Declarations

Ports:

```
BCRES_DAQ               : std_logic
BX                      : std_logic_vector(11 DOWNTO 0)
CLK80_DAQ               : std_logic
CLK_DAQ                 : std_logic
CLK_FBIN_DAQ            : std_logic
EN_DAQ                  : std_logic
FDL                     : std_logic_vector(27 DOWNTO 0)
GMTA                    : std_logic_vector(27 DOWNTO 0)
GMTB                    : std_logic_vector(27 DOWNTO 0)
GPS                     : std_logic_vector(47 DOWNTO 0)
L1A_DAQ                 : std_logic
L1RESET_DAQ             : std_logic
PSB0                    : std_logic_vector(27 DOWNTO 0)
PSB1                    : std_logic_vector(27 DOWNTO 0)
PSB2                    : std_logic_vector(27 DOWNTO 0)
PSB3                    : std_logic_vector(27 DOWNTO 0)
PSB4                    : std_logic_vector(27 DOWNTO 0)
PSB5                    : std_logic_vector(27 DOWNTO 0)
PSB6                    : std_logic_vector(27 DOWNTO 0)
RDRQST                  : std_logic
RESET_DAQ               : std_logic
RESET_DCM_DAQ           : std_logic
RXCLK                   : std_logic_vector(11 DOWNTO 0)
S64_LRL                 : std_logic_vector(3 DOWNTO 0)
S64_NDOWN               : std_logic
S64_NLFF                : std_logic
S64_RESV                : std_logic_vector(2 DOWNTO 0)
STROB                   : std_logic_vector(2 DOWNTO 0)
TCS                     : std_logic_vector(27 DOWNTO 0)
TIM                     : std_logic_vector(27 DOWNTO 0)
VA_I                    : std_logic_vector(19 DOWNTO 1)
WR_DAQ                  : std_logic
CLK_FBOUT_DAQ           : std_logic
DAQ2EVM                 : std_logic_vector(7 DOWNTO 0)
DAQ_TEST                : std_logic_vector(15 DOWNTO 0)
ERR_LED                 : std_logic
GTFE_STATUS             : std_logic_vector(3 DOWNTO 0)
NBERR_DAQ               : std_logic
NDTACK_DAQ              : std_logic
NIRQ_FR_DAQ             : std_logic
NLOCKED_DAQ             : std_logic
OSC_D                   : std_logic_vector(3 DOWNTO 1)
S64_D                   : std_logic_vector(63 DOWNTO 0)
S64_DW                  : std_logic_vector(1 DOWNTO 0)
S64_NCTRL               : std_logic
S64_NLFF_LED            : std_logic
S64_NRESET              : std_logic
S64_NTEST               : std_logic
S64_NWEN                : std_logic
UNUSED_OUT              : std_logic
VD_I                    : std_logic_vector(15 DOWNTO 0)
```

Pre User:

Diagram Signals:

```
SIGNAL BCRESx                   : std_logic
SIGNAL BCRes_int                : std_logic
SIGNAL BCRes_vme                : std_logic
SIGNAL CLK0                     : std_ulogic := '0'
SIGNAL CLK2X                    : std_ulogic := '0'
SIGNAL CLKIN                    : std_logic
SIGNAL Hardres_int              : std_logic
```

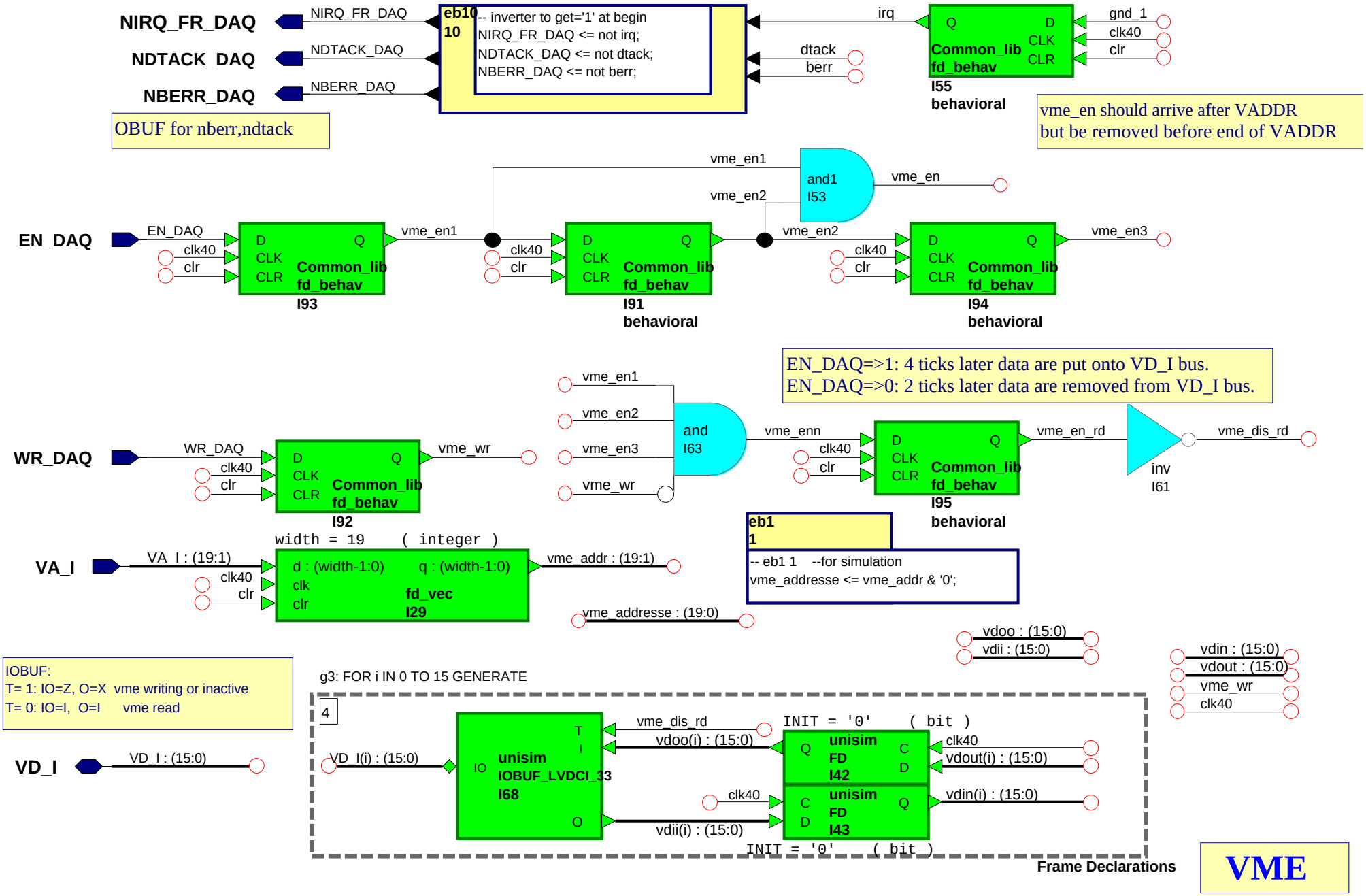
Package List

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_arith.all;
USE ieee.STD_LOGIC_UNSIGNED.all;
USE IEEE.VITAL_Timing.all;
LIBRARY rop_types;
USE rop_types.rop.all;
```

Version 102A

Register DAQ_RECORD_LENGTH

Institut für Hochenergiephysik, Vienna		Project:	rop_chip
Title:	DAQ chip for GTFE board		
Path:	rop_chip_lib/rop_daq/struct		
Edited:	by taurok on 06 Jul 2010		

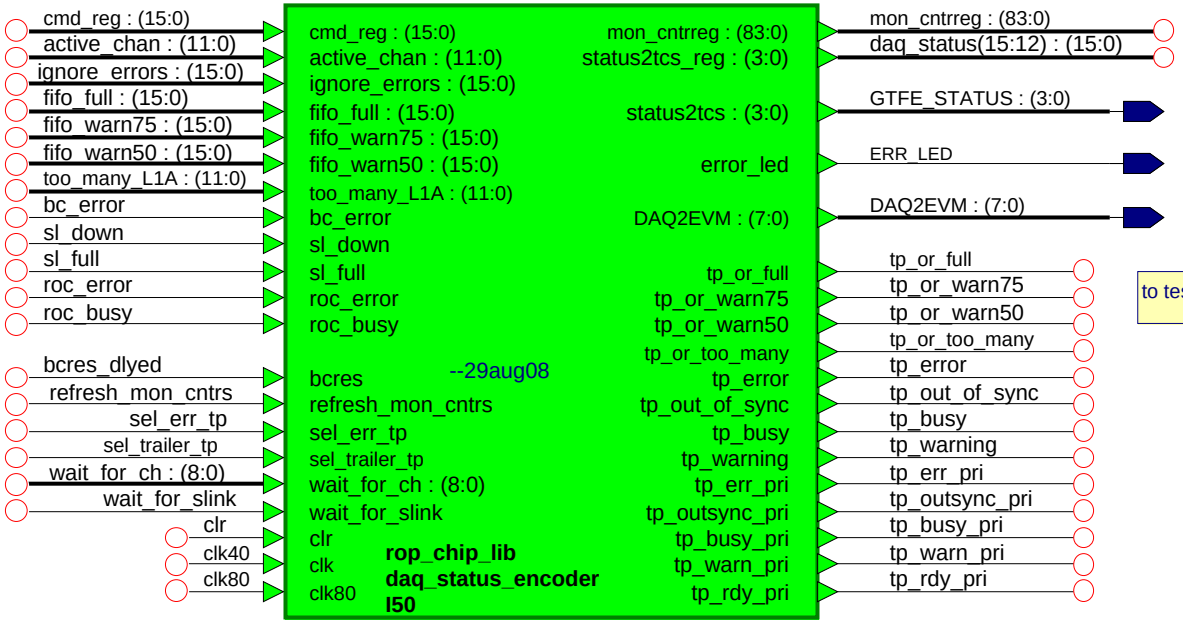

NIRQ_FR_DAQ
NDTACK_DAQ
NBERR_DAQ
NIRQ_FR_DAQ
NDTACK_DAQ
NBERR_DAQ
eb10
10
-- inverter to get='1' at begin
NIRQ_FR_DAQ <= not irq;
NDTACK_DAQ <= not dtack;
NBERR_DAQ <= not berr;
irq
dtack
berr
Common_lib
fd_behav
I55
behavioral
gnd_1
clk40
clr
OBUF for nberr,ndtack
vme_en should arrive after VADDR
but be removed before end of VADDR
EN_DAQ
vme_en1
vme_en2
and1
I53
vme_en
vme_en3
I93
I91
I94
EN_DAQ=>1: 4 ticks later data are put onto VD_I bus.
EN_DAQ=>0: 2 ticks later data are removed from VD_I bus.
WR_DAQ
vme_wr
and
I63
vme_enn
vme_en_rd
vme_dis_rd
inv
I61
I92
I95
width = 19 (integer)
VA_I
VA_I : (19:1)
d : (width-1:0)
q : (width-1:0)
clk
fd_vec
I29
vme_addr : (19:1)
eb1
1
-- eb1 1 --for simulation
vme_addresse <= vme_addr & '0';
vme_addresse : (19:0)
vdoo : (15:0)
vdii : (15:0)
vdin : (15:0)
vdout : (15:0)
vme_wr
clk40
IOBUF:
T= 1: IO=Z, O=X vme writing or inactive
T= 0: IO=I, O=I vme read
g3: FOR i IN 0 TO 15 GENERATE
4
VD_I
VD_I : (15:0)
VD_I(i) : (15:0)
IO
unisim
IOBUF_LVDCI_33
I68
T
I
O
vme_dis_rd
vdoo(i) : (15:0)
vdii(i) : (15:0)
INIT = '0' (bit)
unisim
FD
I42
I43
Q
C
D
clk40
vdout(i) : (15:0)
vdin(i) : (15:0)
INIT = '0' (bit)
Frame Declarations
VME

page 1(2)

VME registers

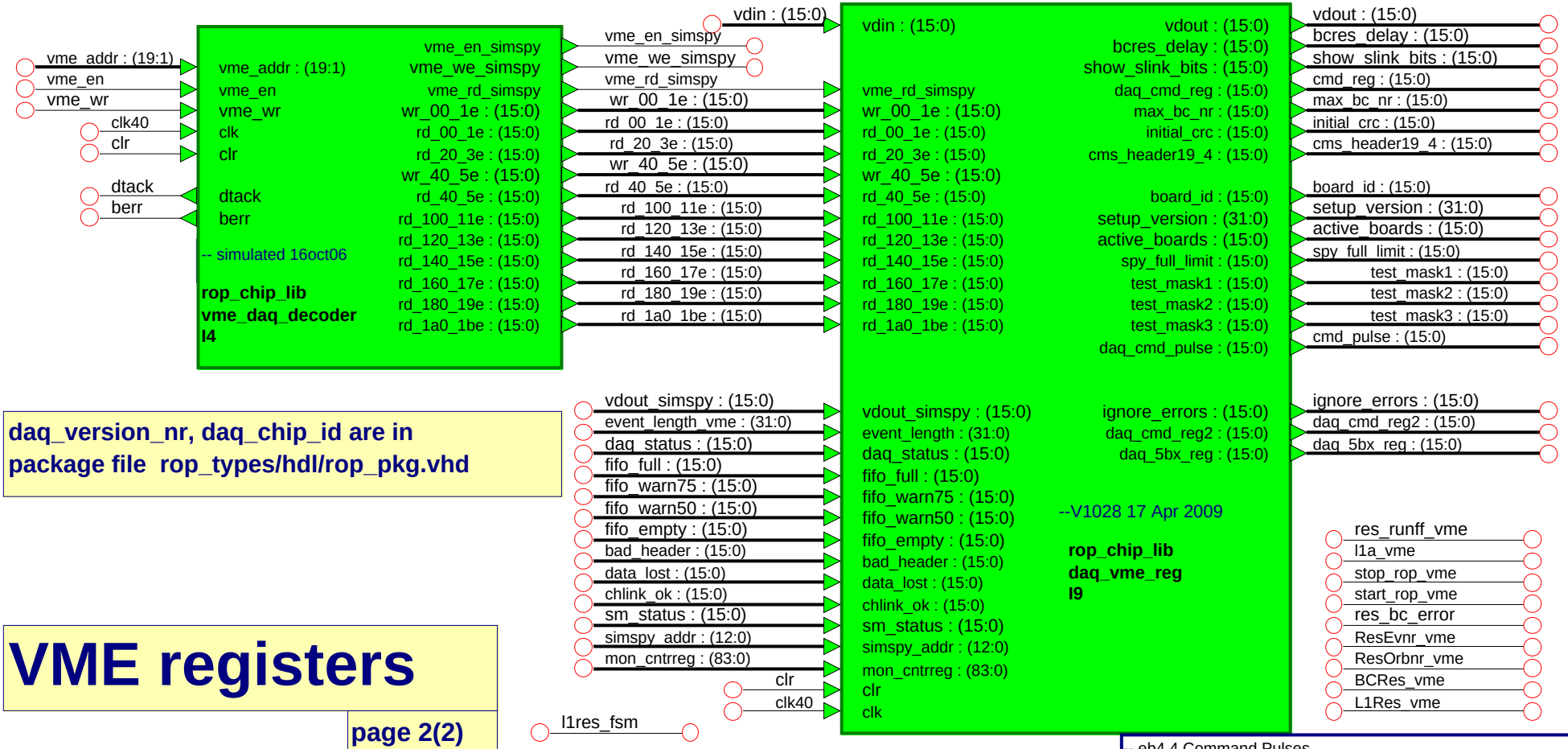
Setup registers
BB10 0000 BCRES_DELAY w/r
BB10 0002 SHOW_SLINK_BITS w/r // mask register to show S-Link64 bits
BB10 0004 DAQ_CMD_REG w/r // sim_mode, enable bits,
// set SLINK mode etc.
BB10 0006 MAX_BC_NUMBER w/r //=orbit length -1
BB10 0008 INITIAL_CRC w/r // bits 15_0 // initial CRC value
BB10 000A CMS_HEADER19_4 w/r //CMS HeaderWord1(19:4)
bits15:4:= GT-identifier in DAQ software: 813 =32D(hex)
bits 3:0:= FORMAT_VERSION currently =0000 ? default value = 32D0
BB10 000C EVENT_LENGTH_15_0 w/r // bits 15_0 : 24 bit length
BB10 000E EVENT_LENGTH_31_16 w/r // bits 31_16 : 24 bit length
// optional: Hardware Adder using ACTIVE_BOARDS
BB10 0010 BOARD_ID w/r //Board_ID(15:8) + length in bx(3 or 5)(7:0)
BB10 0012 SETUP_VERSION_15_0 w/r // bits 15_0 //GFTE HeaderWord1
BB10 0014 SETUP_VERSION_31_16 w/r // bits 31_16 //GFTE HeaderWord1
BB10 0016 ACTIVE_BOARDS w/r //GFTE HeaderWord2
BB10 0018 SPY_FULL_LIMIT w/r // bits 15_0 // Spy control word
BB10 001A TEST_MASK1 w/r // bits 15_0
BB10 001C TEST_MASK2 w/r // bits 15_0
BB10 001E TEST_MASK3 w/r // bits 15_0
Write registers
BB10 0040 CMD_PULSE w/ //
BB10 0042 IGNORE_ERR_FOR_TCS w/r // bits 15_0
BB10 0044 -5E free

Status words read only
BB10 0020 DAQ_CHIP_IDL -/r // bits 15_0
BB10 0022 DAQ_CHIP_IDH -/r // bits 31_16
BB10 0024 DAQ_VERSION_NR -/r // bits 15_0 Firmware Version nr.
BB10 0026 DAQ_STATUS -/r // General status with SLINK
BB10 0028 CHAN_FIFOS_FULL -/r // FULL flags of all channels
BB10 002A CHAN_FIFOS_WARN -/r // WARN flags of all channels
BB10 002C CHAN_FIFOS_EMPTY -/r // EMPTY flags of all channels
BB10 002E CHAN_IS_TOO_LATE -/r // ..of all channels
BB10 0030 CHAN_LINK_BAD_CODE -/r // Bad Header Code flags
BB10 0032 CHAN_LINK_DATA_LOST -/r // ..of all channels
BB10 0034 CHAN_LINK_DOWN -/r // ..of all channels
BB10 0036 STATE_MACHINE_STATUS -/r //actual status of ROP state machine
BB10 0038 xxxxx -/r //
BB10 003A xxxxx -/r //
BB10 003C xxxxx -/r //
BB10 003E xxxxx -/r //



to testpoints

GTFE status



daq_version_nr, daq_chip_id are in package file rop_types/hdl/rop_pkg.vhd

VME registers

page 2(2)

eb8 8

```
-- eb8 8 General status with SLINK
-- daq_status(15:12) <= status to TCS
daq_status(11) <= run_rop;
daq_status(10) <= bc_error;  -- <--gtfe_tim_run_ctrl
daq_status(9)  <= spy_full;   -- <-- gtfe_simspy_ctrl
daq_status(8)  <= sl64_full;     -- SLINK is full
daq_status(7)  <= sl64_down;  -- SLINK is down
daq_status(6 downto 4) <= sl_resv;  -- from SLINK
daq_status(3 downto 0) <= sl_lrl;     -- from SLINK
```

res_link_without_resync

eb2 2

```
-- eb2 2
--          CMD_REG
event_type           <= cmd_reg(15 downto 12);
-- one_event_mode <= cmd_reg(11);  -- to gtfe_simspy_ctrl
-- length_5bx          <= cmd_reg(10); -- not used since V0027
ignore_slink_down  <= cmd_reg(9);
ignore_slink_full    <= cmd_reg(8);
res_link_without_resync <= cmd_reg(7); -- to gtfe_simspy_ctrl
-- spy_every_tick     <= cmd_reg(6); -- to gtfe_simspy_ctrl
sim_mode            <= cmd_reg(5);
test_mode           <= cmd_reg(4);
debug_chanlinks    <= cmd_reg(3);
en_robus            <= cmd_reg(2);
--  cmd_reg(1:0) = status for TCS(daq_status_encoder)
```

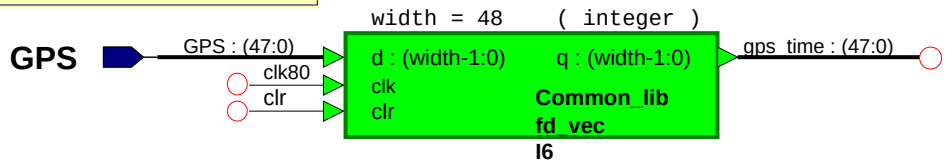
eb4 4

```
-- eb4 4 Command Pulses
-- to gtfe_time_run_control
res_runff_vme  <= cmd_pulse(15);
refresh_mon_cntrs <=cmd_pulse(13);
l1a_vme        <= cmd_pulse(11);
res_roc_vme   <= cmd_pulse(10);
res_slink_vme <= cmd_pulse(9);
stop_spying    <= cmd_pulse(8);   -- to gtfe_simspy_ctrl
start_spying   <= cmd_pulse(7);   -- to gtfe_simspy_ctrl
stop_rop_vme  <= cmd_pulse(6);  -- to gtfe_tim_run_control
start_rop_vme  <= cmd_pulse(5);  --    ----"----
res_bc_error   <= cmd_pulse(4);  --    -----"----
ResEvnr_vme   <= cmd_pulse(3); --    -----"----
ResOrbnr_vme <= cmd_pulse(2); --    -----"----
BCRes_vme    <= cmd_pulse(1);  --    -----"----
L1Res_vme     <= cmd_pulse(0);  --    -----"----
```

Version 102A: 'event_length' for Trailer word corrected
Version 1029: simspy_en applied correctly to simspy meory
Version 1028: vme problem corrected but delivers bad event record. ... do not use it
event_length register shows result of autom.event_length calculation
Version 0027: register 'DAQ_5BX_reg' + automatic event length calculation added.
gtfe_simspy_ctrl : error in one-event- mode corrected
Version 0026: 'vme_en2' added to make 'vme_dis_rd'
'l1a_dlyed' with clk40 as in EVM chip V0018
Version 0025: Monitoring counters: wait_for_chan(i) ...corrected
Version 0024: Monitoring counters; gmt_chan_recvr+chan_recvr: empty cntr corrected
roc: additional outputs for mon_cntrs
Version 0023: gmt_chan_recvr+chan_recvr : active_chan included for flag outputs: chnlink_ok, bad_header
Version 0022: gmt_chan_recvr+chan_recvr : active_chan included for flag outputs: empty, full....
daq_vme_reg_c000c: ignore_error..power-up value=004C
Version 0021: daq_evnr_check+gmt_chan_recvr+chan_recvr : 'too_many_l1a' in chain -->busy in daq_status_decoder
roc: wait until data of one active channel have arrived in fifo
Version 0020: roc: l1res at begin or after end of event, gtfe_simspy_ctrl: sim to sim_sl_ctrl(limitp)
wait 1.6us after l1res(>800n=slink recovery time); Channel FIFOs: warn50,75%
daq_status_decoder, vme_regs simplified
Version 001F: gmt_chan_recvr ... error in empty counter corrected
ROC: res_evnr returns to IDLE to repeat 'clr_fifo' , gmt: full check before each bx-transmission
rd_fifo(8) in wait states only
Version 001E: ROC for GMT: waiting for not empty removed, return with l1res, res_roc
wr_fifo_tp(i) to Testpoint3 (tp2)
Version 001D: ROC for GMT changed at begin of gmt-event, end of event simplified(without l1res_rqst)
Version 001C: New fifo16to64_4k with corrected empty error, ROC: wait 1023 for GMT,
cmd_reg(7)= 0 allows L1Reset to reset S-Link, TESTMASK2 &3: status bits to EVM inserted.
Version 001B: Reset Slink by VME only, not by L1Reset
Version 001A: ###### run_rop = '1' ###### l1a_latency register removed

