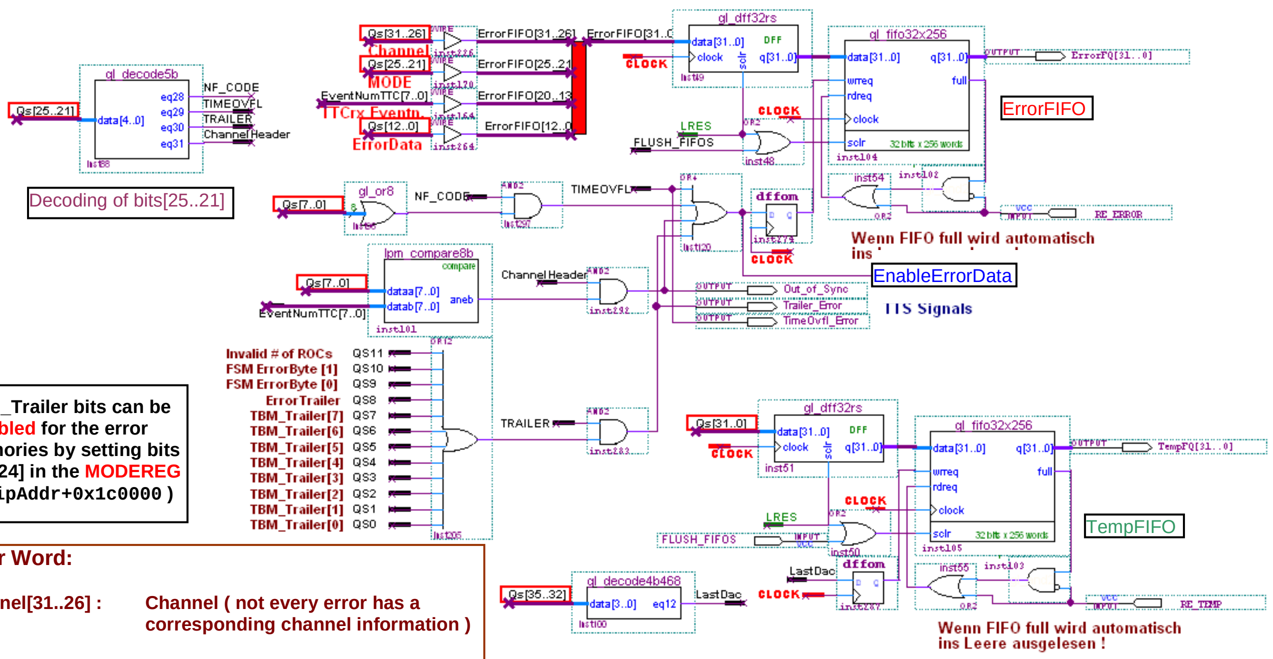
		A20	A19	A18	A17	A16	A15	A14	
CHP_ADDR + 0x20000	Channel#1	0	0	0	1	0	0	0	Number of estimated ROCs D[4..0]
CHP_ADDR + 0x28000	Channel#1	0	0	0	1	0	1	0	UltraBlack Threshold D[29..0]
CHP_ADDR + 0x30000	Channel#1	0	0	0	1	1	0	0	Threshold_012 D[29..0]
CHP_ADDR + 0x34000	Channel#1	0	0	0	1	1	0	1	Threshold_34 D[19..0]
CHP_ADDR + 0x38000	Channel#1	0	0	0	1	1	1	0	READ FIFO1 D[31..0]
CHP_ADDR + 0x40000	Channel#2	0	0	1	0	0	0	0	Number of estimated ROCs D[4..0]
CHP_ADDR + 0x48000	Channel#2	0	0	1	0	0	1	0	UltraBlack Threshold D[29..0]
CHP_ADDR + 0x50000	Channel#2	0	0	1	0	1	0	0	Threshold_012 D[29..0]
CHP_ADDR + 0x54000	Channel#2	0	0	1	0	1	0	1	Threshold_34 D[19..0]
CHP_ADDR + 0x58000	Channel#2	0	0	1	0	1	1	0	READ FIFO1 D[31..0]
CHP_ADDR + 0x60000	Channel#3	0	0	1	1	0	0	0	Number of estimated ROCs D[4..0]
CHP_ADDR + 0x68000	Channel#3	0	0	1	1	0	1	0	UltraBlack Threshold D[29..0]
CHP_ADDR + 0x70000	Channel#3	0	0	1	1	1	0	0	Threshold_012 D[29..0]
CHP_ADDR + 0x74000	Channel#3	0	0	1	1	1	0	1	Threshold_34 D[19..0]
CHP_ADDR + 0x78000	Channel#3	0	0	1	1	1	1	0	READ FIFO1 D[31..0]
CHP_ADDR + 0x80000	Channel#4	0	1	0	0	0	0	0	Number of estimated ROCs D[4..0]
CHP_ADDR + 0x88000	Channel#4	0	1	0	0	0	1	0	UltraBlack Threshold D[29..0]
CHP_ADDR + 0x90000	Channel#4	0	1	0	0	1	0	0	Threshold_012 D[29..0]
CHP_ADDR + 0x94000	Channel#4	0	1	0	0	1	0	1	Threshold_34 D[19..0]
CHP_ADDR + 0x98000	Channel#4	0	1	0	0	1	1	0	READ FIFO1 D[31..0]
CHP_ADDR + 0xa0000	Channel#5	0	1	0	1	0	0	0	Number of estimated ROCs D[4..0]
CHP_ADDR + 0xa8000	Channel#5	0	1	0	1	0	1	0	UltraBlack Threshold D[29..0]
CHP_ADDR + 0xb0000	Channel#5	0	1	0	1	1	0	0	Threshold_012 D[29..0]
CHP_ADDR + 0xb4000	Channel#5	0	1	0	1	1	0	1	Threshold_34 D[19..0]
CHP_ADDR + 0xb8000	Channel#5	0	1	0	1	1	1	0	READ FIFO1 D[31..0]
CHP_ADDR + 0xc0000	Channel#6	0	1	1	0	0	0	0	Number of estimated ROCs D[4..0]
CHP_ADDR + 0xc8000	Channel#6	0	1	1	0	0	1	0	UltraBlack Threshold D[29..0]
CHP_ADDR + 0xd0000	Channel#6	0	1	1	0	1	0	0	Threshold_012 D[29..0]
CHP_ADDR + 0xd4000	Channel#6	0	1	1	0	1	0	1	Threshold_34 D[19..0]
CHP_ADDR + 0xd8000	Channel#6	0	1	1	0	1	1	0	READ FIFO1 D[31..0]
CHP_ADDR + 0xe0000	Channel#7	0	1	1	1	0	0	0	Number of estimated ROCs D[4..0]
CHP_ADDR + 0xe8000	Channel#7	0	1	1	1	0	1	0	UltraBlack Threshold D[29..0]
CHP_ADDR + 0xf0000	Channel#7	0	1	1	1	1	0	0	Threshold_012 D[29..0]
CHP_ADDR + 0xf4000	Channel#7	0	1	1	1	1	0	1	Threshold_34 D[19..0]
CHP_ADDR + 0xf8000	Channel#7	0	1	1	1	1	1	0	READ FIFO1 D[31..0]
CHP_ADDR + 0x100000	Channel#8	1	0	0	0	0	0	0	Number of estimated ROCs D[4..0]
CHP_ADDR + 0x108000	Channel#8	1	0	0	0	0	1	0	UltraBlack Threshold D[29..0]
CHP_ADDR + 0x110000	Channel#8	1	0	0	0	1	0	0	Threshold_012 D[29..0]
CHP_ADDR + 0x114000	Channel#8	1	0	0	0	1	0	1	Threshold_34 D[19..0]
CHP_ADDR + 0x118000	Channel#8	1	0	0	0	1	1	0	READ FIFO1 D[31..0]
CHP_ADDR + 0x120000	Channel#9	1	0	0	1	0	0	0	Number of estimated ROCs D[4..0]
CHP_ADDR + 0x128000	Channel#9	1	0	0	1	0	1	0	UltraBlack Threshold D[29..0]
CHP_ADDR + 0x130000	Channel#9	1	0	0	1	1	0	0	Threshold_012 D[29..0]
CHP_ADDR + 0x134000	Channel#9	1	0	0	1	1	0	1	Threshold_34 D[19..0]
CHP_ADDR + 0x138000	Channel#9	1	0	0	1	1	1	0	READ FIFO1 D[31..0]
CHP_ADDR + 0x140000		1	0	1	0	0	0		READ_ERROR_FIFOup D[31..0]
CHP_ADDR + 0x148000		1	0	1	0	0	1		READ TEMP_FIFOup D[31..0]
CHP_ADDR + 0x150000		1	0	1	0	1	0		READ SPY-FIFO2up D[31..0]
CHP_ADDR + 0x158000		1	0	1	0	1	1		READ SPY-I-up D[31..0]
CHP_ADDR + 0x160000		1	0	1	1	0	0		READ_ERROR_FIFOdown D[31..0]
CHP_ADDR + 0x168000		1	0	1	1	0	1		READ TEMP_FIFOdown D[31..0]
CHP_ADDR + 0x170000		1	0	1	1	1	0		READ SPY-FIFO2down D[31..0]
CHP_ADDR + 0x178000		1	0	1	1	1	1		READ SPY-I-down D[31..0]
CHP_ADDR + 0x180000		1	1	0	0	0	0		OPTOPAR RECEIVER1 D[7..0]
CHP_ADDR + 0x188000		1	1	0	0	0	1		OPTOPAR RECEIVER2 D[7..0]
CHP_ADDR + 0x190000		1	1	0	0	1	0		OPTOPAR RECEIVER3 D[7..0]
CHP_ADDR + 0x1a0000		1	1	0	1	0	0		Write/Read ControlReg D[31..0]
CHP_ADDR + 0x1a8000		1	1	0	1	0	1		Write/Read TestReg D[31..0]
CHP_ADDR + 0x1a0000		1	1	0	1	1	0		Write CLK_Reg D[8..0]
CHP_ADDR + 0x1a8000		1	1	0	1	1	1		Write GAIN_Reg D[3..0]
BROADCAST or CHP_ADDR + 0x1c0000		1	1	1	0	0	0		Write ModeReg D[31..0]
BROADCAST or CHP_ADDR + 0x1c8000		1	1	1	0	0	1		Write ResetPuls D[31..0]
CHP_ADDR + 0x1d0000		1	1	1	0	1	0		AutoBaselEn[D24..16] ValueD [9..0]
CHP_ADDR + 0x1d0000		1	1	1	0	1	0		Read CorrVal ch# 3 2 1 [D29..0]
CHP_ADDR + 0x1d8000		1	1	1	0	1	1		Read CorrVal ch# 6 5 4 [D29..0]
CHP_ADDR + 0x1e0000		1	1	1	1	0	0		Read CorrVal ch# 9 8 7 [D29..0]
CHP_ADDR + 0x1f0000		1	1	1	1	1	0		ReadFirmwareVersion
CHP_ADDR + 0x1e800		1	1	1	1	0	1		Feature Register

