

# D0 Upgrade *Electronics*

D0Note xxxx

# The CPS Axial Trigger - Matching Clusters to Tracks

by Fred Borcharding

In this note I used code first outlined in D0 Note 3514 of 1997, to match tracks found in the Axial CFT with clusters found in the Axial CPS. This matching is done after the tracks are found and formed into lists and the clusters are found and formed into a separate list. This program reads in a list of 4 tracks and a list of 4 clusters, matches them by pairs for sixteen matchings, and outputs some straw man information for the L1 and L2. The entire process is found to require about 1500 Logic Cells in the ALTERA 10k series FPLD. For reference the mid range 10k50 has 2880 LC's. The pin count was also found to be small enough to fit into a relatively economical thin quad-flat-pack.

## 1. Introduction

The CPS has 7680 total channels in 3 layers of triangular cross section scintillators. Two of the layers are stereo and the third is axial and all three layers are split at  $z = 0$ . the axial layer has 2560 channels which are input into VLPC channels in the same cassettes used by the CFT axial. The output of the CPS is divided into the same 80 sectors as the CFT, which gives 32 CPS

channels per sector per layer. The single axial layer is input into the trigger by putting these 32 channels into the CFT trigger board for that sector

The CPS scintillators are triangular in cross section with a base of 7 mm. The mean radius of the CPS detector is 720 mm so each scintillator subtends about  $7\text{mm}/720\text{mm} = 9.7$  mrad. The outer CFT layer is about  $1\text{mm}/550\text{mm} = 1.8$  mrad. So a track through any given outer CFT layer bin points well inside one CPS bin. ( The CPS sector is 8 channels wide, and the CFT outer layer is 44 channels wide. ) MC results have shown that the CPS bin with the maximum pulse height is within 4.9 mrad RMS of the CFT track[1]. That width is about  $\frac{1}{2}$  of a CPS bin. Therefore, we should be able to bin the CPS scintillators together into overlapping groups of 2 and define an electron as a CFT track in coincidence with a CPS bin above threshold:

$$EI = \{ \text{CFT Track} \} \text{ AND } \{ \text{CPS}[\text{bin}_k] \text{ OR } \text{CPS}[\text{bin}_{k+1}] \}$$

### 1.1 Algorithm

The tracking FPLD chips will produce 4 lists of up to 6 tracks each. The Cluster finder FPLD chip will produce 4 lists of clusters each of up to 2 clusters each. The job of the match FPLD chip is to match these up to 24 tracks with the up to 8 clusters.

The matching, in this note, is accomplished by unfolding the 16-bits of a track index into the phi and Pt information. Then unfolding the 16-bits of a cluster index into the original strips hit. And finally matching the track to the cluster. The formula used for this is;

$$PS\_ = H[h\_bin] \text{ AND } P[p\_bin] \text{ AND} \\ ( PS\_in[i\_bin] \text{ OR } PS\_out[o\_bin] )$$

where a particular track phi value, h\_bin, and Pt value, p\_bin, are AND'ed together and then AND'ed again to an OR of two adjacent preshower strips.

To explore this option we had to make several assumptions. The complexity of the track finder logic is very dependent upon the coarseness or road width of the information from the CFT track finder. We assumed one-fiber-wide road resolution from the CFT at both the inner and outer layers. This fine of a road resolution in matching the CFT to the CPS gives an upper limit on the number of equations required and thus the complexity of the logic. About 700 CPS equations are needed to implement this road resolution. We had no MC information to guide us and suspect that with such information we may be able to relax the requirements on the CPS track finder. We do not think that MC studies will indicate that more stringent requirements that are needed. Therefore, the design described below is most likely a maximal design. How much it can be scaled back in hardware and cost can only be determined after specific MC studies.

The CFT tracks come from the CFT track finder loaded into a four deep buffer formatted with the address for each track. The CPS information comes

directly from the CPS cluster finder loaded into a four deep buffer of clusters indexes. The CPS information must be matched separately to each of the CFT tracks. One way to do this would be to use sixteen sets of logic, one for each CFT track CPS cluster combination. Another would be to use a reduced set of logic multiple times each crossing. The first or parallel method requires sixteen copies of the logic in the FPGA, the second or serial method requires that the time for each of the operations be short enough so that all sixteen can be finished within one crossing. The latency for a match was found to be 40ns. So three matches could be made in each crossing. However, since here we are looking a non-optimized case to get a feel for the maximum amount of logic resources required we instantiated 16 sets of the matching code.

Copies of the code are located in the appendix. The SUBDESIGN match\_4to4 is the top-level design file. The four tracks and four clusters are input. The L1 and L2 information is output. This routine calls 16 instantiations of the SUBDESIGN match\_1to1. The input into this sub-design is one track and one cluster. The returned information is a single bit, which signals a match. This routine uses three other sub-design files. The first 16dmux is an ALTERA supplied routine which unfolds a 4 bit count into one of 16 corresponding bits. Loading it with '0110' or 6 for example results in its returning a 16 bit array with the 6<sup>th</sup> bit on. This routine is used to set the bits in the track's phi array and Pt offset array.

The second routine used is SUBDESIGN ps\_slust\_01. It takes the cluster data, which has the phi position, and width encoded; and unfolds it to the exact CPS strips hit. The information is now ready to be sent to the third sub-design. This file called ps\_track\_01 contains the matching equations derived from table 1. This routine contains 704 matching equations. After the 704 terms are found they are OR'ed together in three stages down to a single term - 'el\_tot'. All of the logic in this sub-design is asynchronous and does not use registers. Because of this the optimization provided by the vendor software is given free reign. Consequently, these possible 704 plus logic cells are reduced down to a mere 85.

## 2. Summary

The entire matching problem including sixteen copies of the matching code was found to require 1500 logic cells. This is less than 100 LC's per matched pair. The latency of the process was found to be about 40ns. The results are encouraging. A very non-optimized algorithm was used and found to require a significant but relatively small amount of logic resources. It is therefore felt that whatever matching the physics requires can be incorporated into the Digital Front End board by placing a medium to large FPLD mounted in a thin quad flat pack on the mother board.

## 3. References

- 1 - Brad Abbott, private communication.
- 2 - Fred Borcharding, 'L1 Axial CFT Trigger Hardware and Firmware Design for the Baseline Trigger Algorithm', *D0Note 3082*, September 17, 1996.

<b>CFT inner bin offset</b>	<b>Pt</b>	<b>CPS Bin offset</b>	<b>OR Bins</b>
1	21.0	0.2	0,1
2	10.5	0.5	0,1
3	7.0	0.7	0,1
4	5.3	1.0	1,2
5	4.2	1.2	1,2
6	3.5	1.5	1,2
7	3.0	1.7	1,2
8	2.6	2.0	2,3
9	2.3	2.2	2,3
10	2.1	2.5	2,3
11	1.9	2.7	2,3
12	1.8	3.0	3,4
13	1.6	3.2	3,4
14	1.5	3.5	3,4
15	1.4	3.7	3,4

Table 1. This table calculates the matching between the CFT tracks' offset and the CPS strips for a CFT H bin of 0. For the other 43 H bins the CPS strip numbers must be incremented appropriately.

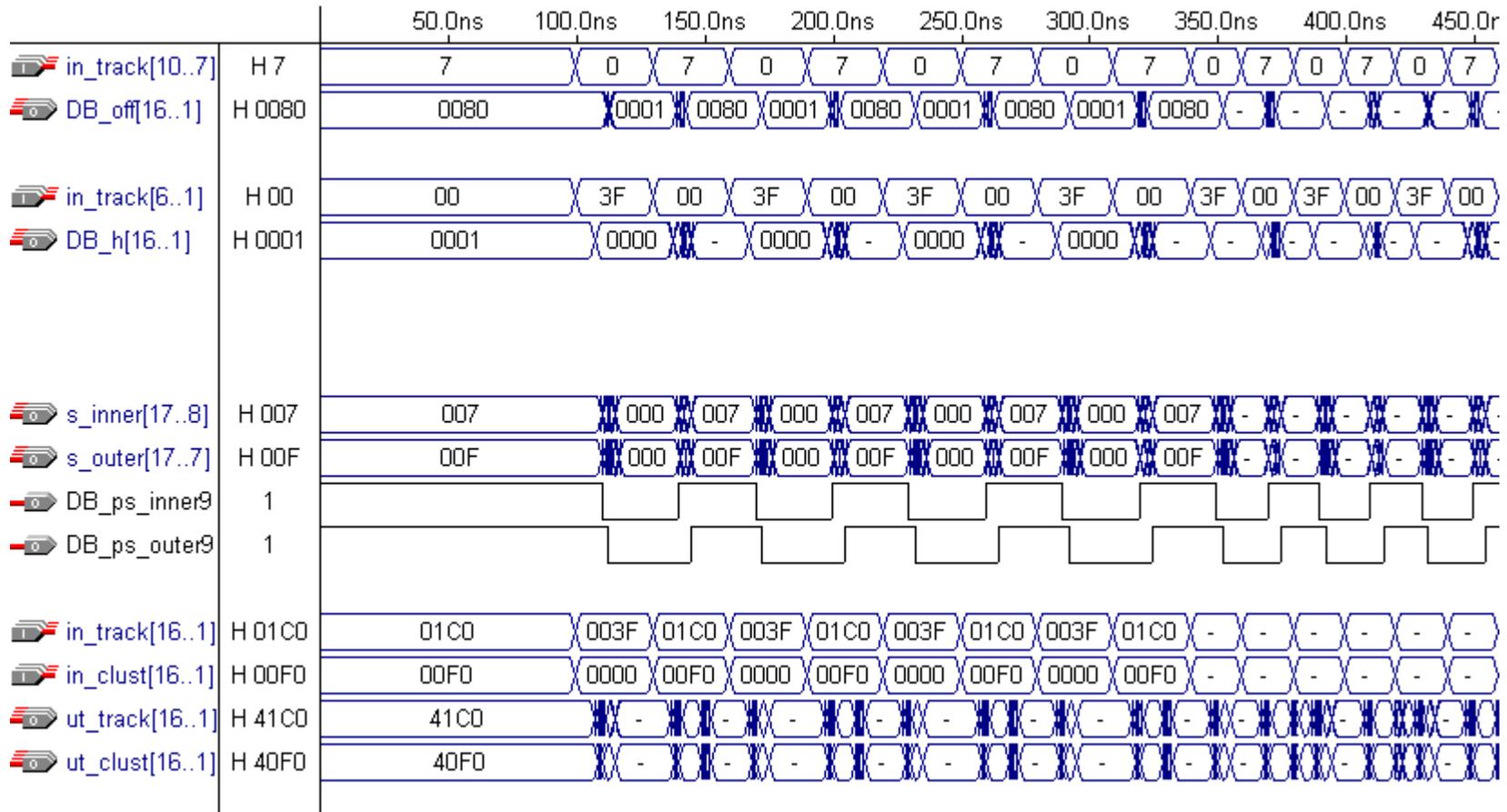


Figure 2. Waveform simulation of CPS\_MAIN\_02 / MATCH\_1TO1. 'in\_track[10..7]' is the Pt sign and value which are unfolded into the offset 'DB\_off[16..1]'. 'in\_track[6..1]' is the h bin which is unfolded into 'DB\_h[16..1]'. In the last four lines the input track/cluster are shown and below them the output track/cluster. This routine has 30ns of latency and can be run with a 30ns cycle time.

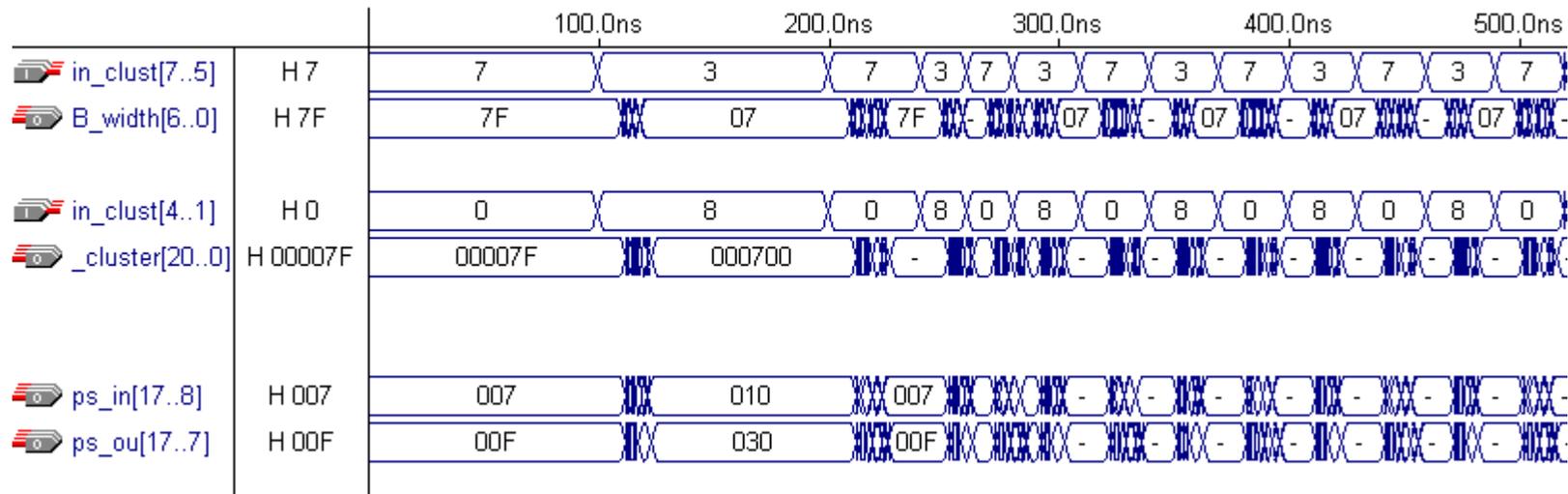


Figure 3. Waveform simulation of PS\_CLUSTER\_01. The input cluster index containing the phi position and width is unfolded into the individual ps strips hit. This routine has 25ns of latency and can operate with a 30ns cycle time.

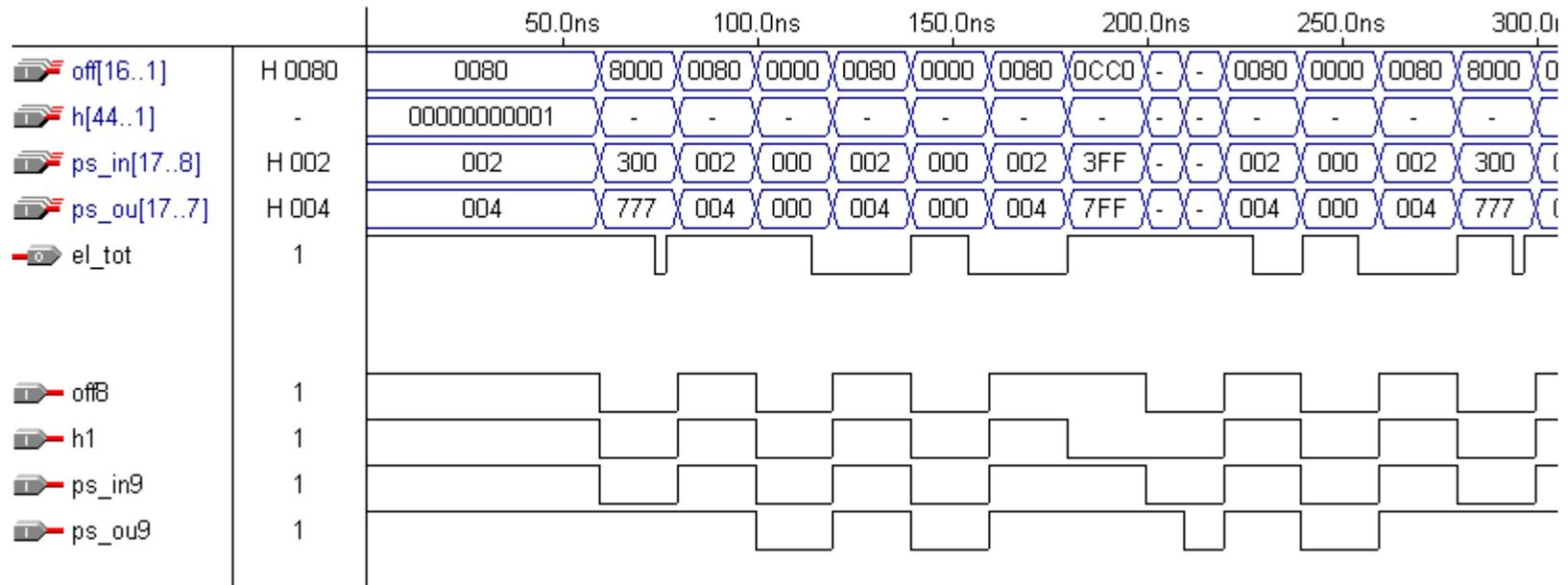


Figure 4. Waveform simulation of PS\_TRACK\_01. The input h and off arrays are matched within 704 equations to the ps\_in and ps\_ou strip arrays. The single bit designating a match - 'el\_tot' is returned. The latency of this routine is 20ns.

```

-----
-- match_4to4.vhd
-----
-- Created:      Wed Feb 10 1999                Fred B.
--              created a top level design file for the back end logic of the Digital FE boards
--              This file takes in the track lists, and cluster lists, and hit count, ect
--              and forms the L1 and L2 outputs.
--              NOTE: The track lists to MUON are pulled off prior to this file
--
--              Bit definitions for the tracks to MUON - here the input bits
--              bits 5-0 > outer layer phi bin
--              bits 11-6 > Pt value, of which
--              bits 11-10 > Pt threshold bin
--              bit 12 sign of track
--              bits 14-13 > not used
--              bit 15 > flags as track
--              Output bits are added as
--              bit 13 > Isolated
--              bit 14 > Matched
--
--              The 6 tracks/ 8 clusters are read in at 53MHz and fifo'ed through in 53MHz stages
--              A state machine is created, reset by the cross clock, to latch values
--              as each track is read in a count is kept of the positive and negative tracks by 4 types
--              as each cluster is read a count is kept of the custers of 4 types
--              Other track/cluster info is gleaned from the incomming track and put aside for processing
--              This processing takes place in a seperate module
--              The results of the processing for L2 are placed on the tracks at the end of the FIFO
--              The results of the L1 processing are put into a seperate L1 output FIFO
-----
--** DEVICE SUMMARY **
--
--Chip/          Input Output Bidir Memory Memory          LCs
--POF   Device   Pins Pins  Pins  Bits % Utilized  LCs  % Utilized
--
--              EPF10K30ETC144-1  69  14  0  0    0 %   1450  83 %
-----