```
SIGNAL hardres_int              : std_logic
SIGNAL L1A_int                  : std_logic
SIGNAL L1Ax                     : std_logic
SIGNAL L1RESETx                 : std_logic
SIGNAL L1Res_int                : std_logic
SIGNAL L1Res_vme                : std_logic
SIGNAL ResEvnr_vme              : std_logic
SIGNAL ResOrbnr_vme             : std_logic
SIGNAL Res_Evnr_int             : std_logic
SIGNAL Res_Orbitnr_int          : std_logic
SIGNAL active_boards            : std_logic_vector(15 DOWNTO 0)
SIGNAL active_chan              : std_logic_vector(11 DOWNTO 0)
SIGNAL apply_err                : std_logic
SIGNAL apply_gtfe               : std_logic_vector( 1 DOWNTO 0 )
SIGNAL apply_header             : std_logic
SIGNAL apply_trailer            : std_logic
SIGNAL bad_header               : std_logic_VECTOR(15 DOWNTO 0)
SIGNAL bc_error                 : std_logic
SIGNAL bcnr                     : std_logic_vector(11 DOWNTO 0)
SIGNAL bcres_delay              : std_logic_vector(15 DOWNTO 0)
SIGNAL bcres_dlyed              : std_logic
SIGNAL berr                     : std_logic
SIGNAL board_id                 : std_logic_vector(15 DOWNTO 0)
SIGNAL chan_link                : vec28_array(11 DOWNTO 0)
SIGNAL chlink_ok                : std_logic_VECTOR(15 DOWNTO 0)
SIGNAL clk40                    : std_logic
SIGNAL clk80                    : std_logic
SIGNAL clk_locked               : std_logic
SIGNAL clk_nlocked              : std_logic
SIGNAL clr                      : std_logic
SIGNAL clr_fifo                 : std_logic
SIGNAL cmd_pulse                : std_logic_vector(15 DOWNTO 0)
SIGNAL cmd_reg                  : std_logic_vector(15 DOWNTO 0)
SIGNAL cms_header19_4           : std_logic_vector(15 DOWNTO 0)
SIGNAL d_mux                    : std_logic_vector(63 DOWNTO 0)
SIGNAL daq_5bx_reg              : std_logic_vector(15 DOWNTO 0)
SIGNAL daq_cmd_reg2             : std_logic_vector(15 DOWNTO 0)
SIGNAL daq_status               : std_logic_vector(15 DOWNTO 0) -- to status register (vme)
SIGNAL data_lost                : std_logic_VECTOR(15 DOWNTO 0)
SIGNAL debug_chanlinks          : std_logic
SIGNAL decr_ch_cntr             : std_logic_vector(11 DOWNTO 0)
SIGNAL decr_ref_cntr            : std_logic
SIGNAL do_crc_trailer           : std_logic
SIGNAL dtack                    : std_logic
SIGNAL en_crc                   : std_logic
SIGNAL en_crc_trailer           : std_logic
-- vme gnerated signals
SIGNAL en_robus                 : std_logic -- enable signals from ROBUS
SIGNAL end_of_event             : std_logic_VECTOR(11 DOWNTO 0)
SIGNAL ev_in_fifo               : vec32_array(11 DOWNTO 0)
SIGNAL event_length             : std_logic_vector(23 DOWNTO 0)
SIGNAL event_length_vme         : std_logic_vector(31 DOWNTO 0)
SIGNAL event_status             : std_logic_vector(7 DOWNTO 0)
SIGNAL event_type               : std_logic_vector(3 DOWNTO 0)
SIGNAL evnr                     : std_logic_vector(31 DOWNTO 0)
SIGNAL evnr_ref_ge_ch           : std_logic_vector(11 DOWNTO 0)
SIGNAL evnr_ref_lt_ch           : std_logic_vector(11 DOWNTO 0)
SIGNAL fifo                     : vec64_array(11 DOWNTO 0)
SIGNAL fifo_empty               : std_logic_vector(15 DOWNTO 0)
SIGNAL fifo_full                : std_logic_VECTOR(15 DOWNTO 0)
SIGNAL fifo_warn50              : std_logic_VECTOR(15 DOWNTO 0)
SIGNAL fifo_warn75              : std_logic_VECTOR(15 DOWNTO 0)
SIGNAL gnd_1                    : std_logic
SIGNAL gps_time                 : std_logic_vector(47 DOWNTO 0)
SIGNAL ground                   : std_logic
SIGNAL gsr                      : STD_ULOGIC
```

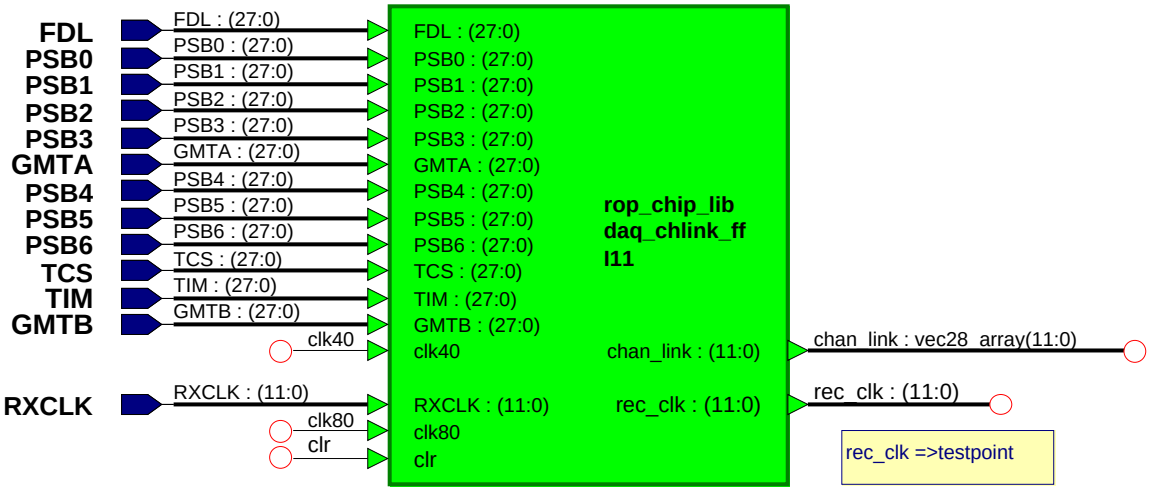
ROP_DAQ: GPS(55:0) = inputs



GPS not used.
V1013: changed to 48 bit vector

Mapping in "evnr_check" module:

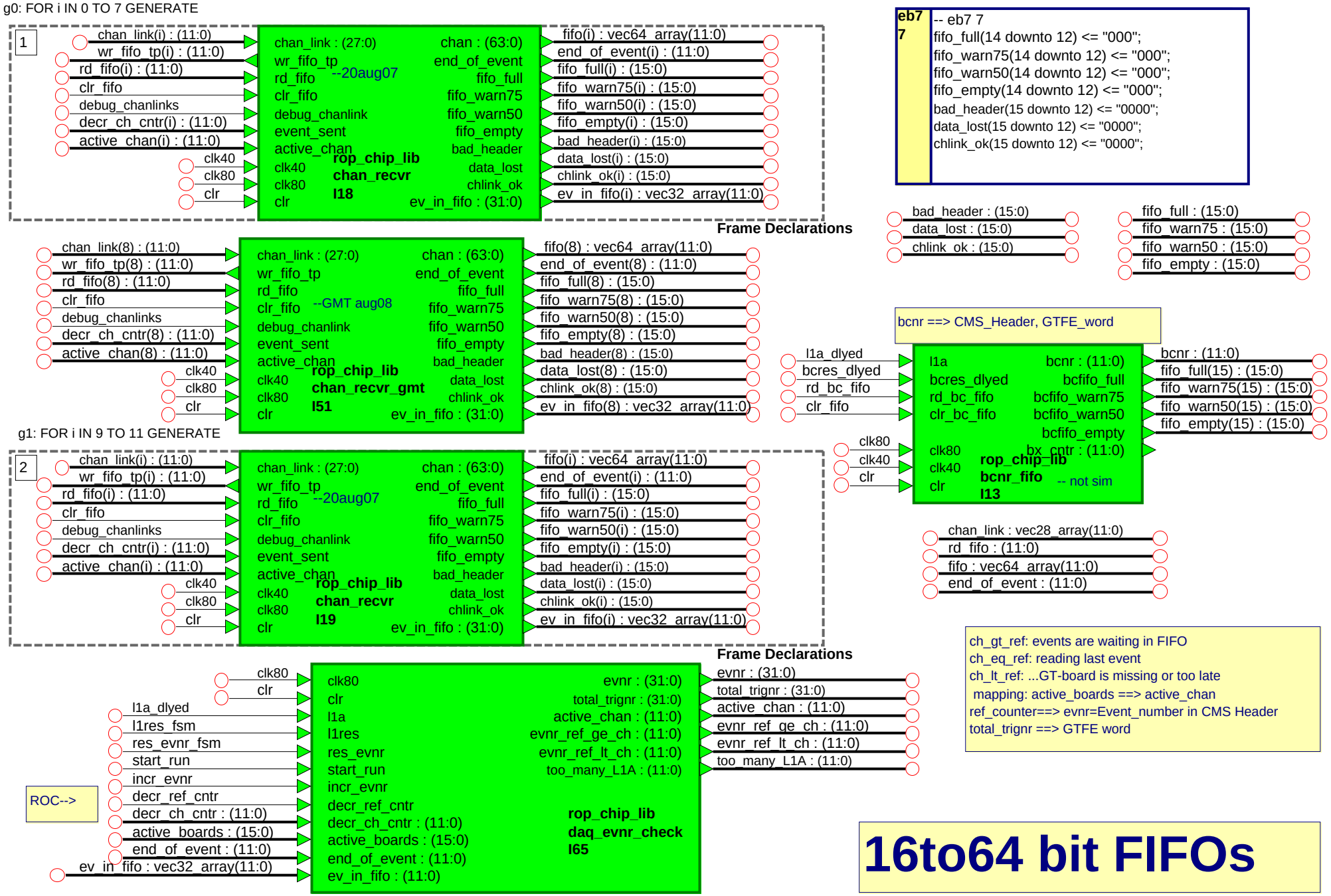
```
active_ch(11) <= active_boards(11);   -- TIM6U ==> channel 11
active_ch(10) <= active_boards(10);   -- TCS_M ==> channel 10
active_ch(9)  <= active_boards(9);    -- GMT_spare ==> channel 9
active_ch(8)  <= active_boards(8);    -- GMT_A  ==> channel 8
active_ch(7)  <= active_boards(7);    -- PSB6 ==> channel 7
active_ch(6)  <= active_boards(6);    -- PSB5 ==> channel 6
active_ch(5)  <= active_boards(5);    -- PSB4 ==> channel 5
active_ch(4)  <= active_boards(4);    -- PSB3 ==> channel 4
active_ch(3)  <= active_boards(3);    -- PSB2 ==> channel 3
active_ch(2)  <= active_boards(2);    -- PSB1 ==> channel 2
active_ch(1)  <= active_boards(1);    -- PSB0 ==> channel 1
active_ch(0)  <= active_boards(0);    -- FDL  ==> channel 0
                 -- GTFE ==> always included as first
```

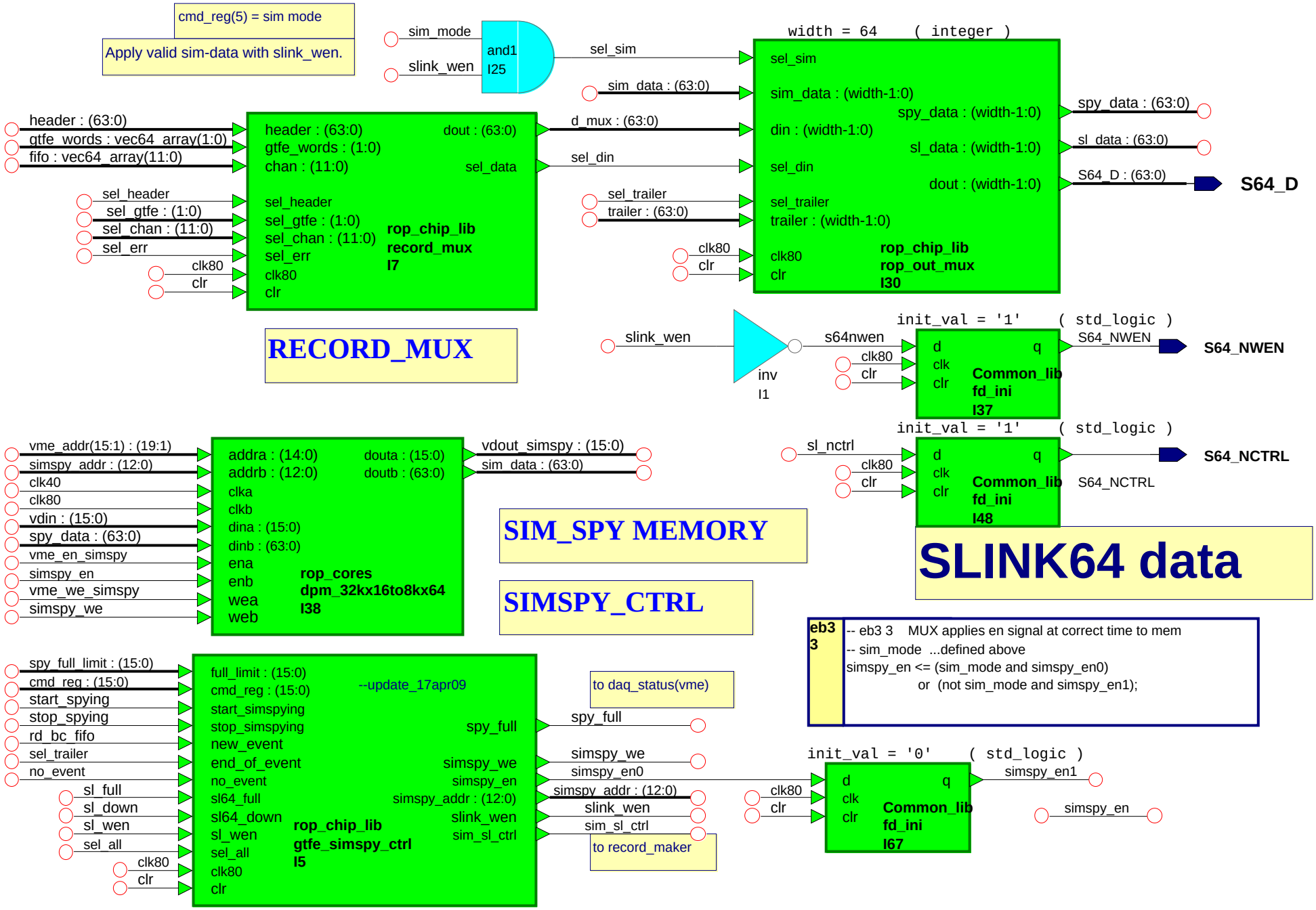


Channel Link mapping

Common LATENCY DELAY for all channels
to read at same time from RingBuffers.
Min.latency = 4/2 = 2bx

BCRes_int = 12.5 ns pulse

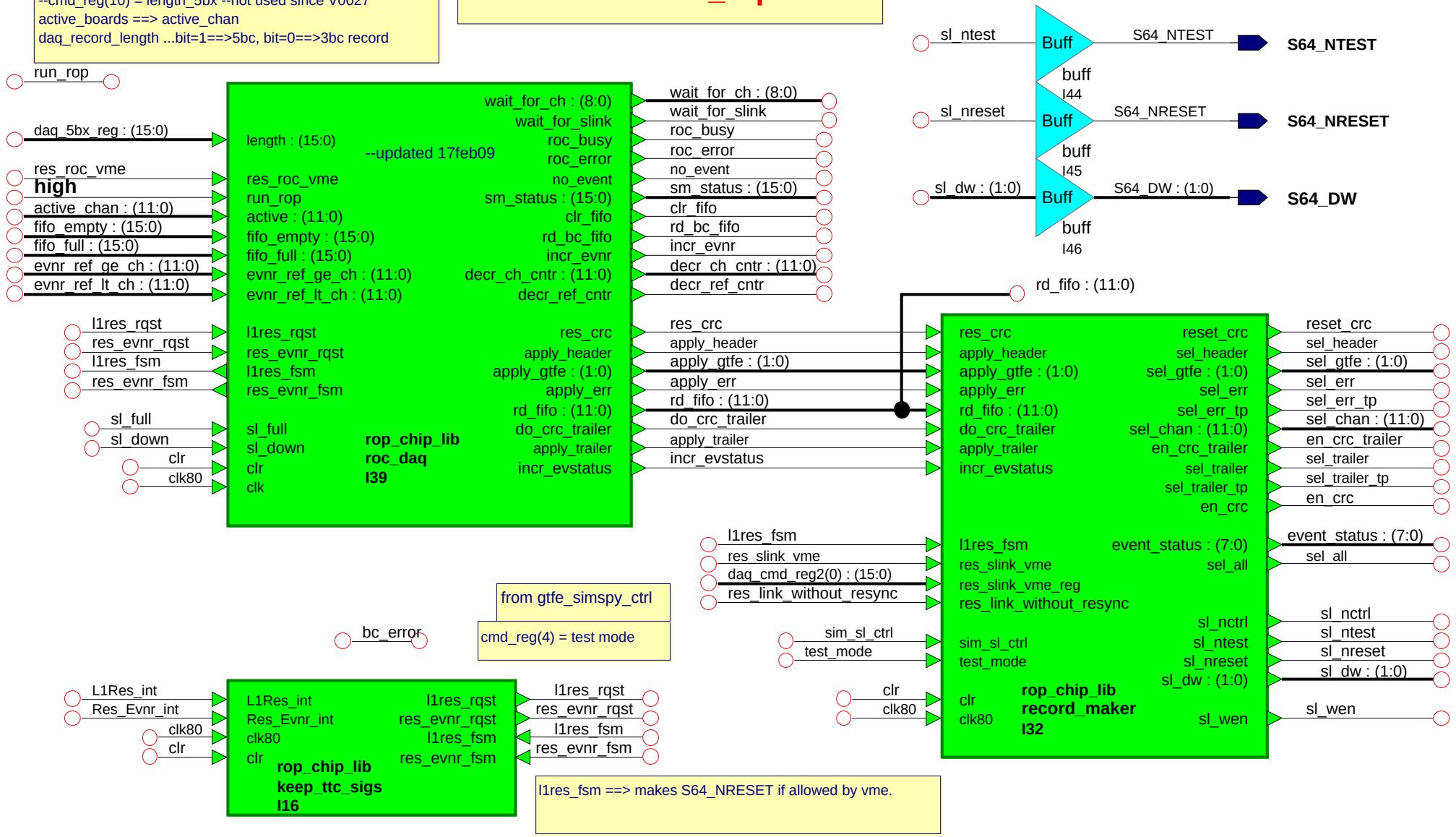

g0: FOR i IN 0 TO 7 GENERATE
1
chan_link(i) : (11:0)
wr_fifo_tp(i) : (11:0)
rd_fifo(i) : (11:0)
clr_fifo
debug_chanlinks
decr_ch_cntr(i) : (11:0)
active_chan(i) : (11:0)
clk40
clk80
clr
chan_link : (27:0)
wr_fifo_tp
rd_fifo
clr_fifo
debug_chanlink
event_sent
active_chan
clk40
clk80
clr
--20aug07
rop_chip_lib
chan_recvr
I18
chan : (63:0)
end_of_event
fifo_full
fifo_warn75
fifo_warn50
fifo_empty
bad_header
data_lost
chlink_ok
ev_in_fifo : (31:0)
fifo(i) : vec64_array(11:0)
end_of_event(i) : (11:0)
fifo_full(i) : (15:0)
fifo_warn75(i) : (15:0)
fifo_warn50(i) : (15:0)
fifo_empty(i) : (15:0)
bad_header(i) : (15:0)
data_lost(i) : (15:0)
chlink_ok(i) : (15:0)
ev_in_fifo(i) : vec32_array(11:0)
eb7 7
-- eb7 7
fifo_full(14 downto 12) <= "000";
fifo_warn75(14 downto 12) <= "000";
fifo_warn50(14 downto 12) <= "000";
fifo_empty(14 downto 12) <= "000";
bad_header(15 downto 12) <= "0000";
data_lost(15 downto 12) <= "0000";
chlink_ok(15 downto 12) <= "0000";
Frame Declarations
bad_header : (15:0)
data_lost : (15:0)
chlink_ok : (15:0)
fifo_full : (15:0)
fifo_warn75 : (15:0)
fifo_warn50 : (15:0)
fifo_empty : (15:0)
chan_link(8) : (11:0)
wr_fifo_tp(8) : (11:0)
rd_fifo(8) : (11:0)
clr_fifo
debug_chanlinks
decr_ch_cntr(8) : (11:0)
active_chan(8) : (11:0)
clk40
clk80
clr
--GMT aug08
rop_chip_lib
chan_recvr_gmt
I51
fifo(8) : vec64_array(11:0)
end_of_event(8) : (11:0)
fifo_full(8) : (15:0)
fifo_warn75(8) : (15:0)
fifo_warn50(8) : (15:0)
fifo_empty(8) : (15:0)
bad_header(8) : (15:0)
data_lost(8) : (15:0)
chlink_ok(8) : (15:0)
ev_in_fifo(8) : vec32_array(11:0)
bcnr ==> CMS_Header, GTFE_word
l1a_dlyed
bcres_dlyed
rd_bc_fifo
clr_fifo
clk80
clk40
clr
l1a
bcres_dlyed
rd_bc_fifo
clr_bc_fifo
clk80
clk40
clr
rop_chip_lib
bcnr_fifo
I13
-- not sim
bcnr : (11:0)
bcfifo_full
bcfifo_warn75
bcfifo_warn50
bcfifo_empty
bx_cntr : (11:0)
bcnr : (11:0)
fifo_full(15) : (15:0)
fifo_warn75(15) : (15:0)
fifo_warn50(15) : (15:0)
fifo_empty(15) : (15:0)
g1: FOR i IN 9 TO 11 GENERATE
2
chan_link(i) : (11:0)
wr_fifo_tp(i) : (11:0)
rd_fifo(i) : (11:0)
clr_fifo
debug_chanlinks
decr_ch_cntr(i) : (11:0)
active_chan(i) : (11:0)
clk40
clk80
clr
--20aug07
rop_chip_lib
chan_recvr
I19
fifo(i) : vec64_array(11:0)
end_of_event(i) : (11:0)
fifo_full(i) : (15:0)
fifo_warn75(i) : (15:0)
fifo_warn50(i) : (15:0)
fifo_empty(i) : (15:0)
bad_header(i) : (15:0)
data_lost(i) : (15:0)
chlink_ok(i) : (15:0)
ev_in_fifo(i) : vec32_array(11:0)
chan_link : vec28_array(11:0)
rd_fifo : (11:0)
fifo : vec64_array(11:0)
end_of_event : (11:0)
Frame Declarations
ch_gt_ref: events are waiting in FIFO
ch_eq_ref: reading last event
ch_lt_ref: ...GT-board is missing or too late
mapping: active_boards ==> active_chan
ref_counter==> evnr=Event_number in CMS Header
total_trignr ==> GTFE word
ROC-->
clk80
clr
l1a_dlyed
l1res_fsm
res_evnr_fsm
start_run
incr_evnr
decr_ref_cntr
decr_ch_cntr : (11:0)
active_boards : (15:0)
end_of_event : (11:0)
ev_in_fifo : vec32_array(11:0)
clk80
clr
l1a
l1res
res_evnr
start_run
incr_evnr
decr_ref_cntr
decr_ch_cntr : (11:0)
active_boards : (15:0)
end_of_event : (11:0)
ev_in_fifo : (11:0)
rop_chip_lib
daq_evnr_check
I65
evnr : (31:0)
total_trignr : (31:0)
active_chan : (11:0)
evnr_ref_ge_ch : (11:0)
evnr_ref_lt_ch : (11:0)
too_many_L1A : (11:0)
evnr : (31:0)
total_trignr : (31:0)
active_chan : (11:0)
evnr_ref_ge_ch : (11:0)
evnr_ref_lt_ch : (11:0)
too_many_L1A : (11:0)
16to64 bit FIFOs


cmd_reg(5) = sim mode
Apply valid sim-data with slink_wen.
sim_mode
slink_wen
and1
I25
sel_sim
sim_data : (63:0)
width = 64 (integer)
sel_sim
sim_data : (width-1:0)
spy_data : (width-1:0)
din : (width-1:0)
sl_data : (width-1:0)
sel_din
dout : (width-1:0)
sel_trailer
trailer : (width-1:0)
clk80
clr
rop_chip_lib
rop_out_mux
I30
spy_data : (63:0)
sl_data : (63:0)
S64_D : (63:0)
S64_D
header : (63:0)
gtfe_words : vec64_array(1:0)
fifo : vec64_array(11:0)
header : (63:0)
gtfe_words : (1:0)
chan : (11:0)
dout : (63:0)
sel_data
d_mux : (63:0)
sel_din
sel_trailer
trailer : (63:0)
sel_header
sel_gtfe : (1:0)
sel_chan : (11:0)
sel_err
clk80
clr
rop_chip_lib
record_mux
I7
RECORD_MUX
slink_wen
inv
I1
s64nwen
init_val = '1' (std_logic)
d
clk
clr
q
Common_lib
fd_ini
I37
S64_NWEN
S64_NWEN
vme_addr(15:1) : (19:1)
simspy_addr : (12:0)
clk40
clk80
vdin : (15:0)
spy_data : (63:0)
vme_en_simspy
simspy_en
vme_we_simspy
simspy_we
addra : (14:0)
addrb : (12:0)
douta : (15:0)
doutb : (63:0)
clka
clkb
dina : (15:0)
dinb : (63:0)
ena
enb
wea
web
rop_cores
dpm_32kx16to8kx64
I38
vdout_simspy : (15:0)
sim_data : (63:0)
SIM_SPY MEMORY
sl_nctrl
init_val = '1' (std_logic)
S64_NCTRL
S64_NCTRL
SLINK64 data
SIMSPY_CTRL
eb3 3
-- eb3 3 MUX applies en signal at correct time to mem
-- sim_mode ...defined above
simspy_en <= (sim_mode and simspy_en0)
or (not sim_mode and simspy_en1);
spy_full_limit : (15:0)
cmd_reg : (15:0)
start_spying
stop_spying
rd_bc_fifo
sel_trailer
no_event
sl_full
sl_down
sl_wen
sel_all
full_limit : (15:0)
cmd_reg : (15:0)
start_simspying
stop_simspying
new_event
end_of_event
no_event
sl64_full
sl64_down
sl_wen
sel_all
--update_17apr09
spy_full
simspy_we
simspy_en
simspy_addr : (12:0)
slink_wen
sim_sl_ctrl
rop_chip_lib
gtfe_simspy_ctrl
I5
to daq_status(vme)
spy_full
simspy_we
simspy_en0
simspy_addr : (12:0)
slink_wen
sim_sl_ctrl
to record_maker
init_val = '0' (std_logic)
simspy_en1
simspy_en
Common_lib
fd_ini
I67

ROC STATE MACHINE

since V001A: run_rop=1

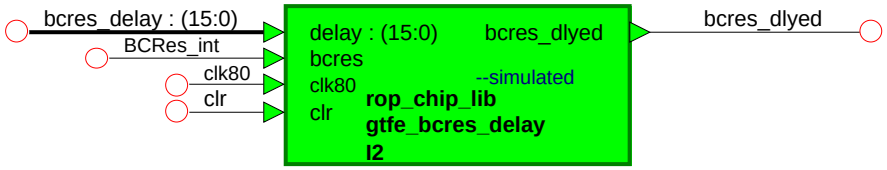
VME commands:
cmd_pulses:
  res_roc_vme
  cmd_pulse(start, stop) ==> run_rop_vme
--cmd_reg(10) = length_5bx --not used since V0027
active_boards ==> active_chan
daq_record_length ...bit=1==>5bc, bit=0==>3bc record

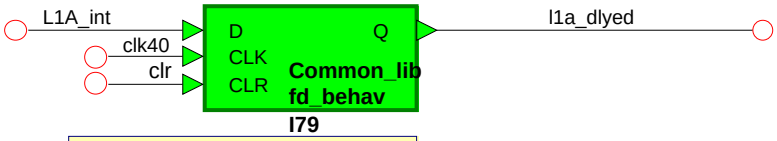