ERROR FIFOs:

Data path from FIFO I to FIFO II :
The EnableBit decides if the data is a normal Pixeldata or if the data is type of error data or LastDac(temp) data.
We have eight Error FIFOs (two per FrontAltera) each responsible for four or five input channels !



Error Coding and Temp FIFO:



TBM_Trailer bits can be disabled for the error memories by setting bits [31..24] in the MODEREG (`ChipAddr+0x1c0000`)

Error Word:

Channel[31..26] :	**Channel (not every error has a corresponding channel information)**	
Code[25..21] :	**ErrorCode:**	**0x1f EventnumberError** **0x1e Trailer Error** **0x1d TimeOut** **0x1c NF MEM** **0x1b Dummy**
TTCrxEvent#[20..13]:	**8 LSBs of TTCrx Eventnumber**	
ErrorData[12..0]:	**ErrorType dependend !**	

When FIFOs are full they will be automatically exhausted !

ERROR FIFO BIT MEANINGS:

D31	D30	D29	D28	D27	D26	D25	D24	D23	D22	D21	D20	D19	D18	D17	D16	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
Channel#						EVnrERROR					8 LSBs ofTTCrx Eventnumber													Data word = Channel event #							
C5	C4	C3	C2	C1	C0	1	1	1	1	1	E7	E6	E5	E4	E3	E2	E1	E0	X	X	X	X	X	E7	E6	E5	E4	E3	E2	E1	E0
						TIMEOUT (word1)					8 LSBs ofTTCrx Eventnumber											Pedestal Correction Value = P#									
X	X	X	X	X	X	29	29	29	29	29	E7	E6	E5	E4	E3	E2	E1	E0	X	X	X	P9	P8	P7	P6	P5	P4	P3	P2	P1	P0
MSB Timeout Counter						TIMEOUT (word2)					8 LSBs ofTTCrx Eventnumber								LSB TimeoutCnt					Data word (Bit = channel with error)							
M5	M4	M3	M2	M1	M0	29	29	29	29	29	E7	E6	E5	E4	E3	E2	E1	E0	L1	L0	B[2]	B[1]	B[0]	X	X	X	CH5	CH4	CH3	CH2	CH1
																					BlockNr: (Block of 4 or 5 Input Channels)										
Channel#						TrailerERROR					8 LSBs ofTTCrx Eventnumber													8Bit TBM trailer							
C5	C4	C3	C2	C1	C0	1	1	1	1	0	E7	E6	E5	E4	E3	E2	E1	E0	X	R#	S[1]	S[0]	OVF	T7	T6	T5	T4	T3	T2	T1	T0
																				invalid # of ROCs	FSM Errors		data stream to long								
						NearlyFULL					8 LSBs ofTTCrx Eventnumber													Data word (Bit = channel FIFO1NF)							
X	X	X	X	X	X	1	1	1	0	0	E7	E6	E5	E4	E3	E2	E1	E0	X	X	X	X	X	L1A	NFII	X	NFe	NFd	NFc	NFb	NFa
Nearly FULL has only limited value of information ! NF error is only stored when FIFO II is emptying and this is not the time when NF occurs !!!!																						Event#FIFO NF				FIFO II NF					

Error words in normal data stream:

ROC numbers 26 to 31 are reserved for error type or special word recognition.