```

```

LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY match_4to4 IS
  PORT(
    trk1,trk2,trk3,trk4          : IN STD_LOGIC_VECTOR (12 downto 0);
    fl1,fl2,fl3,fl4             : IN STD_LOGIC;
    cls1,cls2,cls3,cls4         : IN STD_LOGIC_VECTOR ( 7 downto 0);
    reset, clk                   : IN STD_LOGIC; --
  -- I/O Diagnostic =====
  -- Output =====
    tr_info, cl_info             : OUT STD_LOGIC_VECTOR (2 downto 0);
    tr_match, cl_match           : OUT STD_LOGIC_VECTOR (3 downto 0)
  );
END match_4to4;
-----
ARCHITECTURE a OF match_4to4 IS
  SIGNAL match11,match12,match13,match14 : STD_LOGIC ;
  SIGNAL match21,match22,match23,match24 : STD_LOGIC ;
  SIGNAL match31,match32,match33,match34 : STD_LOGIC ;
  SIGNAL match41,match42,match43,match44 : STD_LOGIC ;
  SIGNAL tr_mat, cl_mat                 : STD_LOGIC_VECTOR (3 downto 0);
----- Component Definitions -----
  COMPONENT match_1to1
  PORT (
    trk          : IN STD_LOGIC_VECTOR (12 downto 0);
    fl           : IN STD_LOGIC;
    cls          : IN STD_LOGIC_VECTOR ( 7 downto 0);
    -- reset, clk : IN STD_LOGIC; --
    match        : OUT STD_LOGIC );
  END COMPONENT;
-----
BEGIN

```

```
-- ++++++ Combine information on tracks and clusters in this section of code ++++++
```

```
----- match 4 tracks with 4 clusters -----
```

```
Match_11: match_1to1
```

```
PORT MAP( -- left is entity, right is from calling design
```

```
  trk => trk1, fl => fl1,
```

```
  cls => cls1,
```

```
--   reset => reset, clk => clk,
```

```
  match => match11); -- returned 1 bits each-
```

```
Match_12: match_1to1
```

```
PORT MAP( -- left is entity, right is from calling design
```

```
  trk => trk1, fl => fl1,
```

```
  cls => cls2,
```

```
--   reset => reset, clk => clk,
```

```
  match => match12); -- returned 1 bits each-
```

```
Match_13: match_1to1
```

```
PORT MAP( -- left is entity, right is from calling design
```

```
  trk => trk1, fl => fl1,
```

```
  cls => cls3,
```

```
--   reset => reset, clk => clk,
```

```
  match => match13); -- returned 1 bits each-
```

```
Match_14: match_1to1
```

```
PORT MAP( -- left is entity, right is from calling design
```

```
  trk => trk1, fl => fl1,
```

```
  cls => cls4,
```

```
--   reset => reset, clk => clk,
```

```
  match => match14); -- returned 1 bits each-
```

```
-----
```

```
Match_21: match_1to1
```

```
PORT MAP( -- left is entity, right is from calling design
```

```
  trk => trk2, fl => fl2,
```

```
  cls => cls1,
```

```
--   reset => reset, clk => clk,
```

```
  match => match21); -- returned 1 bits each-
```

```
Match_22: match_1to1
PORT MAP( -- left is entity, right is from calling design
  trk => trk2, fl => fl2,
  cls => cls2,
--      reset => reset, clk => clk,
        match => match22);    -- returned 1 bits each-
```

```
Match_23: match_1to1
PORT MAP( -- left is entity, right is from calling design
  trk => trk2, fl => fl2,
  cls => cls3,
--      reset => reset, clk => clk,
        match => match23);    -- returned 1 bits each-
```

```
Match_24: match_1to1
PORT MAP( -- left is entity, right is from calling design
  trk => trk2, fl => fl2,
  cls => cls4,
--      reset => reset, clk => clk,
        match => match24);    -- returned 1 bits each-
```

```
-----
Match_31: match_1to1
PORT MAP( -- left is entity, right is from calling design
  trk => trk3, fl => fl3,
  cls => cls1,
--      reset => reset, clk => clk,
        match => match31);    -- returned 1 bits each-
```

```
Match_32: match_1to1
PORT MAP( -- left is entity, right is from calling design
  trk => trk3, fl => fl3,
  cls => cls2,
--      reset => reset, clk => clk,
        match => match32);    -- returned 1 bits each-
```

```
Match_33: match_1to1
```

```
PORT MAP( -- left is entity, right is from calling design
  trk => trk3, fl => fl3,
  cls => cls3,
--      reset => reset, clk => clk,
  match => match33);    -- returned 1 bits each-
```

```
Match_34: match_1to1
PORT MAP( -- left is entity, right is from calling design
  trk => trk3, fl => fl3,
  cls => cls4,
--      reset => reset, clk => clk,
  match => match34);    -- returned 1 bits each-
```

---

```
Match_41: match_1to1
PORT MAP( -- left is entity, right is from calling design
  trk => trk4, fl => fl4,
  cls => cls1,
--      reset => reset, clk => clk,
  match => match41);    -- returned 1 bits each-
```

```
Match_42: match_1to1
PORT MAP( -- left is entity, right is from calling design
  trk => trk4, fl => fl4,
  cls => cls2,
--      reset => reset, clk => clk,
  match => match42);    -- returned 1 bits each-
```

```
Match_43: match_1to1
PORT MAP( -- left is entity, right is from calling design
  trk => trk4, fl => fl4,
  cls => cls3,
--      reset => reset, clk => clk,
  match => match43);    -- returned 1 bits each-
```

```
Match_44: match_1to1
PORT MAP( -- left is entity, right is from calling design
  trk => trk4, fl => fl4,
```

```

        cls => cls4,
--      reset => reset, clk => clk,
        match => match44);    -- returned 1 bits each-
-----
----- Calculate the variious correlations -----
set_tr_match:
PROCESS (    match11,match12,match13,match14,
            match21,match22,match23,match24,
            match31,match32,match33,match34,
            match41,match42,match43,match44 )

BEGIN
    tr_mat(0) <= match11 OR match12 OR match13 OR match14;  --
    tr_mat(1) <= match21 OR match22 OR match23 OR match24;  --
    tr_mat(2) <= match31 OR match32 OR match33 OR match34;  --
    tr_mat(3) <= match41 OR match42 OR match43 OR match44;  --
    cl_mat(0) <= match11 OR match21 OR match31 OR match41;
    cl_mat(1) <= match12 OR match22 OR match32 OR match42;
    cl_mat(2) <= match13 OR match23 OR match33 OR match43;
    cl_mat(3) <= match14 OR match24 OR match34 OR match44;
END PROCESS set_tr_match;

----- Priority endcoder >> put cluster info into track --
load_cl_info_to_track:
PROCESS (tr_mat)
    VARIABLE cl_inf      : STD_LOGIC_VECTOR (2 downto 0);
BEGIN
    IF tr_mat(0) = '1' THEN
        cl_inf := cls1(6 downto 4);
    ELSIF tr_mat(1) = '1' THEN
        cl_inf := cls2(6 downto 4);
    ELSIF tr_mat(2) = '1' THEN
        cl_inf := cls3(6 downto 4);
    ELSIF tr_mat(3) = '1' THEN
        cl_inf := cls4(6 downto 4);
    ELSE
        cl_inf := "000";
    END IF;
END PROCESS load_cl_info_to_track;

```

```
        END IF;
        cl_info <= cl_inf;
        tr_match <= tr_mat;
    END PROCESS load_cl_info_to_track;
----- Priority endcoder >> put track info into cluster --
load_tr_info_to_clust:
PROCESS (cl_mat)
    VARIABLE tr_inf          : STD_LOGIC_VECTOR (2 downto 0);
BEGIN
    IF cl_mat(0) = '1' THEN
        tr_inf := trk1(6 downto 4);
    ELSIF cl_mat(0) = '1' THEN
        tr_inf := trk2(6 downto 4);
    ELSIF cl_mat(0) = '1' THEN
        tr_inf := trk3(6 downto 4);
    ELSIF cl_mat(0) = '1' THEN
        tr_inf := trk4(6 downto 4);
    ELSE
        tr_inf := "000";
    END IF;
    tr_info <= tr_inf;
    cl_match <= cl_mat;
END PROCESS load_tr_info_to_clust;
-----
END;
```

TITLE "Serialization for 2 Pt bins V\_1";

--  
 -- This function takes an input track address and matches it to a CPS hit  
 -- and outputs the track address with the CPS hit bit modified

-- Created 16-Jan-1997 Fred Borcharding  
 -- Modified 16-Feb-1999 Fred Borcharding  
 -- Modify to use the ps\_track file

-- Project Information d:\a4\_trigger\cps\_match\cps\_main\_02.rpt

-- MAX+plus II Compiler Report File  
 -- Version 9.1 10/23/1998  
 -- Compiled: 02/16/1999 13:22:27

Chip/	Input	Output	Bidir	Memory	Memory		LCs	
POF	Device	Pins	Pins	Pins	Bits	% Utilized	LCs	% Utilized
	cps_main_02							
	EPF10K10AQC208-1	30	85	0	0	0 %	280	48 %

FUNCTION 16dmux (d, c, b, a) RETURNS (q[15..0]); -- ALTERA Routine  
 FUNCTION ps\_track\_01 (off[16..1],h[44..1],ps\_in[17..8],ps\_ou[17..7]) RETURNS (el\_tot);  
 FUNCTION ps\_clust\_01 (in\_clust[7..1]) RETURNS (ps\_in[17..8],ps\_ou[17..7]);

SUBDESIGN cps\_main\_02 [ match\_1to1 ]

```
(
    in_track[16..1],in_clust[16..1]      : INPUT;      -- Input hit address word
-- Debug
    DB_h[16..1]                          : OUTPUT;
    DB_off[16..1]                         : OUTPUT;
    DB_ps_inner[17..8]                    : OUTPUT;
    DB_ps_outer[17..7]                   : OUTPUT;
```

```

--      out_track[16..1],out_clust[16..1] : OUTPUT;      -- Output hit address word
)
VARIABLE
  h_add[5..0]                : NODE;      -- phi bin address
  off_add[3..0]              : NODE;      -- Pt bin address
  off[16..1]                 : NODE;
  h[48..1]                   : NODE;      -- H layer phi bins
  el_tot                     : NODE;
  ps_inner[17..8], ps_outer[17..7] : NODE;  --
  --a5[5..0],a6[5..0]        : NODE;      -- Used to store results from step 1
--
-- Macros
  m_16dmux                   : 16dmux;
  mo_16dmux                  : 16dmux;
  ps_cls                      : ps_clust_01;
  ps_trk                      : ps_track_01;

```

-----

BEGIN

```

  h_add[5..0] = in_track[6..1];
  off_add[3..0] = in_track[10..7];

```

----- Find the Track phi values -----

```

CASE h_add[5..4] IS
  WHEN 0 =>
    m_16dmux.a = h_add[0];
    m_16dmux.b = h_add[1];
    m_16dmux.c = h_add[2];
    m_16dmux.d = h_add[3];
    h[16..1] = m_16dmux.q[15..0];
  WHEN 1 =>
    m_16dmux.a = h_add[0];
    m_16dmux.b = h_add[1];

```

```

        m_16dmux.c = h_add[2];
        m_16dmux.d = h_add[3];
        h[32..17] = m_16dmux.q[15..0];
    WHEN 2 =>
        m_16dmux.a = h_add[0];
        m_16dmux.b = h_add[1];
        m_16dmux.c = h_add[2];
        m_16dmux.d = h_add[3];
        h[48..33] = m_16dmux.q[15..0];
    WHEN OTHERS =>
        h[] = H"0";
END CASE;
DB_h[] = h[16..1];
----- Find the Track offset values -----
mo_16dmux.a = off_add[0];
mo_16dmux.b = off_add[1];
mo_16dmux.c = off_add[2];
mo_16dmux.d = off_add[3];
off[16..1] = mo_16dmux.q[15..0];
DB_off[] = off[];
----- Find the Cluster strip values -----
ps_cls.in_clust[7..1]    = in_clust[7..1];
ps_inner[17..8]        = ps_cls.ps_in[17..8];
ps_outer[17..7]       = ps_cls.ps_ou[17..7];
DB_ps_inner[] = ps_inner[];
DB_ps_outer[] = ps_outer[];
----- Combine track with cluster -----
ps_trk.off[16..1] = off[16..1];
ps_trk.h[44..1]  = h[44..1];
ps_trk.ps_in[17..8] = ps_inner[17..8];
ps_trk.ps_ou[17..7] = ps_outer[17..7];
el_tot          = ps_trk.el_tot;
----- Set bits in output -----

out_track[14..1] = in_track[14..1];    -- output the input buffer
out_track[16]   = in_track[16];       -- output the input buffer

```

```
out_track[15]      = el_tot;          --  
  
out_clust[14..1] = in_clust[14..1];  -- output the input buffer  
out_clust[16]     = in_clust[16];    -- output the input buffer  
out_clust[15]     = el_tot;          --
```

-----  
END;

TITLE "Serialization for 2 Pt bins V\_1";

```
--
-- Resurrected          16-Feb-1999          Fred Borcharding
--
-- Project Information   d:\a4_trigger\cps_match\ps_track_01.rpt
--
-- MAX+plus II Compiler Report File
-- Version 9.1 10/23/1998
-- Compiled: 02/16/1999 13:05:49
--
-- Chip/          Input Output Bidir Memory Memory          LCs
-- POF    Device   Pins Pins  Pins  Bits % Utilized  LCs % Utilized
--
-- EPF10K10ATC144-1  81  1  0  0    0 % 85   14 %
--
```

SUBDESIGN ps\_track\_01

```
(
  --load, clock          : INPUT;
  off[16..1]             : INPUT;
  h[44..1]               : INPUT;
  ps_in[17..8]           : INPUT;      -- Input hit address
  ps_ou[17..7]           : INPUT;      --
  el_tot                 : OUTPUT;
)
VARIABLE
  el[704..1]             : NODE;      -- phi bin address
  el1_[70..0]            : NODE;      -- Pt bin address
  el2_[7..0]             : NODE;
--  el_tot                : DFFE;
```

BEGIN

```
--el1_[] .ena    = load;
--el1_[] .clk    = clock;
--el_tot.ena     = load;
--el_tot.clk     = clock;
```