```
SIGNAL gtfe_words              : vec64_array(1 DOWNTO 0)
SIGNAL header                  : std_logic_vector(63 DOWNTO 0)
SIGNAL high                    : std_logic
SIGNAL ignore_errors           : std_logic_vector(15 DOWNTO 0)
SIGNAL ignore_slink_down       : std_logic
SIGNAL ignore_slink_full       : std_logic
SIGNAL incr_evnr               : std_logic
SIGNAL incr_evstatus           : std_logic
SIGNAL initial_crc             : std_logic_vector(15 DOWNTO 0)
SIGNAL irq                     : std_logic
SIGNAL l1a_dlyed               : std_logic
SIGNAL l1a_vme                 : std_logic
SIGNAL l1res_fsm               : std_logic
SIGNAL l1res_rqst              : std_logic
SIGNAL max_bc_nr               : std_logic_vector(15 DOWNTO 0)
SIGNAL mon_cntrreg             : vec16_array(83 DOWNTO 0)
SIGNAL n_clk_locked            : std_logic
SIGNAL no_event                : std_logic
SIGNAL or_unused               : std_logic
SIGNAL rd_00_1e                : std_logic_vector(15 DOWNTO 0)
SIGNAL rd_100_11e              : std_logic_vector(15 DOWNTO 0)
SIGNAL rd_120_13e              : std_logic_vector(15 DOWNTO 0)
SIGNAL rd_140_15e              : std_logic_vector(15 DOWNTO 0)
SIGNAL rd_160_17e              : std_logic_vector(15 DOWNTO 0)
SIGNAL rd_180_19e              : std_logic_vector(15 DOWNTO 0)
SIGNAL rd_1a0_1be              : std_logic_vector(15 DOWNTO 0)
SIGNAL rd_20_3e                : std_logic_vector(15 DOWNTO 0)
SIGNAL rd_40_5e                : std_logic_vector(15 DOWNTO 0)
SIGNAL rd_bc_fifo              : std_logic
SIGNAL rd_fifo                 : std_logic_vector(11 DOWNTO 0)
SIGNAL rec_clk                 : std_logic_vector(11 DOWNTO 0)
SIGNAL refresh_mon_cntrs       : std_logic
SIGNAL res_bc_error            : std_logic
SIGNAL res_crc                 : std_logic
SIGNAL res_evnr_fsm            : std_logic
SIGNAL res_evnr_rqst           : std_logic
SIGNAL res_link_without_resync : std_logic
SIGNAL res_roc_vme             : std_logic
SIGNAL res_runff_vme           : std_logic
SIGNAL res_slink_vme           : std_logic
SIGNAL reset_crc               : std_logic
SIGNAL robus                   : std_logic_vector(15 DOWNTO 0)
SIGNAL roc_busy                : std_logic
SIGNAL roc_error               : std_logic
SIGNAL rst_dcm                 : std_logic
SIGNAL run_rop                 : std_logic
SIGNAL s64nwen                 : std_logic
SIGNAL sel_all                 : std_logic
SIGNAL sel_chan                : std_logic_vector(11 DOWNTO 0)
SIGNAL sel_din                 : std_logic
SIGNAL sel_err                 : std_logic
SIGNAL sel_err_tp              : std_logic
SIGNAL sel_gtfe                : std_logic_vector(1 DOWNTO 0)
SIGNAL sel_header              : std_logic
SIGNAL sel_sim                 : std_logic
SIGNAL sel_trailer             : std_logic
SIGNAL sel_trailer_tp          : std_logic
SIGNAL setup_version           : std_logic_vector(31 DOWNTO 0)
SIGNAL show_slink_bits         : std_logic_vector(15 DOWNTO 0)
SIGNAL sim_data                : std_logic_VECTOR(63 DOWNTO 0)
SIGNAL sim_mode                : std_logic
SIGNAL sim_sl_ctrl             : std_logic
SIGNAL simspy_addr             : std_logic_vector(12 DOWNTO 0)
SIGNAL simspy_en               : std_logic
SIGNAL simspy_en0              : std_logic
```

bcres_delay: tested

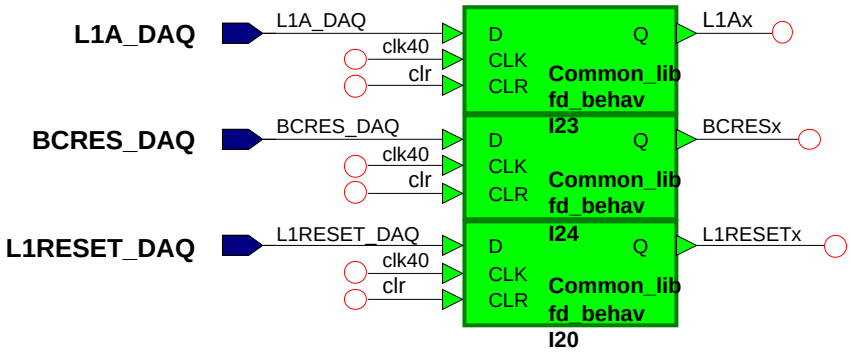
BCRES Delay



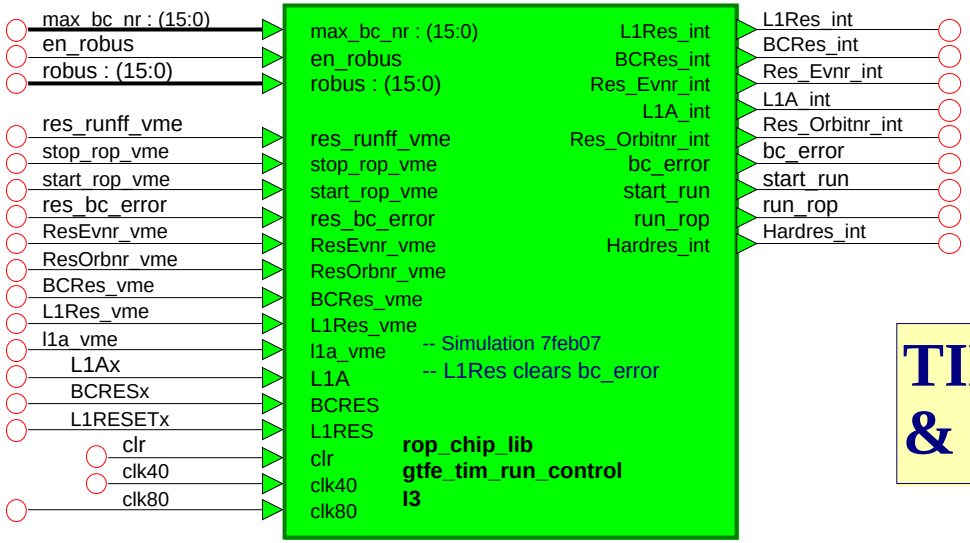


V0026: clk40 for l1a_dlyed

L1A delay seems not to be required.
because the L1A delay for GTFE
on the TIM board could be used to get
equal BC numbers in the event record.



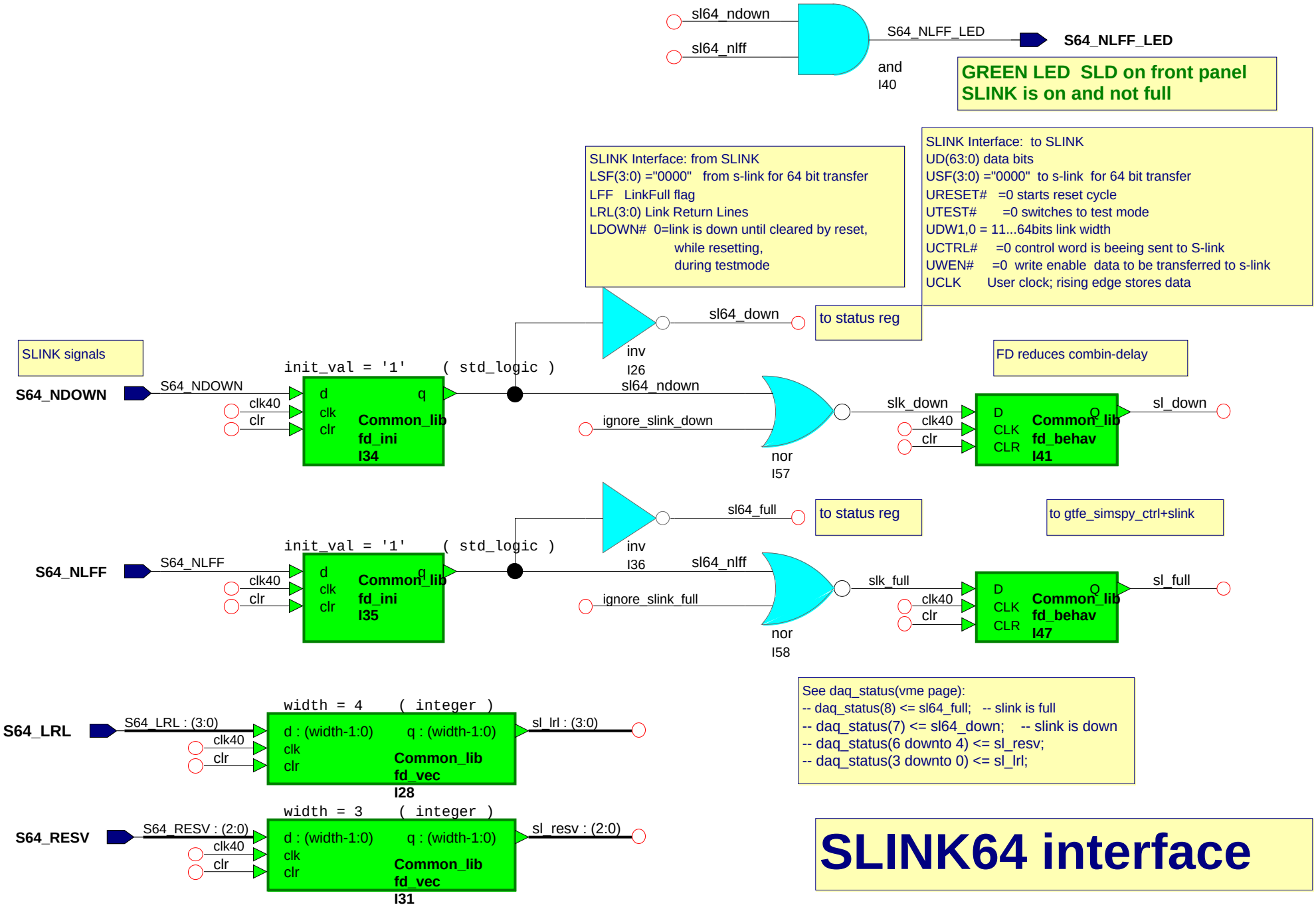
daq_cmd_reg(2)



BCRes_int = 12.5 ns pulse ==> DELAY
Res_Orbitnr_int ...not used

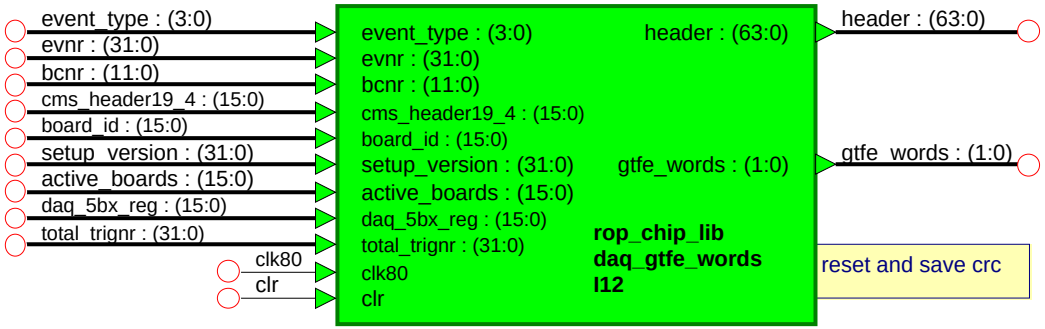
L1A_int = 25 ns pulse ==> ROP
Res_Evnr, L1Res_int ==> ROP
bc_error, run_rop ==> ROP, daq_status
start_run = 12.5ns pulse ==>ROP
Hardres_int ... not used

TIM Signal Decoder & BC_counter

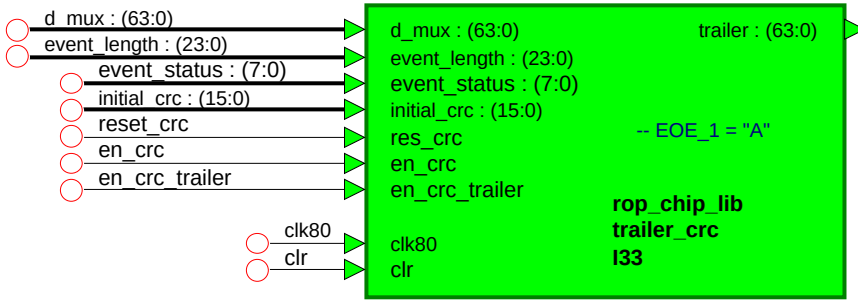

sl64_ndown
sl64_nlff
and
I40
S64_NLFF_LED
S64_NLFF_LED
GREEN LED SLD on front panel
SLINK is on and not full
SLINK Interface: from SLINK
LSF(3:0) ="0000" from s-link for 64 bit transfer
LFF LinkFull flag
LRL(3:0) Link Return Lines
LDOWN# 0=link is down until cleared by reset,
while resetting,
during testmode
SLINK Interface: to SLINK
UD(63:0) data bits
USF(3:0) ="0000" to s-link for 64 bit transfer
URESET# =0 starts reset cycle
UTEST# =0 switches to test mode
UDW1,0 = 11...64bits link width
UCTRL# =0 control word is beeing sent to S-link
UWEN# =0 write enable data to be transferred to s-link
UCLK User clock; rising edge stores data
sl64_down
to status reg
inv
I26
SLINK signals
FD reduces combin-delay
S64_NDOWN
S64_NDOWN
init_val = '1' (std_logic)
d
clk
clr
q
Common_lib
fd_ini
I34
clk40
clr
sl64_ndown
ignore_slink_down
nor
I57
slk_down
clk40
clr
D
CLK
CLR
Q
Common_lib
fd_behav
I41
sl_down
sl64_full
to status reg
to gtfe_simspy_ctrl+slink
inv
I36
S64_NLFF
S64_NLFF
init_val = '1' (std_logic)
Common_lib
fd_ini
I35
sl64_nlff
ignore_slink_full
nor
I58
slk_full
Common_lib
fd_behav
I47
sl_full
See daq_status(vme page):
-- daq_status(8) <= sl64_full; -- slink is full
-- daq_status(7) <= sl64_down; -- slink is down
-- daq_status(6 downto 4) <= sl_resv;
-- daq_status(3 downto 0) <= sl_lrl;
S64_LRL
S64_LRL : (3:0)
width = 4 (integer)
d : (width-1:0)
q : (width-1:0)
Common_lib
fd_vec
I28
sl_lrl : (3:0)
S64_RESV
S64_RESV : (2:0)
width = 3 (integer)
Common_lib
fd_vec
I31
sl_resv : (2:0)
SLINK64 interface

cmd_reg(15:12) = event_type

GTFE WORDS



reset and save crc



TRAILER_CRC

V102A

Include Trailer word with CRC=0 into the CRC calculation
and insert the crc result afterwards.
Slink: After Full flag we have to stop after <32 words.
If full goes inactive again continue with transfer.
Use 'S64_NWEN' to send a word to the S-link.
We can start and stop sending whenever required.
S-LINK f= 40...80 MHz

Automatic EVENT LENGTH CALCULATION

HEADER + GTFE + TRAILER words

```
STANDARD MEMORY SIZES
3k chip: 96 RAMB
SIM_SPY MEMORY: 1 ev=202 W64
 ~40 events =>8kx64 ==> 32 RAMB
FDL:        1kx64   ==>  4 RAMB
7 PSB: 1kx64 =4ramb ==> 28 RAMB
GMT:        4kx64   ==> 16 RAMB
TIM:        1kx64   ==>  4 RAMB
TCSM:       1kx64   ==>  4 RAMB
GMTspare:   1kx64   ==>  4 RAMB
BC-Fifo 1kx16       ==>  1 RAMB
------------------------
                     = 93 RAMB
```

```
Channel FIFOs
3bx case:
1k W64 ==> psb(18W64) : 56 events
1k W64 ==> fdl(21W64) : 48 events
4k W64 ==> gmt(51W64) : 80 events
5bx case:
1k W64 ==> psb(30W64) : 34 events
1k W64 ==> fdl(35W64) : 29 events
4k W64 ==> gmt(85W64) : 48 events
3/5bx case:
1k W64 ==> psb(18W64) : 56 events
1k W64 ==> fdl(35W64) : 29 events
4k W64 ==> gmt(85W64) : 48 events
```

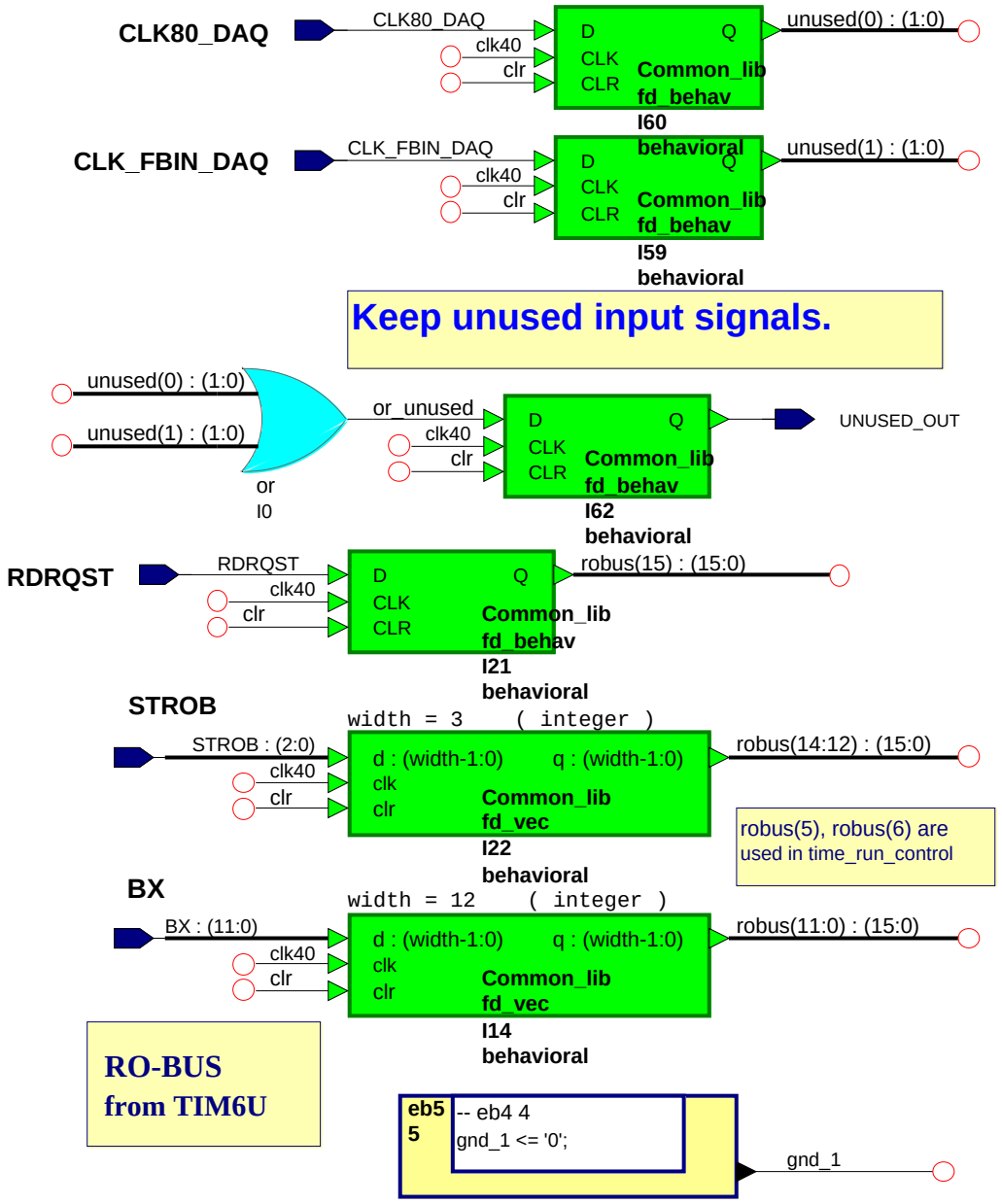
```
1.)ALTERNATIVE MEMORY SIZES
3k chip: 96 RAMB
SIM_SPY MEMORY: 1 ev=202 W64
 ~10 events =>2kx64 ==>  8 RAMB
FDL:        2kx64   ==>  8 RAMB
7 PSB: 2kx64 =8ramb ==> 56 RAMB
GMT         2kx64   ==>  8 RAMB
TIM:        2kx64   ==>  8 RAMB
TCSM:         ---   ==>  0 RAMB
GMTspare:     ---   ==>  0 RAMB
BC-Fifo 1kx16       ==>  1 RAMB
------------------------
                     = 89 RAMB
```

```
Channel FIFOs
3bx case:
2k W64 ==> psb(18W64) : 112 events
2k W64 ==> fdl(21W64) :  96 events
2k W64 ==> gmt(51W64) :  40 events
5bx case:
2k W64 ==> psb(30W64) :  68 events
2k W64 ==> fdl(35W64) :  58 events
2k W64 ==> gmt(85W64) :  24 events
3/5bx case:
2k W64 ==> psb(18W64) : 112 events
2k W64 ==> fdl(35W64) :  58 events
2k W64 ==> gmt(85W64) :  24 events
```

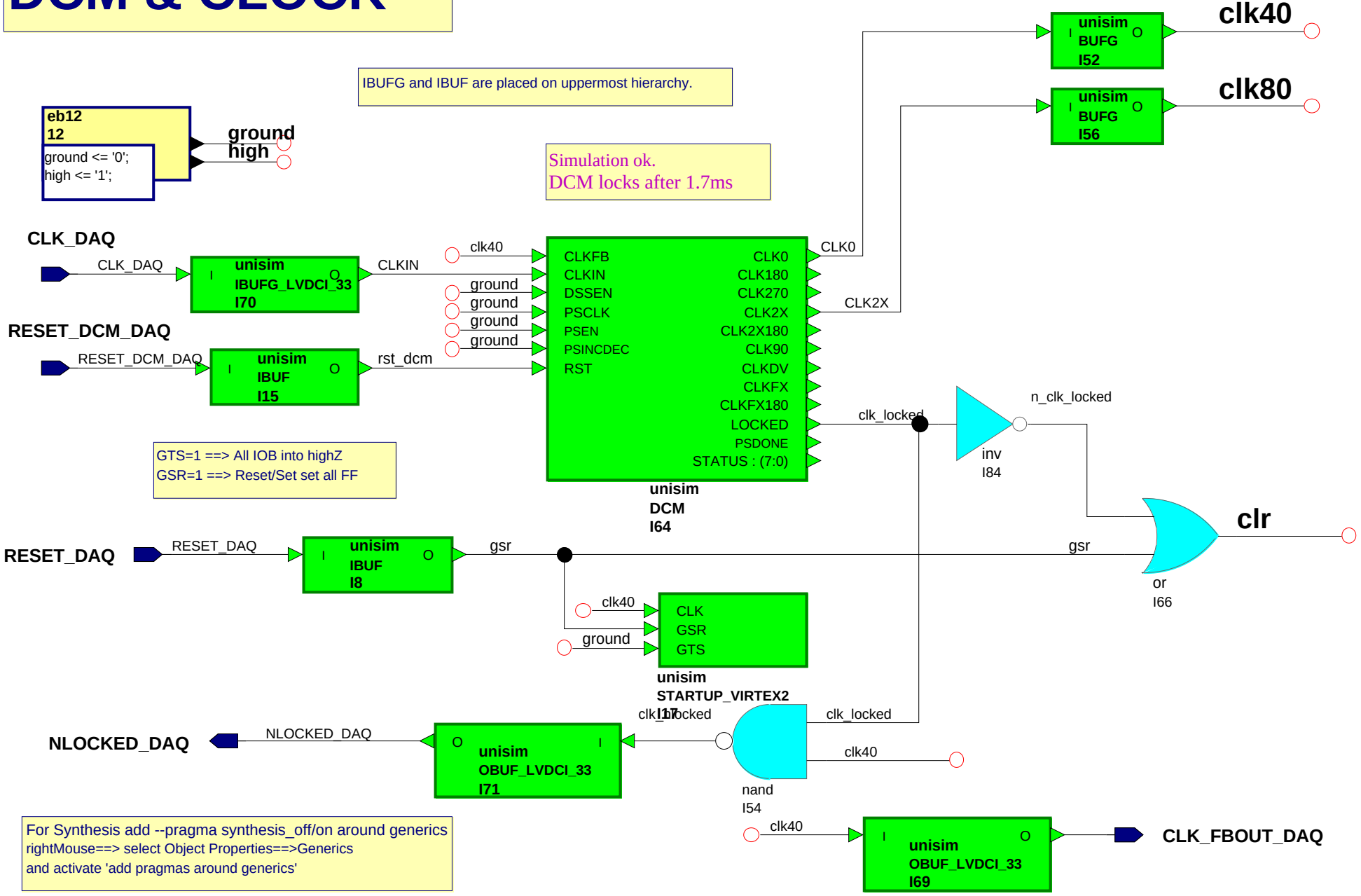
```
2.)ALTERNATIVE MEMORY SIZES
3k chip: 96 RAMB
SIM_SPY MEMORY: 1 ev=202 W64
 ~10 events =>2kx64 ==>  8 RAMB
FDL:        4kx64   ==> 16 RAMB
7 PSB: 2kx64 =8ramb ==> 56 RAMB
GMT         1kx64   ==>  4 RAMB
TIM:        2kx64   ==>  8 RAMB
TCSM:         ---   ==>  0 RAMB
GMTspare:     ---   ==>  0 RAMB
BC-Fifo 1kx16       ==>  1 RAMB
------------------------
                     = 93 RAMB
```

```
Channel FIFOs
3bx case:
2k W64 ==> psb(18W64) : 112 events
4k W64 ==> fdl(21W64) : 192 events
1k W64 ==> gmt(51W64) :  20 events
5bx case:
2k W64 ==> psb(30W64) :  68 events
4k W64 ==> fdl(35W64) : 116 events
1k W64 ==> gmt(85W64) :  12 events
3/5bx case:
2k W64 ==> psb(18W64) : 112 events
4k W64 ==> fdl(35W64) : 116 events
1k W64 ==> gmt(85W64) :  12 events
```

```
SIGNAL simspy_en1              : std_logic
SIGNAL simspy_we               : std_logic
SIGNAL sl64_down               : std_logic
SIGNAL sl64_full               : std_logic
SIGNAL sl64_ndown              : std_logic
SIGNAL sl64_nlff               : std_logic
SIGNAL sl_data                 : std_logic_vector(63 DOWNTO 0)
SIGNAL sl_down                 : std_logic
SIGNAL sl_dw                   : std_logic_vector( 1 DOWNTO 0 )
SIGNAL sl_full                 : std_logic
SIGNAL sl_lrl                  : std_logic_vector( 3 DOWNTO 0 )
SIGNAL sl_nctrl                : std_logic
SIGNAL sl_nreset               : std_logic
SIGNAL sl_ntest                : std_logic
SIGNAL sl_resv                 : std_logic_vector( 2 DOWNTO 0 )
SIGNAL sl_wen                  : std_logic
SIGNAL slink_wen               : std_logic
SIGNAL slk_down                : std_logic
SIGNAL slk_full                : std_logic
SIGNAL sm_status               : std_logic_vector(15 DOWNTO 0 )
SIGNAL spy_data                : std_logic_vector(63 DOWNTO 0)
SIGNAL spy_full                : std_logic
SIGNAL spy_full_limit          : std_logic_vector(15 DOWNTO 0)
SIGNAL start_rop_vme           : std_logic
SIGNAL start_run               : std_logic
SIGNAL start_spying            : std_logic
SIGNAL stop_rop_vme            : std_logic
SIGNAL stop_spying             : std_logic
SIGNAL test_mask1              : std_logic_vector(15 DOWNTO 0)
SIGNAL test_mask2              : std_logic_vector(15 DOWNTO 0)
SIGNAL test_mask3              : std_logic_vector(15 DOWNTO 0)
SIGNAL test_mode               : std_logic
SIGNAL too_many_L1A            : std_logic_vector(11 DOWNTO 0)
SIGNAL total_trignr            : std_logic_vector(31 DOWNTO 0)
SIGNAL tp_busy                 : std_logic
SIGNAL tp_busy_pri             : std_logic
SIGNAL tp_err_pri              : std_logic
SIGNAL tp_error                : std_logic
SIGNAL tp_or_full              : std_logic
SIGNAL tp_or_too_many          : std_logic
SIGNAL tp_or_warn50            : std_logic
SIGNAL tp_or_warn75            : std_logic
SIGNAL tp_out_of_sync          : std_logic
SIGNAL tp_outsync_pri          : std_logic
SIGNAL tp_rdy_pri              : std_logic
SIGNAL tp_warn_pri             : std_logic
SIGNAL tp_warning              : std_logic
SIGNAL trailer                 : std_logic_vector(63 DOWNTO 0)
SIGNAL unused                  : std_logic_vector(1 DOWNTO 0)
SIGNAL vdii                    : std_logic_VECTOR(15 DOWNTO 0)
SIGNAL vdin                    : std_logic_VECTOR(15 DOWNTO 0)
SIGNAL vdoo                    : std_logic_VECTOR(15 DOWNTO 0)
SIGNAL vdout                   : std_logic_vector(15 DOWNTO 0)
SIGNAL vdout_simspy            : std_logic_VECTOR(15 DOWNTO 0)
SIGNAL vme_addr                : std_logic_VECTOR(19 DOWNTO 1)
SIGNAL vme_addresse            : std_logic_vector(19 DOWNTO 0)
SIGNAL vme_dis_rd              : std_logic
SIGNAL vme_en                  : std_logic
SIGNAL vme_en1                 : std_logic
SIGNAL vme_en2                 : std_logic
SIGNAL vme_en3                 : std_logic
SIGNAL vme_en_rd               : std_logic
SIGNAL vme_en_simspy           : std_logic
SIGNAL vme_enn                 : std_logic
SIGNAL vme_rd_simspy           : std_logic
```



DCM & CLOCK

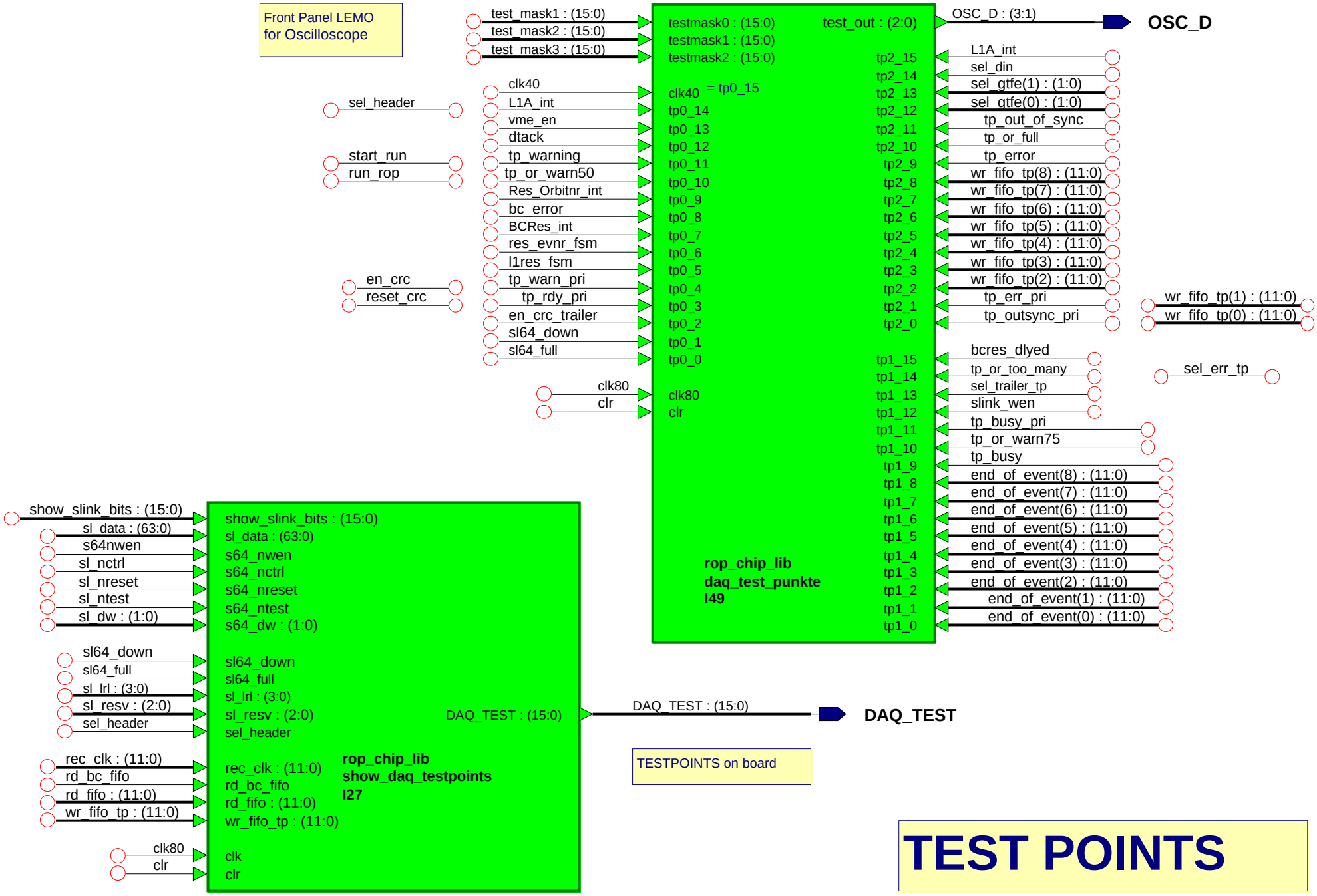

IBUFG and IBUF are placed on uppermost hierarchy.
eb12
12
ground <= '0';
high <= '1';
ground
high
Simulation ok.
DCM locks after 1.7ms
unisim
BUFG
I52
clk40
unisim
BUFG
I56
clk80
CLK_DAQ
CLK_DAQ
unisim
IBUFG_LVDCI_33
I70
CLKIN
clk40
ground
ground
ground
ground
CLKFB
CLKIN
DSSEN
PSCLK
PSEN
PSINCDEC
RST
CLK0
CLK180
CLK270
CLK2X
CLK2X180
CLK90
CLKDV
CLKFX
CLKFX180
LOCKED
PSDONE
STATUS : (7:0)
CLK0
CLK2X
unisim
DCM
I64
RESET_DCM_DAQ
RESET_DCM_DAQ
unisim
IBUF
I15
rst_dcm
clk_locked
n_clk_locked
inv
I84
GTS=1 ==> All IOB into highZ
GSR=1 ==> Reset/Set set all FF
RESET_DAQ
RESET_DAQ
unisim
IBUF
I8
gsr
gsr
clr
or
I66
clk40
ground
CLK
GSR
GTS
unisim
STARTUP_VIRTEX2
I17
clk_locked
clk_locked
clk40
nand
I54
NLOCKED_DAQ
NLOCKED_DAQ
unisim
OBUF_LVDCI_33
I71
clk40
unisim
OBUF_LVDCI_33
I69
CLK_FBOUT_DAQ
For Synthesis add --pragma synthesis_off/on around generics
rightMouse==> select Object Properties==>Generics
and activate 'add pragmas around generics'

DCM parameter