.) Error words are only transmitted when error information is found.
.) Dummy data (**27**)have to be included when there are too less data!
.) When combining two 32bit data streams, gaps are marked with **26**

ROC# **1 ..24** Gap **26** DummyData **27** NearlyFull **28** Timeout **29** TrailerErr **30** EvnrErr **31**

ErrorWord Bit Meaning:

	D31	D30	D29	D28	D27	D26	D25	D24	D23	D22	D21	D20	D19	D18	D17	D16	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
EVnrERROR 0x1f	C5	C4	C3	C2	C1	C0	1	1	1	1	1	X	X	X	X	X	X	X	X	X	X	X	X	X	E7	E6	E5	E4	E3	E2	E1	E0
TrailerERROR0x1e	C5	C4	C3	C2	C1	C0	1	1	1	1	0	X	X	X	X	X	X	X	X	X	R#	S[1]	S[0]	OVF	T7	T6	T5	T4	T3	T2	T1	T0
TIMEOUT 0x1d (word1)	X	X	X	X	X	X	29	29	29	29	29	29	X	X	X	X	X	X	X	X	X	X	P9	P8	P7	P6	P5	P4	P3	P2	P1	P0
TIMEOUT 0x1d (word2)	X	X	X	X	X	X	29	29	29	29	29	X	X	T7	T6	T5	T4	T3	T2	T1	T0	B[2]	B[1]	B[0]	X	X	X	CH5	CH4	CH3	CH2	CH1
NearlyFULL 0x1c	X	X	X	X	X	X	1	1	1	0	0	X	X	X	X	X	X	X	X	X	X	X	X	X	L1A	NFII	X	NFe	NFd	NFc	NFb	NFa
DUMMY 0x1b	X	X	X	X	X	X	1	1	0	1	1	X	X	X	X	X	X	X	X	X	X	X	X	X	P7	P6	P5	P4	P3	P2	P1	P0
GAP 0x1a	X	X	X	X	X	X	1	1	0	1	0	X	X	X	X	X	X	X	X	X	X	X	X	X	P7	P6	P5	P4	P3	P2	P1	P0

Channel#

Data word = Channel event #

8Bit TBM Trailer

invalid # of ROCs

FSM Errors

data stream to long

Pedestal Correction Value = P#

TimeOut Counter = T#

Data word (Bit = channel with error)

BlockNr: (Block of 4 or 5 Input Channels)

Data word (Bit = channel FIFO1NF)

Nearly FULL has only limited value of information ! NF error is only stored when FIFO II is emptying and this is not the time when NF occurs !!!!

TriggerFIFO NF

FIFO II NF

Private Eventnumber written with MODEREG[23..16]
(ChipAddr+0x1c0000)

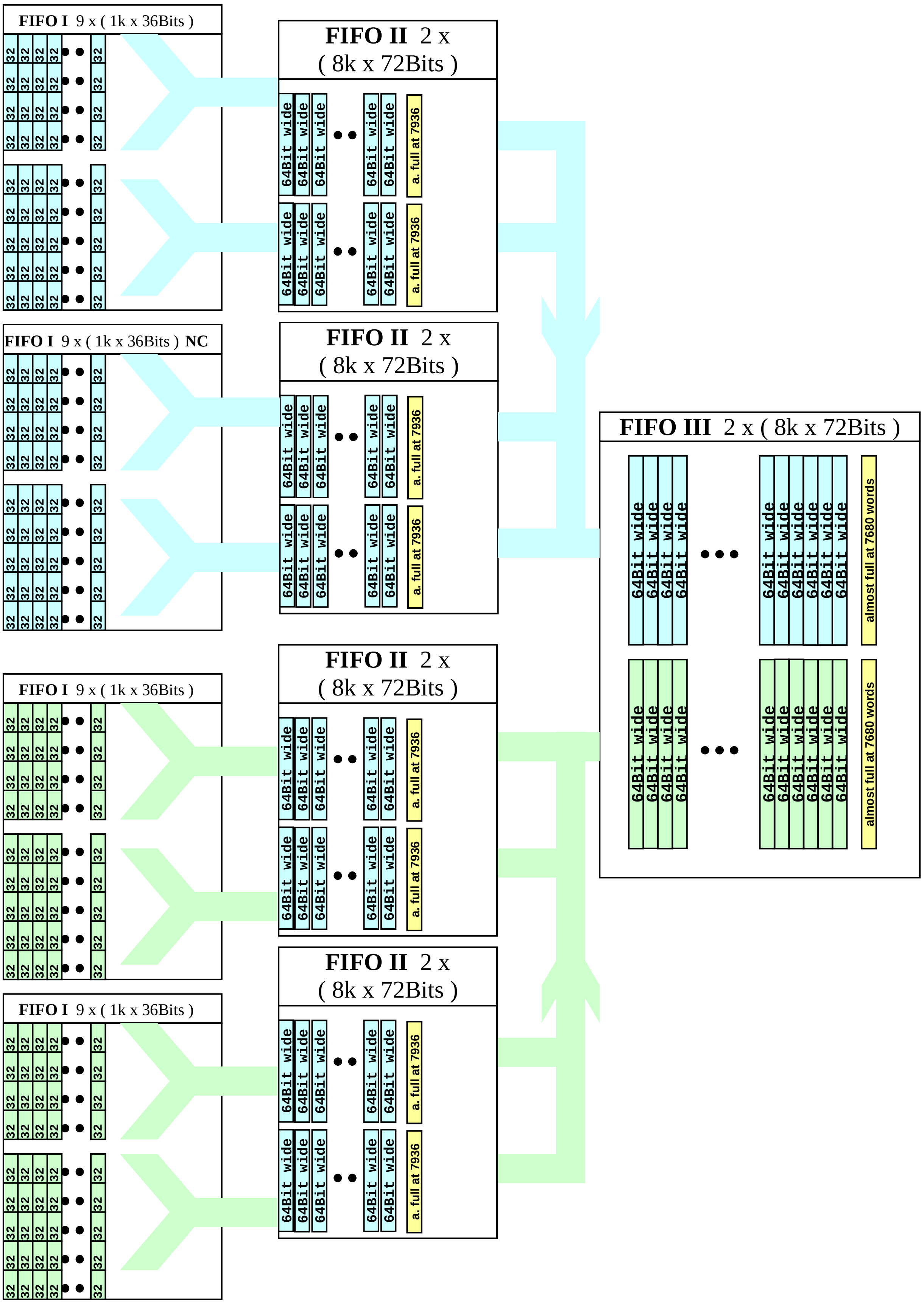
These data are transferred via the Center Chip to the S-link.

Only the EVnrERROR information generates a TTS event under following conditions:

two consecutive event blocks have to have at least 64 Out of Sync Errors to generate a TTS OUTofSYNC signal !!!!!