-----

```
el[ 1] = off[ 1] AND h[ 1] AND (ps_in[10] OR ps_ou[10]); -- -2.6
el[ 2] = off[ 2] AND h[ 1] AND (ps_in[10] OR ps_ou[10]); -- -3.0
el[ 3] = off[ 3] AND h[ 1] AND (ps_in[10] OR ps_ou[ 9]); -- -3.5
el[ 4] = off[ 4] AND h[ 1] AND (ps_in[10] OR ps_ou[ 9]); -- -4.2
el[ 5] = off[ 5] AND h[ 1] AND (ps_in[ 9] OR ps_ou[ 9]); -- -5.3
el[ 6] = off[ 6] AND h[ 1] AND (ps_in[ 9] OR ps_ou[ 9]); -- -7.0
el[ 7] = off[ 7] AND h[ 1] AND (ps_in[ 9] OR ps_ou[ 9]); -- -10.5
el[ 8] = off[ 8] AND h[ 1] AND (ps_in[ 9] OR ps_ou[ 9]); -- -21.0
el[ 9] = off[ 9] AND h[ 1] AND (ps_in[ 9] OR ps_ou[ 9]); -- 99.0
el[10] = off[10] AND h[ 1] AND (ps_in[ 9] OR ps_ou[ 9]); -- 21.0
el[11] = off[11] AND h[ 1] AND (ps_in[ 9] OR ps_ou[ 9]); -- 10.5
el[12] = off[12] AND h[ 1] AND (ps_in[ 9] OR ps_ou[ 8]); -- 7.0
el[13] = off[13] AND h[ 1] AND (ps_in[ 9] OR ps_ou[ 8]); -- 5.3
el[14] = off[14] AND h[ 1] AND (ps_in[ 8] OR ps_ou[ 8]); -- 4.2
el[15] = off[15] AND h[ 1] AND (ps_in[ 8] OR ps_ou[ 8]); -- 3.5
el[16] = off[16] AND h[ 1] AND (ps_in[ 8] OR ps_ou[ 7]); -- 3.0
el[17] = off[ 1] AND h[ 2] AND (ps_in[10] OR ps_ou[10]); -- -2.6
el[18] = off[ 2] AND h[ 2] AND (ps_in[10] OR ps_ou[10]); -- -3.0
el[19] = off[ 3] AND h[ 2] AND (ps_in[10] OR ps_ou[ 9]); -- -3.5
el[20] = off[ 4] AND h[ 2] AND (ps_in[10] OR ps_ou[ 9]); -- -4.2
el[21] = off[ 5] AND h[ 2] AND (ps_in[ 9] OR ps_ou[ 9]); -- -5.3
el[22] = off[ 6] AND h[ 2] AND (ps_in[ 9] OR ps_ou[ 9]); -- -7.0
el[23] = off[ 7] AND h[ 2] AND (ps_in[ 9] OR ps_ou[ 9]); -- -10.5
el[24] = off[ 8] AND h[ 2] AND (ps_in[ 9] OR ps_ou[ 9]); -- -21.0
el[25] = off[ 9] AND h[ 2] AND (ps_in[ 9] OR ps_ou[ 9]); -- 99.0
el[26] = off[10] AND h[ 2] AND (ps_in[ 9] OR ps_ou[ 9]); -- 21.0
el[27] = off[11] AND h[ 2] AND (ps_in[ 9] OR ps_ou[ 9]); -- 10.5
el[28] = off[12] AND h[ 2] AND (ps_in[ 9] OR ps_ou[ 8]); -- 7.0
el[29] = off[13] AND h[ 2] AND (ps_in[ 9] OR ps_ou[ 8]); -- 5.3
el[30] = off[14] AND h[ 2] AND (ps_in[ 8] OR ps_ou[ 8]); -- 4.2
el[31] = off[15] AND h[ 2] AND (ps_in[ 8] OR ps_ou[ 8]); -- 3.5
el[32] = off[16] AND h[ 2] AND (ps_in[ 8] OR ps_ou[ 7]); -- 3.0
el[33] = off[ 1] AND h[ 3] AND (ps_in[10] OR ps_ou[10]); -- -2.6
el[34] = off[ 2] AND h[ 3] AND (ps_in[10] OR ps_ou[10]); -- -3.0
el[35] = off[ 3] AND h[ 3] AND (ps_in[10] OR ps_ou[ 9]); -- -3.5
```