```
TimingChecksOn                        = true                  ( boolean          )
InstancePath                          = "*"                   ( string           )
Xon                                   = true                  ( boolean          )
MsgOn                                 = false                 ( boolean          )
thold_PSEN_PSCLK_negedge_posedge      = 0.000 ns              ( VitalDelayType   )
thold_PSEN_PSCLK_posedge_posedge      = 0.000 ns              ( VitalDelayType   )
thold_PSINCDEC_PSCLK_negedge_posedge  = 0.000 ns              ( VitalDelayType   )
thold_PSINCDEC_PSCLK_posedge_posedge  = 0.000 ns              ( VitalDelayType   )
tipd_CLKFB                            = (0.000 ns, 0.000 ns)  ( VitalDelayType01 )
tipd_CLKIN                            = (0.000 ns, 0.000 ns)  ( VitalDelayType01 )
tipd_DSSEN                            = (0.000 ns, 0.000 ns)  ( VitalDelayType01 )
tipd_PSCLK                            = (0.000 ns, 0.000 ns)  ( VitalDelayType01 )
tipd_PSEN                             = (0.000 ns, 0.000 ns)  ( VitalDelayType01 )
tipd_PSINCDEC                         = (0.000 ns, 0.000 ns)  ( VitalDelayType01 )
tipd_RST                              = (0.000 ns, 0.000 ns)  ( VitalDelayType01 )
tpd_CLKIN_LOCKED                      = (0.000 ns, 0.000 ns)  ( VitalDelayType01 )
tpd_PSCLK_PSDONE                      = (0.000 ns, 0.000 ns)  ( VitalDelayType01 )
tperiod_CLKIN_POSEDGE                 = 0.000 ns              ( VitalDelayType   )
tperiod_PSCLK_POSEDGE                 = 0.000 ns              ( VitalDelayType   )
tpw_CLKIN_negedge                     = 0.000 ns              ( VitalDelayType   )
tpw_CLKIN_posedge                     = 0.000 ns              ( VitalDelayType   )
tpw_PSCLK_negedge                     = 0.000 ns              ( VitalDelayType   )
tpw_PSCLK_posedge                     = 0.000 ns              ( VitalDelayType   )
tpw_RST_posedge                       = 0.000 ns              ( VitalDelayType   )
tsetup_PSEN_PSCLK_negedge_posedge     = 0.000 ns              ( VitalDelayType   )
tsetup_PSEN_PSCLK_posedge_posedge     = 0.000 ns              ( VitalDelayType   )
tsetup_PSINCDEC_PSCLK_negedge_posedge = 0.000 ns              ( VitalDelayType   )
tsetup_PSINCDEC_PSCLK_posedge_posedge = 0.000 ns              ( VitalDelayType   )
CLKDV_DIVIDE                          = 2.0                   ( real             )
CLKFX_DIVIDE                          = 1                     ( integer          )
CLKFX_MULTIPLY                        = 4                     ( integer          )
CLKIN_DIVIDE_BY_2                     = false                 ( boolean          )
CLKIN_PERIOD                          = 23.0                  ( real             )
CLKOUT_PHASE_SHIFT                    = "NONE"                ( string           )
CLK_FEEDBACK                          = "1X"                  ( string           )
DESKEW_ADJUST                         = "SYSTEM_SYNCHRONOUS"  ( string           )
DFS_FREQUENCY_MODE                    = "LOW"                 ( string           )
DLL_FREQUENCY_MODE                    = "LOW"                 ( string           )
DSS_MODE                              = "NONE"                ( string           )
DUTY_CYCLE_CORRECTION                 = true                  ( boolean          )
FACTORY_JF                            = X"C080"               ( bit_vector       )
MAXPERCLKIN                           = 1000000 ps            ( time             )
MAXPERPSCLK                           = 100000000 ps          ( time             )
PHASE_SHIFT                           = 0                     ( integer          )
SIM_CLKIN_CYCLE_JITTER                = 300 ps                ( time             )
SIM_CLKIN_PERIOD_JITTER               = 1000 ps               ( time             )
STARTUP_WAIT                          = false                 ( boolean          )
```

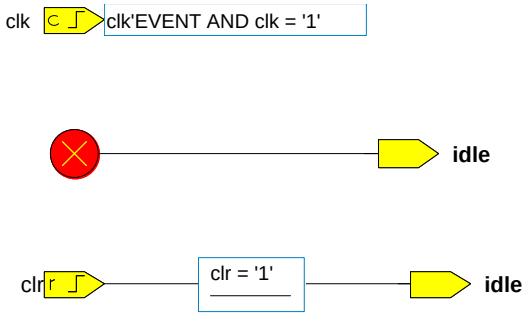
Bit 0 : =1 Show S64_D(15:0) on testpoints DAQ_TEST(15:0)
Bit 1 : =1 Show S64_D(31:16) on testpoints DAQ_TEST(15:0)
Bit 2 : =1 Show S64_D(47:32) on testpoints DAQ_TEST(15:0)
Bit 3 : =1 Show S64_D(63:48) on testpoints DAQ_TEST(15:0)

Bit 4 : =1 Show S64_NWEN on DAQ_TEST(0) // to S-Link64
    Show S64_NCTRL on DAQ_TEST(1) // to S-Link64
    Show S64_NRESET on DAQ_TEST(2) // to S-Link64
    Show S64_NTEST on DAQ_TEST(3) // to S-Link64
    Show S64_DW(0) on DAQ_TEST(4) // to S-Link64
    Show S64_DW(1) on DAQ_TEST(5) // to S-Link64
    Show S64_DOWN on DAQ_TEST(6) // from S-Link64
    Show S64_FULL on DAQ_TEST(7) // from S-Link64
    Show S64_LRL(3:0) on DAQ_TEST(11:8) // from S-Link64
    Show S64_RESV(2:0) on DAQ_TEST(14:12) // from S-Link64
    Show SEL_HEADER on DAQ_TEST(15) // begin of record
Bit 5: =1 Show REC_CLK signals(11 downto 0)


Front Panel LEMO
for Oscilloscope
test_mask1 : (15:0)
test_mask2 : (15:0)
test_mask3 : (15:0)
sel_header
start_run
run_rop
en_crc
reset_crc
clk40
L1A_int
vme_en
dtack
tp_warning
tp_or_warn50
Res_Orbitnr_int
bc_error
BCRes_int
res_evnr_fsm
l1res_fsm
tp_warn_pri
tp_rdy_pri
en_crc_trailer
sl64_down
sl64_full
clk80
clr
testmask0 : (15:0)
testmask1 : (15:0)
testmask2 : (15:0)
clk40 = tp0_15
tp0_14
tp0_13
tp0_12
tp0_11
tp0_10
tp0_9
tp0_8
tp0_7
tp0_6
tp0_5
tp0_4
tp0_3
tp0_2
tp0_1
tp0_0
clk80
clr
rop_chip_lib
daq_test_punkte
I49
test_out : (2:0)
tp2_15
tp2_14
tp2_13
tp2_12
tp2_11
tp2_10
tp2_9
tp2_8
tp2_7
tp2_6
tp2_5
tp2_4
tp2_3
tp2_2
tp2_1
tp2_0
tp1_15
tp1_14
tp1_13
tp1_12
tp1_11
tp1_10
tp1_9
tp1_8
tp1_7
tp1_6
tp1_5
tp1_4
tp1_3
tp1_2
tp1_1
tp1_0
OSC_D : (3:1)
OSC_D
L1A_int
sel_din
sel_gtfe(1) : (1:0)
sel_gtfe(0) : (1:0)
tp_out_of_sync
tp_or_full
tp_error
wr_fifo_tp(8) : (11:0)
wr_fifo_tp(7) : (11:0)
wr_fifo_tp(6) : (11:0)
wr_fifo_tp(5) : (11:0)
wr_fifo_tp(4) : (11:0)
wr_fifo_tp(3) : (11:0)
wr_fifo_tp(2) : (11:0)
tp_err_pri
tp_outsync_pri
wr_fifo_tp(1) : (11:0)
wr_fifo_tp(0) : (11:0)
bcres_dlyed
tp_or_too_many
sel_trailer_tp
slink_wen
tp_busy_pri
tp_or_warn75
tp_busy
end_of_event(8) : (11:0)
end_of_event(7) : (11:0)
end_of_event(6) : (11:0)
end_of_event(5) : (11:0)
end_of_event(4) : (11:0)
end_of_event(3) : (11:0)
end_of_event(2) : (11:0)
end_of_event(1) : (11:0)
end_of_event(0) : (11:0)
sel_err_tp
show_slink_bits : (15:0)
sl_data : (63:0)
s64nwen
sl_nctrl
sl_nreset
sl_ntest
sl_dw : (1:0)
sl64_down
sl64_full
sl_lrl : (3:0)
sl_resv : (2:0)
sel_header
rec_clk : (11:0)
rd_bc_fifo
rd_fifo : (11:0)
wr_fifo_tp : (11:0)
clk80
clr
show_slink_bits : (15:0)
sl_data : (63:0)
s64_nwen
s64_nctrl
s64_nreset
s64_ntest
s64_dw : (1:0)
sl64_down
sl64_full
sl_lrl : (3:0)
sl_resv : (2:0)
sel_header
rec_clk : (11:0)
rd_bc_fifo
rd_fifo : (11:0)
wr_fifo_tp : (11:0)
clk
clr
rop_chip_lib
show_daq_testpoints
I27
DAQ_TEST : (15:0)
DAQ_TEST : (15:0)
DAQ_TEST
TESTPOINTS on board
TEST POINTS

```
old HISTORY:
Version 0019:   l1a_latency register + l1a_delay added!!  (l1a_dlyed -->evnr_check, bcnr_fifo)
                fifo_16to64_4k corrected
Version 0018: chan_recvr : 11x fifo_1k =44ramb, gmt: fifo_4k =16 ramb ==> 60ramb;  simspy+bcnr_fifo =33ramb
                           60+33= 93 <96ramb(max)
Version 0017: ROC : GMT logic simplified (send 1bxdata, then check Slink)
Version 0016: daq_evnr_check: changed to 32 bit counters
Version 0015: daq_status_encoder: sl_full(inkl.ignore bit) included
Version 0014: chan_recvr_gmt:  Empty logic for GMT per event, not per word
Version 0013: Send GT-Status to EVM chip: GPS 56-->48 bits, ROC,daq_evnr_check modified
Version 0012: ROC: length for GMT extended to 3x17, 5x17 w64
Version 0011: chlink_ok register, chan_receiver_gmt, ROC:res_roc_vme
Version 0010: corrected simspyctrl avoids 'spurious' events when switching from spy to sim mode
Version 000f: ROC changed, new: res_runff_vme, daq_status(11):=run_rop
Version 000e: ROC: roc_busy  -->daq_status_encoder // for TCS
Version 000d: ROC: stop_run
Version 000c:status flag logic corrected; chlink_too_late flag not included as error anymore
Version 000b: initial values corrected, reset_slink extended....1st working version!!
Version 000a: DAQ_TEST points: with clk80, s64nwen, new SM-states
Version 0009: FIFO logic corrected
Version 0008: L1Res_int ...error corrected, S64_NCTRL becomes 0 for Header+Trailer
                CRC sum gives same result as Janos VHDL code, but that is different to
                CRC in janos example file.
                 New 'IGNORE_SL64_DOWN,FULL' reg bits.
                 LED   SLD= Green if Slink is not down and not full
Version 0007 - 0001: several improvements
```

```
SIGNAL vme_we_simspy           : std_logic
SIGNAL vme_wr                  : std_logic
SIGNAL wait_for_ch             : std_logic_vector(8 DOWNTO 0)
SIGNAL wait_for_slink          : std_logic
SIGNAL wr_00_1e                : std_logic_vector(15 DOWNTO 0)
SIGNAL wr_40_5e                : std_logic_vector(15 DOWNTO 0)
SIGNAL wr_fifo_tp              : std_logic_vector(11 DOWNTO 0)
```


clk
clk'EVENT AND clk = '1'
idle
clr
clr = '1'
idle

SM_STATUS
==========
8000 = IDLE status because run_rop =0
8001 = ongoing L1RES
0000 = ROC is transmitting an event
Begin of an event:
10FF = waiting for SLINK64 before waiting for a new event
20FF = waiting for a new event= waiting for a L1A signal
4000 = FIFO FULL error, // L1Reset(=Resync) or res_roc_vme reset ROC to IDLE status
FDL =channel 0 (vme slot 10)
1000 = waiting for SLINK
2000 = waiting for FDL data // evnr_ch_lt_ref(0) = '1' OR fifo_empty(0) = '1
3000 = starting transfer of FDL data
PSB0 =channel 1 //Technical Triggers
1001 = waiting for SLINK
2001 = waiting for PSB0 data (vme slot 9)
3001 = starting transfer of PSB0 data
PSB1 =channel 2 // Calo data
1002 = waiting for SLINK
2002 = waiting for PSB1 data (vme slot 13)
3002 = starting transfer of PSB1 data
PSB2 =channel 3 // Calo data
1003 = waiting for SLINK
2003 = waiting for PSB2 data (vme slot 14)
3003 = starting transfer of PSB2 data
PSB3 =channel 4 // Calo data
1004 = waiting for SLINK
2004 = waiting for PSB3 data (vme slot 15)
3004 = starting transfer of PSB3 data
PSB4 =channel 5 // M/Q bits
1005 = waiting for SLINK
2005 = waiting for PSB4 data (vme slot 19)
3005 = starting transfer of PSB4 data
PSB5 =channel 6 // M/Q bits
1006 = waiting for SLINK
2006 = waiting for PSB5 data (vme slot 20)
3006 = starting transfer of PSB5 data
PSB6 =channel 7 // M/Q bits
1007 = waiting for SLINK
2007 = waiting for PSB6 data (vme slot 21)
3007 = starting transfer of PSB6 data
GMT =channel 8 // Muons
1081 = waiting for SLINK before bx1 1082 = waiting for SLINK before bx2 1083 = waiting for SLINK before bx3
1084 = waiting for SLINK before bx4 1085 = waiting for SLINK before bx5
1091 = waiting for SLINK before bx1-err 1092 = waiting for SLINK before bx2-err 1093 = waiting for SLINK before bx3-err
1094 = waiting for SLINK before bx4-err 1095 = waiting for SLINK before bx5-err
2008 = waiting for GMT data (vme slot 18)
3008 = starting transfer of GMT data
TIM =channel 11 // test data
1011 = waiting for SLINK
2011 = waiting for TIM data (vme slot 16)

Global Actions **Concurrent Statements** **Architecture Declarations**
Pre Actions:
Post Actions:

Package List

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_arith.all;
USE ieee.STD_LOGIC_UNSIGNED.all;
```

Signals Status **State Register Statements** **Process Declarations**
Clocked Process:
Output Process:

SIGNAL	MODE	DEFAULT	RESET	SCHEME
apply_err	OUT	'0'	'0'	CLKD
apply_gtfe	OUT	(OTHERS => '0')	(OTHERS => '0')	CLKD
apply_header	OUT	'0'	'0'	CLKD
apply_trailer	OUT	'0'	'0'	CLKD
clr_fifo	OUT	'0'	'0'	CLKD
decr_ch_cntr	OUT	(OTHERS => '0')	(OTHERS => '0')	CLKD
decr_ref_cntr	OUT	'0'	'0'	CLKD
do_crc_trailer	OUT	'0'	'0'	CLKD
incr_evnr	OUT	'0'	'0'	CLKD
incr_evstatus	OUT	'0'	'0'	CLKD
l1res_fsm	OUT	'0'	'0'	CLKD
no_event	OUT	'0'	'0'	CLKD
rd_bc_fifo	OUT	'0'	'0'	CLKD
rd_fifo	OUT	(OTHERS => '0')	(OTHERS => '0')	CLKD
res_crc	OUT	'0'	'0'	CLKD
res_evnr_fsm	OUT	'0'	'0'	CLKD
roc_busy	OUT	'0'	'0'	CLKD
roc_error	OUT	'0'	'0'	CLKD
sm_status	OUT	(OTHERS => '0')	(OTHERS => '0')	CLKD
wait_for_ch	OUT	(OTHERS => '0')	(OTHERS => '0')	CLKD
wait_for_slink	OUT	'0'	'0'	CLKD

ROC behavior:
==========
- The ROC skips non active boards. In that case the EVENT_LENGTH register has to be programmed accordingly.
- If Channel Link data do not arrive within 255*12.5 ns = 3.187,5 ns then the ROC sends 'FFFF' instead and proceeds to the next channel.
- If the SLINK64 is not ready the ROC waits for an infinite time.
  Either a VME 'res_roc' command or a L1RESET(=RESYNC) can reset the ROC to the IDLE state.
- According to the 'length(i)' register bit the ROC expects data form 3 or 5 bunchcrossings.
  If the record lengths of boards and the DAQ chip disagree bad data will be sent.
  Never change 'length' register during run !!

* ROC sends 'roc_error' when FIFOs are full
* ROC sends 'roc_busy' while resetting after a L1Reset/Resync command.
* ROC resets SLINK64 after a L1Reset/Resync command.

28.Aug08:
Outputs: wait_for_slink, wait_for_ch(8:0)

Return from ERROR, wait_slink, wait_ev states with:
res_roc_vme = '1' only.
Aug.2007: L1RESET_RQST serviced only before or after an event.

1=highest priority condition

state= combinatorial
clk-edge switches from old to new state
Clocked Signal switches back to '0' =default value.
===> 1 assignment makes a 1T pulse

res_roc_vme removed at end_ev

res_roc_vme or l1res immediately stop transmission.
--> 1.) make L1RESET/resync
--> 2.) 'clr_fifo'

clr_fifo =25ns signal for FIFO-write side(40MHz).

<company name>		Project:	rop_chip
Title:	ROC_DAQ State Machine	lengths are hardwired with wait statesfor 3bx and 5bx	
Path:	rop_chip_lib/roc_daq/fsm		
Edited:	by taurok on 17 Feb 2009		

3011 = starting transfer of TIM data

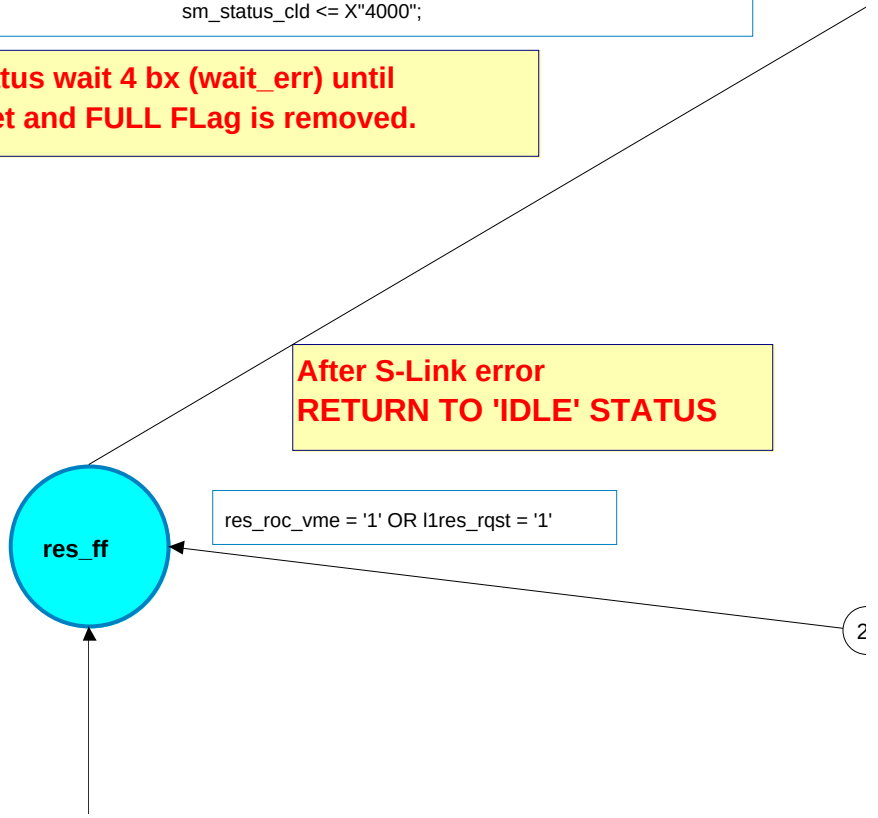
V00254 (Aug08): wait_for_slink, wait_for_ch(8:0) added
Version 001F: res_evnr returns to idle to make a 'clr_fifo'
fifo_empty(8) = '0' AND evnr_ref_ge_ch(8) = '1' required
before sending each GMT_bx.
Version 001D: gmt changed, end_of event simplified
Version 0013: NEW logic with ...ge
Version 000E: hard_res removed, new: roc_busy

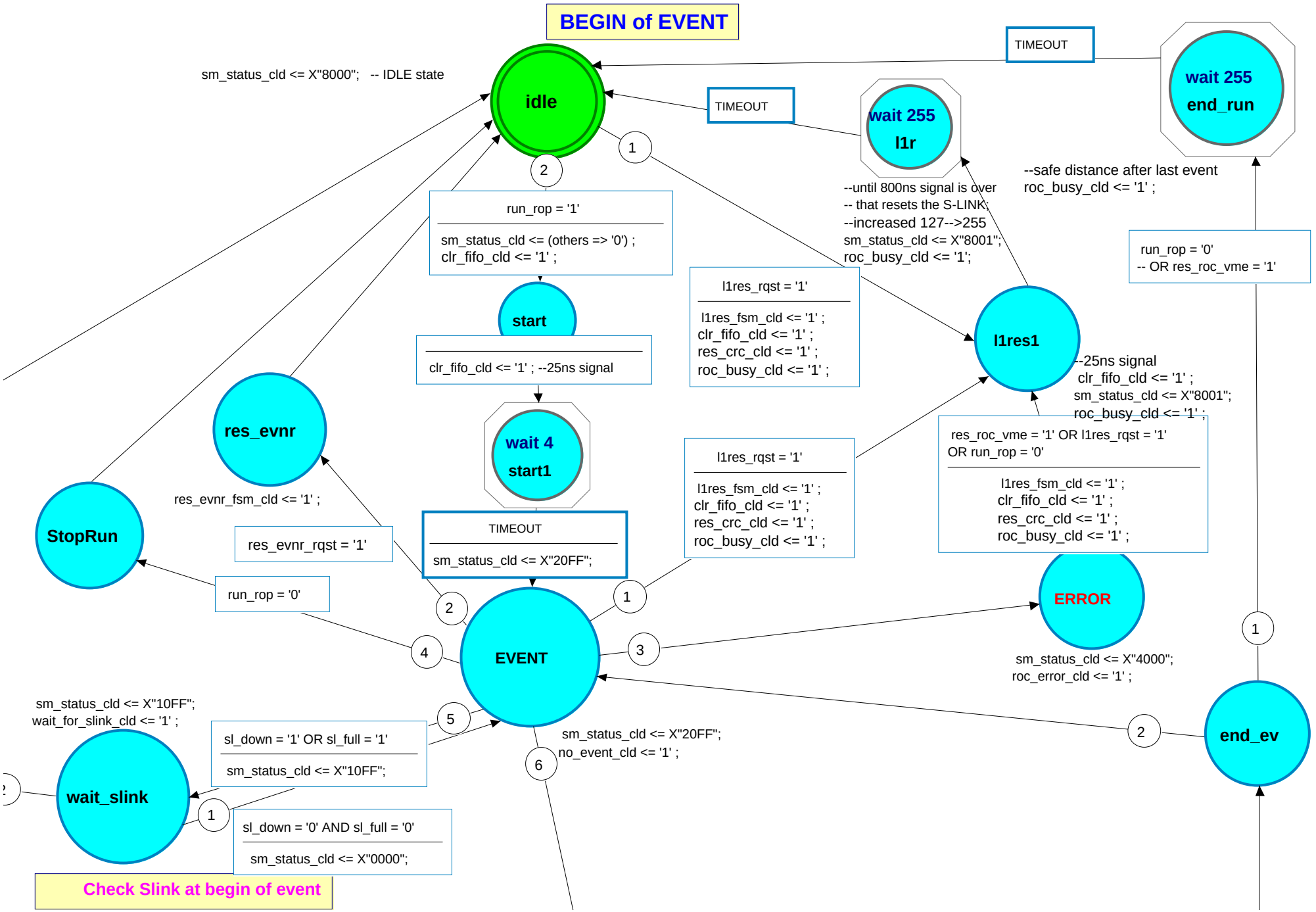
Conditions for ==> ERROR state'4000'
(on next page)

```
fifo_full(15) = '1'
    OR ( fifo_full(0) = '1' AND active(0) = '1' ) OR ( fifo_full(1) = '1' AND active(1) = '1' )
    OR ( fifo_full(2) = '1' AND active(2) = '1' ) OR ( fifo_full(3) = '1' AND active(3) = '1' )
    OR ( fifo_full(4) = '1' AND active(4) = '1' ) OR ( fifo_full(5) = '1' AND active(5) = '1' )
    OR ( fifo_full(6) = '1' AND active(6) = '1' ) OR ( fifo_full(7) = '1' AND active(7) = '1' )
    OR ( fifo_full(8) = '1' AND active(8) = '1' )
```

```
-- FIFO FULL ERROR
 sm_status_cld <= X"4000";
```

After Error status wait 4 bx (wait_err) until
FIFOs are reset and FULL FLag is removed.