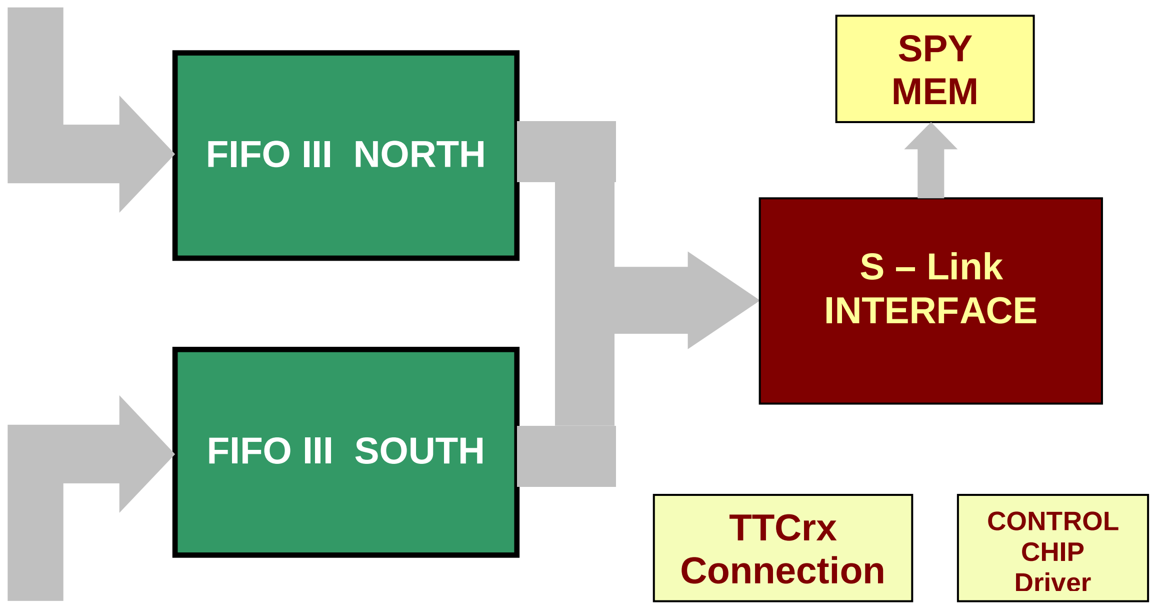
or consecutive event blocks have 64 Out of Sync Errors ! (**a healthy event in between will clear the Out of Sync Error Counter**)

FIFO DEPTHS:


FIFO I 9 x (1k x 36Bits)
FIFO I 9 x (1k x 36Bits) NC
FIFO I 9 x (1k x 36Bits)
FIFO I 9 x (1k x 36Bits)
32
FIFO II 2 x (8k x 72Bits)
64Bit wide
a. full at 7936
FIFO III 2 x (8k x 72Bits)
almost full at 7680 words

“ CENTER “ FPGA Data Path Block Diagram :

Data from NORTH and NORTHCENTER



Data from SOUTH and SOUTHTHCENTER

CENTER LOCAL BUS ADDRESSES:

	A20	A19	A18	A17	A16	A15	
CHP_ADDR + 0x20000	0	0	0	1	0	0	CLEAR_HIST (needs 12.8us)
CHP_ADDR + 0x28000	0	0	0	1	0	1	EnHistogramming D[0]
CHP_ADDR + 0x30000	0	0	0	1	1	0	TRIPPLE select UP D[2..0]
CHP_ADDR + 0x38000	0	0	0	1	1	1	TRIPPLE select DOWN D[2..0]
CHP_ADDR + 0x40000	0	0	1	0	0	0	ROC_READ_UP A[6..2]
CHP_ADDR + 0x48000	0	0	1	0	0	1	ROC_READ_DOWN A[6..2]
CHP_ADDR + 0x60000	0	0	1	1	0	0	READ TTS-FIFO D[31..0]
CHP_ADDR + 0xe0000	0	1	1	1	0	0	Write SourceID D[5..0]
CHP_ADDR + 0xf0000	0	1	1	1	1	0	WR TTS_Linet. D[31],[3..0]
CHP_ADDR + 0x140000	1	0	1	0	0	0	READ SPY-FIFOup D[31..0]
CHP_ADDR + 0x160000	1	0	1	1	0	0	READ SPY-FIFOdown D[31..0]
CHP_ADDR + 0x180000	1	1	0	0	0	0	WR Event#, D[8]=1 + D[7..0]
CHP_ADDR + 0x180000	1	1	0	0	0	0	StartTrp D[9]=1
CHP_ADDR + 0x188000	1	1	0	0	0	1	RD BUNCH Counter D[31..0]
CHP_ADDR + 0x190000	1	1	0	0	1	0	RD EVENT Counter D[31..0]
CHP_ADDR + 0x198000	1	1	0	0	1	1	RD L1A Counter D[31..0]
CHP_ADDR + 0x1a0000	1	1	0	1	0	0	W/R ControlReg D[31..0]
CHP_ADDR + 0x1a8000	1	1	0	1	0	1	W/R TestReg D[31..0]
BROADCAST or CHP_ADDR + 0x1c0000	1	1	1	0	0	0	WRITE ModeReg D[31..0]
BROADCAST or CHP_ADDR + 0x1c8000	1	1	1	0	0	1	WRITE ResetPuls D[31..0]
CHP_ADDR + 0x1f0000	1	1	1	1	1	0	ReadFirmwareVersion
CHP_ADDR + 0x1e0000	1	1	1	1	0	0	Feature Register D[3..0]

Control, ModeReg, and Feature Meaning:

```
CtrlReg[0]=0/1  ==> Disable/Enable Transparent Mode
CtrlReg[1]=0/1  ==> TranspGateStart by L1A / TranspGateStart by VME or EFT(OPTO Module)
CtrlReg[2]=0/1  ==> Disable/Enable DAC Data for Transparent Mode
CtrlReg[3]=0/1  ==> TTC Event# / VME Event#
CtrlReg[4]=0/1  ==> Disable/Enable L1A from TTCrx
CtrlReg[5]=0/1  ==> Disable/Enable EFT Signal from OPTO Module

CtrlReg[16]=0/1 ==> TTSReady dis/en
CtrlReg[17]=0/1 ==> TTSError dis/en
CtrlReg[18]=0/1 ==> OUTofSYN dis/en
CtrlReg[19]=0/1 ==> Disable/Enable Timeout counter overflow timeout
CtrlReg[20]=0/1 ==> Disable/Enable Broken Token mechanism for Type 1 ROCs

ModeReg[0]=1  ==> S-Link disable
ModeReg[1]=1  ==> Write Disable SPY MEM
ModeReg[2]=1  ==> URESET ( S-Link Reset )
ModeReg[3]=1  ==> Ignore Slink LFF

ModeReg[31]=1 ==> Select Slink Test Pattern
```

Feature Register address: `LAD_C + 0x1e0000`

```
FeatureReg[0]=1 ==> Enables the FED stuck-in-reset (a more thorough scrubbing of FIFOs)
FeatureReg[1]=1 ==> No longer active
FeatureReg[2]=1 ==> Turns on the Continuous Busy --> OOS feature
FeatureReg[3]=1 ==> Turns on the Continuous Warn --> OOS feature
                   (use this last feature only if bit 20 in the control register is set)
```

LADfront + 0x1e800

Fifo-1 hit Limit
Maximum number of hits allowed in a channel for each event. If this number is exceeded, the remaining hits are dumped and an error word is written.
LADfront + 0x1e800 bits [9...0].
If there is nothing written to the register, a limit of 192 is used.