el[ 36] = off[ 4] AND h[ 3] AND (ps\_in[10] OR ps\_ou[ 9]); -- -4.2  
el[ 37] = off[ 5] AND h[ 3] AND (ps\_in[ 9] OR ps\_ou[ 9]); -- -5.3  
el[ 38] = off[ 6] AND h[ 3] AND (ps\_in[ 9] OR ps\_ou[ 9]); -- -7.0  
el[ 39] = off[ 7] AND h[ 3] AND (ps\_in[ 9] OR ps\_ou[ 9]); -- -10.5  
el[ 40] = off[ 8] AND h[ 3] AND (ps\_in[ 9] OR ps\_ou[ 9]); -- -21.0  
el[ 41] = off[ 9] AND h[ 3] AND (ps\_in[ 9] OR ps\_ou[ 9]); -- 99.0  
el[ 42] = off[10] AND h[ 3] AND (ps\_in[ 9] OR ps\_ou[ 9]); -- 21.0  
el[ 43] = off[11] AND h[ 3] AND (ps\_in[ 9] OR ps\_ou[ 9]); -- 10.5  
el[ 44] = off[12] AND h[ 3] AND (ps\_in[ 9] OR ps\_ou[ 8]); -- 7.0  
el[ 45] = off[13] AND h[ 3] AND (ps\_in[ 9] OR ps\_ou[ 8]); -- 5.3  
el[ 46] = off[14] AND h[ 3] AND (ps\_in[ 8] OR ps\_ou[ 8]); -- 4.2  
el[ 47] = off[15] AND h[ 3] AND (ps\_in[ 8] OR ps\_ou[ 8]); -- 3.5  
el[ 48] = off[16] AND h[ 3] AND (ps\_in[ 8] OR ps\_ou[ 7]); -- 3.0  
el[ 49] = off[ 1] AND h[ 4] AND (ps\_in[10] OR ps\_ou[10]); -- -2.6  
el[ 50] = off[ 2] AND h[ 4] AND (ps\_in[10] OR ps\_ou[10]); -- -3.0  
el[ 51] = off[ 3] AND h[ 4] AND (ps\_in[10] OR ps\_ou[ 9]); -- -3.5  
el[ 52] = off[ 4] AND h[ 4] AND (ps\_in[10] OR ps\_ou[ 9]); -- -4.2  
el[ 53] = off[ 5] AND h[ 4] AND (ps\_in[ 9] OR ps\_ou[ 9]); -- -5.3  
el[ 54] = off[ 6] AND h[ 4] AND (ps\_in[ 9] OR ps\_ou[ 9]); -- -7.0  
el[ 55] = off[ 7] AND h[ 4] AND (ps\_in[ 9] OR ps\_ou[ 9]); -- -10.5  
el[ 56] = off[ 8] AND h[ 4] AND (ps\_in[ 9] OR ps\_ou[ 9]); -- -21.0  
el[ 57] = off[ 9] AND h[ 4] AND (ps\_in[ 9] OR ps\_ou[ 9]); -- 99.0  
el[ 58] = off[10] AND h[ 4] AND (ps\_in[ 9] OR ps\_ou[ 9]); -- 21.0  
el[ 59] = off[11] AND h[ 4] AND (ps\_in[ 9] OR ps\_ou[ 9]); -- 10.5  
el[ 60] = off[12] AND h[ 4] AND (ps\_in[ 9] OR ps\_ou[ 8]); -- 7.0  
el[ 61] = off[13] AND h[ 4] AND (ps\_in[ 9] OR ps\_ou[ 8]); -- 5.3  
el[ 62] = off[14] AND h[ 4] AND (ps\_in[ 8] OR ps\_ou[ 8]); -- 4.2  
el[ 63] = off[15] AND h[ 4] AND (ps\_in[ 8] OR ps\_ou[ 8]); -- 3.5  
el[ 64] = off[16] AND h[ 4] AND (ps\_in[ 8] OR ps\_ou[ 7]); -- 3.0  
el[ 65] = off[ 1] AND h[ 5] AND (ps\_in[10] OR ps\_ou[10]); -- -2.6  
el[ 66] = off[ 2] AND h[ 5] AND (ps\_in[10] OR ps\_ou[10]); -- -3.0  
el[ 67] = off[ 3] AND h[ 5] AND (ps\_in[10] OR ps\_ou[ 9]); -- -3.5  
el[ 68] = off[ 4] AND h[ 5] AND (ps\_in[10] OR ps\_ou[ 9]); -- -4.2  
el[ 69] = off[ 5] AND h[ 5] AND (ps\_in[ 9] OR ps\_ou[ 9]); -- -5.3  
el[ 70] = off[ 6] AND h[ 5] AND (ps\_in[ 9] OR ps\_ou[ 9]); -- -7.0  
el[ 71] = off[ 7] AND h[ 5] AND (ps\_in[ 9] OR ps\_ou[ 9]); -- -10.5  
el[ 72] = off[ 8] AND h[ 5] AND (ps\_in[ 9] OR ps\_ou[ 9]); -- -21.0

```
el[ 73] = off[ 9] AND h[ 5] AND (ps_in[ 9] OR ps_ou[ 9]); -- 99.0
el[ 74] = off[10] AND h[ 5] AND (ps_in[ 9] OR ps_ou[ 9]); -- 21.0
el[ 75] = off[11] AND h[ 5] AND (ps_in[ 9] OR ps_ou[ 9]); -- 10.5
el[ 76] = off[12] AND h[ 5] AND (ps_in[ 9] OR ps_ou[ 8]); -- 7.0
el[ 77] = off[13] AND h[ 5] AND (ps_in[ 9] OR ps_ou[ 8]); -- 5.3
el[ 78] = off[14] AND h[ 5] AND (ps_in[ 8] OR ps_ou[ 8]); -- 4.2
el[ 79] = off[15] AND h[ 5] AND (ps_in[ 8] OR ps_ou[ 8]); -- 3.5
el[ 80] = off[16] AND h[ 5] AND (ps_in[ 8] OR ps_ou[ 7]); -- 3.0
el[ 81] = off[ 1] AND h[ 6] AND (ps_in[10] OR ps_ou[10]); -- -2.6
el[ 82] = off[ 2] AND h[ 6] AND (ps_in[10] OR ps_ou[10]); -- -3.0
el[ 83] = off[ 3] AND h[ 6] AND (ps_in[10] OR ps_ou[ 9]); -- -3.5
el[ 84] = off[ 4] AND h[ 6] AND (ps_in[10] OR ps_ou[ 9]); -- -4.2
el[ 85] = off[ 5] AND h[ 6] AND (ps_in[ 9] OR ps_ou[ 9]); -- -5.3
el[ 86] = off[ 6] AND h[ 6] AND (ps_in[ 9] OR ps_ou[ 9]); -- -7.0
el[ 87] = off[ 7] AND h[ 6] AND (ps_in[ 9] OR ps_ou[ 9]); -- -10.5
el[ 88] = off[ 8] AND h[ 6] AND (ps_in[ 9] OR ps_ou[ 9]); -- -21.0
el[ 89] = off[ 9] AND h[ 6] AND (ps_in[ 9] OR ps_ou[ 9]); -- 99.0
el[ 90] = off[10] AND h[ 6] AND (ps_in[ 9] OR ps_ou[ 9]); -- 21.0
el[ 91] = off[11] AND h[ 6] AND (ps_in[ 9] OR ps_ou[ 9]); -- 10.5
el[ 92] = off[12] AND h[ 6] AND (ps_in[ 9] OR ps_ou[ 8]); -- 7.0
el[ 93] = off[13] AND h[ 6] AND (ps_in[ 9] OR ps_ou[ 8]); -- 5.3
el[ 94] = off[14] AND h[ 6] AND (ps_in[ 8] OR ps_ou[ 8]); -- 4.2
el[ 95] = off[15] AND h[ 6] AND (ps_in[ 8] OR ps_ou[ 8]); -- 3.5
el[ 96] = off[16] AND h[ 6] AND (ps_in[ 8] OR ps_ou[ 7]); -- 3.0
el[ 97] = off[ 1] AND h[ 7] AND (ps_in[11] OR ps_ou[11]); -- -2.6
el[ 98] = off[ 2] AND h[ 7] AND (ps_in[11] OR ps_ou[11]); -- -3.0
el[ 99] = off[ 3] AND h[ 7] AND (ps_in[11] OR ps_ou[10]); -- -3.5
el[100] = off[ 4] AND h[ 7] AND (ps_in[11] OR ps_ou[10]); -- -4.2
el[101] = off[ 5] AND h[ 7] AND (ps_in[10] OR ps_ou[10]); -- -5.3
el[102] = off[ 6] AND h[ 7] AND (ps_in[10] OR ps_ou[10]); -- -7.0
el[103] = off[ 7] AND h[ 7] AND (ps_in[10] OR ps_ou[10]); -- -10.5
el[104] = off[ 8] AND h[ 7] AND (ps_in[10] OR ps_ou[10]); -- -21.0
el[105] = off[ 9] AND h[ 7] AND (ps_in[10] OR ps_ou[10]); -- 99.0
el[106] = off[10] AND h[ 7] AND (ps_in[10] OR ps_ou[10]); -- 21.0
el[107] = off[11] AND h[ 7] AND (ps_in[10] OR ps_ou[10]); -- 10.5
el[108] = off[12] AND h[ 7] AND (ps_in[10] OR ps_ou[ 9]); -- 7.0
el[109] = off[13] AND h[ 7] AND (ps_in[10] OR ps_ou[ 9]); -- 5.3
```

```
el[ 110] = off[14] AND h[ 7] AND (ps_in[ 9] OR ps_ou[ 9]); -- 4.2
el[ 111] = off[15] AND h[ 7] AND (ps_in[ 9] OR ps_ou[ 9]); -- 3.5
el[ 112] = off[16] AND h[ 7] AND (ps_in[ 9] OR ps_ou[ 8]); -- 3.0
el[ 113] = off[ 1] AND h[ 8] AND (ps_in[11] OR ps_ou[11]); -- -2.6
el[ 114] = off[ 2] AND h[ 8] AND (ps_in[11] OR ps_ou[11]); -- -3.0
el[ 115] = off[ 3] AND h[ 8] AND (ps_in[11] OR ps_ou[10]); -- -3.5
el[ 116] = off[ 4] AND h[ 8] AND (ps_in[11] OR ps_ou[10]); -- -4.2
el[ 117] = off[ 5] AND h[ 8] AND (ps_in[10] OR ps_ou[10]); -- -5.3
el[ 118] = off[ 6] AND h[ 8] AND (ps_in[10] OR ps_ou[10]); -- -7.0
el[ 119] = off[ 7] AND h[ 8] AND (ps_in[10] OR ps_ou[10]); -- -10.5
el[ 120] = off[ 8] AND h[ 8] AND (ps_in[10] OR ps_ou[10]); -- -21.0
el[ 121] = off[ 9] AND h[ 8] AND (ps_in[10] OR ps_ou[10]); -- 99.0
el[ 122] = off[10] AND h[ 8] AND (ps_in[10] OR ps_ou[10]); -- 21.0
el[ 123] = off[11] AND h[ 8] AND (ps_in[10] OR ps_ou[10]); -- 10.5
el[ 124] = off[12] AND h[ 8] AND (ps_in[10] OR ps_ou[ 9]); -- 7.0
el[ 125] = off[13] AND h[ 8] AND (ps_in[10] OR ps_ou[ 9]); -- 5.3
el[ 126] = off[14] AND h[ 8] AND (ps_in[ 9] OR ps_ou[ 9]); -- 4.2
el[ 127] = off[15] AND h[ 8] AND (ps_in[ 9] OR ps_ou[ 9]); -- 3.5
el[ 128] = off[16] AND h[ 8] AND (ps_in[ 9] OR ps_ou[ 8]); -- 3.0
el[ 129] = off[ 1] AND h[ 9] AND (ps_in[11] OR ps_ou[11]); -- -2.6
el[ 130] = off[ 2] AND h[ 9] AND (ps_in[11] OR ps_ou[11]); -- -3.0
el[ 131] = off[ 3] AND h[ 9] AND (ps_in[11] OR ps_ou[10]); -- -3.5
el[ 132] = off[ 4] AND h[ 9] AND (ps_in[11] OR ps_ou[10]); -- -4.2
el[ 133] = off[ 5] AND h[ 9] AND (ps_in[10] OR ps_ou[10]); -- -5.3
el[ 134] = off[ 6] AND h[ 9] AND (ps_in[10] OR ps_ou[10]); -- -7.0
el[ 135] = off[ 7] AND h[ 9] AND (ps_in[10] OR ps_ou[10]); -- -10.5
el[ 136] = off[ 8] AND h[ 9] AND (ps_in[10] OR ps_ou[10]); -- -21.0
el[ 137] = off[ 9] AND h[ 9] AND (ps_in[10] OR ps_ou[10]); -- 99.0
el[ 138] = off[10] AND h[ 9] AND (ps_in[10] OR ps_ou[10]); -- 21.0
el[ 139] = off[11] AND h[ 9] AND (ps_in[10] OR ps_ou[10]); -- 10.5
el[ 140] = off[12] AND h[ 9] AND (ps_in[10] OR ps_ou[ 9]); -- 7.0
el[ 141] = off[13] AND h[ 9] AND (ps_in[10] OR ps_ou[ 9]); -- 5.3
el[ 142] = off[14] AND h[ 9] AND (ps_in[ 9] OR ps_ou[ 9]); -- 4.2
el[ 143] = off[15] AND h[ 9] AND (ps_in[ 9] OR ps_ou[ 9]); -- 3.5
el[ 144] = off[16] AND h[ 9] AND (ps_in[ 9] OR ps_ou[ 8]); -- 3.0
el[ 145] = off[ 1] AND h[10] AND (ps_in[11] OR ps_ou[11]); -- -2.6
el[ 146] = off[ 2] AND h[10] AND (ps_in[11] OR ps_ou[11]); -- -3.0
```

```
el[ 147] = off[ 3] AND h[10] AND (ps_in[11] OR ps_ou[10]); -- -3.5
el[ 148] = off[ 4] AND h[10] AND (ps_in[11] OR ps_ou[10]); -- -4.2
el[ 149] = off[ 5] AND h[10] AND (ps_in[10] OR ps_ou[10]); -- -5.3
el[ 150] = off[ 6] AND h[10] AND (ps_in[10] OR ps_ou[10]); -- -7.0
el[ 151] = off[ 7] AND h[10] AND (ps_in[10] OR ps_ou[10]); -- -10.5
el[ 152] = off[ 8] AND h[10] AND (ps_in[10] OR ps_ou[10]); -- -21.0
el[ 153] = off[ 9] AND h[10] AND (ps_in[10] OR ps_ou[10]); -- 99.0
el[ 154] = off[10] AND h[10] AND (ps_in[10] OR ps_ou[10]); -- 21.0
el[ 155] = off[11] AND h[10] AND (ps_in[10] OR ps_ou[10]); -- 10.5
el[ 156] = off[12] AND h[10] AND (ps_in[10] OR ps_ou[ 9]); -- 7.0
el[ 157] = off[13] AND h[10] AND (ps_in[10] OR ps_ou[ 9]); -- 5.3
el[ 158] = off[14] AND h[10] AND (ps_in[ 9] OR ps_ou[ 9]); -- 4.2
el[ 159] = off[15] AND h[10] AND (ps_in[ 9] OR ps_ou[ 9]); -- 3.5
el[ 160] = off[16] AND h[10] AND (ps_in[ 9] OR ps_ou[ 8]); -- 3.0
el[ 161] = off[ 1] AND h[11] AND (ps_in[11] OR ps_ou[11]); -- -2.6
el[ 162] = off[ 2] AND h[11] AND (ps_in[11] OR ps_ou[11]); -- -3.0
el[ 163] = off[ 3] AND h[11] AND (ps_in[11] OR ps_ou[10]); -- -3.5
el[ 164] = off[ 4] AND h[11] AND (ps_in[11] OR ps_ou[10]); -- -4.2
el[ 165] = off[ 5] AND h[11] AND (ps_in[10] OR ps_ou[10]); -- -5.3
el[ 166] = off[ 6] AND h[11] AND (ps_in[10] OR ps_ou[10]); -- -7.0
el[ 167] = off[ 7] AND h[11] AND (ps_in[10] OR ps_ou[10]); -- -10.5
el[ 168] = off[ 8] AND h[11] AND (ps_in[10] OR ps_ou[10]); -- -21.0
el[ 169] = off[ 9] AND h[11] AND (ps_in[10] OR ps_ou[10]); -- 99.0
el[ 170] = off[10] AND h[11] AND (ps_in[10] OR ps_ou[10]); -- 21.0
el[ 171] = off[11] AND h[11] AND (ps_in[10] OR ps_ou[10]); -- 10.5
el[ 172] = off[12] AND h[11] AND (ps_in[10] OR ps_ou[ 9]); -- 7.0
el[ 173] = off[13] AND h[11] AND (ps_in[10] OR ps_ou[ 9]); -- 5.3
el[ 174] = off[14] AND h[11] AND (ps_in[ 9] OR ps_ou[ 9]); -- 4.2
el[ 175] = off[15] AND h[11] AND (ps_in[ 9] OR ps_ou[ 9]); -- 3.5
el[ 176] = off[16] AND h[11] AND (ps_in[ 9] OR ps_ou[ 8]); -- 3.0
el[ 177] = off[ 1] AND h[12] AND (ps_in[12] OR ps_ou[12]); -- -2.6
el[ 178] = off[ 2] AND h[12] AND (ps_in[12] OR ps_ou[12]); -- -3.0
el[ 179] = off[ 3] AND h[12] AND (ps_in[12] OR ps_ou[11]); -- -3.5
el[ 180] = off[ 4] AND h[12] AND (ps_in[12] OR ps_ou[11]); -- -4.2
el[ 181] = off[ 5] AND h[12] AND (ps_in[11] OR ps_ou[11]); -- -5.3
el[ 182] = off[ 6] AND h[12] AND (ps_in[11] OR ps_ou[11]); -- -7.0
el[ 183] = off[ 7] AND h[12] AND (ps_in[11] OR ps_ou[11]); -- -10.5
```

```
el[ 184] = off[ 8] AND h[12] AND (ps_in[11] OR ps_ou[11]); -- -21.0
el[ 185] = off[ 9] AND h[12] AND (ps_in[11] OR ps_ou[11]); -- 99.0
el[ 186] = off[10] AND h[12] AND (ps_in[11] OR ps_ou[11]); -- 21.0
el[ 187] = off[11] AND h[12] AND (ps_in[11] OR ps_ou[11]); -- 10.5
el[ 188] = off[12] AND h[12] AND (ps_in[11] OR ps_ou[10]); -- 7.0
el[ 189] = off[13] AND h[12] AND (ps_in[11] OR ps_ou[10]); -- 5.3
el[ 190] = off[14] AND h[12] AND (ps_in[10] OR ps_ou[10]); -- 4.2
el[ 191] = off[15] AND h[12] AND (ps_in[10] OR ps_ou[10]); -- 3.5
el[ 192] = off[16] AND h[12] AND (ps_in[10] OR ps_ou[ 9]); -- 3.0
el[ 193] = off[ 1] AND h[13] AND (ps_in[12] OR ps_ou[12]); -- -2.6
el[ 194] = off[ 2] AND h[13] AND (ps_in[12] OR ps_ou[12]); -- -3.0
el[ 195] = off[ 3] AND h[13] AND (ps_in[12] OR ps_ou[11]); -- -3.5
el[ 196] = off[ 4] AND h[13] AND (ps_in[12] OR ps_ou[11]); -- -4.2
el[ 197] = off[ 5] AND h[13] AND (ps_in[11] OR ps_ou[11]); -- -5.3
el[ 198] = off[ 6] AND h[13] AND (ps_in[11] OR ps_ou[11]); -- -7.0
el[ 199] = off[ 7] AND h[13] AND (ps_in[11] OR ps_ou[11]); -- -10.5
el[ 200] = off[ 8] AND h[13] AND (ps_in[11] OR ps_ou[11]); -- -21.0
el[ 201] = off[ 9] AND h[13] AND (ps_in[11] OR ps_ou[11]); -- 99.0
el[ 202] = off[10] AND h[13] AND (ps_in[11] OR ps_ou[11]); -- 21.0
el[ 203] = off[11] AND h[13] AND (ps_in[11] OR ps_ou[11]); -- 10.5
el[ 204] = off[12] AND h[13] AND (ps_in[11] OR ps_ou[10]); -- 7.0
el[ 205] = off[13] AND h[13] AND (ps_in[11] OR ps_ou[10]); -- 5.3
el[ 206] = off[14] AND h[13] AND (ps_in[10] OR ps_ou[10]); -- 4.2
el[ 207] = off[15] AND h[13] AND (ps_in[10] OR ps_ou[10]); -- 3.5
el[ 208] = off[16] AND h[13] AND (ps_in[10] OR ps_ou[ 9]); -- 3.0
el[ 209] = off[ 1] AND h[14] AND (ps_in[12] OR ps_ou[12]); -- -2.6
el[ 210] = off[ 2] AND h[14] AND (ps_in[12] OR ps_ou[12]); -- -3.0
el[ 211] = off[ 3] AND h[14] AND (ps_in[12] OR ps_ou[11]); -- -3.5
el[ 212] = off[ 4] AND h[14] AND (ps_in[12] OR ps_ou[11]); -- -4.2
el[ 213] = off[ 5] AND h[14] AND (ps_in[11] OR ps_ou[11]); -- -5.3
el[ 214] = off[ 6] AND h[14] AND (ps_in[11] OR ps_ou[11]); -- -7.0
el[ 215] = off[ 7] AND h[14] AND (ps_in[11] OR ps_ou[11]); -- -10.5
el[ 216] = off[ 8] AND h[14] AND (ps_in[11] OR ps_ou[11]); -- -21.0
el[ 217] = off[ 9] AND h[14] AND (ps_in[11] OR ps_ou[11]); -- 99.0
el[ 218] = off[10] AND h[14] AND (ps_in[11] OR ps_ou[11]); -- 21.0
el[ 219] = off[11] AND h[14] AND (ps_in[11] OR ps_ou[11]); -- 10.5
el[ 220] = off[12] AND h[14] AND (ps_in[11] OR ps_ou[10]); -- 7.0
```

```
el[ 221] = off[13] AND h[14] AND (ps_in[11] OR ps_ou[10]); -- 5.3
el[ 222] = off[14] AND h[14] AND (ps_in[10] OR ps_ou[10]); -- 4.2
el[ 223] = off[15] AND h[14] AND (ps_in[10] OR ps_ou[10]); -- 3.5
el[ 224] = off[16] AND h[14] AND (ps_in[10] OR ps_ou[ 9]); -- 3.0
el[ 225] = off[ 1] AND h[15] AND (ps_in[12] OR ps_ou[12]); -- -2.6
el[ 226] = off[ 2] AND h[15] AND (ps_in[12] OR ps_ou[12]); -- -3.0
el[ 227] = off[ 3] AND h[15] AND (ps_in[12] OR ps_ou[11]); -- -3.5
el[ 228] = off[ 4] AND h[15] AND (ps_in[12] OR ps_ou[11]); -- -4.2
el[ 229] = off[ 5] AND h[15] AND (ps_in[11] OR ps_ou[11]); -- -5.3
el[ 230] = off[ 6] AND h[15] AND (ps_in[11] OR ps_ou[11]); -- -7.0
el[ 231] = off[ 7] AND h[15] AND (ps_in[11] OR ps_ou[11]); -- -10.5
el[ 232] = off[ 8] AND h[15] AND (ps_in[11] OR ps_ou[11]); -- -21.0
el[ 233] = off[ 9] AND h[15] AND (ps_in[11] OR ps_ou[11]); -- 99.0
el[ 234] = off[10] AND h[15] AND (ps_in[11] OR ps_ou[11]); -- 21.0
el[ 235] = off[11] AND h[15] AND (ps_in[11] OR ps_ou[11]); -- 10.5
el[ 236] = off[12] AND h[15] AND (ps_in[11] OR ps_ou[10]); -- 7.0
el[ 237] = off[13] AND h[15] AND (ps_in[11] OR ps_ou[10]); -- 5.3
el[ 238] = off[14] AND h[15] AND (ps_in[10] OR ps_ou[10]); -- 4.2
el[ 239] = off[15] AND h[15] AND (ps_in[10] OR ps_ou[10]); -- 3.5
el[ 240] = off[16] AND h[15] AND (ps_in[10] OR ps_ou[ 9]); -- 3.0
el[ 241] = off[ 1] AND h[16] AND (ps_in[12] OR ps_ou[12]); -- -2.6
el[ 242] = off[ 2] AND h[16] AND (ps_in[12] OR ps_ou[12]); -- -3.0
el[ 243] = off[ 3] AND h[16] AND (ps_in[12] OR ps_ou[11]); -- -3.5
el[ 244] = off[ 4] AND h[16] AND (ps_in[12] OR ps_ou[11]); -- -4.2
el[ 245] = off[ 5] AND h[16] AND (ps_in[11] OR ps_ou[11]); -- -5.3
el[ 246] = off[ 6] AND h[16] AND (ps_in[11] OR ps_ou[11]); -- -7.0
el[ 247] = off[ 7] AND h[16] AND (ps_in[11] OR ps_ou[11]); -- -10.5
el[ 248] = off[ 8] AND h[16] AND (ps_in[11] OR ps_ou[11]); -- -21.0
el[ 249] = off[ 9] AND h[16] AND (ps_in[11] OR ps_ou[11]); -- 99.0
el[ 250] = off[10] AND h[16] AND (ps_in[11] OR ps_ou[11]); -- 21.0
el[ 251] = off[11] AND h[16] AND (ps_in[11] OR ps_ou[11]); -- 10.5
el[ 252] = off[12] AND h[16] AND (ps_in[11] OR ps_ou[10]); -- 7.0
el[ 253] = off[13] AND h[16] AND (ps_in[11] OR ps_ou[10]); -- 5.3
el[ 254] = off[14] AND h[16] AND (ps_in[10] OR ps_ou[10]); -- 4.2
el[ 255] = off[15] AND h[16] AND (ps_in[10] OR ps_ou[10]); -- 3.5
el[ 256] = off[16] AND h[16] AND (ps_in[10] OR ps_ou[ 9]); -- 3.0
el[ 257] = off[ 1] AND h[17] AND (ps_in[12] OR ps_ou[12]); -- -2.6
```

```
el[ 258] = off[ 2] AND h[17] AND (ps_in[12] OR ps_ou[12]); -- -3.0
el[ 259] = off[ 3] AND h[17] AND (ps_in[12] OR ps_ou[11]); -- -3.5
el[ 260] = off[ 4] AND h[17] AND (ps_in[12] OR ps_ou[11]); -- -4.2
el[ 261] = off[ 5] AND h[17] AND (ps_in[11] OR ps_ou[11]); -- -5.3
el[ 262] = off[ 6] AND h[17] AND (ps_in[11] OR ps_ou[11]); -- -7.0
el[ 263] = off[ 7] AND h[17] AND (ps_in[11] OR ps_ou[11]); -- -10.5
el[ 264] = off[ 8] AND h[17] AND (ps_in[11] OR ps_ou[11]); -- -21.0
el[ 265] = off[ 9] AND h[17] AND (ps_in[11] OR ps_ou[11]); -- 99.0
el[ 266] = off[10] AND h[17] AND (ps_in[11] OR ps_ou[11]); -- 21.0
el[ 267] = off[11] AND h[17] AND (ps_in[11] OR ps_ou[11]); -- 10.5
el[ 268] = off[12] AND h[17] AND (ps_in[11] OR ps_ou[10]); -- 7.0
el[ 269] = off[13] AND h[17] AND (ps_in[11] OR ps_ou[10]); -- 5.3
el[ 270] = off[14] AND h[17] AND (ps_in[10] OR ps_ou[10]); -- 4.2
el[ 271] = off[15] AND h[17] AND (ps_in[10] OR ps_ou[10]); -- 3.5
el[ 272] = off[16] AND h[17] AND (ps_in[10] OR ps_ou[ 9]); -- 3.0
el[ 273] = off[ 1] AND h[18] AND (ps_in[13] OR ps_ou[13]); -- -2.6
el[ 274] = off[ 2] AND h[18] AND (ps_in[13] OR ps_ou[13]); -- -3.0
el[ 275] = off[ 3] AND h[18] AND (ps_in[13] OR ps_ou[12]); -- -3.5
el[ 276] = off[ 4] AND h[18] AND (ps_in[13] OR ps_ou[12]); -- -4.2
el[ 277] = off[ 5] AND h[18] AND (ps_in[12] OR ps_ou[12]); -- -5.3
el[ 278] = off[ 6] AND h[18] AND (ps_in[12] OR ps_ou[12]); -- -7.0
el[ 279] = off[ 7] AND h[18] AND (ps_in[12] OR ps_ou[12]); -- -10.5
el[ 280] = off[ 8] AND h[18] AND (ps_in[12] OR ps_ou[12]); -- -21.0
el[ 281] = off[ 9] AND h[18] AND (ps_in[12] OR ps_ou[12]); -- 99.0
el[ 282] = off[10] AND h[18] AND (ps_in[12] OR ps_ou[12]); -- 21.0
el[ 283] = off[11] AND h[18] AND (ps_in[12] OR ps_ou[12]); -- 10.5
el[ 284] = off[12] AND h[18] AND (ps_in[12] OR ps_ou[11]); -- 7.0
el[ 285] = off[13] AND h[18] AND (ps_in[12] OR ps_ou[11]); -- 5.3
el[ 286] = off[14] AND h[18] AND (ps_in[11] OR ps_ou[11]); -- 4.2
el[ 287] = off[15] AND h[18] AND (ps_in[11] OR ps_ou[11]); -- 3.5
el[ 288] = off[16] AND h[18] AND (ps_in[11] OR ps_ou[10]); -- 3.0
el[ 289] = off[ 1] AND h[19] AND (ps_in[13] OR ps_ou[13]); -- -2.6
el[ 290] = off[ 2] AND h[19] AND (ps_in[13] OR ps_ou[13]); -- -3.0
el[ 291] = off[ 3] AND h[19] AND (ps_in[13] OR ps_ou[12]); -- -3.5
el[ 292] = off[ 4] AND h[19] AND (ps_in[13] OR ps_ou[12]); -- -4.2
el[ 293] = off[ 5] AND h[19] AND (ps_in[12] OR ps_ou[12]); -- -5.3
el[ 294] = off[ 6] AND h[19] AND (ps_in[12] OR ps_ou[12]); -- -7.0
```

```
el[ 295] = off[ 7] AND h[19] AND (ps_in[12] OR ps_ou[12]); -- -10.5
el[ 296] = off[ 8] AND h[19] AND (ps_in[12] OR ps_ou[12]); -- -21.0
el[ 297] = off[ 9] AND h[19] AND (ps_in[12] OR ps_ou[12]); -- 99.0
el[ 298] = off[10] AND h[19] AND (ps_in[12] OR ps_ou[12]); -- 21.0
el[ 299] = off[11] AND h[19] AND (ps_in[12] OR ps_ou[12]); -- 10.5
el[ 300] = off[12] AND h[19] AND (ps_in[12] OR ps_ou[11]); -- 7.0
el[ 301] = off[13] AND h[19] AND (ps_in[12] OR ps_ou[11]); -- 5.3
el[ 302] = off[14] AND h[19] AND (ps_in[11] OR ps_ou[11]); -- 4.2
el[ 303] = off[15] AND h[19] AND (ps_in[11] OR ps_ou[11]); -- 3.5
el[ 304] = off[16] AND h[19] AND (ps_in[11] OR ps_ou[10]); -- 3.0
el[ 305] = off[ 1] AND h[20] AND (ps_in[13] OR ps_ou[13]); -- -2.6
el[ 306] = off[ 2] AND h[20] AND (ps_in[13] OR ps_ou[13]); -- -3.0
el[ 307] = off[ 3] AND h[20] AND (ps_in[13] OR ps_ou[12]); -- -3.5
el[ 308] = off[ 4] AND h[20] AND (ps_in[13] OR ps_ou[12]); -- -4.2
el[ 309] = off[ 5] AND h[20] AND (ps_in[12] OR ps_ou[12]); -- -5.3
el[ 310] = off[ 6] AND h[20] AND (ps_in[12] OR ps_ou[12]); -- -7.0
el[ 311] = off[ 7] AND h[20] AND (ps_in[12] OR ps_ou[12]); -- -10.5
el[ 312] = off[ 8] AND h[20] AND (ps_in[12] OR ps_ou[12]); -- -21.0
el[ 313] = off[ 9] AND h[20] AND (ps_in[12] OR ps_ou[12]); -- 99.0
el[ 314] = off[10] AND h[20] AND (ps_in[12] OR ps_ou[12]); -- 21.0
el[ 315] = off[11] AND h[20] AND (ps_in[12] OR ps_ou[12]); -- 10.5
el[ 316] = off[12] AND h[20] AND (ps_in[12] OR ps_ou[11]); -- 7.0
el[ 317] = off[13] AND h[20] AND (ps_in[12] OR ps_ou[11]); -- 5.3
el[ 318] = off[14] AND h[20] AND (ps_in[11] OR ps_ou[11]); -- 4.2
el[ 319] = off[15] AND h[20] AND (ps_in[11] OR ps_ou[11]); -- 3.5
el[ 320] = off[16] AND h[20] AND (ps_in[11] OR ps_ou[10]); -- 3.0
el[ 321] = off[ 1] AND h[21] AND (ps_in[13] OR ps_ou[13]); -- -2.6
el[ 322] = off[ 2] AND h[21] AND (ps_in[13] OR ps_ou[13]); -- -3.0
el[ 323] = off[ 3] AND h[21] AND (ps_in[13] OR ps_ou[12]); -- -3.5
el[ 324] = off[ 4] AND h[21] AND (ps_in[13] OR ps_ou[12]); -- -4.2
el[ 325] = off[ 5] AND h[21] AND (ps_in[12] OR ps_ou[12]); -- -5.3
el[ 326] = off[ 6] AND h[21] AND (ps_in[12] OR ps_ou[12]); -- -7.0
el[ 327] = off[ 7] AND h[21] AND (ps_in[12] OR ps_ou[12]); -- -10.5
el[ 328] = off[ 8] AND h[21] AND (ps_in[12] OR ps_ou[12]); -- -21.0
el[ 329] = off[ 9] AND h[21] AND (ps_in[12] OR ps_ou[12]); -- 99.0
el[ 330] = off[10] AND h[21] AND (ps_in[12] OR ps_ou[12]); -- 21.0
el[ 331] = off[11] AND h[21] AND (ps_in[12] OR ps_ou[12]); -- 10.5
```

```
el[ 332] = off[12] AND h[21] AND (ps_in[12] OR ps_ou[11]); -- 7.0
el[ 333] = off[13] AND h[21] AND (ps_in[12] OR ps_ou[11]); -- 5.3
el[ 334] = off[14] AND h[21] AND (ps_in[11] OR ps_ou[11]); -- 4.2
el[ 335] = off[15] AND h[21] AND (ps_in[11] OR ps_ou[11]); -- 3.5
el[ 336] = off[16] AND h[21] AND (ps_in[11] OR ps_ou[10]); -- 3.0
el[ 337] = off[ 1] AND h[22] AND (ps_in[13] OR ps_ou[13]); -- -2.6
el[ 338] = off[ 2] AND h[22] AND (ps_in[13] OR ps_ou[13]); -- -3.0
el[ 339] = off[ 3] AND h[22] AND (ps_in[13] OR ps_ou[12]); -- -3.5
el[ 340] = off[ 4] AND h[22] AND (ps_in[13] OR ps_ou[12]); -- -4.2
el[ 341] = off[ 5] AND h[22] AND (ps_in[12] OR ps_ou[12]); -- -5.3
el[ 342] = off[ 6] AND h[22] AND (ps_in[12] OR ps_ou[12]); -- -7.0
el[ 343] = off[ 7] AND h[22] AND (ps_in[12] OR ps_ou[12]); -- -10.5
el[ 344] = off[ 8] AND h[22] AND (ps_in[12] OR ps_ou[12]); -- -21.0
el[ 345] = off[ 9] AND h[22] AND (ps_in[12] OR ps_ou[12]); -- 99.0
el[ 346] = off[10] AND h[22] AND (ps_in[12] OR ps_ou[12]); -- 21.0
el[ 347] = off[11] AND h[22] AND (ps_in[12] OR ps_ou[12]); -- 10.5
el[ 348] = off[12] AND h[22] AND (ps_in[12] OR ps_ou[11]); -- 7.0
el[ 349] = off[13] AND h[22] AND (ps_in[12] OR ps_ou[11]); -- 5.3
el[ 350] = off[14] AND h[22] AND (ps_in[11] OR ps_ou[11]); -- 4.2
el[ 351] = off[15] AND h[22] AND (ps_in[11] OR ps_ou[11]); -- 3.5
el[ 352] = off[16] AND h[22] AND (ps_in[11] OR ps_ou[10]); -- 3.0
el[ 353] = off[ 1] AND h[23] AND (ps_in[14] OR ps_ou[14]); -- -2.6
el[ 354] = off[ 2] AND h[23] AND (ps_in[14] OR ps_ou[14]); -- -3.0
el[ 355] = off[ 3] AND h[23] AND (ps_in[14] OR ps_ou[13]); -- -3.5
el[ 356] = off[ 4] AND h[23] AND (ps_in[14] OR ps_ou[13]); -- -4.2
el[ 357] = off[ 5] AND h[23] AND (ps_in[13] OR ps_ou[13]); -- -5.3
el[ 358] = off[ 6] AND h[23] AND (ps_in[13] OR ps_ou[13]); -- -7.0
el[ 359] = off[ 7] AND h[23] AND (ps_in[13] OR ps_ou[13]); -- -10.5
el[ 360] = off[ 8] AND h[23] AND (ps_in[13] OR ps_ou[13]); -- -21.0
el[ 361] = off[ 9] AND h[23] AND (ps_in[13] OR ps_ou[13]); -- 99.0
el[ 362] = off[10] AND h[23] AND (ps_in[13] OR ps_ou[13]); -- 21.0
el[ 363] = off[11] AND h[23] AND (ps_in[13] OR ps_ou[13]); -- 10.5
el[ 364] = off[12] AND h[23] AND (ps_in[13] OR ps_ou[12]); -- 7.0
el[ 365] = off[13] AND h[23] AND (ps_in[13] OR ps_ou[12]); -- 5.3
el[ 366] = off[14] AND h[23] AND (ps_in[12] OR ps_ou[12]); -- 4.2
el[ 367] = off[15] AND h[23] AND (ps_in[12] OR ps_ou[12]); -- 3.5
el[ 368] = off[16] AND h[23] AND (ps_in[12] OR ps_ou[11]); -- 3.0