BEGIN of EVENT
sm_status_cld <= X"8000"; -- IDLE state
idle
TIMEOUT
wait 255
end_run
TIMEOUT
wait 255
l1r
--until 800ns signal is over
-- that resets the S-LINK;
--increased 127-->255
sm_status_cld <= X"8001";
roc_busy_cld <= '1';
--safe distance after last event
roc_busy_cld <= '1' ;
1
2
run_rop = '1'
sm_status_cld <= (others => '0') ;
clr_fifo_cld <= '1' ;
run_rop = '0'
-- OR res_roc_vme = '1'
start
clr_fifo_cld <= '1' ; --25ns signal
l1res_rqst = '1'
l1res_fsm_cld <= '1' ;
clr_fifo_cld <= '1' ;
res_crc_cld <= '1' ;
roc_busy_cld <= '1' ;
l1res1
--25ns signal
clr_fifo_cld <= '1' ;
sm_status_cld <= X"8001";
roc_busy_cld <= '1' ;
res_evnr
res_evnr_fsm_cld <= '1' ;
wait 4
start1
res_roc_vme = '1' OR l1res_rqst = '1'
OR run_rop = '0'
l1res_fsm_cld <= '1' ;
clr_fifo_cld <= '1' ;
res_crc_cld <= '1' ;
roc_busy_cld <= '1' ;
StopRun
res_evnr_rqst = '1'
TIMEOUT
sm_status_cld <= X"20FF";
l1res_rqst = '1'
l1res_fsm_cld <= '1' ;
clr_fifo_cld <= '1' ;
res_crc_cld <= '1' ;
roc_busy_cld <= '1' ;
run_rop = '0'
ERROR
sm_status_cld <= X"4000";
roc_error_cld <= '1' ;
EVENT
3
4
5
6
sm_status_cld <= X"10FF";
wait_for_slink_cld <= '1' ;
sm_status_cld <= X"20FF";
no_event_cld <= '1' ;
end_ev
sl_down = '1' OR sl_full = '1'
sm_status_cld <= X"10FF";
wait_slink
sl_down = '0' AND sl_full = '0'
sm_status_cld <= X"0000";
Check Slink at begin of event

RETURN TO 'IDLE' STATUS

HEADER + GTFE data

```
sl_down = '0' AND sl_full = '0' AND fifo_full(15) = '0'  AND
l1res_rqst = '0' AND res_evnr_rqst = '0'
AND  fifo_empty(15) = '0' AND
(       (fifo_empty(0) = '0' AND active(0) = '1' )   -- wait for ChanLink
  OR (fifo_empty(1) = '0' AND active(1) = '1' )
  OR (fifo_empty(2) = '0' AND active(2) = '1' )
  OR (fifo_empty(3) = '0' AND active(3) = '1' )
  OR (fifo_empty(4) = '0' AND active(4) = '1' )
  OR (fifo_empty(5) = '0' AND active(5) = '1' )
  OR (fifo_empty(6) = '0' AND active(6) = '1' )
  OR (fifo_empty(7) = '0' AND active(7) = '1' )
  OR (fifo_empty(8) = '0' AND active(8) = '1' )
  OR (fifo_empty(11) = '0' AND active(11) = '1' ) )
```

```
sm_status_cld <= (others => '0') ; no_event_cld <= '0' ;
incr_evnr_cld <= '1' ; res_crc_cld <= '1' ;
rd_bc_fifo_cld <= '1' ; -- 1tick
```

EVENT_STATUS:
- res_crc also resets event_status
- every error increments the event_status

prepare

-- Signals return to default status =0
-- ==> 1T pulses:
-- incr_evnr_cld , rd_bc_fifo_cld, res_crc_cld

clk_edge ==>

```
apply_header_cld <= '1' ;
```

header word

header

clk_edge ==>

```
apply_gtfe_cld(0) <= '1' ;
```

1st gtfe word

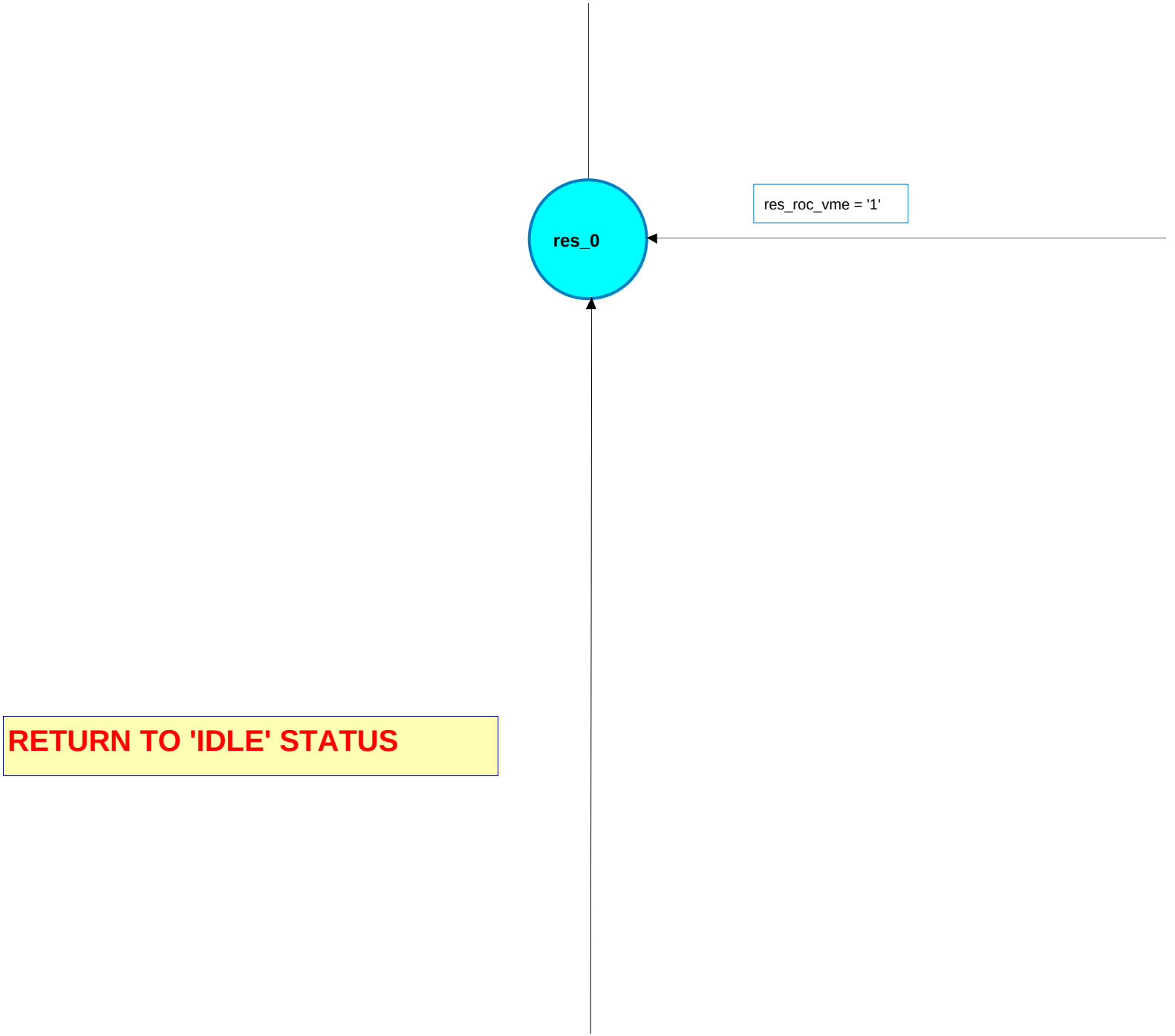
gtfe

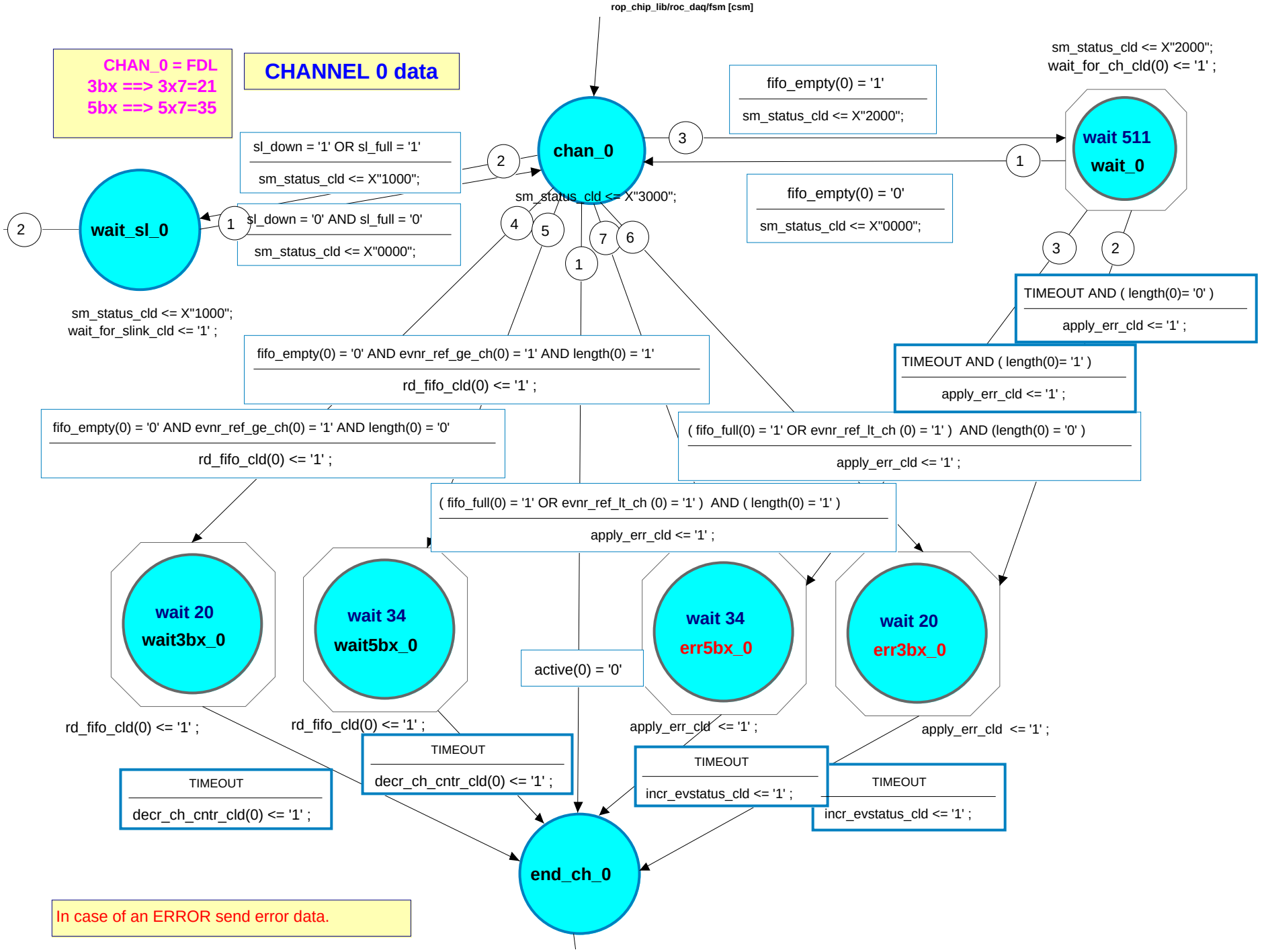
```
apply_gtfe_cld(1) <= '1' ;
```

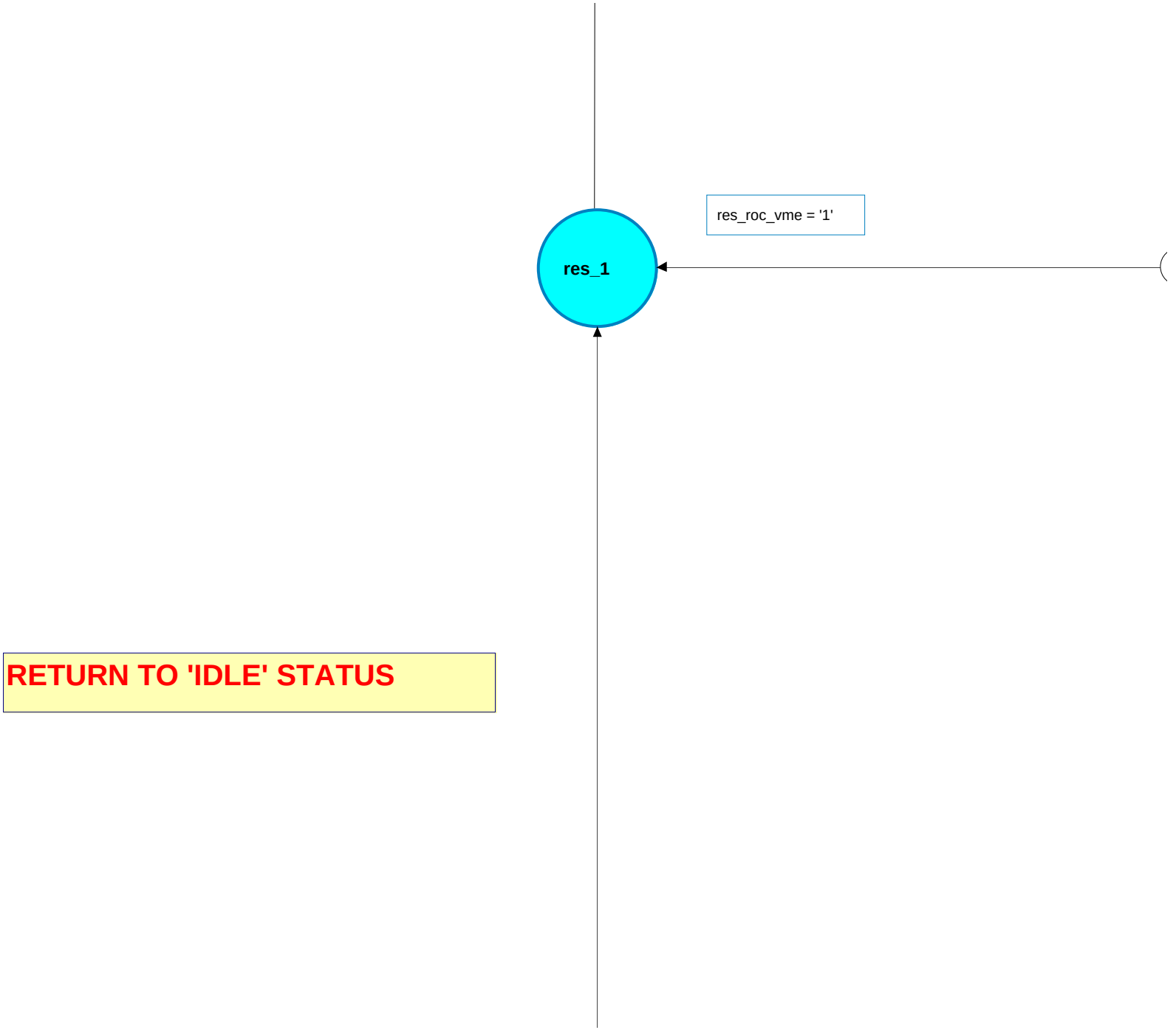
2nd gtfe word

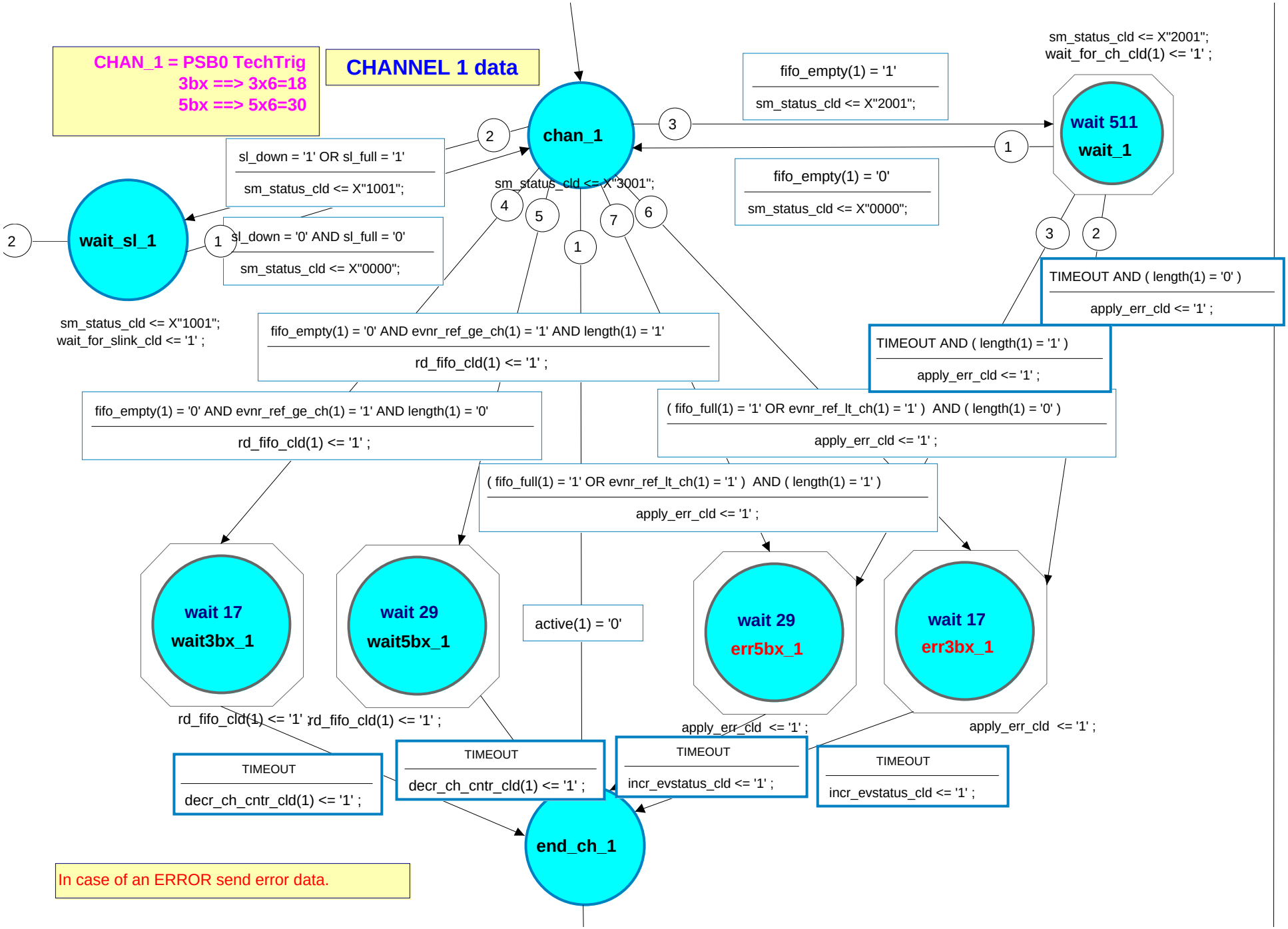
Wait infinitly for the SLINK64.
Emergy exit by res_roc_vme or

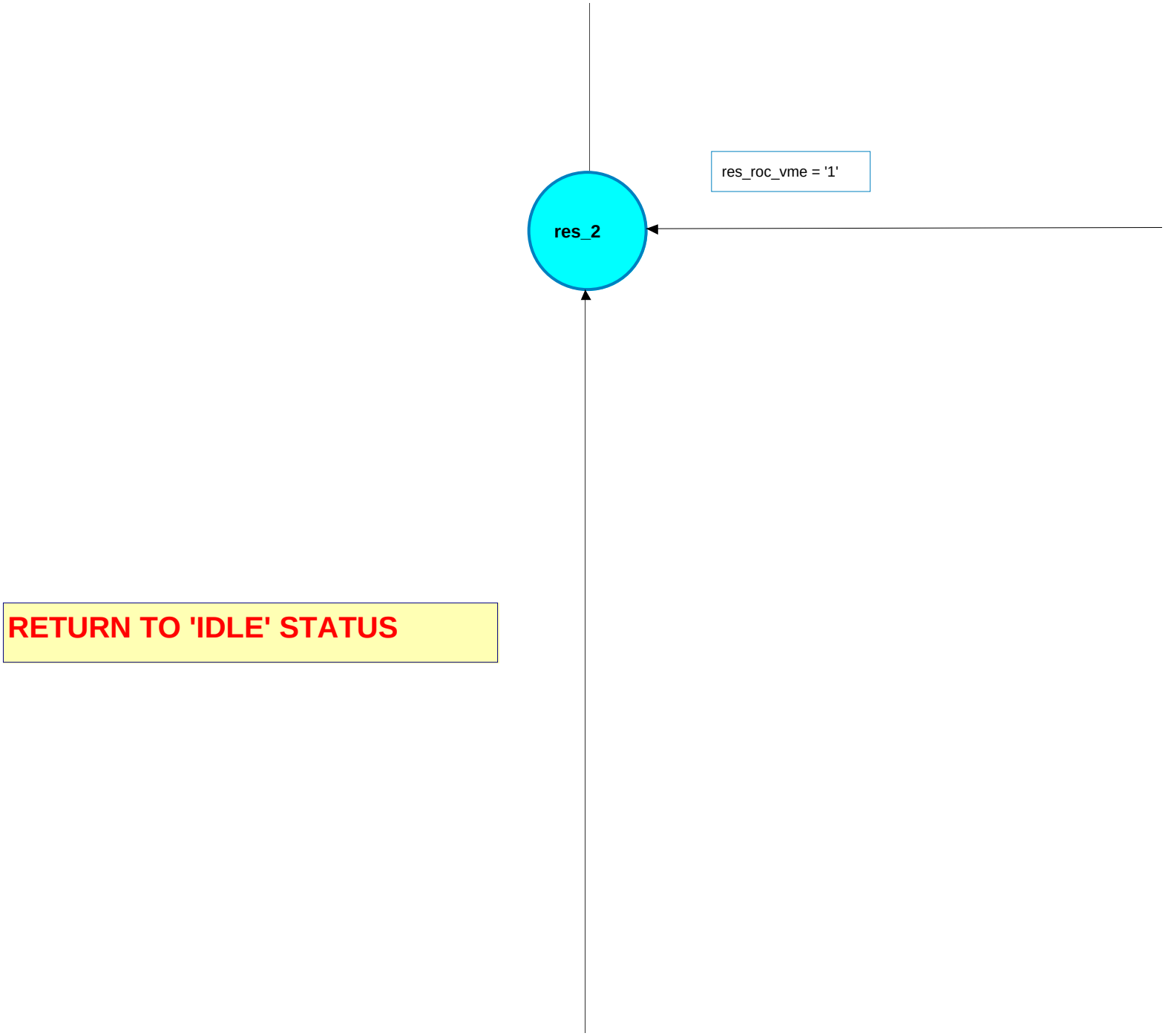
Send error data, when no
Channel Link data arrive
within 512 ticks = 6,25 us
and increment event_status

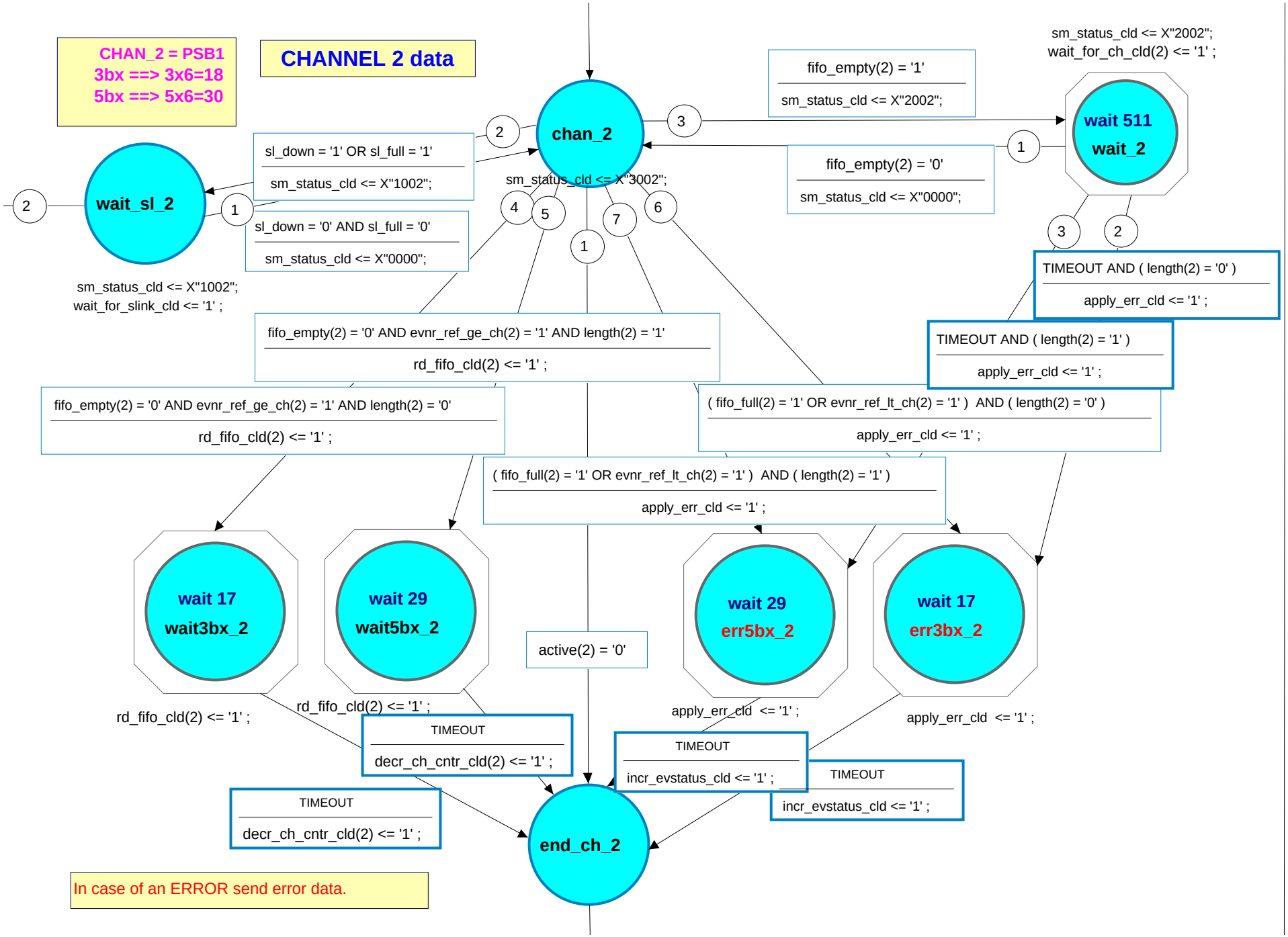

res_0
res_roc_vme = '1'
RETURN TO 'IDLE' STATUS

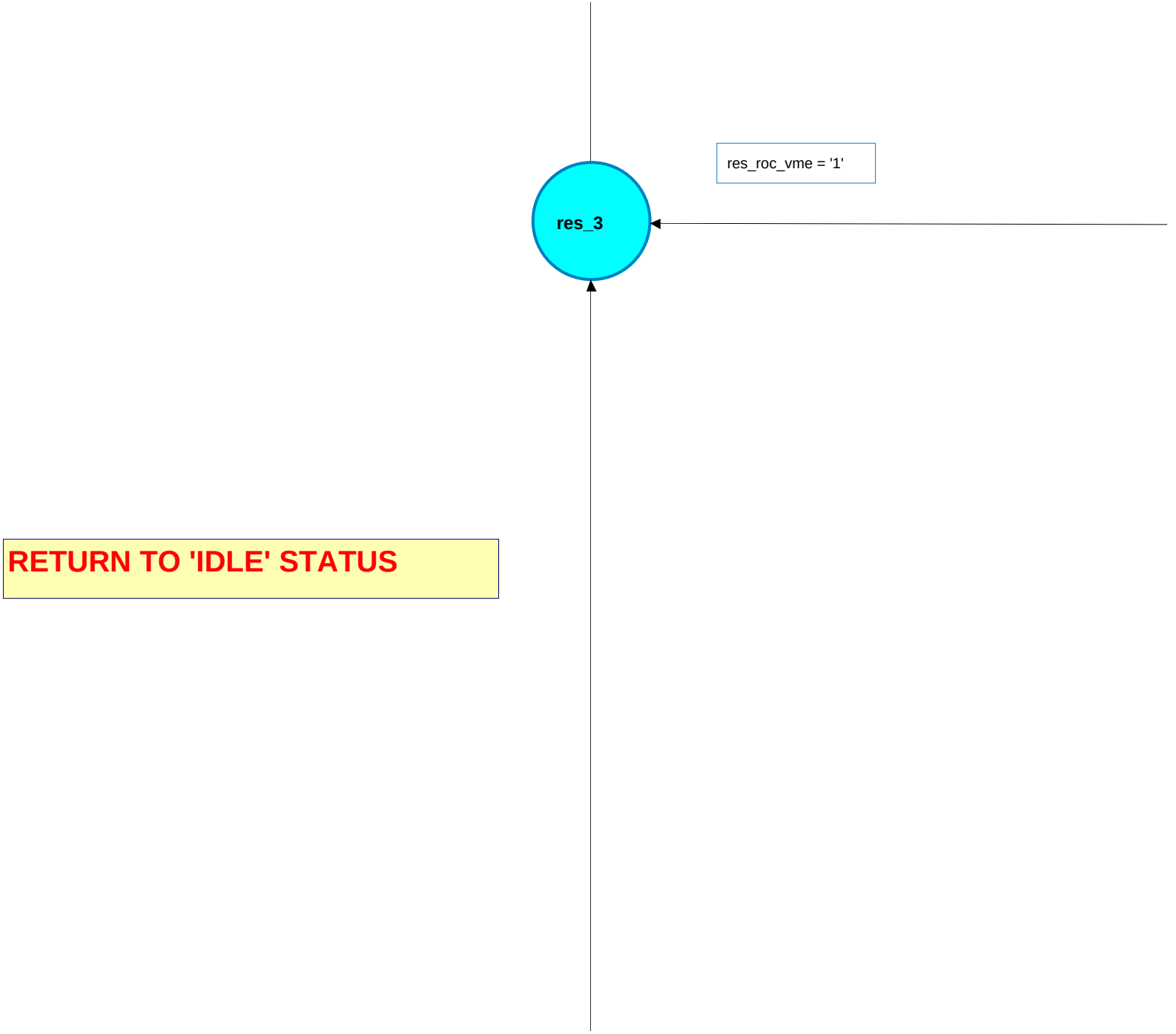

CHAN_0 = FDL
3bx ==> 3x7=21
5bx ==> 5x7=35
CHANNEL 0 data
chan_0
wait_sl_0
wait 511
wait_0
wait 20
wait3bx_0
wait 34
wait5bx_0
wait 34
err5bx_0
wait 20
err3bx_0
end_ch_0
sl_down = '1' OR sl_full = '1'
sm_status_cld <= X"1000";
sl_down = '0' AND sl_full = '0'
sm_status_cld <= X"0000";
sm_status_cld <= X"3000";
fifo_empty(0) = '1'
sm_status_cld <= X"2000";
fifo_empty(0) = '0'
sm_status_cld <= X"0000";
sm_status_cld <= X"2000";
wait_for_ch_cld(0) <= '1' ;
sm_status_cld <= X"1000";
wait_for_slink_cld <= '1' ;
TIMEOUT AND (length(0)= '0')
apply_err_cld <= '1' ;
TIMEOUT AND (length(0)= '1')
apply_err_cld <= '1' ;
fifo_empty(0) = '0' AND evnr_ref_ge_ch(0) = '1' AND length(0) = '1'
rd_fifo_cld(0) <= '1' ;
fifo_empty(0) = '0' AND evnr_ref_ge_ch(0) = '1' AND length(0) = '0'
rd_fifo_cld(0) <= '1' ;
(fifo_full(0) = '1' OR evnr_ref_lt_ch (0) = '1') AND (length(0) = '0')
apply_err_cld <= '1' ;
(fifo_full(0) = '1' OR evnr_ref_lt_ch (0) = '1') AND (length(0) = '1')
apply_err_cld <= '1' ;
active(0) = '0'
rd_fifo_cld(0) <= '1' ;
rd_fifo_cld(0) <= '1' ;
apply_err_cld <= '1' ;
apply_err_cld <= '1' ;
TIMEOUT
decr_ch_cntr_cld(0) <= '1' ;
TIMEOUT
decr_ch_cntr_cld(0) <= '1' ;
TIMEOUT
incr_evstatus_cld <= '1' ;
TIMEOUT
incr_evstatus_cld <= '1' ;
In case of an ERROR send error data.

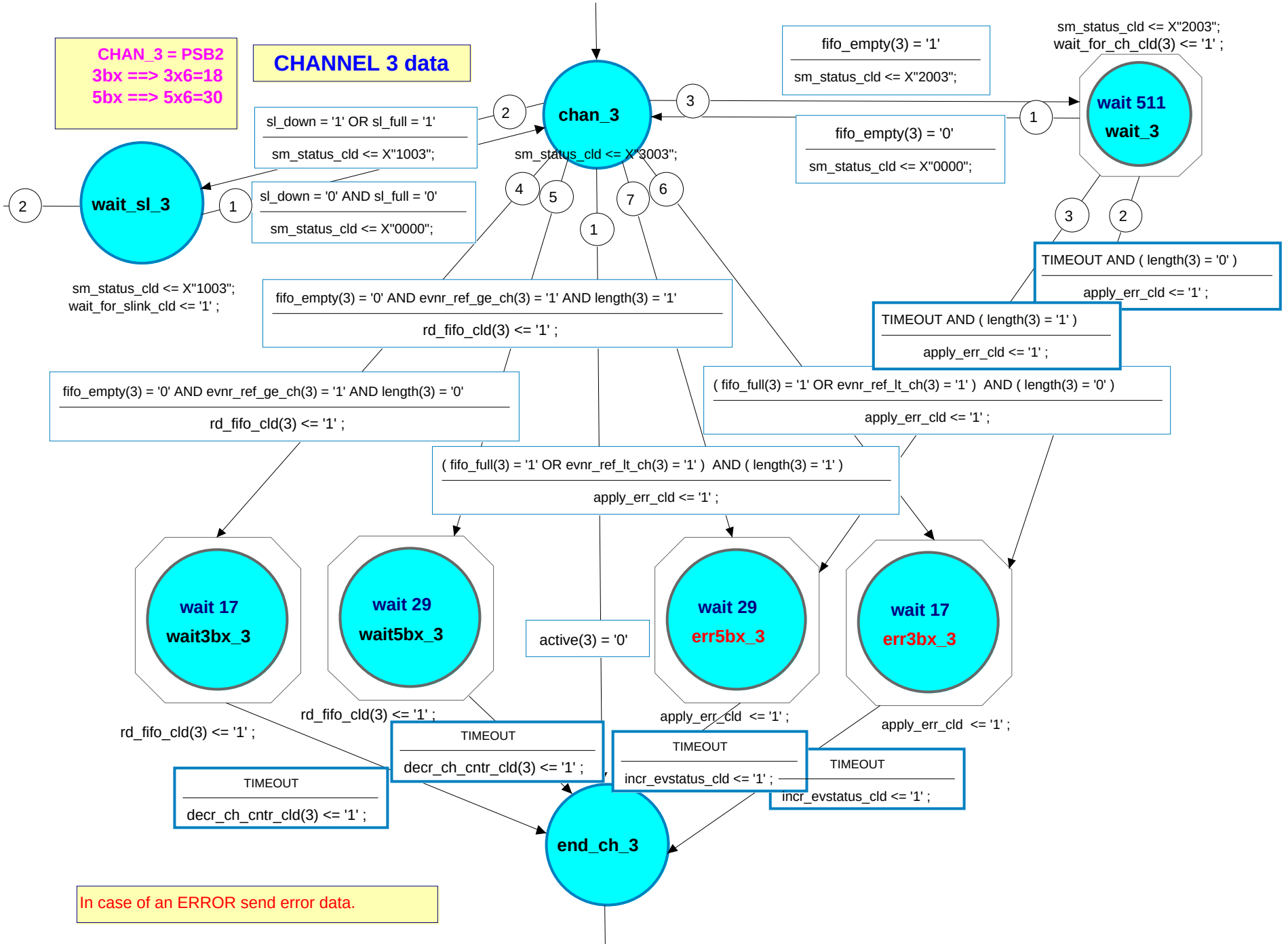

res_roc_vme = '1'
res_1
RETURN TO 'IDLE' STATUS

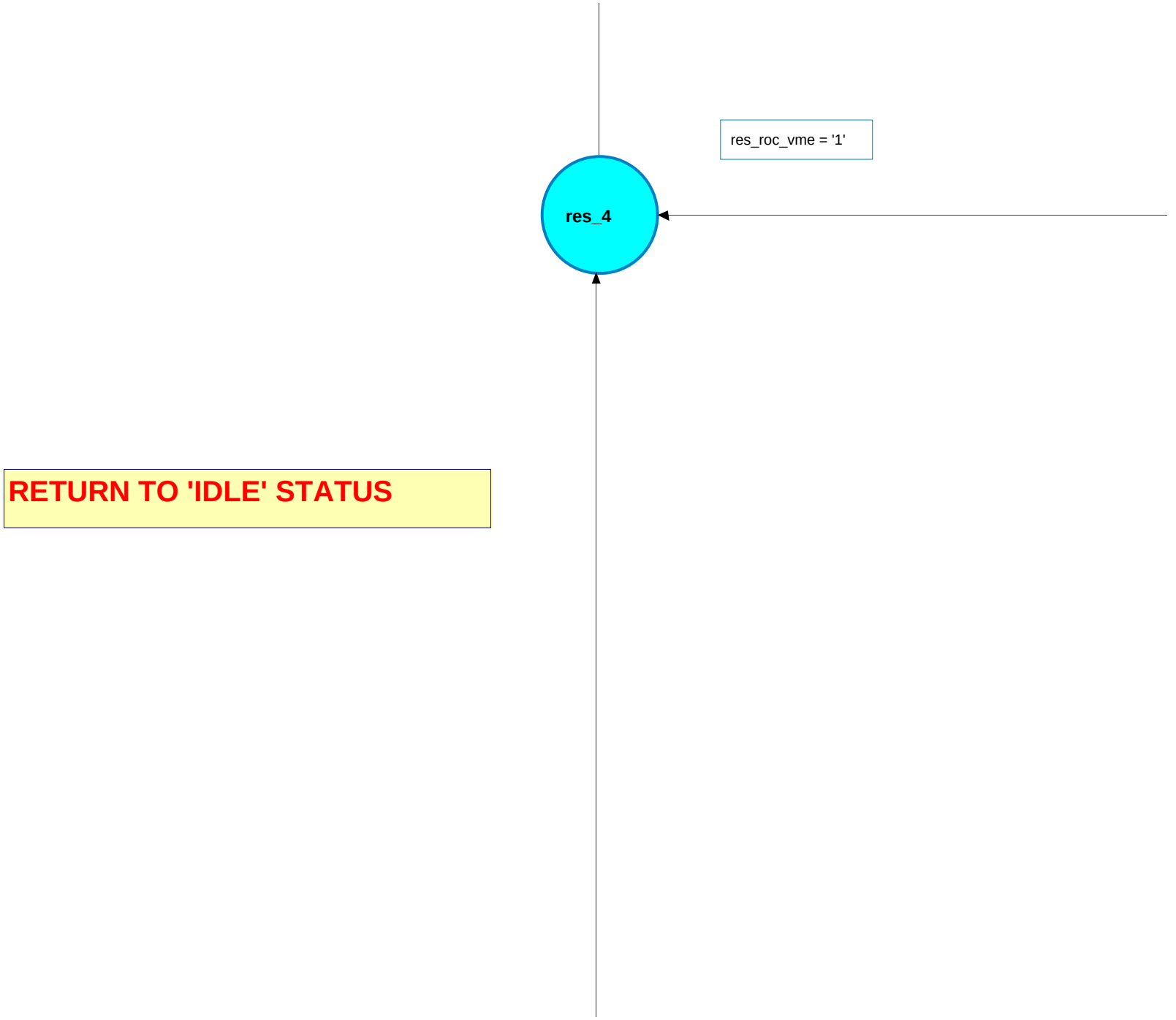

CHAN_1 = PSB0 TechTrig
3bx ==> 3x6=18
5bx ==> 5x6=30
CHANNEL 1 data
chan_1
wait_sl_1
wait 511
wait_1
wait 17
wait3bx_1
wait 29
wait5bx_1
wait 29
err5bx_1
wait 17
err3bx_1
end_ch_1
fifo_empty(1) = '1'
sm_status_cld <= X"2001";
sm_status_cld <= X"2001";
wait_for_ch_cld(1) <= '1' ;
fifo_empty(1) = '0'
sm_status_cld <= X"0000";
sl_down = '1' OR sl_full = '1'
sm_status_cld <= X"1001";
sl_down = '0' AND sl_full = '0'
sm_status_cld <= X"0000";
sm_status_cld <= X"3001";
sm_status_cld <= X"1001";
wait_for_slink_cld <= '1' ;
TIMEOUT AND (length(1) = '0')
apply_err_cld <= '1' ;
TIMEOUT AND (length(1) = '1')
apply_err_cld <= '1' ;
fifo_empty(1) = '0' AND evnr_ref_ge_ch(1) = '1' AND length(1) = '1'
rd_fifo_cld(1) <= '1' ;
fifo_empty(1) = '0' AND evnr_ref_ge_ch(1) = '1' AND length(1) = '0'
rd_fifo_cld(1) <= '1' ;
(fifo_full(1) = '1' OR evnr_ref_lt_ch(1) = '1') AND (length(1) = '0')
apply_err_cld <= '1' ;
(fifo_full(1) = '1' OR evnr_ref_lt_ch(1) = '1') AND (length(1) = '1')
apply_err_cld <= '1' ;
active(1) = '0'
rd_fifo_cld(1) <= '1' ;
rd_fifo_cld(1) <= '1' ;
apply_err_cld <= '1' ;
apply_err_cld <= '1' ;
TIMEOUT
decr_ch_cntr_cld(1) <= '1' ;
TIMEOUT
decr_ch_cntr_cld(1) <= '1' ;
TIMEOUT
incr_evstatus_cld <= '1' ;
TIMEOUT
incr_evstatus_cld <= '1' ;
In case of an ERROR send error data.

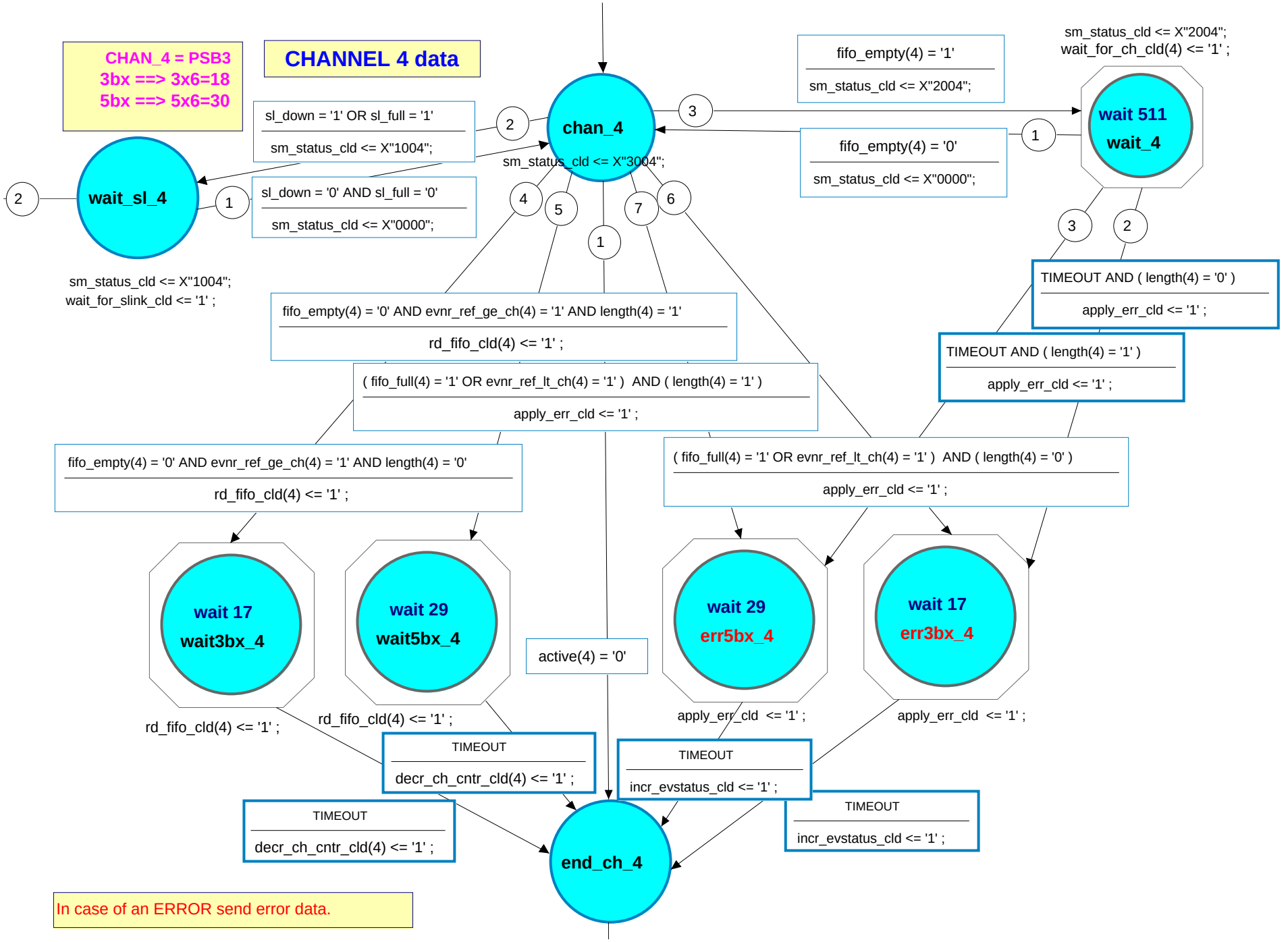

res_roc_vme = '1'
res_2
RETURN TO 'IDLE' STATUS

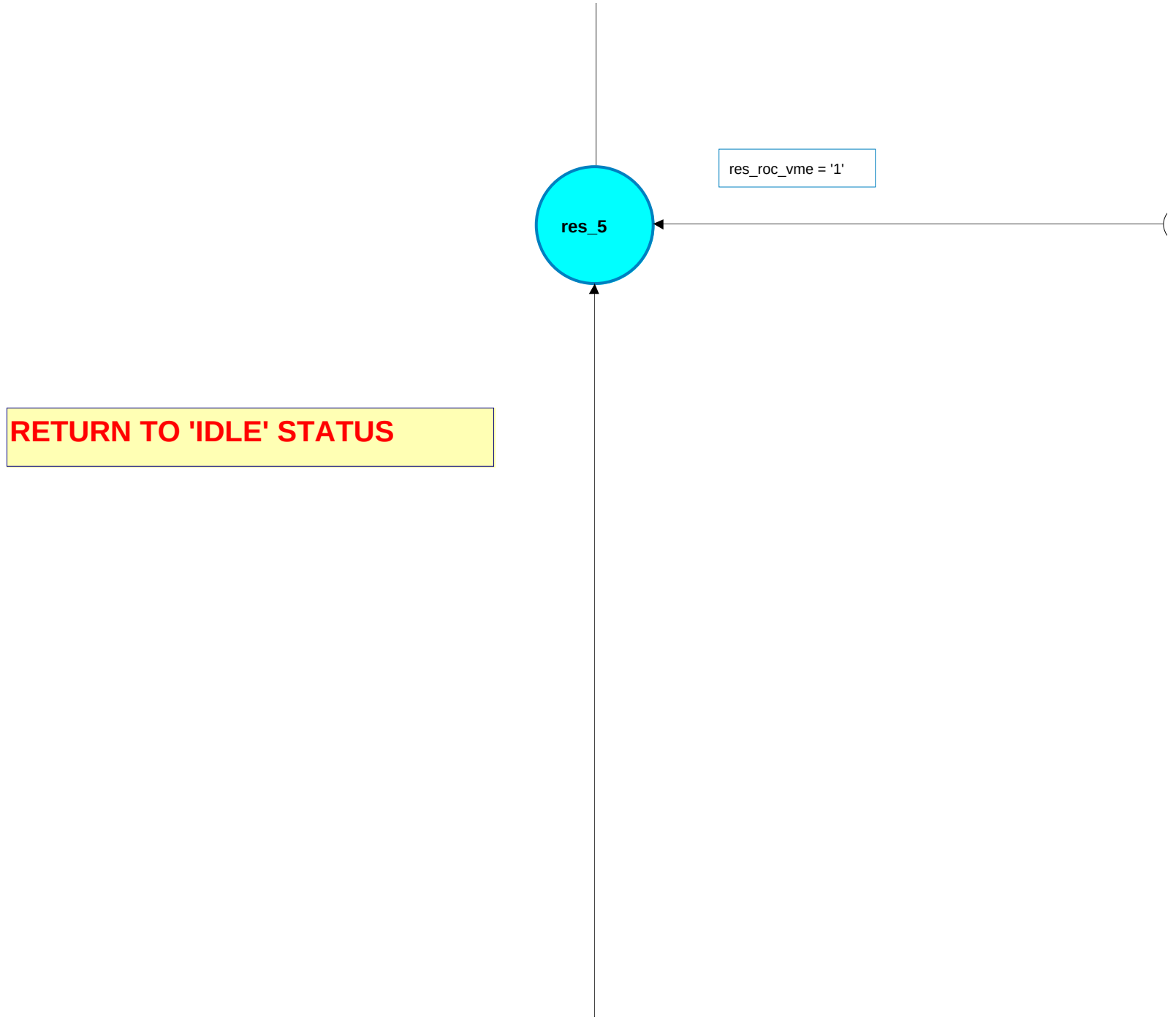

CHAN_2 = PSB1
3bx ==> 3x6=18
5bx ==> 5x6=30
CHANNEL 2 data
chan_2
wait_sl_2
wait 511
wait_2
wait 17
wait3bx_2
wait 29
wait5bx_2
wait 29
err5bx_2
wait 17
err3bx_2
end_ch_2
fifo_empty(2) = '1'
sm_status_cld <= X"2002";
sm_status_cld <= X"2002";
wait_for_ch_cld(2) <= '1' ;
fifo_empty(2) = '0'
sm_status_cld <= X"0000";
sl_down = '1' OR sl_full = '1'
sm_status_cld <= X"1002";
sm_status_cld <= X"3002";
sl_down = '0' AND sl_full = '0'
sm_status_cld <= X"0000";
sm_status_cld <= X"1002";
wait_for_slink_cld <= '1' ;
TIMEOUT AND (length(2) = '0')
apply_err_cld <= '1' ;
TIMEOUT AND (length(2) = '1')
apply_err_cld <= '1' ;
fifo_empty(2) = '0' AND evnr_ref_ge_ch(2) = '1' AND length(2) = '1'
rd_fifo_cld(2) <= '1' ;
fifo_empty(2) = '0' AND evnr_ref_ge_ch(2) = '1' AND length(2) = '0'
rd_fifo_cld(2) <= '1' ;
(fifo_full(2) = '1' OR evnr_ref_lt_ch(2) = '1') AND (length(2) = '0')
apply_err_cld <= '1' ;
(fifo_full(2) = '1' OR evnr_ref_lt_ch(2) = '1') AND (length(2) = '1')
apply_err_cld <= '1' ;
active(2) = '0'
rd_fifo_cld(2) <= '1' ;
rd_fifo_cld(2) <= '1' ;
apply_err_cld <= '1' ;
apply_err_cld <= '1' ;
TIMEOUT
decr_ch_cntr_cld(2) <= '1' ;
TIMEOUT
incr_evstatus_cld <= '1' ;
TIMEOUT
incr_evstatus_cld <= '1' ;
TIMEOUT
decr_ch_cntr_cld(2) <= '1' ;
In case of an ERROR send error data.

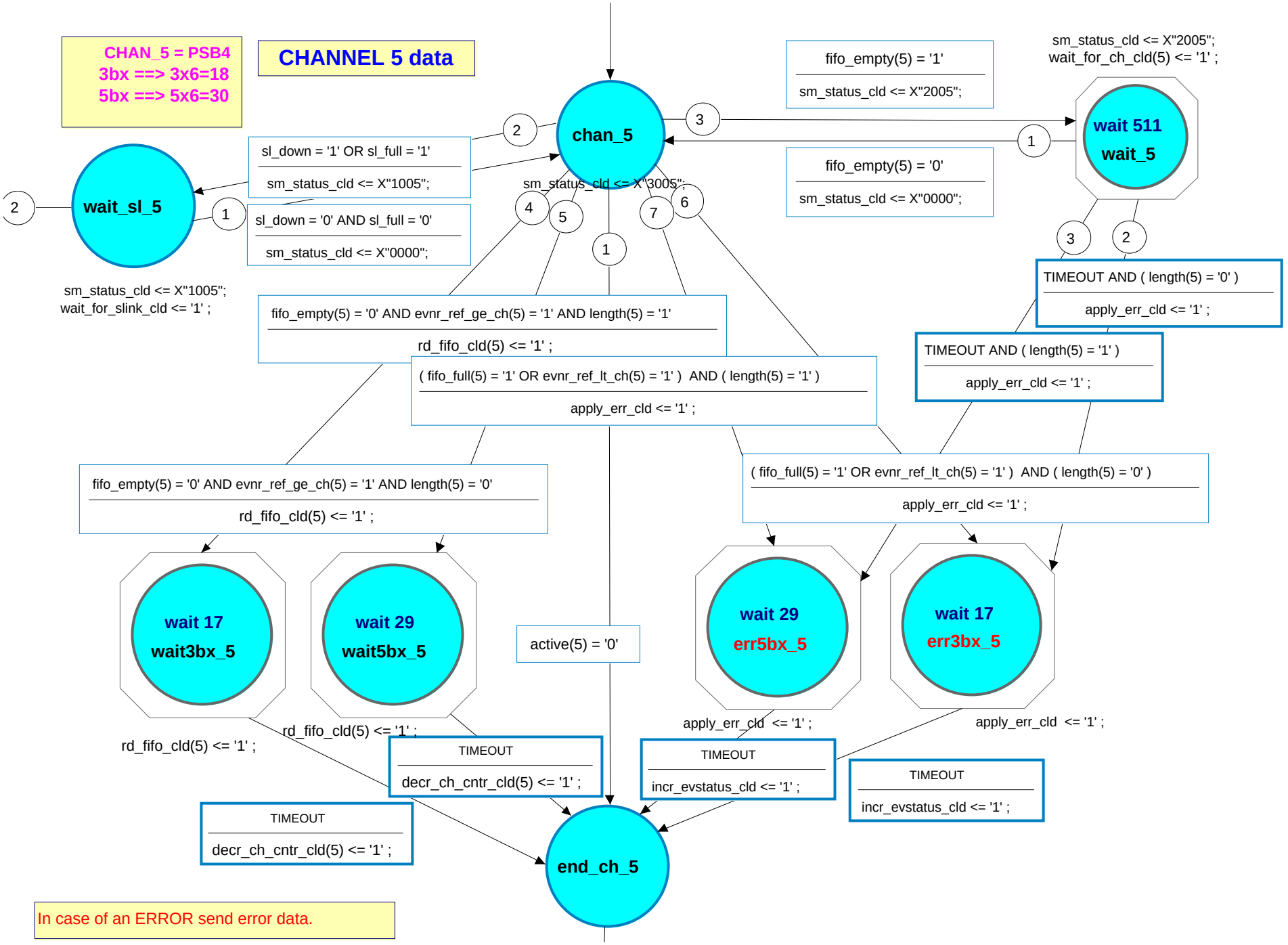

res_roc_vme = '1'
res_3
RETURN TO 'IDLE' STATUS

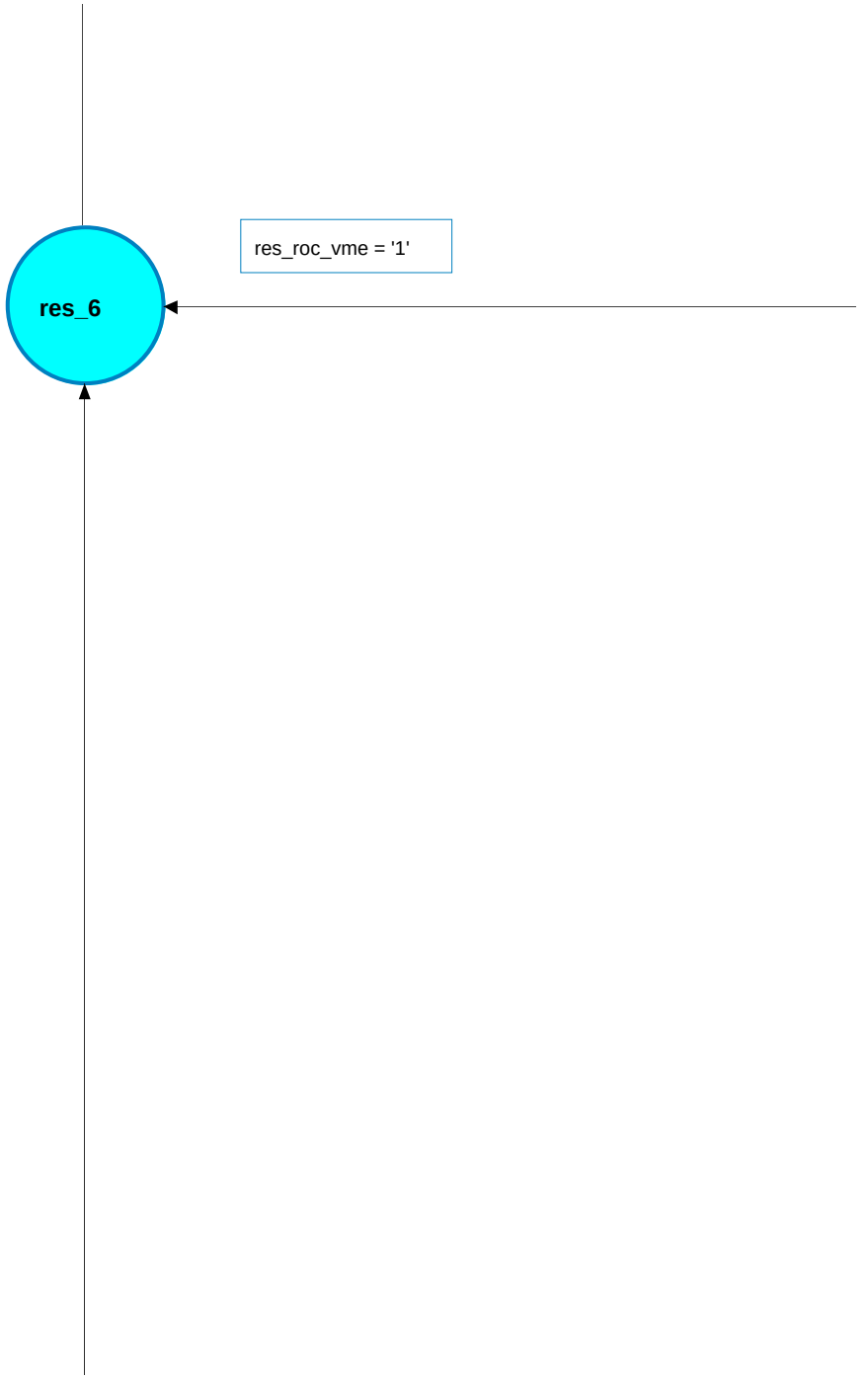

CHAN_3 = PSB2
3bx ==> 3x6=18
5bx ==> 5x6=30
CHANNEL 3 data
chan_3
wait_sl_3
wait 511
wait_3
sl_down = '1' OR sl_full = '1'
sm_status_cld <= X"1003";
sl_down = '0' AND sl_full = '0'
sm_status_cld <= X"0000";
sm_status_cld <= X"3003";
fifo_empty(3) = '1'
sm_status_cld <= X"2003";
fifo_empty(3) = '0'
sm_status_cld <= X"0000";
sm_status_cld <= X"2003";
wait_for_ch_cld(3) <= '1' ;
sm_status_cld <= X"1003";
wait_for_slink_cld <= '1' ;
fifo_empty(3) = '0' AND evnr_ref_ge_ch(3) = '1' AND length(3) = '1'
rd_fifo_cld(3) <= '1' ;
TIMEOUT AND (length(3) = '0')
apply_err_cld <= '1' ;
TIMEOUT AND (length(3) = '1')
apply_err_cld <= '1' ;
fifo_empty(3) = '0' AND evnr_ref_ge_ch(3) = '1' AND length(3) = '0'
rd_fifo_cld(3) <= '1' ;
(fifo_full(3) = '1' OR evnr_ref_lt_ch(3) = '1') AND (length(3) = '0')
apply_err_cld <= '1' ;
(fifo_full(3) = '1' OR evnr_ref_lt_ch(3) = '1') AND (length(3) = '1')
apply_err_cld <= '1' ;
wait 17
wait3bx_3
wait 29
wait5bx_3
active(3) = '0'
wait 29
err5bx_3
wait 17
err3bx_3
rd_fifo_cld(3) <= '1' ;
rd_fifo_cld(3) <= '1' ;
apply_err_cld <= '1' ;
apply_err_cld <= '1' ;
TIMEOUT
decr_ch_cntr_cld(3) <= '1' ;
TIMEOUT
incr_evstatus_cld <= '1' ;
TIMEOUT
incr_evstatus_cld <= '1' ;
TIMEOUT
decr_ch_cntr_cld(3) <= '1' ;
end_ch_3
In case of an ERROR send error data.

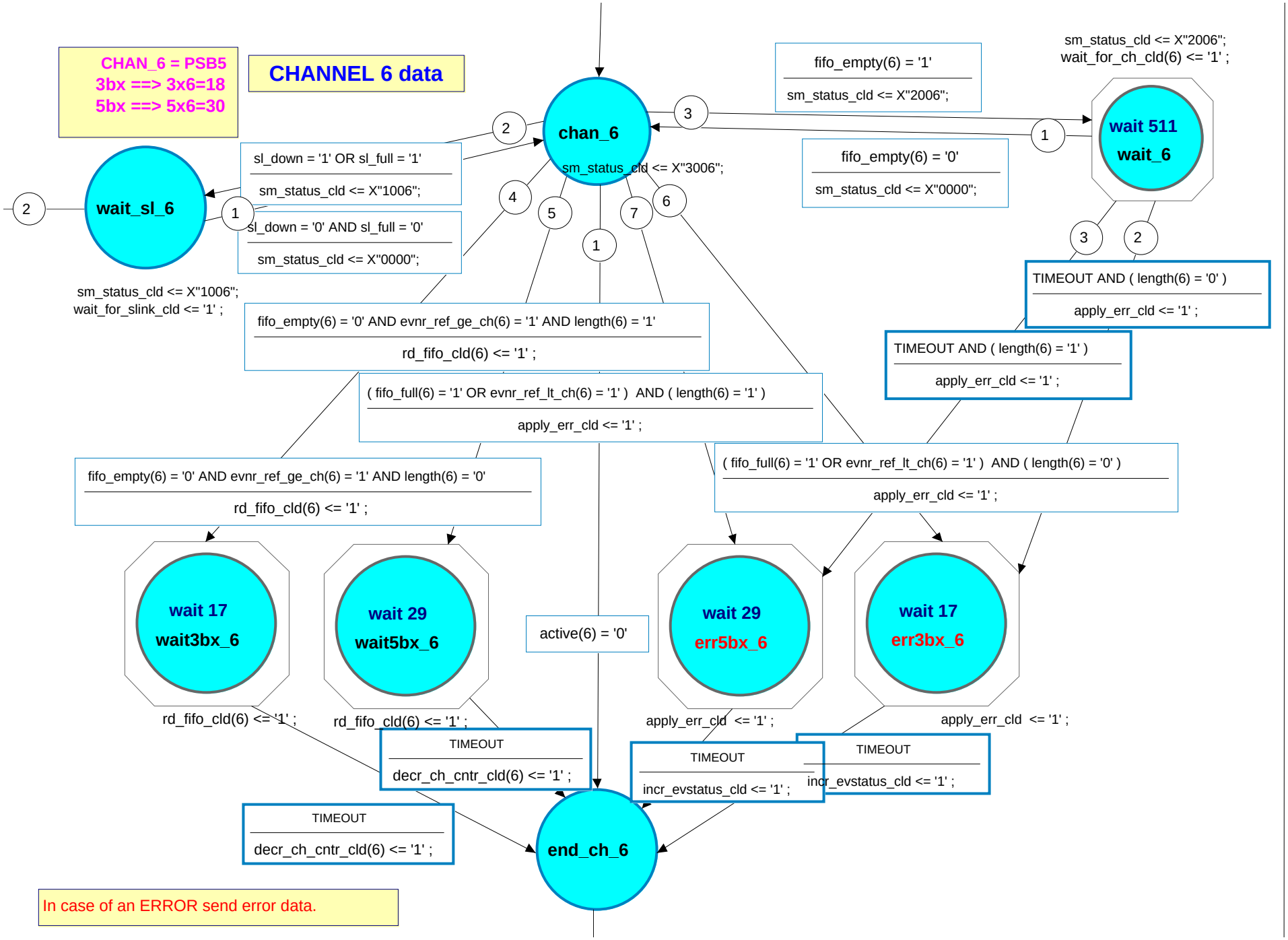

res_roc_vme = '1'
res_4
RETURN TO 'IDLE' STATUS

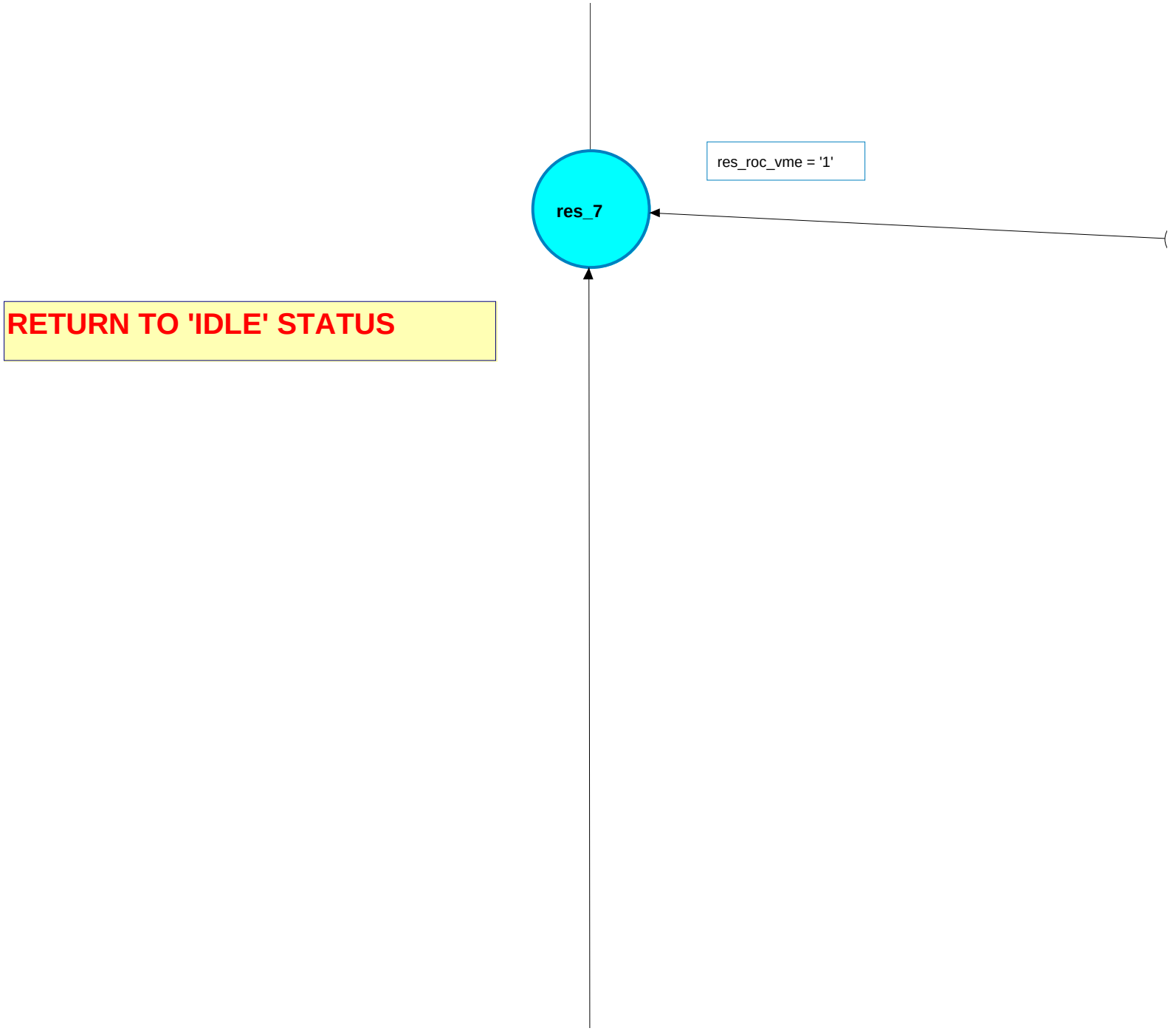

CHAN_4 = PSB3
3bx ==> 3x6=18
5bx ==> 5x6=30
CHANNEL 4 data
chan_4
sm_status_cld <= X"3004";
sl_down = '1' OR sl_full = '1'
sm_status_cld <= X"1004";
sl_down = '0' AND sl_full = '0'
sm_status_cld <= X"0000";
wait_sl_4
sm_status_cld <= X"1004";
wait_for_slink_cld <= '1' ;
fifo_empty(4) = '1'
sm_status_cld <= X"2004";
fifo_empty(4) = '0'
sm_status_cld <= X"0000";
sm_status_cld <= X"2004";
wait_for_ch_cld(4) <= '1' ;
wait 511
wait_4
TIMEOUT AND (length(4) = '0')
apply_err_cld <= '1' ;
TIMEOUT AND (length(4) = '1')
apply_err_cld <= '1' ;
fifo_empty(4) = '0' AND evnr_ref_ge_ch(4) = '1' AND length(4) = '1'
rd_fifo_cld(4) <= '1' ;
(fifo_full(4) = '1' OR evnr_ref_lt_ch(4) = '1') AND (length(4) = '1')
apply_err_cld <= '1' ;
fifo_empty(4) = '0' AND evnr_ref_ge_ch(4) = '1' AND length(4) = '0'
rd_fifo_cld(4) <= '1' ;
(fifo_full(4) = '1' OR evnr_ref_lt_ch(4) = '1') AND (length(4) = '0')
apply_err_cld <= '1' ;
wait 17
wait3bx_4
wait 29
wait5bx_4
wait 29
err5bx_4
wait 17
err3bx_4
rd_fifo_cld(4) <= '1' ;
rd_fifo_cld(4) <= '1' ;
apply_err_cld <= '1' ;
apply_err_cld <= '1' ;
active(4) = '0'
TIMEOUT
decr_ch_cntr_cld(4) <= '1' ;
TIMEOUT
incr_evstatus_cld <= '1' ;
TIMEOUT
decr_ch_cntr_cld(4) <= '1' ;
TIMEOUT
incr_evstatus_cld <= '1' ;
end_ch_4
In case of an ERROR send error data.

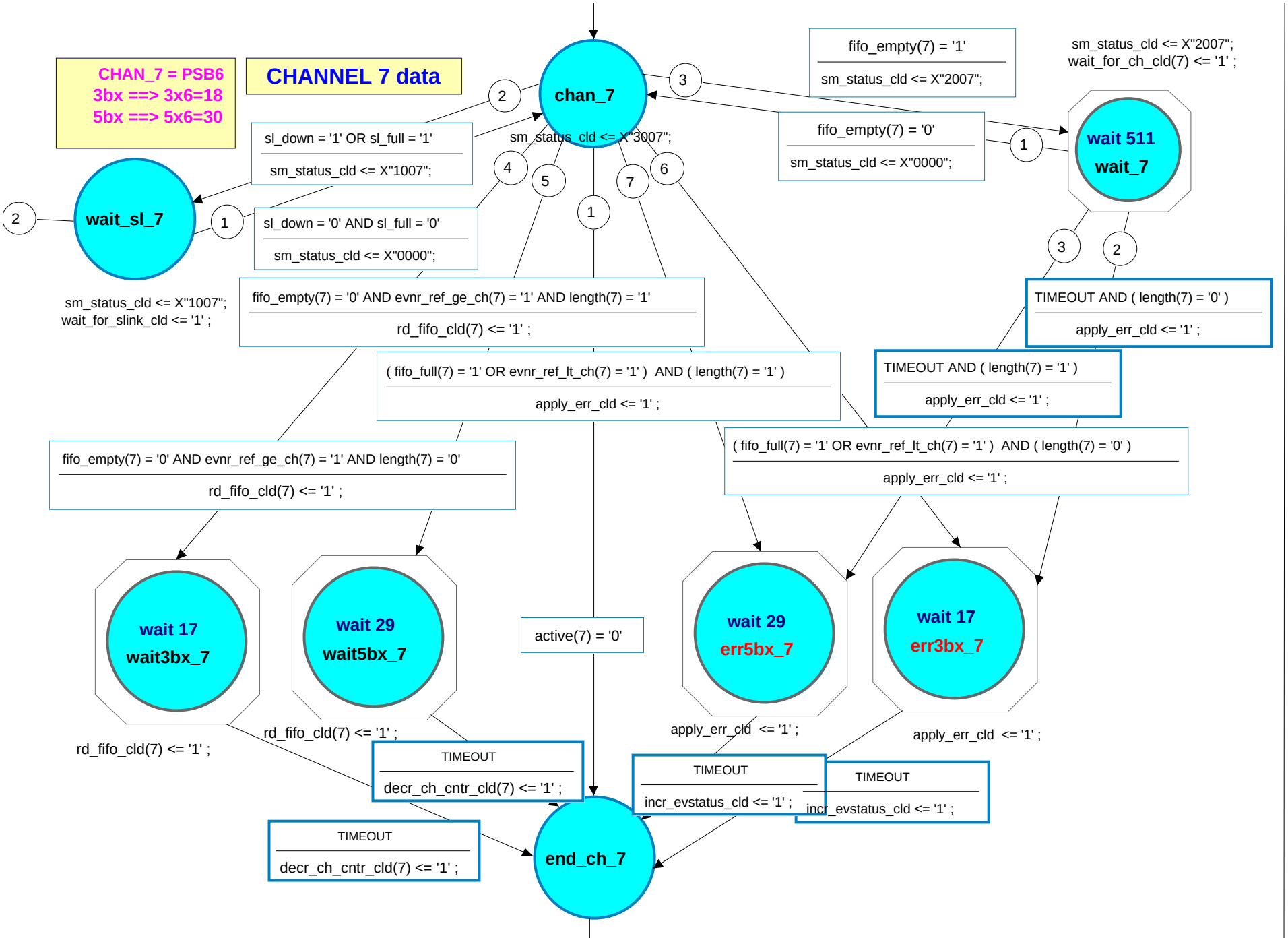

res_roc_vme = '1'
res_5
RETURN TO 'IDLE' STATUS


CHAN_5 = PSB4
3bx ==> 3x6=18
5bx ==> 5x6=30
CHANNEL 5 data
chan_5
wait_sl_5
wait 511
wait_5
wait 17
wait3bx_5
wait 29
wait5bx_5
wait 29
err5bx_5
wait 17
err3bx_5
end_ch_5
fifo_empty(5) = '1'
sm_status_cld <= X"2005";
sm_status_cld <= X"2005";
wait_for_ch_cld(5) <= '1' ;
fifo_empty(5) = '0'
sm_status_cld <= X"0000";
sl_down = '1' OR sl_full = '1'
sm_status_cld <= X"1005";
sl_down = '0' AND sl_full = '0'
sm_status_cld <= X"0000";
sm_status_cld <= X"3005";
sm_status_cld <= X"1005";
wait_for_slink_cld <= '1' ;
TIMEOUT AND (length(5) = '0')
apply_err_cld <= '1' ;
TIMEOUT AND (length(5) = '1')
apply_err_cld <= '1' ;
fifo_empty(5) = '0' AND evnr_ref_ge_ch(5) = '1' AND length(5) = '1'
rd_fifo_cld(5) <= '1' ;
(fifo_full(5) = '1' OR evnr_ref_lt_ch(5) = '1') AND (length(5) = '1')
apply_err_cld <= '1' ;
fifo_empty(5) = '0' AND evnr_ref_ge_ch(5) = '1' AND length(5) = '0'
rd_fifo_cld(5) <= '1' ;
(fifo_full(5) = '1' OR evnr_ref_lt_ch(5) = '1') AND (length(5) = '0')
apply_err_cld <= '1' ;
active(5) = '0'
rd_fifo_cld(5) <= '1' ;
rd_fifo_cld(5) <= '1' ;
apply_err_cld <= '1' ;
apply_err_cld <= '1' ;
TIMEOUT
decr_ch_cntr_cld(5) <= '1' ;
TIMEOUT
incr_evstatus_cld <= '1' ;
TIMEOUT
incr_evstatus_cld <= '1' ;
TIMEOUT
decr_ch_cntr_cld(5) <= '1' ;
In case of an ERROR send error data.

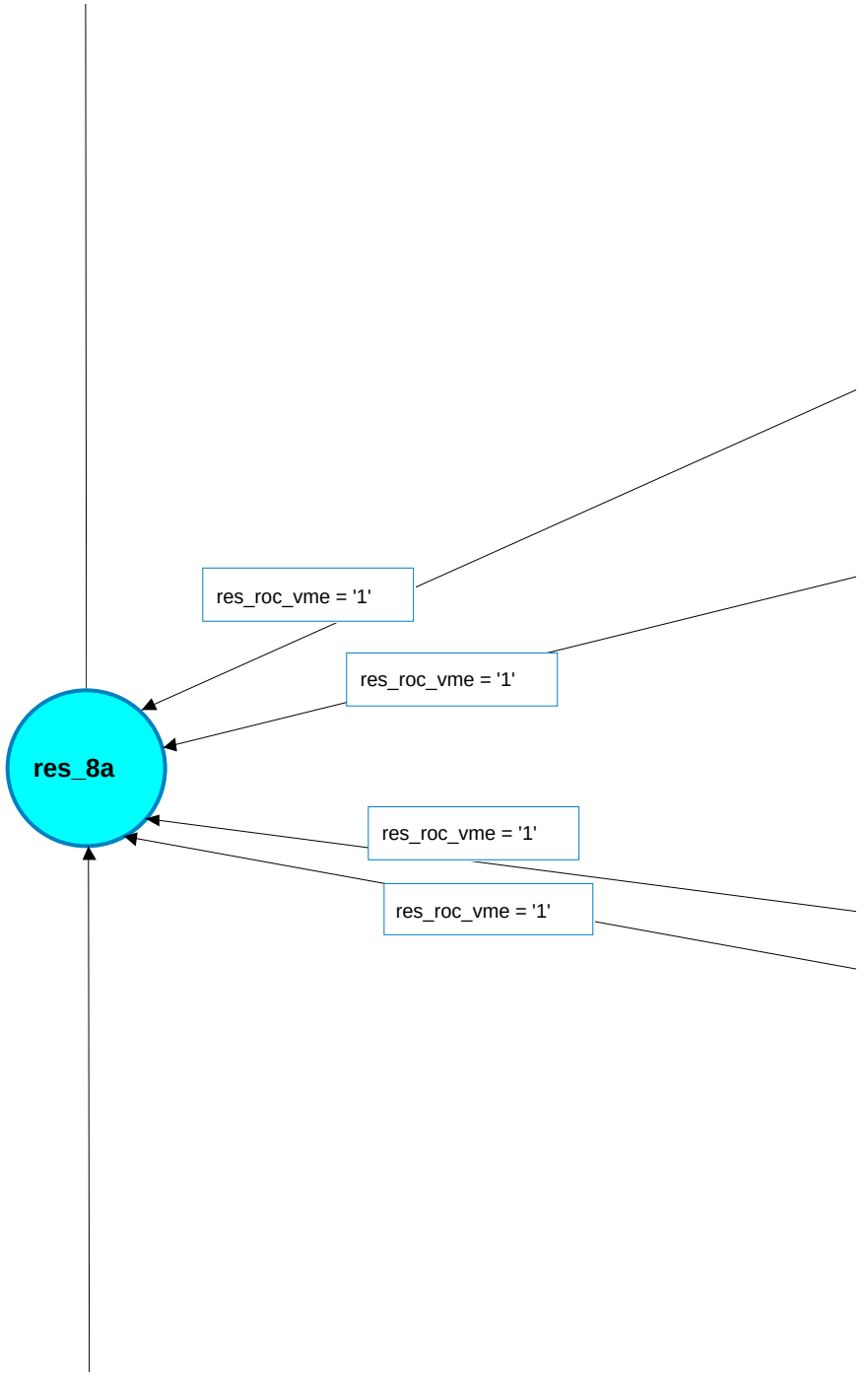

res_roc_vme = '1'
res_6
RETURN TO 'IDLE' STATUS

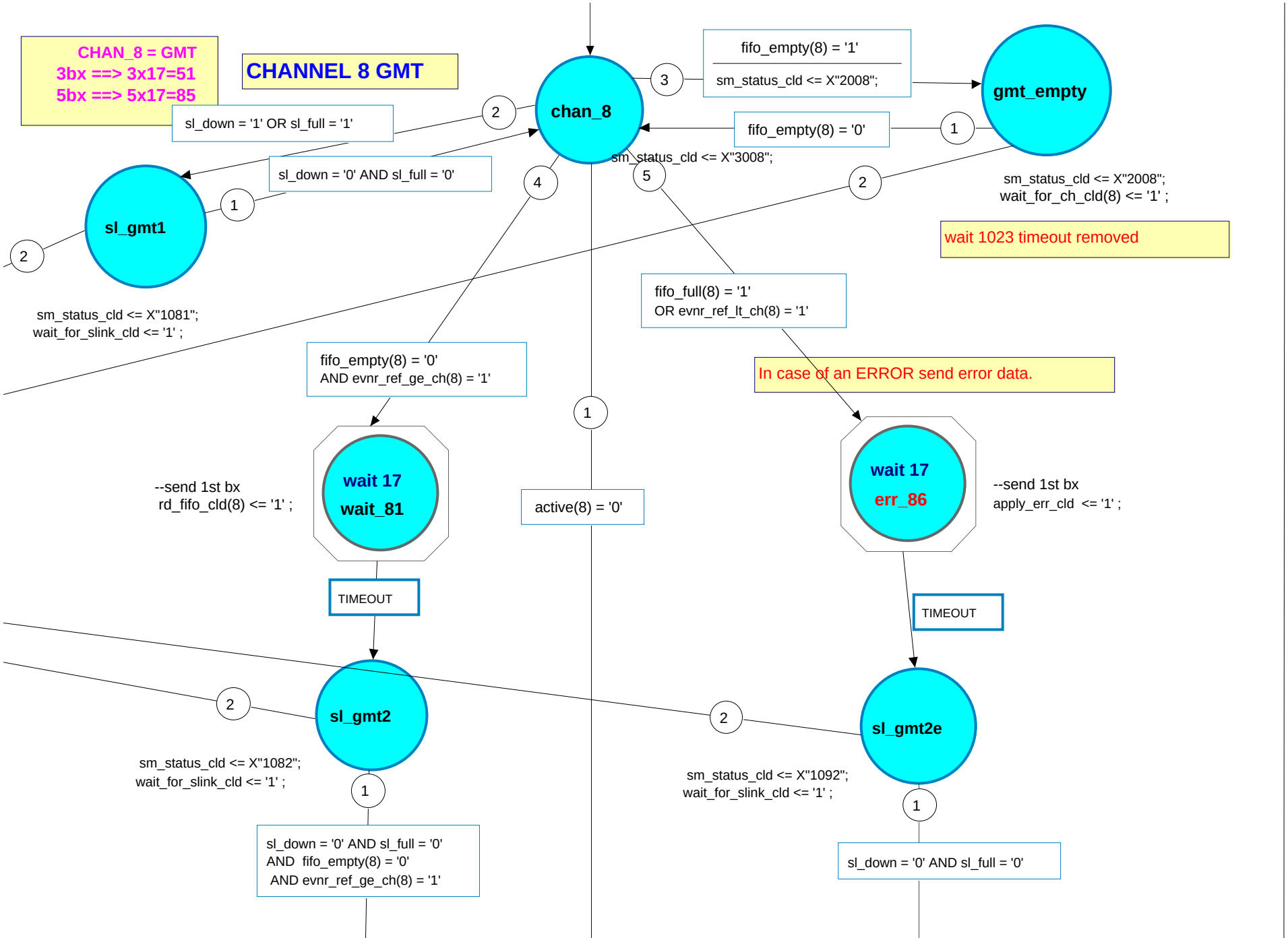

CHAN_6 = PSB5
3bx ==> 3x6=18
5bx ==> 5x6=30
CHANNEL 6 data
chan_6
wait_sl_6
wait 511
wait_6
sl_down = '1' OR sl_full = '1'
sm_status_cld <= X"1006";
sl_down = '0' AND sl_full = '0'
sm_status_cld <= X"0000";
sm_status_cld <= X"3006";
fifo_empty(6) = '1'
sm_status_cld <= X"2006";
fifo_empty(6) = '0'
sm_status_cld <= X"0000";
sm_status_cld <= X"2006";
wait_for_ch_cld(6) <= '1' ;
sm_status_cld <= X"1006";
wait_for_slink_cld <= '1' ;
TIMEOUT AND (length(6) = '0')
apply_err_cld <= '1' ;
TIMEOUT AND (length(6) = '1')
apply_err_cld <= '1' ;
fifo_empty(6) = '0' AND evnr_ref_ge_ch(6) = '1' AND length(6) = '1'
rd_fifo_cld(6) <= '1' ;
(fifo_full(6) = '1' OR evnr_ref_lt_ch(6) = '1') AND (length(6) = '1')
apply_err_cld <= '1' ;
fifo_empty(6) = '0' AND evnr_ref_ge_ch(6) = '1' AND length(6) = '0'
rd_fifo_cld(6) <= '1' ;
(fifo_full(6) = '1' OR evnr_ref_lt_ch(6) = '1') AND (length(6) = '0')
apply_err_cld <= '1' ;
wait 17
wait3bx_6
wait 29
wait5bx_6
active(6) = '0'
wait 29
err5bx_6
wait 17
err3bx_6
rd_fifo_cld(6) <= '1' ;
rd_fifo_cld(6) <= '1' ;
apply_err_cld <= '1' ;
apply_err_cld <= '1' ;
TIMEOUT
decr_ch_cntr_cld(6) <= '1' ;
TIMEOUT
incr_evstatus_cld <= '1' ;
TIMEOUT
incr_evstatus_cld <= '1' ;
TIMEOUT
decr_ch_cntr_cld(6) <= '1' ;
end_ch_6
In case of an ERROR send error data.


res_roc_vme = '1'
res_7
RETURN TO 'IDLE' STATUS

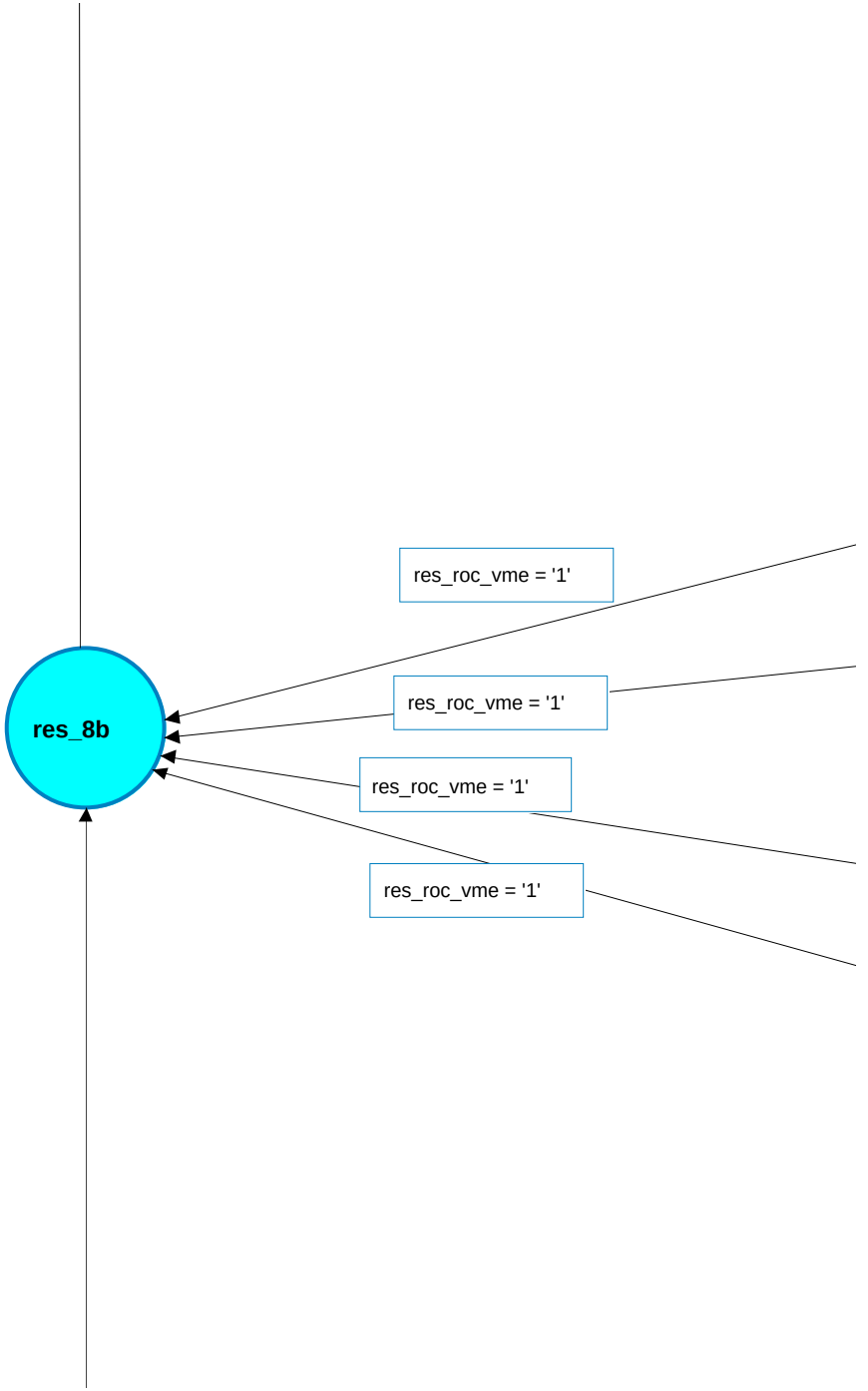

CHAN_7 = PSB6
3bx ==> 3x6=18
5bx ==> 5x6=30
CHANNEL 7 data
chan_7
wait_sl_7
wait 511
wait_7
wait 17
wait3bx_7
wait 29
wait5bx_7
wait 29
err5bx_7
wait 17
err3bx_7
end_ch_7
fifo_empty(7) = '1'
sm_status_cld <= X"2007";
sm_status_cld <= X"2007";
wait_for_ch_cld(7) <= '1' ;
fifo_empty(7) = '0'
sm_status_cld <= X"0000";
sl_down = '1' OR sl_full = '1'
sm_status_cld <= X"1007";
sm_status_cld <= X"3007";
sl_down = '0' AND sl_full = '0'
sm_status_cld <= X"0000";
sm_status_cld <= X"1007";
wait_for_slink_cld <= '1' ;
fifo_empty(7) = '0' AND evnr_ref_ge_ch(7) = '1' AND length(7) = '1'
rd_fifo_cld(7) <= '1' ;
TIMEOUT AND (length(7) = '0')
apply_err_cld <= '1' ;
(fifo_full(7) = '1' OR evnr_ref_lt_ch(7) = '1') AND (length(7) = '1')
apply_err_cld <= '1' ;
TIMEOUT AND (length(7) = '1')
apply_err_cld <= '1' ;
fifo_empty(7) = '0' AND evnr_ref_ge_ch(7) = '1' AND length(7) = '0'
rd_fifo_cld(7) <= '1' ;
(fifo_full(7) = '1' OR evnr_ref_lt_ch(7) = '1') AND (length(7) = '0')
apply_err_cld <= '1' ;
active(7) = '0'
rd_fifo_cld(7) <= '1' ;
rd_fifo_cld(7) <= '1' ;
apply_err_cld <= '1' ;
apply_err_cld <= '1' ;
TIMEOUT
decr_ch_cntr_cld(7) <= '1' ;
TIMEOUT
incr_evstatus_cld <= '1' ;
TIMEOUT
incr_evstatus_cld <= '1' ;
TIMEOUT
decr_ch_cntr_cld(7) <= '1' ;

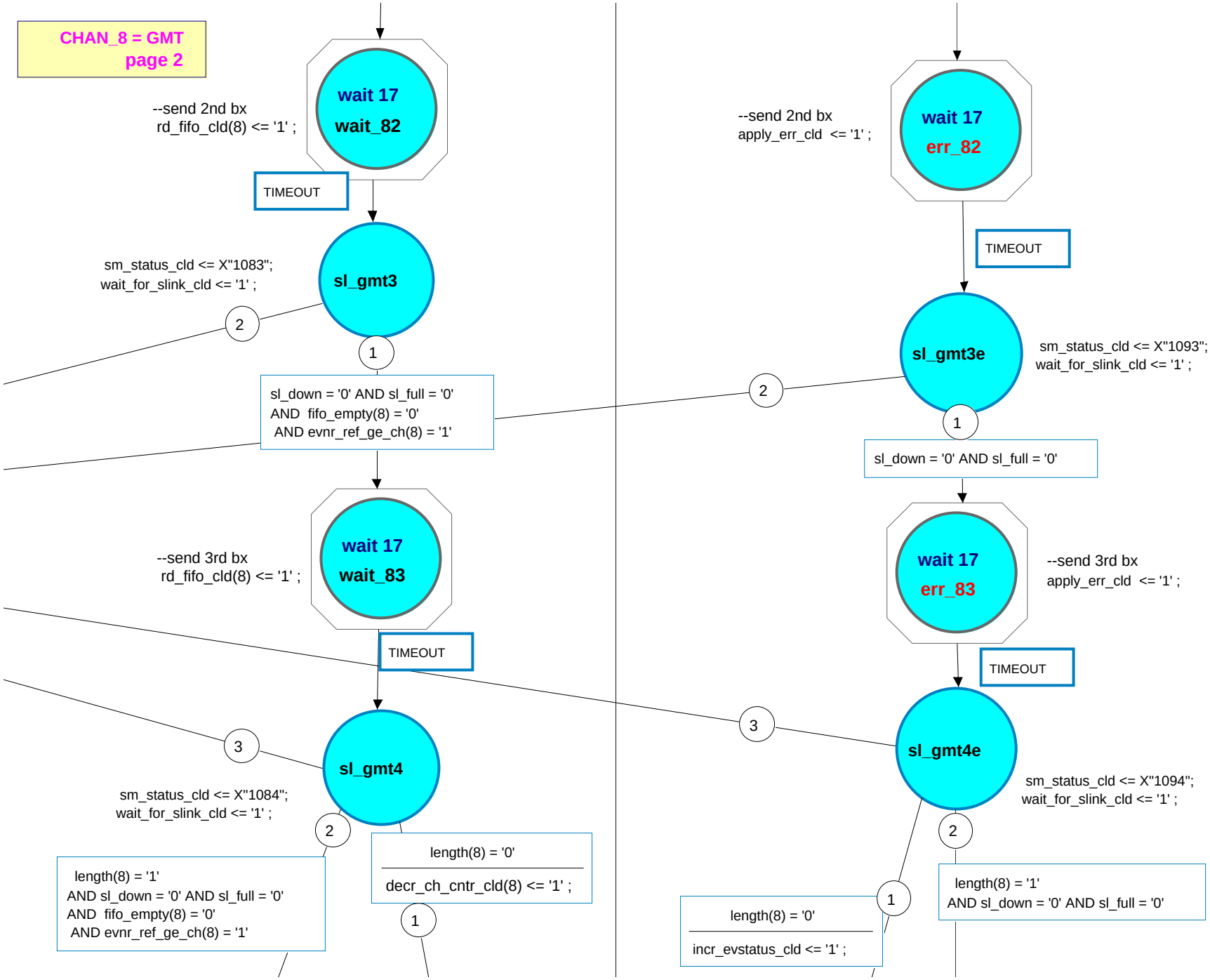
RETURN TO 'IDLE' STATUS



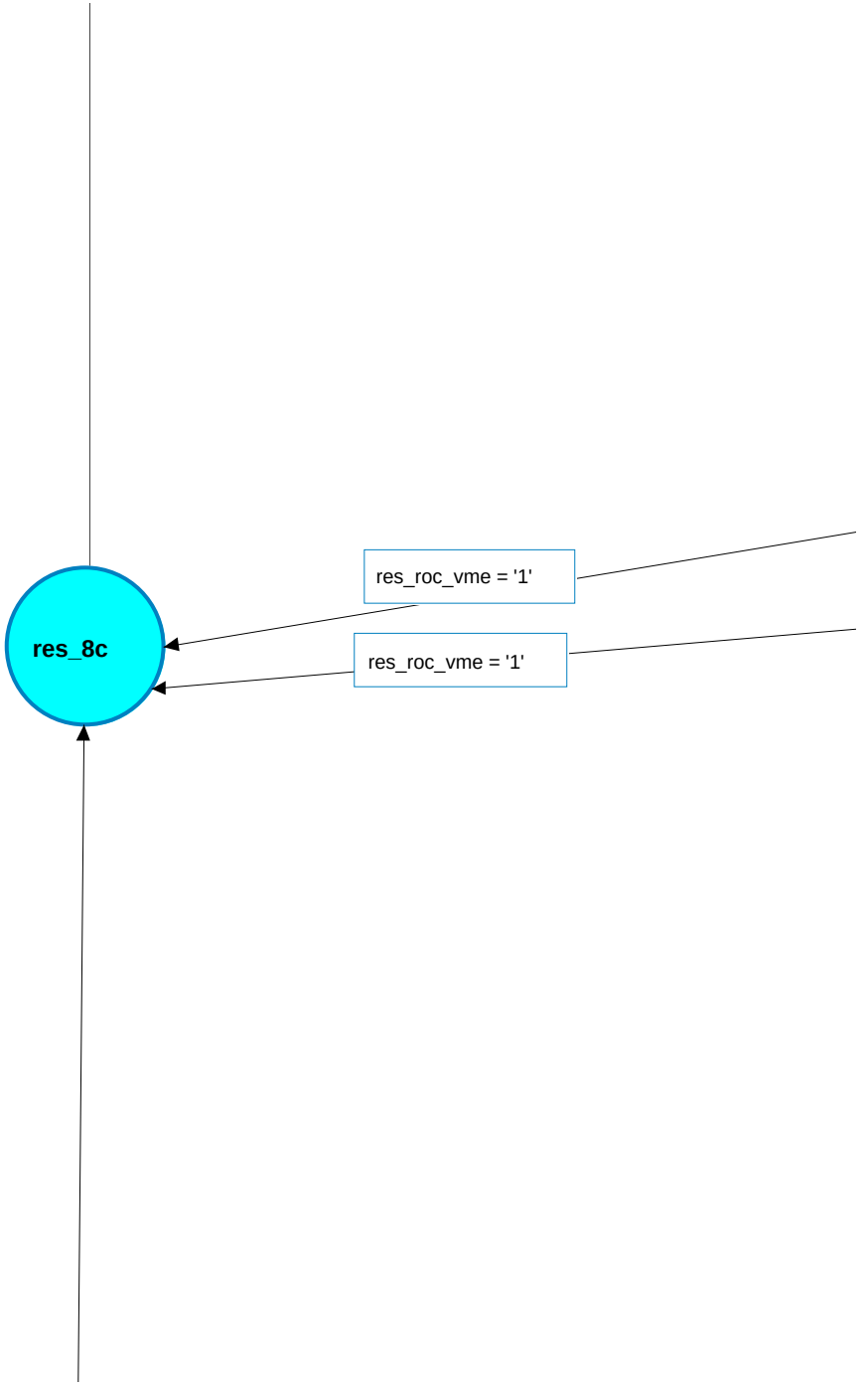

CHAN_8 = GMT
3bx ==> 3x17=51
5bx ==> 5x17=85
CHANNEL 8 GMT
chan_8
gmt_empty
sl_gmt1
sl_gmt2
sl_gmt2e
wait 17
wait_81
wait 17
err_86
fifo_empty(8) = '1'
sm_status_cld <= X"2008";
fifo_empty(8) = '0'
sm_status_cld <= X"3008";
sl_down = '1' OR sl_full = '1'
sl_down = '0' AND sl_full = '0'
sm_status_cld <= X"2008";
wait_for_ch_cld(8) <= '1' ;
wait 1023 timeout removed
sm_status_cld <= X"1081";
wait_for_slink_cld <= '1' ;
fifo_full(8) = '1'
OR evnr_ref_lt_ch(8) = '1'
fifo_empty(8) = '0'
AND evnr_ref_ge_ch(8) = '1'
In case of an ERROR send error data.
--send 1st bx
rd_fifo_cld(8) <= '1' ;
active(8) = '0'
--send 1st bx
apply_err_cld <= '1' ;
TIMEOUT
TIMEOUT
sm_status_cld <= X"1082";
wait_for_slink_cld <= '1' ;
sm_status_cld <= X"1092";
wait_for_slink_cld <= '1' ;
sl_down = '0' AND sl_full = '0'
AND fifo_empty(8) = '0'
AND evnr_ref_ge_ch(8) = '1'
sl_down = '0' AND sl_full = '0'

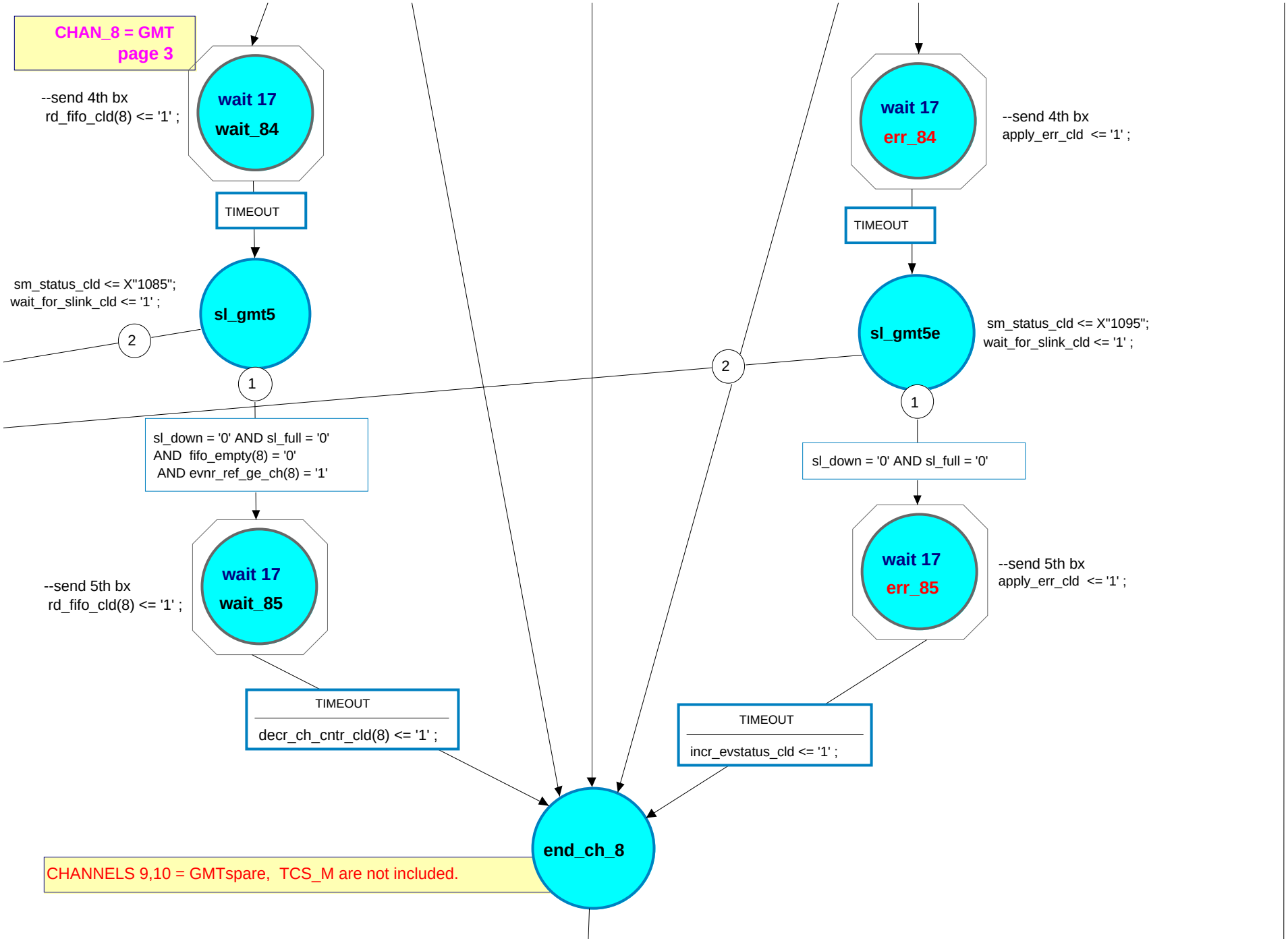
RETURN TO 'IDLE' STATUS

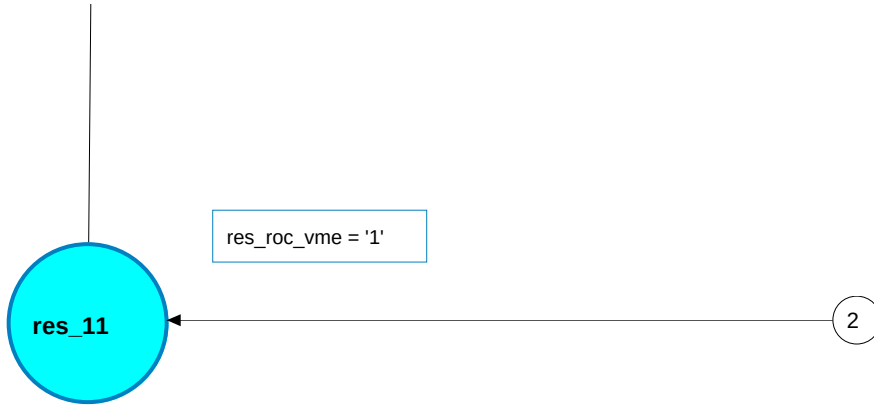


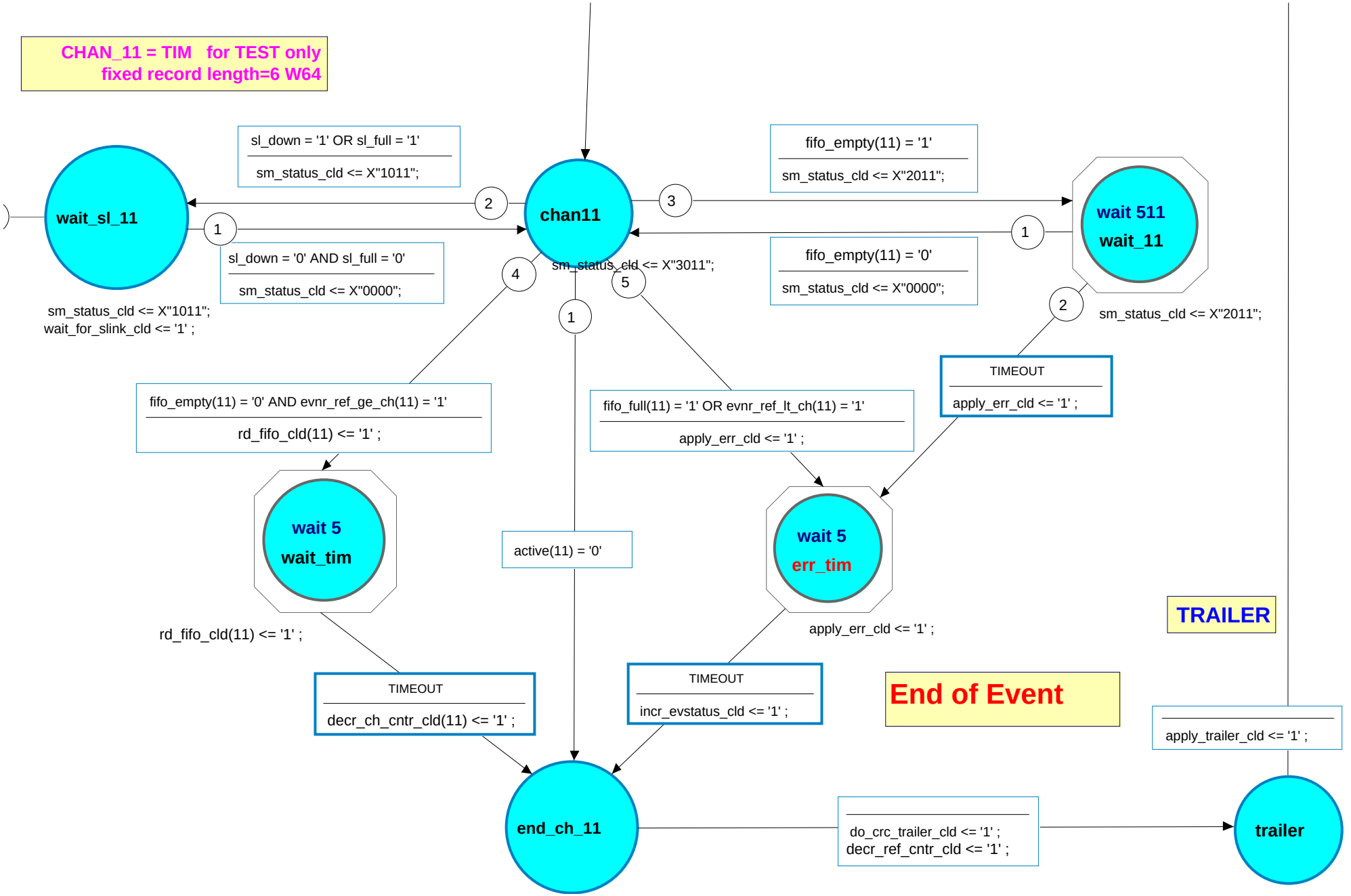

CHAN_8 = GMT
page 2
wait 17
wait_82
--send 2nd bx
rd_fifo_cld(8) <= '1' ;
TIMEOUT
sl_gmt3
sm_status_cld <= X"1083";
wait_for_slink_cld <= '1' ;
2
1
sl_down = '0' AND sl_full = '0'
AND fifo_empty(8) = '0'
AND evnr_ref_ge_ch(8) = '1'
wait 17
wait_83
--send 3rd bx
rd_fifo_cld(8) <= '1' ;
TIMEOUT
3
sl_gmt4
sm_status_cld <= X"1084";
wait_for_slink_cld <= '1' ;
2
length(8) = '1'
AND sl_down = '0' AND sl_full = '0'
AND fifo_empty(8) = '0'
AND evnr_ref_ge_ch(8) = '1'
length(8) = '0'
decr_ch_cntr_cld(8) <= '1' ;
1
wait 17
err_82
--send 2nd bx
apply_err_cld <= '1' ;
TIMEOUT
sl_gmt3e
sm_status_cld <= X"1093";
wait_for_slink_cld <= '1' ;
2
1
sl_down = '0' AND sl_full = '0'
wait 17
err_83
--send 3rd bx
apply_err_cld <= '1' ;
TIMEOUT
3
sl_gmt4e
sm_status_cld <= X"1094";
wait_for_slink_cld <= '1' ;
2
length(8) = '1'
AND sl_down = '0' AND sl_full = '0'
length(8) = '0'
incr_evstatus_cld <= '1' ;
1

RETURN TO 'IDLE' STATUS




CHAN_8 = GMT
page 3
--send 4th bx
rd_fifo_cld(8) <= '1' ;
wait 17
wait_84
TIMEOUT
sm_status_cld <= X"1085";
wait_for_slink_cld <= '1' ;
sl_gmt5
2
1
sl_down = '0' AND sl_full = '0'
AND fifo_empty(8) = '0'
AND evnr_ref_ge_ch(8) = '1'
--send 5th bx
rd_fifo_cld(8) <= '1' ;
wait 17
wait_85
TIMEOUT
decr_ch_cntr_cld(8) <= '1' ;
wait 17
err_84
--send 4th bx
apply_err_cld <= '1' ;
TIMEOUT
sl_gmt5e
sm_status_cld <= X"1095";
wait_for_slink_cld <= '1' ;
2
1
sl_down = '0' AND sl_full = '0'
wait 17
err_85
--send 5th bx
apply_err_cld <= '1' ;
TIMEOUT
incr_evstatus_cld <= '1' ;
end_ch_8
CHANNELS 9,10 = GMTspare, TCS_M are not included.


res_roc_vme = '1'
res_11
2


CHAN_11 = TIM for TEST only
fixed record length=6 W64
wait_sl_11
sl_down = '1' OR sl_full = '1'
sm_status_cld <= X"1011";
sm_status_cld <= X"1011";
wait_for_slink_cld <= '1' ;
sl_down = '0' AND sl_full = '0'
sm_status_cld <= X"0000";
chan11
sm_status_cld <= X"3011";
fifo_empty(11) = '1'
sm_status_cld <= X"2011";
fifo_empty(11) = '0'
sm_status_cld <= X"0000";
wait 511
wait_11
sm_status_cld <= X"2011";
TIMEOUT
apply_err_cld <= '1' ;
fifo_empty(11) = '0' AND evnr_ref_ge_ch(11) = '1'
rd_fifo_cld(11) <= '1' ;
fifo_full(11) = '1' OR evnr_ref_lt_ch(11) = '1'
apply_err_cld <= '1' ;
wait 5
wait_tim
rd_fifo_cld(11) <= '1' ;
active(11) = '0'
wait 5
err_tim
apply_err_cld <= '1' ;
TRAILER
TIMEOUT
decr_ch_cntr_cld(11) <= '1' ;
TIMEOUT
incr_evstatus_cld <= '1' ;
End of Event
apply_trailer_cld <= '1' ;
end_ch_11
do_crc_trailer_cld <= '1' ;
decr_ref_cntr_cld <= '1' ;
trailer