BusyWhenBehind
Number of timeouts before asserting busy (min busy below, max = 100ms)
LADfront + 0x1e800 bits [23..16].
If there is nothing written to the register, a hard coded value of 200 will be used.

MinBusyWidth
Minimum Busy width for UP/Down counter is selected via
LADfront + 0x1e800 bits[25..24]

Value	Time (μs)
00	100
01	50
02	25
03	12.5

If there is nothing written to the register, a value of 0 (100μs) will be used.

Special Test Pattern DAC Mode:

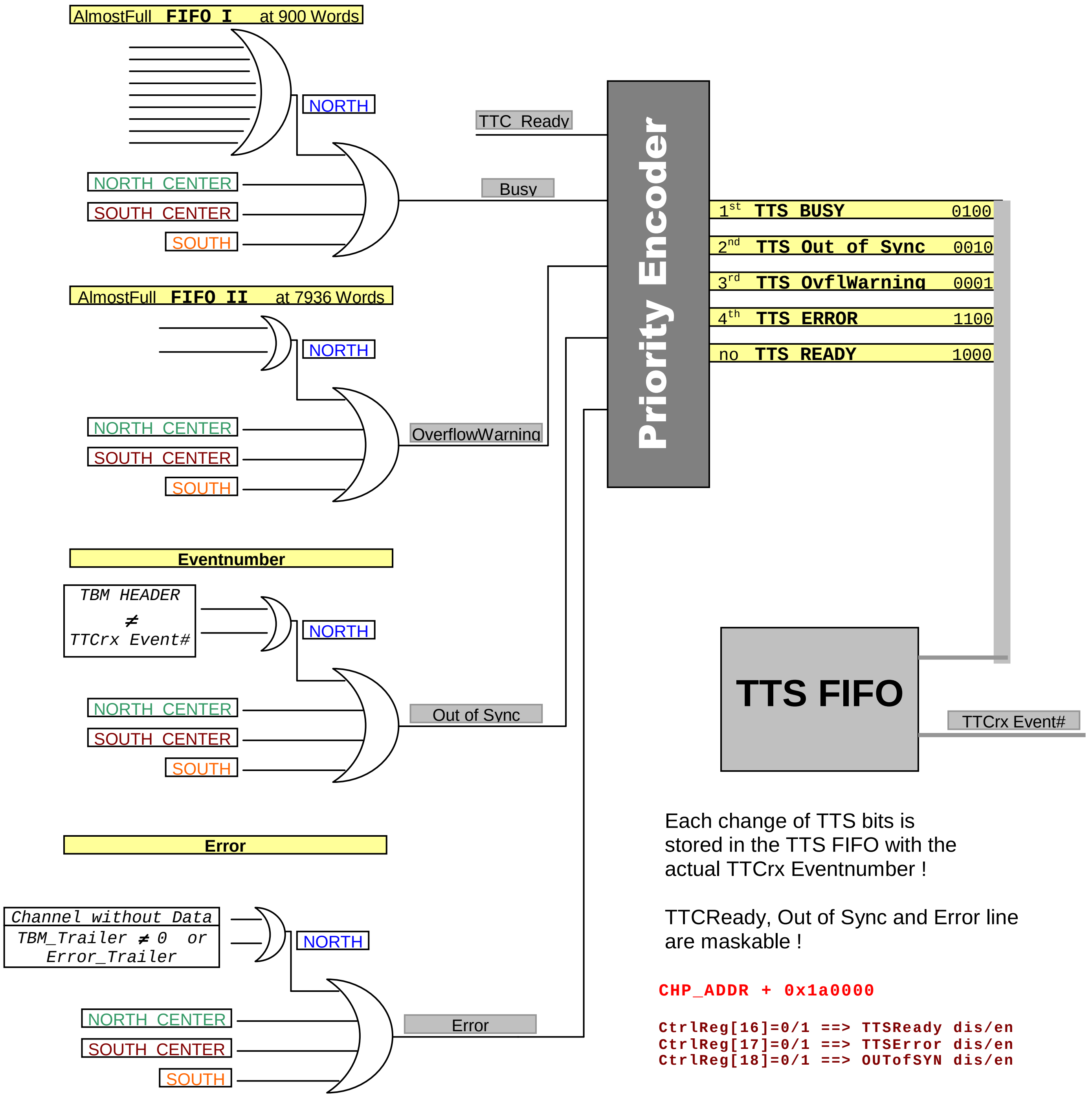
Note: A special mode should be used when random triggers are anticipated and the testDAC test Pattern is desired. In this case set:

`CtrlReg[2]=0 in the CENTER Control Register (Disables normal DAC data mode)`

And

`ModeReg[0]=1 in the NORTH Mode Register (Enables Special DAC data mode)`

TTS System:



Each change of TTS bits is stored in the TTS FIFO with the actual TTCrx Eventnumber !

TTCReady, Out of Sync and Error line are maskable !

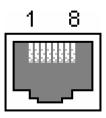
CHP_ADDR + 0x1a0000

```
CtrlReg[16]=0/1 ==> TTSReady dis/en
CtrlReg[17]=0/1 ==> TTSError dis/en
CtrlReg[18]=0/1 ==> OUTofSYN dis/en
```

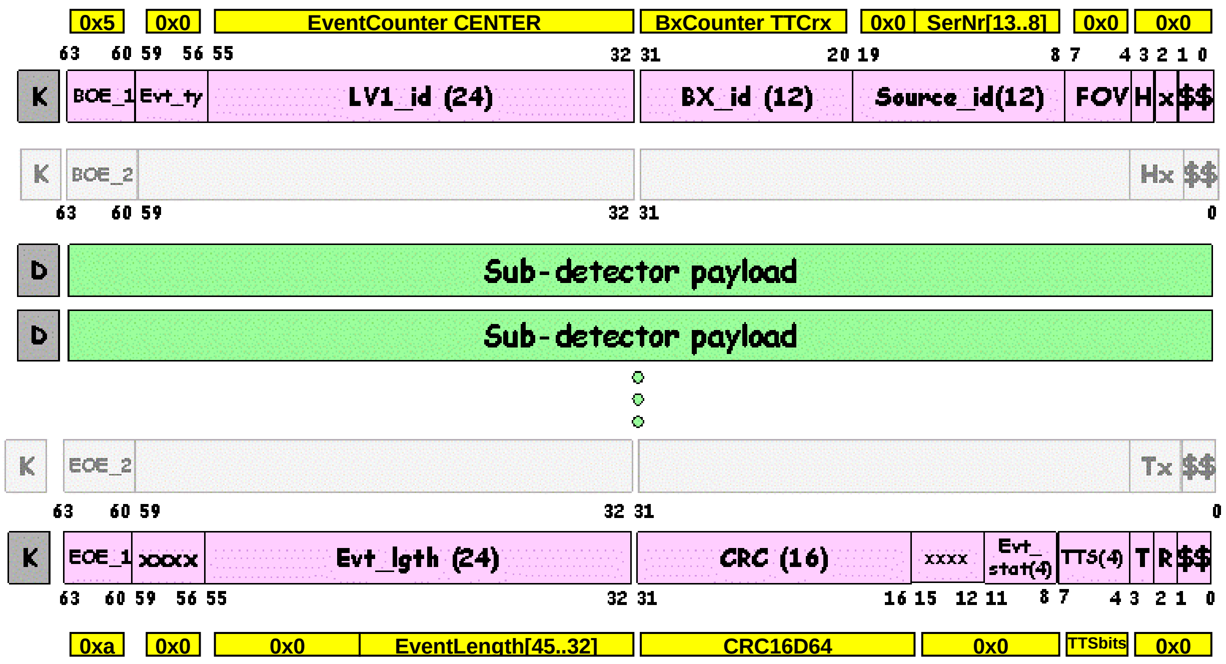
TTS FIFO read address: **LAD_C + 0x60000**