```

```
el[ 369] = off[ 1] AND h[24] AND (ps_in[14] OR ps_ou[14]); -- -2.6
el[ 370] = off[ 2] AND h[24] AND (ps_in[14] OR ps_ou[14]); -- -3.0
el[ 371] = off[ 3] AND h[24] AND (ps_in[14] OR ps_ou[13]); -- -3.5
el[ 372] = off[ 4] AND h[24] AND (ps_in[14] OR ps_ou[13]); -- -4.2
el[ 373] = off[ 5] AND h[24] AND (ps_in[13] OR ps_ou[13]); -- -5.3
el[ 374] = off[ 6] AND h[24] AND (ps_in[13] OR ps_ou[13]); -- -7.0
el[ 375] = off[ 7] AND h[24] AND (ps_in[13] OR ps_ou[13]); -- -10.5
el[ 376] = off[ 8] AND h[24] AND (ps_in[13] OR ps_ou[13]); -- -21.0
el[ 377] = off[ 9] AND h[24] AND (ps_in[13] OR ps_ou[13]); -- 99.0
el[ 378] = off[10] AND h[24] AND (ps_in[13] OR ps_ou[13]); -- 21.0
el[ 379] = off[11] AND h[24] AND (ps_in[13] OR ps_ou[13]); -- 10.5
el[ 380] = off[12] AND h[24] AND (ps_in[13] OR ps_ou[12]); -- 7.0
el[ 381] = off[13] AND h[24] AND (ps_in[13] OR ps_ou[12]); -- 5.3
el[ 382] = off[14] AND h[24] AND (ps_in[12] OR ps_ou[12]); -- 4.2
el[ 383] = off[15] AND h[24] AND (ps_in[12] OR ps_ou[12]); -- 3.5
el[ 384] = off[16] AND h[24] AND (ps_in[12] OR ps_ou[11]); -- 3.0
el[ 385] = off[ 1] AND h[25] AND (ps_in[14] OR ps_ou[14]); -- -2.6
el[ 386] = off[ 2] AND h[25] AND (ps_in[14] OR ps_ou[14]); -- -3.0
el[ 387] = off[ 3] AND h[25] AND (ps_in[14] OR ps_ou[13]); -- -3.5
el[ 388] = off[ 4] AND h[25] AND (ps_in[14] OR ps_ou[13]); -- -4.2
el[ 389] = off[ 5] AND h[25] AND (ps_in[13] OR ps_ou[13]); -- -5.3
el[ 390] = off[ 6] AND h[25] AND (ps_in[13] OR ps_ou[13]); -- -7.0
el[ 391] = off[ 7] AND h[25] AND (ps_in[13] OR ps_ou[13]); -- -10.5
el[ 392] = off[ 8] AND h[25] AND (ps_in[13] OR ps_ou[13]); -- -21.0
el[ 393] = off[ 9] AND h[25] AND (ps_in[13] OR ps_ou[13]); -- 99.0
el[ 394] = off[10] AND h[25] AND (ps_in[13] OR ps_ou[13]); -- 21.0
el[ 395] = off[11] AND h[25] AND (ps_in[13] OR ps_ou[13]); -- 10.5
el[ 396] = off[12] AND h[25] AND (ps_in[13] OR ps_ou[12]); -- 7.0
el[ 397] = off[13] AND h[25] AND (ps_in[13] OR ps_ou[12]); -- 5.3
el[ 398] = off[14] AND h[25] AND (ps_in[12] OR ps_ou[12]); -- 4.2
el[ 399] = off[15] AND h[25] AND (ps_in[12] OR ps_ou[12]); -- 3.5
el[ 400] = off[16] AND h[25] AND (ps_in[12] OR ps_ou[11]); -- 3.0
el[ 401] = off[ 1] AND h[26] AND (ps_in[14] OR ps_ou[14]); -- -2.6
el[ 402] = off[ 2] AND h[26] AND (ps_in[14] OR ps_ou[14]); -- -3.0
el[ 403] = off[ 3] AND h[26] AND (ps_in[14] OR ps_ou[13]); -- -3.5
el[ 404] = off[ 4] AND h[26] AND (ps_in[14] OR ps_ou[13]); -- -4.2
el[ 405] = off[ 5] AND h[26] AND (ps_in[13] OR ps_ou[13]); -- -5.3
```

```
el[ 406] = off[ 6] AND h[26] AND (ps_in[13] OR ps_ou[13]); -- -7.0
el[ 407] = off[ 7] AND h[26] AND (ps_in[13] OR ps_ou[13]); -- -10.5
el[ 408] = off[ 8] AND h[26] AND (ps_in[13] OR ps_ou[13]); -- -21.0
el[ 409] = off[ 9] AND h[26] AND (ps_in[13] OR ps_ou[13]); -- 99.0
el[ 410] = off[10] AND h[26] AND (ps_in[13] OR ps_ou[13]); -- 21.0
el[ 411] = off[11] AND h[26] AND (ps_in[13] OR ps_ou[13]); -- 10.5
el[ 412] = off[12] AND h[26] AND (ps_in[13] OR ps_ou[12]); -- 7.0
el[ 413] = off[13] AND h[26] AND (ps_in[13] OR ps_ou[12]); -- 5.3
el[ 414] = off[14] AND h[26] AND (ps_in[12] OR ps_ou[12]); -- 4.2
el[ 415] = off[15] AND h[26] AND (ps_in[12] OR ps_ou[12]); -- 3.5
el[ 416] = off[16] AND h[26] AND (ps_in[12] OR ps_ou[11]); -- 3.0
el[ 417] = off[ 1] AND h[27] AND (ps_in[14] OR ps_ou[14]); -- -2.6
el[ 418] = off[ 2] AND h[27] AND (ps_in[14] OR ps_ou[14]); -- -3.0
el[ 419] = off[ 3] AND h[27] AND (ps_in[14] OR ps_ou[13]); -- -3.5
el[ 420] = off[ 4] AND h[27] AND (ps_in[14] OR ps_ou[13]); -- -4.2
el[ 421] = off[ 5] AND h[27] AND (ps_in[13] OR ps_ou[13]); -- -5.3
el[ 422] = off[ 6] AND h[27] AND (ps_in[13] OR ps_ou[13]); -- -7.0
el[ 423] = off[ 7] AND h[27] AND (ps_in[13] OR ps_ou[13]); -- -10.5
el[ 424] = off[ 8] AND h[27] AND (ps_in[13] OR ps_ou[13]); -- -21.0
el[ 425] = off[ 9] AND h[27] AND (ps_in[13] OR ps_ou[13]); -- 99.0
el[ 426] = off[10] AND h[27] AND (ps_in[13] OR ps_ou[13]); -- 21.0
el[ 427] = off[11] AND h[27] AND (ps_in[13] OR ps_ou[13]); -- 10.5
el[ 428] = off[12] AND h[27] AND (ps_in[13] OR ps_ou[12]); -- 7.0
el[ 429] = off[13] AND h[27] AND (ps_in[13] OR ps_ou[12]); -- 5.3
el[ 430] = off[14] AND h[27] AND (ps_in[12] OR ps_ou[12]); -- 4.2
el[ 431] = off[15] AND h[27] AND (ps_in[12] OR ps_ou[12]); -- 3.5
el[ 432] = off[16] AND h[27] AND (ps_in[12] OR ps_ou[11]); -- 3.0
el[ 433] = off[ 1] AND h[28] AND (ps_in[14] OR ps_ou[14]); -- -2.6
el[ 434] = off[ 2] AND h[28] AND (ps_in[14] OR ps_ou[14]); -- -3.0
el[ 435] = off[ 3] AND h[28] AND (ps_in[14] OR ps_ou[13]); -- -3.5
el[ 436] = off[ 4] AND h[28] AND (ps_in[14] OR ps_ou[13]); -- -4.2
el[ 437] = off[ 5] AND h[28] AND (ps_in[13] OR ps_ou[13]); -- -5.3
el[ 438] = off[ 6] AND h[28] AND (ps_in[13] OR ps_ou[13]); -- -7.0
el[ 439] = off[ 7] AND h[28] AND (ps_in[13] OR ps_ou[13]); -- -10.5
el[ 440] = off[ 8] AND h[28] AND (ps_in[13] OR ps_ou[13]); -- -21.0
el[ 441] = off[ 9] AND h[28] AND (ps_in[13] OR ps_ou[13]); -- 99.0
el[ 442] = off[10] AND h[28] AND (ps_in[13] OR ps_ou[13]); -- 21.0
```

```
el[ 443] = off[11] AND h[28] AND (ps_in[13] OR ps_ou[13]); -- 10.5
el[ 444] = off[12] AND h[28] AND (ps_in[13] OR ps_ou[12]); -- 7.0
el[ 445] = off[13] AND h[28] AND (ps_in[13] OR ps_ou[12]); -- 5.3
el[ 446] = off[14] AND h[28] AND (ps_in[12] OR ps_ou[12]); -- 4.2
el[ 447] = off[15] AND h[28] AND (ps_in[12] OR ps_ou[12]); -- 3.5
el[ 448] = off[16] AND h[28] AND (ps_in[12] OR ps_ou[11]); -- 3.0
el[ 449] = off[ 1] AND h[29] AND (ps_in[15] OR ps_ou[15]); -- -2.6
el[ 450] = off[ 2] AND h[29] AND (ps_in[15] OR ps_ou[15]); -- -3.0
el[ 451] = off[ 3] AND h[29] AND (ps_in[15] OR ps_ou[14]); -- -3.5
el[ 452] = off[ 4] AND h[29] AND (ps_in[15] OR ps_ou[14]); -- -4.2
el[ 453] = off[ 5] AND h[29] AND (ps_in[14] OR ps_ou[14]); -- -5.3
el[ 454] = off[ 6] AND h[29] AND (ps_in[14] OR ps_ou[14]); -- -7.0
el[ 455] = off[ 7] AND h[29] AND (ps_in[14] OR ps_ou[14]); -- -10.5
el[ 456] = off[ 8] AND h[29] AND (ps_in[14] OR ps_ou[14]); -- -21.0
el[ 457] = off[ 9] AND h[29] AND (ps_in[14] OR ps_ou[14]); -- 99.0
el[ 458] = off[10] AND h[29] AND (ps_in[14] OR ps_ou[14]); -- 21.0
el[ 459] = off[11] AND h[29] AND (ps_in[14] OR ps_ou[14]); -- 10.5
el[ 460] = off[12] AND h[29] AND (ps_in[14] OR ps_ou[13]); -- 7.0
el[ 461] = off[13] AND h[29] AND (ps_in[14] OR ps_ou[13]); -- 5.3
el[ 462] = off[14] AND h[29] AND (ps_in[13] OR ps_ou[13]); -- 4.2
el[ 463] = off[15] AND h[29] AND (ps_in[13] OR ps_ou[13]); -- 3.5
el[ 464] = off[16] AND h[29] AND (ps_in[13] OR ps_ou[12]); -- 3.0
el[ 465] = off[ 1] AND h[30] AND (ps_in[15] OR ps_ou[15]); -- -2.6
el[ 466] = off[ 2] AND h[30] AND (ps_in[15] OR ps_ou[15]); -- -3.0
el[ 467] = off[ 3] AND h[30] AND (ps_in[15] OR ps_ou[14]); -- -3.5
el[ 468] = off[ 4] AND h[30] AND (ps_in[15] OR ps_ou[14]); -- -4.2
el[ 469] = off[ 5] AND h[30] AND (ps_in[14] OR ps_ou[14]); -- -5.3
el[ 470] = off[ 6] AND h[30] AND (ps_in[14] OR ps_ou[14]); -- -7.0
el[ 471] = off[ 7] AND h[30] AND (ps_in[14] OR ps_ou[14]); -- -10.5
el[ 472] = off[ 8] AND h[30] AND (ps_in[14] OR ps_ou[14]); -- -21.0
el[ 473] = off[ 9] AND h[30] AND (ps_in[14] OR ps_ou[14]); -- 99.0
el[ 474] = off[10] AND h[30] AND (ps_in[14] OR ps_ou[14]); -- 21.0
el[ 475] = off[11] AND h[30] AND (ps_in[14] OR ps_ou[14]); -- 10.5
el[ 476] = off[12] AND h[30] AND (ps_in[14] OR ps_ou[13]); -- 7.0
el[ 477] = off[13] AND h[30] AND (ps_in[14] OR ps_ou[13]); -- 5.3
el[ 478] = off[14] AND h[30] AND (ps_in[13] OR ps_ou[13]); -- 4.2
el[ 479] = off[15] AND h[30] AND (ps_in[13] OR ps_ou[13]); -- 3.5
```

```
el[ 480] = off[16] AND h[30] AND (ps_in[13] OR ps_ou[12]); -- 3.0
el[ 481] = off[ 1] AND h[31] AND (ps_in[15] OR ps_ou[15]); -- -2.6
el[ 482] = off[ 2] AND h[31] AND (ps_in[15] OR ps_ou[15]); -- -3.0
el[ 483] = off[ 3] AND h[31] AND (ps_in[15] OR ps_ou[14]); -- -3.5
el[ 484] = off[ 4] AND h[31] AND (ps_in[15] OR ps_ou[14]); -- -4.2
el[ 485] = off[ 5] AND h[31] AND (ps_in[14] OR ps_ou[14]); -- -5.3
el[ 486] = off[ 6] AND h[31] AND (ps_in[14] OR ps_ou[14]); -- -7.0
el[ 487] = off[ 7] AND h[31] AND (ps_in[14] OR ps_ou[14]); -- -10.5
el[ 488] = off[ 8] AND h[31] AND (ps_in[14] OR ps_ou[14]); -- -21.0
el[ 489] = off[ 9] AND h[31] AND (ps_in[14] OR ps_ou[14]); -- 99.0
el[ 490] = off[10] AND h[31] AND (ps_in[14] OR ps_ou[14]); -- 21.0
el[ 491] = off[11] AND h[31] AND (ps_in[14] OR ps_ou[14]); -- 10.5
el[ 492] = off[12] AND h[31] AND (ps_in[14] OR ps_ou[13]); -- 7.0
el[ 493] = off[13] AND h[31] AND (ps_in[14] OR ps_ou[13]); -- 5.3
el[ 494] = off[14] AND h[31] AND (ps_in[13] OR ps_ou[13]); -- 4.2
el[ 495] = off[15] AND h[31] AND (ps_in[13] OR ps_ou[13]); -- 3.5
el[ 496] = off[16] AND h[31] AND (ps_in[13] OR ps_ou[12]); -- 3.0
el[ 497] = off[ 1] AND h[32] AND (ps_in[15] OR ps_ou[15]); -- -2.6
el[ 498] = off[ 2] AND h[32] AND (ps_in[15] OR ps_ou[15]); -- -3.0
el[ 499] = off[ 3] AND h[32] AND (ps_in[15] OR ps_ou[14]); -- -3.5
el[ 500] = off[ 4] AND h[32] AND (ps_in[15] OR ps_ou[14]); -- -4.2
el[ 501] = off[ 5] AND h[32] AND (ps_in[14] OR ps_ou[14]); -- -5.3
el[ 502] = off[ 6] AND h[32] AND (ps_in[14] OR ps_ou[14]); -- -7.0
el[ 503] = off[ 7] AND h[32] AND (ps_in[14] OR ps_ou[14]); -- -10.5
el[ 504] = off[ 8] AND h[32] AND (ps_in[14] OR ps_ou[14]); -- -21.0
el[ 505] = off[ 9] AND h[32] AND (ps_in[14] OR ps_ou[14]); -- 99.0
el[ 506] = off[10] AND h[32] AND (ps_in[14] OR ps_ou[14]); -- 21.0
el[ 507] = off[11] AND h[32] AND (ps_in[14] OR ps_ou[14]); -- 10.5
el[ 508] = off[12] AND h[32] AND (ps_in[14] OR ps_ou[13]); -- 7.0
el[ 509] = off[13] AND h[32] AND (ps_in[14] OR ps_ou[13]); -- 5.3
el[ 510] = off[14] AND h[32] AND (ps_in[13] OR ps_ou[13]); -- 4.2
el[ 511] = off[15] AND h[32] AND (ps_in[13] OR ps_ou[13]); -- 3.5
el[ 512] = off[16] AND h[32] AND (ps_in[13] OR ps_ou[12]); -- 3.0
el[ 513] = off[ 1] AND h[33] AND (ps_in[15] OR ps_ou[15]); -- -2.6
el[ 514] = off[ 2] AND h[33] AND (ps_in[15] OR ps_ou[15]); -- -3.0
el[ 515] = off[ 3] AND h[33] AND (ps_in[15] OR ps_ou[14]); -- -3.5
el[ 516] = off[ 4] AND h[33] AND (ps_in[15] OR ps_ou[14]); -- -4.2
```

```
el[ 517] = off[ 5] AND h[33] AND (ps_in[14] OR ps_ou[14]); -- -5.3
el[ 518] = off[ 6] AND h[33] AND (ps_in[14] OR ps_ou[14]); -- -7.0
el[ 519] = off[ 7] AND h[33] AND (ps_in[14] OR ps_ou[14]); -- -10.5
el[ 520] = off[ 8] AND h[33] AND (ps_in[14] OR ps_ou[14]); -- -21.0
el[ 521] = off[ 9] AND h[33] AND (ps_in[14] OR ps_ou[14]); -- 99.0
el[ 522] = off[10] AND h[33] AND (ps_in[14] OR ps_ou[14]); -- 21.0
el[ 523] = off[11] AND h[33] AND (ps_in[14] OR ps_ou[14]); -- 10.5
el[ 524] = off[12] AND h[33] AND (ps_in[14] OR ps_ou[13]); -- 7.0
el[ 525] = off[13] AND h[33] AND (ps_in[14] OR ps_ou[13]); -- 5.3
el[ 526] = off[14] AND h[33] AND (ps_in[13] OR ps_ou[13]); -- 4.2
el[ 527] = off[15] AND h[33] AND (ps_in[13] OR ps_ou[13]); -- 3.5
el[ 528] = off[16] AND h[33] AND (ps_in[13] OR ps_ou[12]); -- 3.0
el[ 529] = off[ 1] AND h[34] AND (ps_in[16] OR ps_ou[16]); -- -2.6
el[ 530] = off[ 2] AND h[34] AND (ps_in[16] OR ps_ou[16]); -- -3.0
el[ 531] = off[ 3] AND h[34] AND (ps_in[16] OR ps_ou[15]); -- -3.5
el[ 532] = off[ 4] AND h[34] AND (ps_in[16] OR ps_ou[15]); -- -4.2
el[ 533] = off[ 5] AND h[34] AND (ps_in[15] OR ps_ou[15]); -- -5.3
el[ 534] = off[ 6] AND h[34] AND (ps_in[15] OR ps_ou[15]); -- -7.0
el[ 535] = off[ 7] AND h[34] AND (ps_in[15] OR ps_ou[15]); -- -10.5
el[ 536] = off[ 8] AND h[34] AND (ps_in[15] OR ps_ou[15]); -- -21.0
el[ 537] = off[ 9] AND h[34] AND (ps_in[15] OR ps_ou[15]); -- 99.0
el[ 538] = off[10] AND h[34] AND (ps_in[15] OR ps_ou[15]); -- 21.0
el[ 539] = off[11] AND h[34] AND (ps_in[15] OR ps_ou[15]); -- 10.5
el[ 540] = off[12] AND h[34] AND (ps_in[15] OR ps_ou[14]); -- 7.0
el[ 541] = off[13] AND h[34] AND (ps_in[15] OR ps_ou[14]); -- 5.3
el[ 542] = off[14] AND h[34] AND (ps_in[14] OR ps_ou[14]); -- 4.2
el[ 543] = off[15] AND h[34] AND (ps_in[14] OR ps_ou[14]); -- 3.5
el[ 544] = off[16] AND h[34] AND (ps_in[14] OR ps_ou[13]); -- 3.0
el[ 545] = off[ 1] AND h[35] AND (ps_in[16] OR ps_ou[16]); -- -2.6
el[ 546] = off[ 2] AND h[35] AND (ps_in[16] OR ps_ou[16]); -- -3.0
el[ 547] = off[ 3] AND h[35] AND (ps_in[16] OR ps_ou[15]); -- -3.5
el[ 548] = off[ 4] AND h[35] AND (ps_in[16] OR ps_ou[15]); -- -4.2
el[ 549] = off[ 5] AND h[35] AND (ps_in[15] OR ps_ou[15]); -- -5.3
el[ 550] = off[ 6] AND h[35] AND (ps_in[15] OR ps_ou[15]); -- -7.0
el[ 551] = off[ 7] AND h[35] AND (ps_in[15] OR ps_ou[15]); -- -10.5
el[ 552] = off[ 8] AND h[35] AND (ps_in[15] OR ps_ou[15]); -- -21.0
el[ 553] = off[ 9] AND h[35] AND (ps_in[15] OR ps_ou[15]); -- 99.0
```

```
el[ 554] = off[10] AND h[35] AND (ps_in[15] OR ps_ou[15]); -- 21.0
el[ 555] = off[11] AND h[35] AND (ps_in[15] OR ps_ou[15]); -- 10.5
el[ 556] = off[12] AND h[35] AND (ps_in[15] OR ps_ou[14]); -- 7.0
el[ 557] = off[13] AND h[35] AND (ps_in[15] OR ps_ou[14]); -- 5.3
el[ 558] = off[14] AND h[35] AND (ps_in[14] OR ps_ou[14]); -- 4.2
el[ 559] = off[15] AND h[35] AND (ps_in[14] OR ps_ou[14]); -- 3.5
el[ 560] = off[16] AND h[35] AND (ps_in[14] OR ps_ou[13]); -- 3.0
el[ 561] = off[ 1] AND h[36] AND (ps_in[16] OR ps_ou[16]); -- -2.6
el[ 562] = off[ 2] AND h[36] AND (ps_in[16] OR ps_ou[16]); -- -3.0
el[ 563] = off[ 3] AND h[36] AND (ps_in[16] OR ps_ou[15]); -- -3.5
el[ 564] = off[ 4] AND h[36] AND (ps_in[16] OR ps_ou[15]); -- -4.2
el[ 565] = off[ 5] AND h[36] AND (ps_in[15] OR ps_ou[15]); -- -5.3
el[ 566] = off[ 6] AND h[36] AND (ps_in[15] OR ps_ou[15]); -- -7.0
el[ 567] = off[ 7] AND h[36] AND (ps_in[15] OR ps_ou[15]); -- -10.5
el[ 568] = off[ 8] AND h[36] AND (ps_in[15] OR ps_ou[15]); -- -21.0
el[ 569] = off[ 9] AND h[36] AND (ps_in[15] OR ps_ou[15]); -- 99.0
el[ 570] = off[10] AND h[36] AND (ps_in[15] OR ps_ou[15]); -- 21.0
el[ 571] = off[11] AND h[36] AND (ps_in[15] OR ps_ou[15]); -- 10.5
el[ 572] = off[12] AND h[36] AND (ps_in[15] OR ps_ou[14]); -- 7.0
el[ 573] = off[13] AND h[36] AND (ps_in[15] OR ps_ou[14]); -- 5.3
el[ 574] = off[14] AND h[36] AND (ps_in[14] OR ps_ou[14]); -- 4.2
el[ 575] = off[15] AND h[36] AND (ps_in[14] OR ps_ou[14]); -- 3.5
el[ 576] = off[16] AND h[36] AND (ps_in[14] OR ps_ou[13]); -- 3.0
el[ 577] = off[ 1] AND h[37] AND (ps_in[16] OR ps_ou[16]); -- -2.6
el[ 578] = off[ 2] AND h[37] AND (ps_in[16] OR ps_ou[16]); -- -3.0
el[ 579] = off[ 3] AND h[37] AND (ps_in[16] OR ps_ou[15]); -- -3.5
el[ 580] = off[ 4] AND h[37] AND (ps_in[16] OR ps_ou[15]); -- -4.2
el[ 581] = off[ 5] AND h[37] AND (ps_in[15] OR ps_ou[15]); -- -5.3
el[ 582] = off[ 6] AND h[37] AND (ps_in[15] OR ps_ou[15]); -- -7.0
el[ 583] = off[ 7] AND h[37] AND (ps_in[15] OR ps_ou[15]); -- -10.5
el[ 584] = off[ 8] AND h[37] AND (ps_in[15] OR ps_ou[15]); -- -21.0
el[ 585] = off[ 9] AND h[37] AND (ps_in[15] OR ps_ou[15]); -- 99.0
el[ 586] = off[10] AND h[37] AND (ps_in[15] OR ps_ou[15]); -- 21.0
el[ 587] = off[11] AND h[37] AND (ps_in[15] OR ps_ou[15]); -- 10.5
el[ 588] = off[12] AND h[37] AND (ps_in[15] OR ps_ou[14]); -- 7.0
el[ 589] = off[13] AND h[37] AND (ps_in[15] OR ps_ou[14]); -- 5.3
el[ 590] = off[14] AND h[37] AND (ps_in[14] OR ps_ou[14]); -- 4.2
```

```
el[ 591] = off[15] AND h[37] AND (ps_in[14] OR ps_ou[14]); -- 3.5
el[ 592] = off[16] AND h[37] AND (ps_in[14] OR ps_ou[13]); -- 3.0
el[ 593] = off[ 1] AND h[38] AND (ps_in[16] OR ps_ou[16]); -- -2.6
el[ 594] = off[ 2] AND h[38] AND (ps_in[16] OR ps_ou[16]); -- -3.0
el[ 595] = off[ 3] AND h[38] AND (ps_in[16] OR ps_ou[15]); -- -3.5
el[ 596] = off[ 4] AND h[38] AND (ps_in[16] OR ps_ou[15]); -- -4.2
el[ 597] = off[ 5] AND h[38] AND (ps_in[15] OR ps_ou[15]); -- -5.3
el[ 598] = off[ 6] AND h[38] AND (ps_in[15] OR ps_ou[15]); -- -7.0
el[ 599] = off[ 7] AND h[38] AND (ps_in[15] OR ps_ou[15]); -- -10.5
el[ 600] = off[ 8] AND h[38] AND (ps_in[15] OR ps_ou[15]); -- -21.0
el[ 601] = off[ 9] AND h[38] AND (ps_in[15] OR ps_ou[15]); -- 99.0
el[ 602] = off[10] AND h[38] AND (ps_in[15] OR ps_ou[15]); -- 21.0
el[ 603] = off[11] AND h[38] AND (ps_in[15] OR ps_ou[15]); -- 10.5
el[ 604] = off[12] AND h[38] AND (ps_in[15] OR ps_ou[14]); -- 7.0
el[ 605] = off[13] AND h[38] AND (ps_in[15] OR ps_ou[14]); -- 5.3
el[ 606] = off[14] AND h[38] AND (ps_in[14] OR ps_ou[14]); -- 4.2
el[ 607] = off[15] AND h[38] AND (ps_in[14] OR ps_ou[14]); -- 3.5
el[ 608] = off[16] AND h[38] AND (ps_in[14] OR ps_ou[13]); -- 3.0
el[ 609] = off[ 1] AND h[39] AND (ps_in[16] OR ps_ou[16]); -- -2.6
el[ 610] = off[ 2] AND h[39] AND (ps_in[16] OR ps_ou[16]); -- -3.0
el[ 611] = off[ 3] AND h[39] AND (ps_in[16] OR ps_ou[15]); -- -3.5
el[ 612] = off[ 4] AND h[39] AND (ps_in[16] OR ps_ou[15]); -- -4.2
el[ 613] = off[ 5] AND h[39] AND (ps_in[15] OR ps_ou[15]); -- -5.3
el[ 614] = off[ 6] AND h[39] AND (ps_in[15] OR ps_ou[15]); -- -7.0
el[ 615] = off[ 7] AND h[39] AND (ps_in[15] OR ps_ou[15]); -- -10.5
el[ 616] = off[ 8] AND h[39] AND (ps_in[15] OR ps_ou[15]); -- -21.0
el[ 617] = off[ 9] AND h[39] AND (ps_in[15] OR ps_ou[15]); -- 99.0
el[ 618] = off[10] AND h[39] AND (ps_in[15] OR ps_ou[15]); -- 21.0
el[ 619] = off[11] AND h[39] AND (ps_in[15] OR ps_ou[15]); -- 10.5
el[ 620] = off[12] AND h[39] AND (ps_in[15] OR ps_ou[14]); -- 7.0
el[ 621] = off[13] AND h[39] AND (ps_in[15] OR ps_ou[14]); -- 5.3
el[ 622] = off[14] AND h[39] AND (ps_in[14] OR ps_ou[14]); -- 4.2
el[ 623] = off[15] AND h[39] AND (ps_in[14] OR ps_ou[14]); -- 3.5
el[ 624] = off[16] AND h[39] AND (ps_in[14] OR ps_ou[13]); -- 3.0
el[ 625] = off[ 1] AND h[40] AND (ps_in[17] OR ps_ou[17]); -- -2.6
el[ 626] = off[ 2] AND h[40] AND (ps_in[17] OR ps_ou[17]); -- -3.0
el[ 627] = off[ 3] AND h[40] AND (ps_in[17] OR ps_ou[16]); -- -3.5
```

```
el[ 628] = off[ 4] AND h[40] AND (ps_in[17] OR ps_ou[16]); -- -4.2
el[ 629] = off[ 5] AND h[40] AND (ps_in[16] OR ps_ou[16]); -- -5.3
el[ 630] = off[ 6] AND h[40] AND (ps_in[16] OR ps_ou[16]); -- -7.0
el[ 631] = off[ 7] AND h[40] AND (ps_in[16] OR ps_ou[16]); -- -10.5
el[ 632] = off[ 8] AND h[40] AND (ps_in[16] OR ps_ou[16]); -- -21.0
el[ 633] = off[ 9] AND h[40] AND (ps_in[16] OR ps_ou[16]); -- 99.0
el[ 634] = off[10] AND h[40] AND (ps_in[16] OR ps_ou[16]); -- 21.0
el[ 635] = off[11] AND h[40] AND (ps_in[16] OR ps_ou[16]); -- 10.5
el[ 636] = off[12] AND h[40] AND (ps_in[16] OR ps_ou[15]); -- 7.0
el[ 637] = off[13] AND h[40] AND (ps_in[16] OR ps_ou[15]); -- 5.3
el[ 638] = off[14] AND h[40] AND (ps_in[15] OR ps_ou[15]); -- 4.2
el[ 639] = off[15] AND h[40] AND (ps_in[15] OR ps_ou[15]); -- 3.5
el[ 640] = off[16] AND h[40] AND (ps_in[15] OR ps_ou[14]); -- 3.0
el[ 641] = off[ 1] AND h[41] AND (ps_in[17] OR ps_ou[17]); -- -2.6
el[ 642] = off[ 2] AND h[41] AND (ps_in[17] OR ps_ou[17]); -- -3.0
el[ 643] = off[ 3] AND h[41] AND (ps_in[17] OR ps_ou[16]); -- -3.5
el[ 644] = off[ 4] AND h[41] AND (ps_in[17] OR ps_ou[16]); -- -4.2
el[ 645] = off[ 5] AND h[41] AND (ps_in[16] OR ps_ou[16]); -- -5.3
el[ 646] = off[ 6] AND h[41] AND (ps_in[16] OR ps_ou[16]); -- -7.0
el[ 647] = off[ 7] AND h[41] AND (ps_in[16] OR ps_ou[16]); -- -10.5
el[ 648] = off[ 8] AND h[41] AND (ps_in[16] OR ps_ou[16]); -- -21.0
el[ 649] = off[ 9] AND h[41] AND (ps_in[16] OR ps_ou[16]); -- 99.0
el[ 650] = off[10] AND h[41] AND (ps_in[16] OR ps_ou[16]); -- 21.0
el[ 651] = off[11] AND h[41] AND (ps_in[16] OR ps_ou[16]); -- 10.5
el[ 652] = off[12] AND h[41] AND (ps_in[16] OR ps_ou[15]); -- 7.0
el[ 653] = off[13] AND h[41] AND (ps_in[16] OR ps_ou[15]); -- 5.3
el[ 654] = off[14] AND h[41] AND (ps_in[15] OR ps_ou[15]); -- 4.2
el[ 655] = off[15] AND h[41] AND (ps_in[15] OR ps_ou[15]); -- 3.5
el[ 656] = off[16] AND h[41] AND (ps_in[15] OR ps_ou[14]); -- 3.0
el[ 657] = off[ 1] AND h[42] AND (ps_in[17] OR ps_ou[17]); -- -2.6
el[ 658] = off[ 2] AND h[42] AND (ps_in[17] OR ps_ou[17]); -- -3.0
el[ 659] = off[ 3] AND h[42] AND (ps_in[17] OR ps_ou[16]); -- -3.5
el[ 660] = off[ 4] AND h[42] AND (ps_in[17] OR ps_ou[16]); -- -4.2
el[ 661] = off[ 5] AND h[42] AND (ps_in[16] OR ps_ou[16]); -- -5.3
el[ 662] = off[ 6] AND h[42] AND (ps_in[16] OR ps_ou[16]); -- -7.0
el[ 663] = off[ 7] AND h[42] AND (ps_in[16] OR ps_ou[16]); -- -10.5
el[ 664] = off[ 8] AND h[42] AND (ps_in[16] OR ps_ou[16]); -- -21.0
```

```
el[ 665] = off[ 9] AND h[42] AND (ps_in[16] OR ps_ou[16]); -- 99.0
el[ 666] = off[10] AND h[42] AND (ps_in[16] OR ps_ou[16]); -- 21.0
el[ 667] = off[11] AND h[42] AND (ps_in[16] OR ps_ou[16]); -- 10.5
el[ 668] = off[12] AND h[42] AND (ps_in[16] OR ps_ou[15]); -- 7.0
el[ 669] = off[13] AND h[42] AND (ps_in[16] OR ps_ou[15]); -- 5.3
el[ 670] = off[14] AND h[42] AND (ps_in[15] OR ps_ou[15]); -- 4.2
el[ 671] = off[15] AND h[42] AND (ps_in[15] OR ps_ou[15]); -- 3.5
el[ 672] = off[16] AND h[42] AND (ps_in[15] OR ps_ou[14]); -- 3.0
el[ 673] = off[ 1] AND h[43] AND (ps_in[17] OR ps_ou[17]); -- -2.6
el[ 674] = off[ 2] AND h[43] AND (ps_in[17] OR ps_ou[17]); -- -3.0
el[ 675] = off[ 3] AND h[43] AND (ps_in[17] OR ps_ou[16]); -- -3.5
el[ 676] = off[ 4] AND h[43] AND (ps_in[17] OR ps_ou[16]); -- -4.2
el[ 677] = off[ 5] AND h[43] AND (ps_in[16] OR ps_ou[16]); -- -5.3
el[ 678] = off[ 6] AND h[43] AND (ps_in[16] OR ps_ou[16]); -- -7.0
el[ 679] = off[ 7] AND h[43] AND (ps_in[16] OR ps_ou[16]); -- -10.5
el[ 680] = off[ 8] AND h[43] AND (ps_in[16] OR ps_ou[16]); -- -21.0
el[ 681] = off[ 9] AND h[43] AND (ps_in[16] OR ps_ou[16]); -- 99.0
el[ 682] = off[10] AND h[43] AND (ps_in[16] OR ps_ou[16]); -- 21.0
el[ 683] = off[11] AND h[43] AND (ps_in[16] OR ps_ou[16]); -- 10.5
el[ 684] = off[12] AND h[43] AND (ps_in[16] OR ps_ou[15]); -- 7.0
el[ 685] = off[13] AND h[43] AND (ps_in[16] OR ps_ou[15]); -- 5.3
el[ 686] = off[14] AND h[43] AND (ps_in[15] OR ps_ou[15]); -- 4.2
el[ 687] = off[15] AND h[43] AND (ps_in[15] OR ps_ou[15]); -- 3.5
el[ 688] = off[16] AND h[43] AND (ps_in[15] OR ps_ou[14]); -- 3.0
el[ 689] = off[ 1] AND h[44] AND (ps_in[17] OR ps_ou[17]); -- -2.6
el[ 690] = off[ 2] AND h[44] AND (ps_in[17] OR ps_ou[17]); -- -3.0
el[ 691] = off[ 3] AND h[44] AND (ps_in[17] OR ps_ou[16]); -- -3.5
el[ 692] = off[ 4] AND h[44] AND (ps_in[17] OR ps_ou[16]); -- -4.2
el[ 693] = off[ 5] AND h[44] AND (ps_in[16] OR ps_ou[16]); -- -5.3
el[ 694] = off[ 6] AND h[44] AND (ps_in[16] OR ps_ou[16]); -- -7.0
el[ 695] = off[ 7] AND h[44] AND (ps_in[16] OR ps_ou[16]); -- -10.5
el[ 696] = off[ 8] AND h[44] AND (ps_in[16] OR ps_ou[16]); -- -21.0
el[ 697] = off[ 9] AND h[44] AND (ps_in[16] OR ps_ou[16]); -- 99.0
el[ 698] = off[10] AND h[44] AND (ps_in[16] OR ps_ou[16]); -- 21.0
el[ 699] = off[11] AND h[44] AND (ps_in[16] OR ps_ou[16]); -- 10.5
el[ 700] = off[12] AND h[44] AND (ps_in[16] OR ps_ou[15]); -- 7.0
el[ 701] = off[13] AND h[44] AND (ps_in[16] OR ps_ou[15]); -- 5.3
```