For connectivity tests a register is foreseen to set all TTS lines by VME command:

```
LAD_C+0xe0000        D[31]  enable TTS line test

                     D[3]   ready                pins 6,3
                     D[2]   busy                 pins 2,1
                     D[1]   out of synch         pins 8,7
                     D[0]   overflow warning     pins 4,5
```



Slink Header and Trailer words:



EventCounter CENTER

24bit event counter incremented by L1Accept .
__Reset possibilities:__ *LocalBus Reset or B-channel Command BRCST[5..0] = 0x2 (= reset EventCounter) or B-channel Command BRCST[5..0] = 0x14 (= FullReset)*

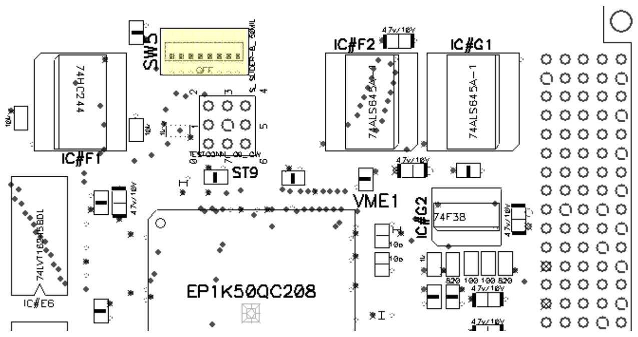
BxCounter TTCrx

BxCounter information from TTCrx chip.

SerNr[13..8]

FED Serial Number set by dipswitch SW5. Switch ON(OFF) Means bit = 0(1). Switch #1 (or closest to front of board) Corresponds to bit position 0 of bits 0-5 in the Serial Number Register.

This Serial Number can be read with: `FedBase+0xa00000` *(D[5..0]) and has to be written to the CENTER Chip register* `LAD_C + 0xe0000` *at module initialisation time.*



EventLength[45..32]

Our EvenLengthCounter is only 14 bit wide ! (two times 8192 words FIFO III depth)

Small events:

The next picture shows a Slink data dump for a small event (every third channel has one hit).
The gray marked fillwords originate from our ambition to avoid too short blocks in our data path.

A fillword can be identified by:
CH = 0 ; ROC = 0x1b ; DC = 0 ; PXL = 0 ; PH = private event type

On the way from FIFO I to FIFO II short blocks are filled up until they have at least six words.

Small Event Test:

```
[000]          50000008               87100700   BOE_1  LV1_ID BX_I Source_ID
[001]   4  7  9  56  af     1  7  9  56  af
[002]   0 1b  0   0   1     0 1b  0   0   1   N up      six  32 bit  words !
[003]   0 1b  0   0   1     0 1b  0   0   1
[004]   0 1b  0   0   1     7  7  9  56  b0
[005]   0 1b  0   0   1     0 1b  0   0   1   N down
[006]   0 1b  0   0   1     0 1b  0   0   1
[007]  13  7  9  56  ae    10  7  9  56  af
[008]   0 1b  0   0   1     0 1b  0   0   1   NC up              Fillwords
[009]   0 1b  0   0   1     0 1b  0   0   1                      (PrivEvTyp=1)
[010]   0 1b  0   0   1    16  7  9  56  ae
[011]   0 1b  0   0   1     0 1b  0   0   1   NC
[012]   0 1b  0   0   1     0 1b  0   0   1
[013]  22  7  9  56  ae    19  7  9  56  b1
[014]   0 1b  0   0   1     0 1b  0   0   1   SC up
[015]   0 1b  0   0   1     0 1b  0   0   1
[016]   0 1b  0   0   1    25  7  9  56  b0
[017]   0 1b  0   0   1     0 1b  0   0   1   SC down
[018]   0 1b  0   0   1     0 1b  0   0   1
[019]  31  7  9  56  b0    28  7  9  56  af
[020]   0 1b  0   0   1     0 1b  0   0   1   S up
[021]   0 1b  0   0   1     0 1b  0   0   1
[022]   0 1b  0   0   1    34  7  9  56  ae
[023]   0 1b  0   0   1     0 1b  0   0   1   S down
[024]   0 1b  0   0   1     0 1b  0   0   1
[025]          a000001a               0f510000   EOE_1  EvtLength   CRC16


received :      6 -> data_size = 2
```

Slink Test Pattern:

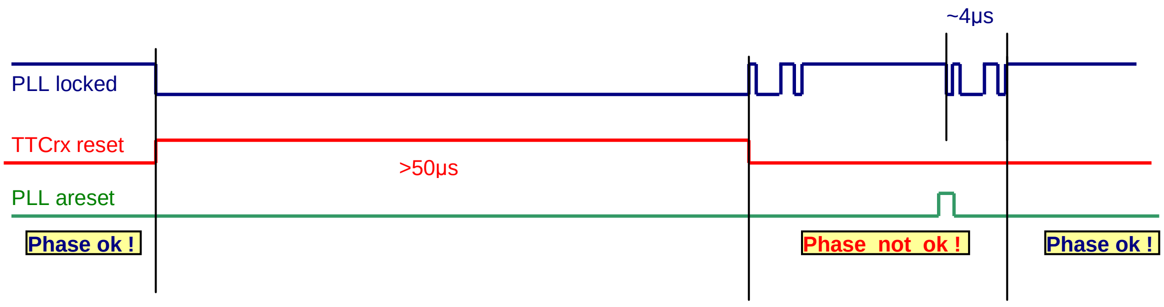
Setting ModeReg[31] (`LAD_C + 0x1c0000`) sends the following test pattern over the Slink cable:

```
0x 50## EV##  0000 0000
0x aaaa aaaa  aaaa aaaa
0x 5555 5555  5555 5555
0x 0000 0000  0000 0000
0x ffff ffff  ffff ffff
0x 0000 0000  0000 0000
0x ffff ffff  ffff ffff
0x 0f0f 0f0f  0f0f 0f0f
0x f0f0 f0f0  f0f0 f0f0
0x cccc cccc  cccc cccc
0x 3333 3333  3333 3333
0x ffff ffff  0000 0000
0x 0000 0000  ffff ffff
0x aaaa aaaa  aaaa aaaa
0x 5555 5555  5555 5555
0x 0000 0000  0000 0000
0x ffff ffff  ffff ffff
0x a000 0012  -CRC 0000
```

Eventnumber starts with one after LocalBus Reset !   Length of Event = 0x12

ALTERA PLL and reset TTCrx:

Phase relationships between PLL output clocks need to be maintained after a loss of lock condition !

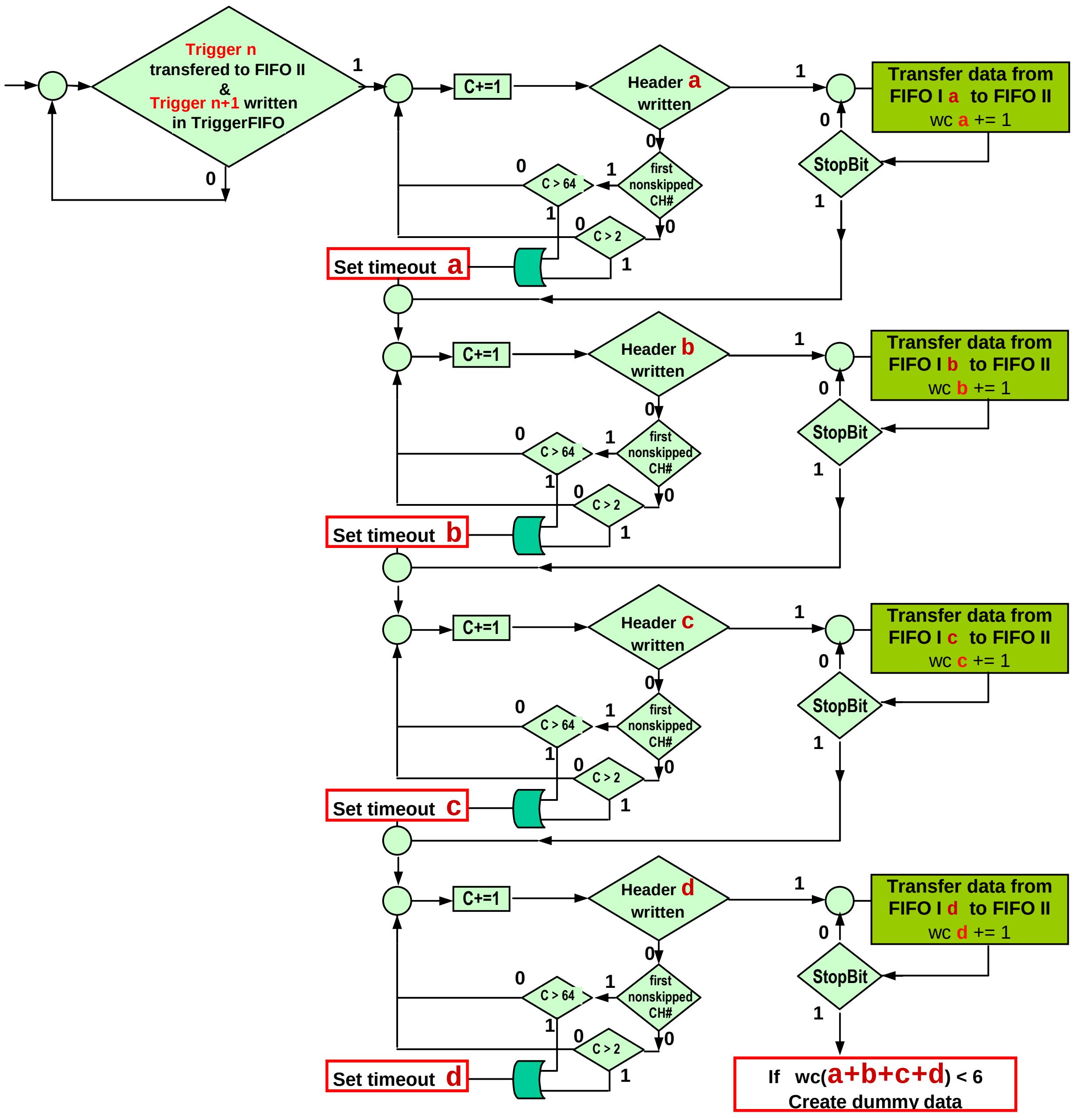


In V4 firmware a PLL areset is done with each LocalBusReset or individually with *CHIPaddr+0x1c8000 D[29]=1*

A LocalBusReset doesn't influence the PLL for the 80MHz clock (Slink) in the Center Chip, a PLL resync can only be done with the individual Reset *0x1c8000 D[29]=1*

A PLL areset is also recommended after power up !

Timeout mechanism:



Set timeout means that an error word with error code 0x6 is written in the corresponding error FIFO !

Firmware Versions:

Each FRONT ALTERA and the CENTER ALTERA has a read only version date register

BaseAddr + ChipAddr + 0x1f0000

The VME ALTERA has its version date register at *BaseAddr + 0xa0003c*

day								month								year								year							
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00

[hex] coded

(for example: 0x 16 08 14 07 means 22.8.2007)

Fifo Status:

It is possible to check the status of the fifos I, II & III by reading the Control Register.
LAD_C + 0x1a0000

Bit **Meaning**

```
[0] .. AlmostFull FIFO I   North ( all 9 Input Channels or – ed )
[1] .. NearlyFull FIFO II  North ( or of both FIFO II )
[2] .. AlmostFull FIFO I   NorthCenter
[3] .. NearlyFull FIFO II  NorthCenter
[4] .. AlmostFull FIFO I   SouthCenter
[5] .. NearlyFull FIFO II  SouthCenter
[6] .. AlmostFull FIFO I   South
[7] .. NearlyFull FIFO II  South
[8] .. AlmostFull FIFO III UP
[9] .. AlmostFull FIFO III DOWN
```

It is possible to check whether SpyFifo3 has an entire event in its buffer.
LAD_C + 0x190000

Bit **Meaning**

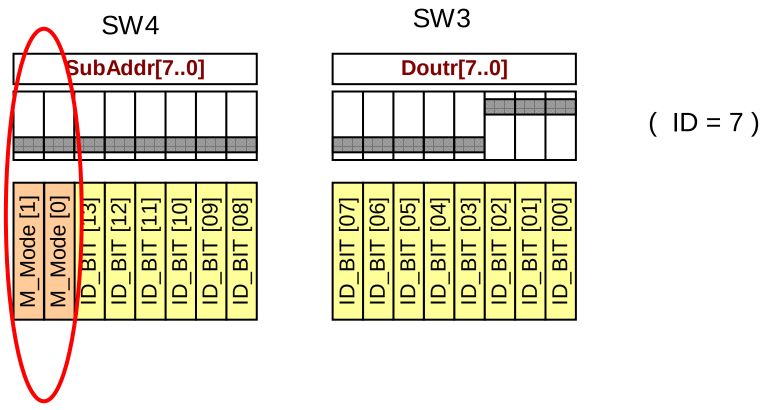
```
[31]   = 1 (Whole event in SpyFifo3)
[31]   = 0 (No or partial event in SpyFifo3)
```

Caution: To ensure a whole event, the SpyFifo3 loads one event *after* the SpyFifo3 disable is received. One should read the event *after* disable is set. Otherwise another event can try to fill the SpyFifo3 even when the SpyFifo3 disable is received.