el[ 702] = off[14] AND h[44] AND (ps\_in[15] OR ps\_ou[15]); -- 4.2  
 el[ 703] = off[15] AND h[44] AND (ps\_in[15] OR ps\_ou[15]); -- 3.5  
 el[ 704] = off[16] AND h[44] AND (ps\_in[15] OR ps\_ou[14]); -- 3.0

-----  
 el1\_[0] = el[1] OR el[2] OR el[3] OR el[4] OR el[5] OR el[6] OR el[7] OR el[8] OR el[9];  
 el1\_[1] = el[10] OR el[11] OR el[12] OR el[13] OR el[14] OR el[15] OR el[16] OR el[17] OR el[18] OR el[19];  
 el1\_[2] = el[20] OR el[21] OR el[22] OR el[23] OR el[24] OR el[25] OR el[26] OR el[27] OR el[28] OR el[29];  
 el1\_[3] = el[30] OR el[31] OR el[32] OR el[33] OR el[34] OR el[35] OR el[36] OR el[37] OR el[38] OR el[39];  
 el1\_[4] = el[40] OR el[41] OR el[42] OR el[43] OR el[44] OR el[45] OR el[46] OR el[47] OR el[48] OR el[49];  
 el1\_[5] = el[50] OR el[51] OR el[52] OR el[53] OR el[54] OR el[55] OR el[56] OR el[57] OR el[58] OR el[59];  
 el1\_[6] = el[60] OR el[61] OR el[62] OR el[63] OR el[64] OR el[65] OR el[66] OR el[67] OR el[68] OR el[69];  
 el1\_[7] = el[70] OR el[71] OR el[72] OR el[73] OR el[74] OR el[75] OR el[76] OR el[77] OR el[78] OR el[79];  
 el1\_[8] = el[80] OR el[81] OR el[82] OR el[83] OR el[84] OR el[85] OR el[86] OR el[87] OR el[88] OR el[89];  
 el1\_[9] = el[90] OR el[91] OR el[92] OR el[93] OR el[94] OR el[95] OR el[96] OR el[97] OR el[98] OR el[99];

-----  
 el1\_[10] = el[100] OR el[101] OR el[102] OR el[103] OR el[104] OR el[105] OR el[106] OR el[107] OR el[108] OR el[109];  
 el1\_[11] = el[110] OR el[111] OR el[112] OR el[113] OR el[114] OR el[115] OR el[116] OR el[117] OR el[118] OR el[119];  
 el1\_[12] = el[120] OR el[121] OR el[122] OR el[123] OR el[124] OR el[125] OR el[126] OR el[127] OR el[128] OR el[129];  
 el1\_[13] = el[130] OR el[131] OR el[132] OR el[133] OR el[134] OR el[135] OR el[136] OR el[137] OR el[138] OR el[139];  
 el1\_[14] = el[140] OR el[141] OR el[142] OR el[143] OR el[144] OR el[145] OR el[146] OR el[147] OR el[148] OR el[149];  
 el1\_[15] = el[150] OR el[151] OR el[152] OR el[153] OR el[154] OR el[155] OR el[156] OR el[157] OR el[158] OR el[159];  
 el1\_[16] = el[160] OR el[161] OR el[162] OR el[163] OR el[164] OR el[165] OR el[166] OR el[167] OR el[168] OR el[169];  
 el1\_[17] = el[170] OR el[171] OR el[172] OR el[173] OR el[174] OR el[175] OR el[176] OR el[177] OR el[178] OR el[179];  
 el1\_[18] = el[180] OR el[181] OR el[182] OR el[183] OR el[184] OR el[185] OR el[186] OR el[187] OR el[188] OR el[189];  
 el1\_[19] = el[190] OR el[191] OR el[192] OR el[193] OR el[194] OR el[195] OR el[196] OR el[197] OR el[198] OR el[199];

-----  
 el1\_[20] = el[200] OR el[201] OR el[202] OR el[203] OR el[204] OR el[205] OR el[206] OR el[207] OR el[208] OR el[209];  
 el1\_[21] = el[210] OR el[211] OR el[212] OR el[213] OR el[214] OR el[215] OR el[216] OR el[217] OR el[218] OR el[219];  
 el1\_[22] = el[220] OR el[221] OR el[222] OR el[223] OR el[224] OR el[225] OR el[226] OR el[227] OR el[228] OR el[229];  
 el1\_[23] = el[230] OR el[231] OR el[232] OR el[233] OR el[234] OR el[235] OR el[236] OR el[237] OR el[238] OR el[239];  
 el1\_[24] = el[240] OR el[241] OR el[242] OR el[243] OR el[244] OR el[245] OR el[246] OR el[247] OR el[248] OR el[249];  
 el1\_[25] = el[250] OR el[251] OR el[252] OR el[253] OR el[254] OR el[255] OR el[256] OR el[257] OR el[258] OR el[259];  
 el1\_[26] = el[260] OR el[261] OR el[262] OR el[263] OR el[264] OR el[265] OR el[266] OR el[267] OR el[268] OR el[269];  
 el1\_[27] = el[270] OR el[271] OR el[272] OR el[273] OR el[274] OR el[275] OR el[276] OR el[277] OR el[278] OR el[279];  
 el1\_[28] = el[280] OR el[281] OR el[282] OR el[283] OR el[284] OR el[285] OR el[286] OR el[287] OR el[288] OR el[289];  
 el1\_[29] = el[290] OR el[291] OR el[292] OR el[293] OR el[294] OR el[295] OR el[296] OR el[297] OR el[298] OR el[299];

-----  
el1\_[30] = el[300] OR el[301] OR el[302] OR el[303] OR el[304] OR el[305] OR el[306] OR el[307] OR el[308] OR el[309];  
el1\_[31] = el[310] OR el[311] OR el[312] OR el[313] OR el[314] OR el[315] OR el[316] OR el[317] OR el[318] OR el[319];  
el1\_[32] = el[320] OR el[321] OR el[322] OR el[323] OR el[324] OR el[325] OR el[326] OR el[327] OR el[328] OR el[329];  
el1\_[33] = el[330] OR el[331] OR el[332] OR el[333] OR el[334] OR el[335] OR el[336] OR el[337] OR el[338] OR el[339];  
el1\_[34] = el[340] OR el[341] OR el[342] OR el[343] OR el[344] OR el[345] OR el[346] OR el[347] OR el[348] OR el[349];  
el1\_[35] = el[350] OR el[351] OR el[352] OR el[353] OR el[354] OR el[355] OR el[356] OR el[357] OR el[358] OR el[359];  
el1\_[36] = el[360] OR el[361] OR el[362] OR el[363] OR el[364] OR el[365] OR el[366] OR el[367] OR el[368] OR el[369];  
el1\_[37] = el[370] OR el[371] OR el[372] OR el[373] OR el[374] OR el[375] OR el[376] OR el[377] OR el[378] OR el[379];  
el1\_[38] = el[380] OR el[381] OR el[382] OR el[383] OR el[384] OR el[385] OR el[386] OR el[387] OR el[388] OR el[389];  
el1\_[39] = el[390] OR el[391] OR el[392] OR el[393] OR el[394] OR el[395] OR el[396] OR el[397] OR el[398] OR el[399];  
-----

el1\_[40] = el[400] OR el[401] OR el[402] OR el[403] OR el[404] OR el[405] OR el[406] OR el[407] OR el[408] OR el[409];  
el1\_[41] = el[410] OR el[411] OR el[412] OR el[413] OR el[414] OR el[415] OR el[416] OR el[417] OR el[418] OR el[419];  
el1\_[42] = el[420] OR el[421] OR el[422] OR el[423] OR el[424] OR el[425] OR el[426] OR el[427] OR el[428] OR el[429];  
el1\_[43] = el[430] OR el[431] OR el[432] OR el[433] OR el[434] OR el[435] OR el[436] OR el[437] OR el[438] OR el[439];  
el1\_[44] = el[440] OR el[441] OR el[442] OR el[443] OR el[444] OR el[445] OR el[446] OR el[447] OR el[448] OR el[449];  
el1\_[45] = el[450] OR el[451] OR el[452] OR el[453] OR el[454] OR el[455] OR el[456] OR el[457] OR el[458] OR el[459];  
el1\_[46] = el[460] OR el[461] OR el[462] OR el[463] OR el[464] OR el[465] OR el[466] OR el[467] OR el[468] OR el[469];  
el1\_[47] = el[470] OR el[471] OR el[472] OR el[473] OR el[474] OR el[475] OR el[476] OR el[477] OR el[478] OR el[479];  
el1\_[48] = el[480] OR el[481] OR el[482] OR el[483] OR el[484] OR el[485] OR el[486] OR el[487] OR el[488] OR el[489];  
el1\_[49] = el[490] OR el[491] OR el[492] OR el[493] OR el[494] OR el[495] OR el[496] OR el[497] OR el[498] OR el[499];  
-----

el1\_[50] = el[500] OR el[501] OR el[502] OR el[503] OR el[504] OR el[505] OR el[506] OR el[507] OR el[508] OR el[509];  
el1\_[51] = el[510] OR el[511] OR el[512] OR el[513] OR el[514] OR el[515] OR el[516] OR el[517] OR el[518] OR el[519];  
el1\_[52] = el[520] OR el[521] OR el[522] OR el[523] OR el[524] OR el[525] OR el[526] OR el[527] OR el[528] OR el[529];  
el1\_[53] = el[530] OR el[531] OR el[532] OR el[533] OR el[534] OR el[535] OR el[536] OR el[537] OR el[538] OR el[539];  
el1\_[54] = el[540] OR el[541] OR el[542] OR el[543] OR el[544] OR el[545] OR el[546] OR el[547] OR el[548] OR el[549];  
el1\_[55] = el[550] OR el[551] OR el[552] OR el[553] OR el[554] OR el[555] OR el[556] OR el[557] OR el[558] OR el[559];  
el1\_[56] = el[560] OR el[561] OR el[562] OR el[563] OR el[564] OR el[565] OR el[566] OR el[567] OR el[568] OR el[569];  
el1\_[57] = el[570] OR el[571] OR el[572] OR el[573] OR el[574] OR el[575] OR el[576] OR el[577] OR el[578] OR el[579];  
el1\_[58] = el[580] OR el[581] OR el[582] OR el[583] OR el[584] OR el[585] OR el[586] OR el[587] OR el[588] OR el[589];  
el1\_[59] = el[590] OR el[591] OR el[592] OR el[593] OR el[594] OR el[595] OR el[596] OR el[597] OR el[598] OR el[599];  
-----

el1\_[60] = el[600] OR el[601] OR el[602] OR el[603] OR el[604] OR el[605] OR el[606] OR el[607] OR el[608] OR el[609];  
el1\_[61] = el[610] OR el[611] OR el[612] OR el[613] OR el[614] OR el[615] OR el[616] OR el[617] OR el[618] OR el[619];  
el1\_[62] = el[620] OR el[621] OR el[622] OR el[623] OR el[624] OR el[625] OR el[626] OR el[627] OR el[628] OR el[629];

el1\_[63] = el[630] OR el[631] OR el[632] OR el[633] OR el[634] OR el[635] OR el[636] OR el[637] OR el[638] OR el[639];  
el1\_[64] = el[640] OR el[641] OR el[642] OR el[643] OR el[644] OR el[645] OR el[646] OR el[647] OR el[648] OR el[649];  
el1\_[65] = el[650] OR el[651] OR el[652] OR el[653] OR el[654] OR el[655] OR el[656] OR el[657] OR el[658] OR el[659];  
el1\_[66] = el[660] OR el[661] OR el[662] OR el[663] OR el[664] OR el[665] OR el[666] OR el[667] OR el[668] OR el[669];  
el1\_[67] = el[670] OR el[671] OR el[672] OR el[673] OR el[674] OR el[675] OR el[676] OR el[677] OR el[678] OR el[679];  
el1\_[68] = el[680] OR el[681] OR el[682] OR el[683] OR el[684] OR el[685] OR el[686] OR el[687] OR el[688] OR el[689];  
el1\_[69] = el[690] OR el[691] OR el[692] OR el[693] OR el[694] OR el[695] OR el[696] OR el[697] OR el[698] OR el[699];

-----  
el1\_[70] = el[700] OR el[701] OR el[702] OR el[703] OR el[704];  
-----

el2\_[0] = el1\_[0] OR el1\_[1] OR el1\_[2] OR el1\_[3] OR el1\_[4] OR el1\_[5] OR el1\_[6] OR el1\_[7] OR el1\_[8] OR el1\_[9];  
el2\_[1] = el1\_[10] OR el1\_[11] OR el1\_[12] OR el1\_[13] OR el1\_[14] OR el1\_[15] OR el1\_[16] OR el1\_[17] OR el1\_[18] OR el1\_[19];  
el2\_[2] = el1\_[20] OR el1\_[21] OR el1\_[22] OR el1\_[23] OR el1\_[24] OR el1\_[25] OR el1\_[26] OR el1\_[27] OR el1\_[28] OR el1\_[29];  
el2\_[3] = el1\_[30] OR el1\_[31] OR el1\_[32] OR el1\_[33] OR el1\_[34] OR el1\_[35] OR el1\_[36] OR el1\_[37] OR el1\_[38] OR el1\_[39];  
el2\_[4] = el1\_[40] OR el1\_[41] OR el1\_[42] OR el1\_[43] OR el1\_[44] OR el1\_[45] OR el1\_[46] OR el1\_[47] OR el1\_[48] OR el1\_[49];  
el2\_[5] = el1\_[50] OR el1\_[51] OR el1\_[52] OR el1\_[53] OR el1\_[54] OR el1\_[55] OR el1\_[56] OR el1\_[57] OR el1\_[58] OR el1\_[59];  
el2\_[6] = el1\_[60] OR el1\_[61] OR el1\_[62] OR el1\_[63] OR el1\_[64] OR el1\_[65] OR el1\_[66] OR el1\_[67] OR el1\_[68] OR el1\_[69];  
el2\_[7] = el1\_[70];

-----  
el\_tot = el2\_[0] OR el2\_[1] OR el2\_[2] OR el2\_[3] OR el2\_[4] OR el2\_[5] OR el2\_[6] OR el2\_[7];  
-----

END;

TITLE "Pt address to Pt bin translator";

```
%
% This function takes an input Pt address and fills a Pt bit array
% The mapping of address to pt bit is given elsewhere
%
% Created      15-Nov-1996      Fred Borcharding
%
```

```
--
--Project Information      d:\a4_trigger\cps_match\ps_clust_01.rpt
--
--MAX+plus II Compiler Report File
--Version 9.1 10/23/1998
--Compiled: 02/16/1999 13:58:02
```

```
--
--Chip/      Input Output Bidir Memory Memory          LCs
--POF  Device      Pins Pins  Pins  Bits % Utilized LCs % Utilized
--
--ps_clust_01
--  EPF10K10ATC100-1  7   21  0  0   0 %  62  10 %
--
```

SUBDESIGN ps\_clust\_01

```
(
  in_clust[7..1]          : INPUT;      -- Phi address of cluster
  DB_width[6..0]         : OUTPUT;
  DB_cluster[20..0]      : OUTPUT;
  ps_in[17..8],ps_ou[17..7] : OUTPUT;  --
)
```

VARIABLE

```
width[6..0]      : NODE;
cluster[20..0]   : NODE;
```

BEGIN

```
--DB_width[] = width[];
--DB_cluster[] = cluster[];
```

-----  
TABLE

```

in_clust[7..5] => width[];
H"0" => B"0000000";
H"1" => B"0000001";
H"2" => B"0000011";
H"3" => B"0000111";
H"4" => B"0001111";
H"5" => B"0011111";
H"6" => B"0111111";
H"7" => B"1111111";

```

END TABLE;

```

-----
CASE in_clust[4..1] IS
  WHEN 0 =>
    cluster[ 6.. 0] = width[6..0];
    cluster[20.. 7] = B"00000000000000";
  WHEN 1 =>
    cluster[ 0] = B"0";
    cluster[ 7.. 1] = width[6..0];
    cluster[20.. 8] = B"00000000000000";
  WHEN 2 =>
    cluster[ 1.. 0] = B"00";
    cluster[ 8.. 2] = width[6..0];
    cluster[20.. 9] = B"00000000000000";
  WHEN 3 =>
    cluster[ 2.. 0] = B"000";
    cluster[ 9.. 3] = width[6..0];
    cluster[20..10] = B"00000000000000";
  WHEN 4 =>
    cluster[ 3.. 0] = B"0000";
    cluster[10.. 4] = width[6..0];
    cluster[20..11] = B"00000000000000";
  WHEN 5 =>
    cluster[ 4.. 0] = B"00000";
    cluster[11.. 5] = width[6..0];
    cluster[20..12] = B"00000000000000";
  WHEN 6 =>
    cluster[ 5.. 0] = B"000000";

```

```
        cluster[12.. 6] = width[6..0];
        cluster[20..13] = B"00000000";
WHEN 7 =>
        cluster[ 6.. 0] = B"00000000";
        cluster[13.. 7] = width[6..0];
        cluster[20..14] = B"00000000";
WHEN 8 =>
        cluster[ 7.. 0] = B"00000000";
        cluster[14.. 8] = width[6..0];
        cluster[20..15] = B"00000000";
WHEN 9 =>
        cluster[ 8.. 0] = B"0000000000";
        cluster[15.. 9] = width[6..0];
        cluster[20..16] = B"000000";
WHEN 10 =>
        cluster[ 9.. 0] = B"000000000000";
        cluster[16..10] = width[6..0];
        cluster[20..17] = B"00000";
WHEN 11 =>
        cluster[10.. 0] = B"00000000000000";
        cluster[17..11] = width[6..0];
        cluster[20..18] = B"000";
WHEN 12 =>
        cluster[11.. 0] = B"0000000000000000";
        cluster[18..12] = width[6..0];
        cluster[20..19] = B"00";
WHEN 13 =>
        cluster[12.. 0] = B"000000000000000000";
        cluster[19..13] = width[6..0];
        cluster[ 20] = B"0";
WHEN 14 =>
        cluster[13.. 0] = B"00000000000000000000";
        cluster[20..14] = width[6..0];
WHEN 15 =>
        cluster[14.. 0] = B"0000000000000000000000";
        cluster[20..15] = width[5..0];
WHEN OTHERS =>
```

```
cluster[20.. 0] = B"00000000000000000000";  
END CASE;
```

```
-----  
ps_in[ 8] = cluster[ 1];  
ps_in[ 9] = cluster[ 3];  
ps_in[10] = cluster[ 5];  
ps_in[11] = cluster[ 7];  
ps_in[12] = cluster[ 9];  
ps_in[13] = cluster[11];  
ps_in[14] = cluster[13];  
ps_in[15] = cluster[15];  
ps_in[16] = cluster[17];  
ps_in[17] = cluster[19];
```

```
ps_ou[ 7] = cluster[ 0];  
ps_ou[ 8] = cluster[ 2];  
ps_ou[ 9] = cluster[ 4];  
ps_ou[10] = cluster[ 6];  
ps_ou[11] = cluster[ 8];  
ps_ou[12] = cluster[10];  
ps_ou[13] = cluster[12];  
ps_ou[14] = cluster[14];  
ps_ou[15] = cluster[16];  
ps_ou[16] = cluster[18];  
ps_ou[17] = cluster[20];
```

```
-----  
END;
```