TTCrxSwitches:

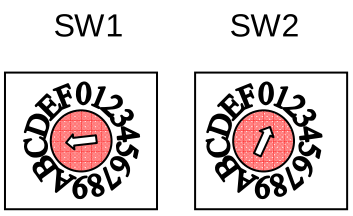
It is possible to set TTCrx pins Dout[7..0] high or low using SW3 and TTCrx pins SubAddr[7..0] high or low using SW4.
These switches can therefore be used to set an ID and Master Mode for the TTCrx chip.

The bits for the Master Mode should be let in the **lower position**. This means MasterMode[1..0] = 0 (See the TTCrx manual for more Information.)



Base Address Switches:

There are 2 rotary switches located below the VME FPGA. These switches provide the VME Base Address (A31-A24) for the board. For example, the Base Address used during the testing of the boards was typically set to 0x1c000000. This looks like



Other Switches/Jumpers:

L1_CK_RDY_EN: Switch to “open”. Signal is used in conjunction with a special test board.
(Open in this context means pin 12 of IC # H3 should be pulled “high”.)

VME_64X: Not Implemented. Can be left Closed (Jumpered).

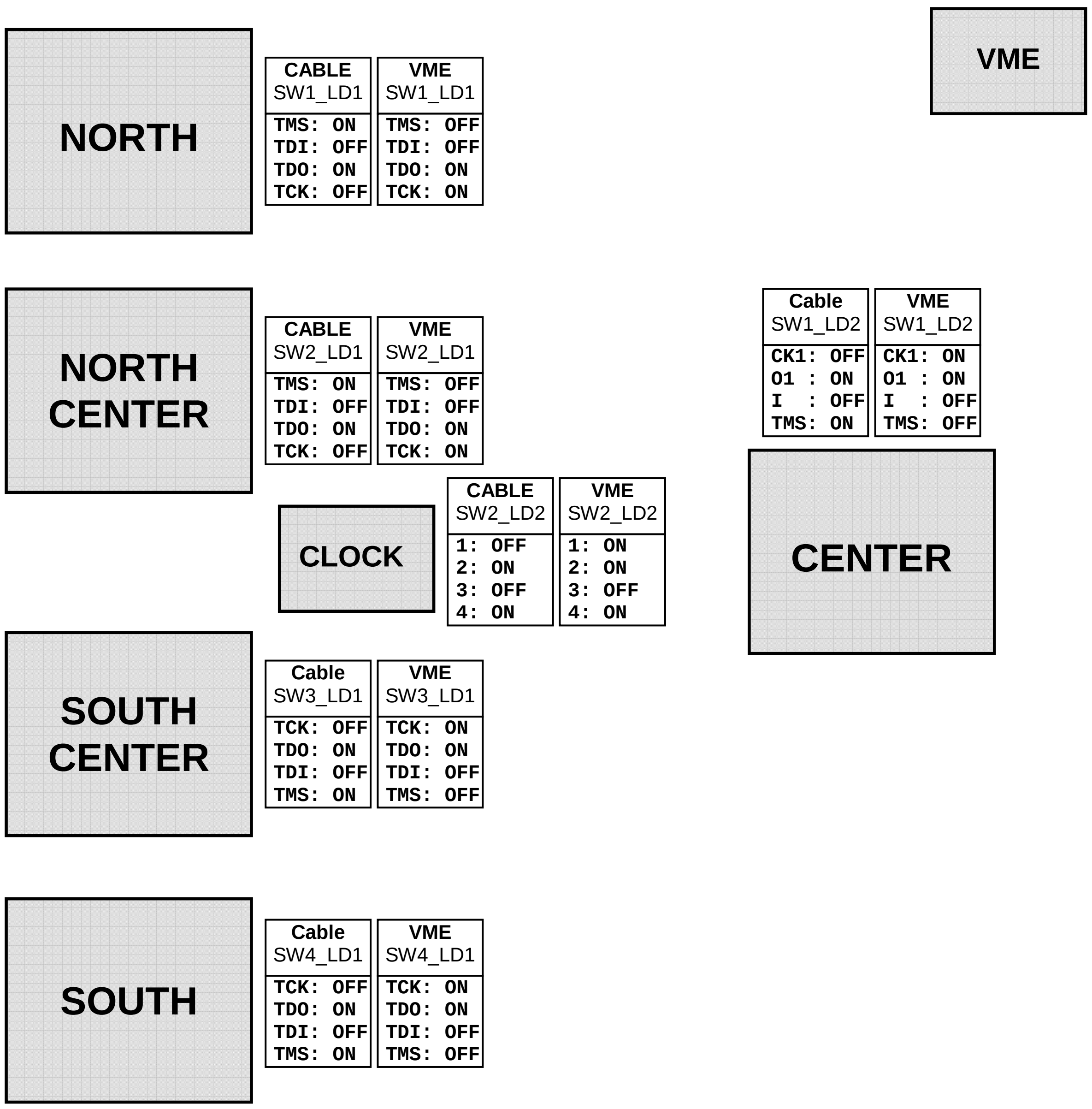
Firmware Programming:

The FRONT ALTERA FPGA's (N, NC, SC & S) and the CENTER ALTERA FPGA's (CENTER & CLOCK) are re-programmable either using a JTAG programmer, like the USB Blaster, with the proper cable and connector, or through the internal VME interface (mentioned earlier).

One just needs to set switches to indicate which mode (Cable or VME) is to be used.

Note 1: The physical switches for the Front FPGA's are arranged so that the positions look the same without having to refer to the switch label (use caution!).

Note 2: Prior to V4, switches IC4_OE and IC9_OE need to be switched ON(OFF) for VME(CABLE) programming .



NORTH

CABLE SW1_LD1	**VME** SW1_LD1
TMS: ON	TMS: OFF
TDI: OFF	TDI: OFF
TDO: ON	TDO: ON
TCK: OFF	TCK: ON

NORTH CENTER

CABLE SW2_LD1	**VME** SW2_LD1
TMS: ON	TMS: OFF
TDI: OFF	TDI: OFF
TDO: ON	TDO: ON
TCK: OFF	TCK: ON

CENTER

Cable SW1_LD2	**VME** SW1_LD2
CK1: OFF	CK1: ON
O1 : ON	O1 : ON
I : OFF	I : OFF
TMS: ON	TMS: OFF

CLOCK

CABLE SW2_LD2	**VME** SW2_LD2
1: OFF	1: ON
2: ON	2: ON
3: OFF	3: OFF
4: ON	4: ON

SOUTH CENTER

Cable SW3_LD1	**VME** SW3_LD1
TCK: OFF	TCK: ON
TDO: ON	TDO: ON
TDI: OFF	TDI: OFF
TMS: ON	TMS: OFF

SOUTH

Cable SW4_LD1	**VME** SW4_LD1
TCK: OFF	TCK: ON
TDO: ON	TDO: ON
TDI: OFF	TDI: OFF
TMS: ON	TMS: OFF

NOTES: