ADVANCDED DIGITAL IC DESIGN

ENGR – 852

FALL 2016

ASIC Implementation of Motion Estimator in 32/28nm CMOS

Report Submitted by

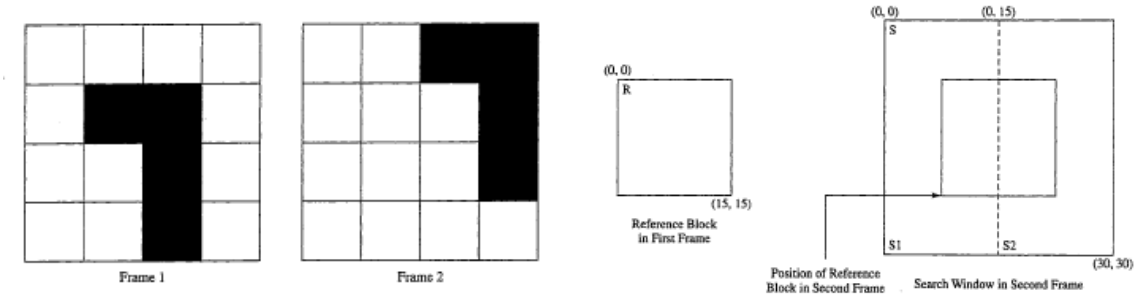
Anoja Rajalakshmi

Sudha Mahadik

Table of Content:

INTRODUCTION:

Motion estimation is the process of determining the movement of blocks between adjacent video frames. This technique utilizes the concept of motion and moving objects in a video and eliminates the redundancy between successive video frames. In this project, the motion estimation is illustrated by a simple example in which two successive video frames are considered as shown. Candidates for compression via motion estimation are found by doing a search on a set of possible motion vectors between two subsequent frames and looking for a good match. Two frames are compared by taking each 16 pixel x16 pixel reference block in the first frame and searching for a good match for it within a search window in the second frame. In this case, search window in the second frame is a 31 x 31 window centered around the position of the reference block in the first frame.



(a) (b)

Figure 1: (a) Consecutive frames in which the second can be coded as a motion vector applied to the first (b) Search is performed by comparing the reference block in the first frame with a set of possible positions of the reference block in the second frame.

A total of (31-15) x (31-15) = 256 comparisons performed from top left to bottom right of the search window. For each possible comparison block (i,j) , a distortion figure is calculated as below:

$$D(i,j) = \sum_{m=0}^{15} \sum_{n=0}^{15} |r_{m,n} - S_{m+i,n+j}|$$

The architecture considered for this project is shown below:

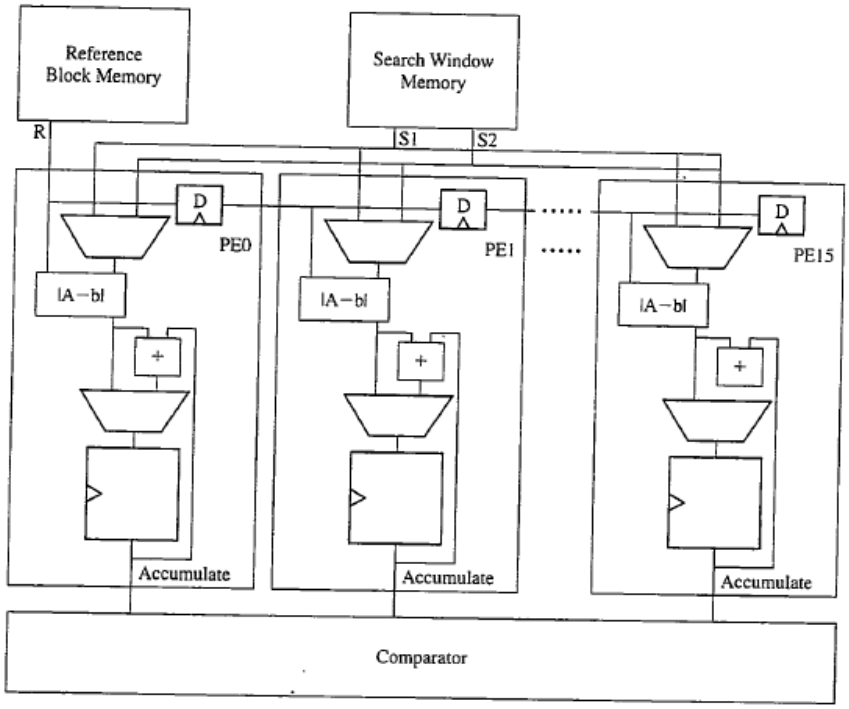


Figure 2:Architecture

Design Requirements:

- 16x16 Reference Block
- 31x31 Search Window
- Search memory is two-read-ported memory
- One memory per search
- Grey-scale coded pixels (8 bits/block)
- 4096 reference blocks in a frame
- 16x16=256 add-accumulates per search
- At 260 MHz At least 16 adders in parallel (4027/260=15.5)
- Timing requirements:
    - search speed at 15 frames per second
    - Encoding is not in real time
    - Clock: 260 MHz
- 4096*15*256*256 = 4.027E9 add-accumulates/second
- Target Library: 32 nm CMOS library (standard max db)

Data Path Design:

1. Processing Elements:

Performs the computation of the SAD “Sum of Absolute Difference”

2. Comparator

Compares the Distortions obtained from the PE's and determines the best distortion with minimum SAD.

3. Controller

Controls the operations of the flow of the signals from Memories to PE and Comparator.

- **System Design in Detail:**

1. Processing element:

It is the module that performs the computation of differences between the reference block and search block. It performs 256 SAD "Sum of Absolute Differences" per block and results the final distortion for each comparison out of 256 comparisons. There are total 16 PE modules that are required to meet the design specifications and the computation takes place in a way that same reference block is compared with the different locations with in the search block. Since the same reference block is being used by the 16 PE modules it is pipelined through a R pipe register with one clock cycle delay. It's the reason why each PE starts computation with one clock cycle delay.

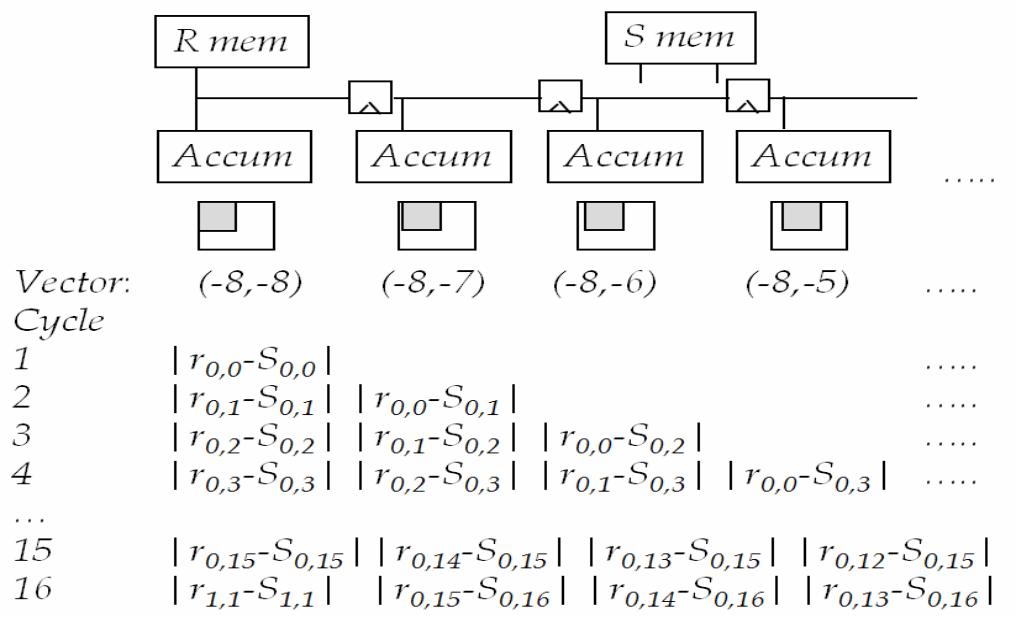


Figure 3:Search computation & pipelining registers

- PE module Verilog Design code:

```
module pe(clock,R,s1,s2,s1s2mux,newDist,Accumulate,Rpipe);
input clock;
input [7:0] R,s1,s2;
input s1s2mux,newDist;
output [7:0] Accumulate,Rpipe;
reg [7:0] Accumulate,AccumulateIn,difference,Rpipe,s;
reg Carry;

always@(posedge clock)Rpipe= R;
always@(posedge clock) Accumulate=AccumulateIn;

always@(R or s1 or s2 or s1s2mux or newDist or Accumulate)
  begin
  if(s1s2mux)s=s1;
  else s=s2;

  if(R>s)
  difference =R -s;
  else
   difference=s-R;
  {Carry,AccumulateIn}=Accumulate+difference;
  if(Carry==1) AccumulateIn=8'hFF;
   else if(newDist==1) AccumulateIn=difference;
  end
  endmodule
```

2. Comparator Module:

It is the module that results the final best distortion along with its motion vectors. The output from the 16 PE's is compared against each other and whenever the new best is found it updates the best distortion so far with the new best distortion and this process continues till the end of the all the comparisons and the final result will be outputted as the Best distortion with the location.

- Verilog Code for Comparator design:

```
module Comparator(clock,CompStart,PEout,PEready,vectorX,vectorY,BestDist,motionX,motionY, pflag);
input clock;
input CompStart;
input [8*16-1:0] PEout;
input [15:0] PEready;
input [3:0] vectorX,vectorY;
input pflag;
output [7:0] BestDist;
output [3:0] motionX,motionY;
reg [7:0] BestDist,newDist;
reg [3:0] motionX,motionY;
reg newBest;
reg [8*16-1:0] tempPEout;

always@(posedge clock)
  if(CompStart==0) BestDist<=8'hff;
  else if(newBest==1)
  begin
     BestDist<=newDist;
     motionX<=vectorX;
     motionY<=vectorY;
  end
always@(posedge clock)
    if(pflag == 1) tempPEout[127:0] <= PEout[127:0] ;

always@(BestDist or tempPEout or PEready)
begin

       case(PEready)
         16'b0000000000000001: newDist=tempPEout[7:0];
         16'b0000000000000010: newDist=tempPEout[15:8];
         16'b0000000000000100: newDist=tempPEout[23:16];
         16'b0000000000001000: newDist=tempPEout[31:24];
         16'b0000000000010000: newDist=tempPEout[39:32];
         16'b0000000000100000: newDist=tempPEout[47:40];
         16'b0000000001000000: newDist=tempPEout[55:48];
         16'b0000000010000000: newDist=tempPEout[63:56];
         16'b0000000100000000: newDist=tempPEout[71:64];
         16'b0000001000000000: newDist=tempPEout[79:72];
         16'b0000010000000000: newDist=tempPEout[87:80];
         16'b0000100000000000: newDist=tempPEout[95:88];
         16'b0001000000000000: newDist=tempPEout[103:96];
         16'b0010000000000000: newDist=tempPEout[111:104];
         16'b0100000000000000: newDist=tempPEout[119:112];
         16'b1000000000000000: newDist=tempPEout[127:120];
         endcase
       if( (CompStart==0)) newBest=0;
     else if (newDist<BestDist) newBest=1;
 else newBest=0;
end
endmodule
```

3. Controller Module:

The controller design strategy is to build one counter and decode logic to toggle the various control lines. It is the most important part of the design which determines the flow of execution of the design and which maintains the integrity of the overall design. It controls the flow starting from the dual ported memories and ending at the Comparator. The counter has to go high enough to calculate all of the distortion vectors. Controller controls the flow of 16 PE's which defines the order of computation of the reference frame with respect to the search frame. Here controller follows a mechanism of a counter.

- *Control points:*

1. *PE control lines:*

S1S2mux [15:0]; // S1-S2 mux control
NewDist [15:0]; // =1 when PE is starting a new distortion calculation
Comparator control lines:
Comp Start = 1 = // when PEs running
PEready [15:0]; // PEready[i] =1 when PE (i) has a new distortion vector
Vector X [3:0], Vector Y [3:0]; // Motion vector being evaluated

2. *Memory control lines:*

Memories organized in row-major format
Address R [7:0]; // address for Reference memory (0, 0) - (15, 15)
AddressS1 [9:0]; // address for first read port of Search mem
AddressS2 [9:0]; // second read port of Search mem (0, 0) - (30, 30)

- Verilog Code for Controller design:

```
module control (clock,start,S1S2mux,NewDist,CompStart,PEready,VectorX,VectorY,AddressR,AddressS1,AddressS2, pflag);
input clock;
input start;
output [15:0] S1S2mux;
output  [15:0] NewDist ;
output CompStart;
output [15:0] PEready;
output [3:0] VectorX,VectorY;
output [7:0] AddressR;
output [9:0] AddressS1,AddressS2;
output pflag;
reg pflag;
reg [15:0] S1S2mux;
reg [15:0] NewDist;
reg CompStart;
reg [15:0] PEready;
reg [3:0] VectorX,VectorY;
reg [7:0] AddressR;
reg [9:0] AddressS1,AddressS2;
reg [12:0] count;
//reg [12:0] count_temp;
reg completed;
reg [11:0] temp;
integer i;
integer j;


always@(posedge clock) begin
  if(start ==0)
    count <= 12'b0;
  else if (completed==0)
   begin
     count<=count+1'b1;
   end


end
```

```
   always@(count)
    begin

         for(i=0;i<16;i=i+1)
        begin
         NewDist[i]=(count[7:0]==i);

         PEready[i]=(NewDist[i] && !(count<256));
         S1S2mux[i]=(count[3:0] >= i);
      CompStart = (!(count<256));
        if( (count & 255)==0 )
            pflag = 1;
        else
            pflag = 0;

        end
      AddressR=count[7:0];
      AddressS1=(count[11:8] + count[7:4])*32 +count[3:0];

      temp = count[11:0]-16;
      AddressS2=(temp[11:8] + temp[7:4])*32 + temp[3:0] + 16;

      VectorX=count[3:0] - 8;
      VectorY=count[11:8] - 8;
      completed=(count == 4111);
      end

   endmodule
```

- *Top Module of the design code:*

Top module instantiation:

Top (topmodule. v) encloses the datapath (PE. v comparator. v) and control (control. v) modules. The control signals are given by the controller and the status signals are given by the data path module.

The inputs to the Top module are clock, start, R, s1 and s2.

The outputs are motionx, motiony, AddressR, AddressS1, AddressS2, BestDist.

The control signals given by the controller are Address R; AddressS1, Address S2, vector x, vector y, Newdist, Comp start, PEready.

```
module topmodule (clock, start, R, s1, s2, motionx, motiony,AddressR, AddressS1, AddressS2,BestDist);
input clock;
input start;
input[7:0] R,s1,s2;
output [3:0]motionx;
output [3:0]motiony;
output [7:0]BestDist;
output [7:0]AddressR;
output [9:0]AddressS1;
output [9:0]AddressS2;

wire[15:0] s1s2mux;
wire[15:0] newDist;
wire[15:0] PEready;
wire       CompStart;
wire[3:0]  VectorX;
wire[3:0]  VectorY;
wire[7:0]  AddressR;
wire[9:0]  AddressS1;
wire[9:0]  AddressS2;
wire[7:0]  bestdist;
wire[8*16-1:0] peout;
wire[7:0] Rpipe0,Rpipe1,Rpipe2,Rpipe3,Rpipe4,Rpipe5,Rpipe6,Rpipe7,Rpipe8,Rpipe9,Rpipe10,Rpipe11,Rpipe12,Rpipe13,Rpipe14 ;
wire pflag;
```

```
control c (.clock(clock), .start(start), .S1S2mux(s1s2mux[15:0]), .NewDist(newDist[15:0]), .CompStart(CompStart), .PEready(PEready[15:0]), .VectorX(VectorX[3:0]), .VectorY(Ve

Comparator comparator1 (.clock(clock), .CompStart(CompStart), .PEout(peout[8*16-1:0]) , .PEready(PEready[15:0]),.vectorX(VectorX[3:0]), .vectorY(VectorY[3:0]), .BestDist(Best

pe pe0(.clock(clock), .R(R[7:0]), .s1(s1[7:0]), .s2(s2[7:0]), .s1s2mux(s1s2mux[0]), .newDist(newDist[0]), .Accumulate(peout [7 : 0]) , .Rpipe(Rpipe0));
pe pe1(.clock(clock), .R(Rpipe0), .s1(s1[7:0]), .s2(s2[7:0]), .s1s2mux(s1s2mux[1]), .newDist(newDist[1]), .Accumulate(peout [15 : 8]) , .Rpipe(Rpipe1));
pe pe2(.clock(clock), .R(Rpipe1), .s1(s1[7:0]), .s2(s2[7:0]), .s1s2mux(s1s2mux[2]), .newDist(newDist[2]), .Accumulate(peout [23 : 16]) , .Rpipe(Rpipe2));
pe pe3(.clock(clock), .R(Rpipe2), .s1(s1[7:0]), .s2(s2[7:0]), .s1s2mux(s1s2mux[3]), .newDist(newDist[3]), .Accumulate(peout [31 : 24]) , .Rpipe(Rpipe3));
pe pe4(.clock(clock), .R(Rpipe3), .s1(s1[7:0]), .s2(s2[7:0]), .s1s2mux(s1s2mux[4]), .newDist(newDist[4]), .Accumulate(peout [39 : 32]) , .Rpipe(Rpipe4));
pe pe5(.clock(clock), .R(Rpipe4), .s1(s1[7:0]), .s2(s2[7:0]), .s1s2mux(s1s2mux[5]), .newDist(newDist[5]), .Accumulate(peout [47 : 40]) , .Rpipe(Rpipe5));
pe pe6(.clock(clock), .R(Rpipe5), .s1(s1[7:0]), .s2(s2[7:0]), .s1s2mux(s1s2mux[6]), .newDist(newDist[6]), .Accumulate(peout [55 : 48]) , .Rpipe(Rpipe6));
pe pe7(.clock(clock), .R(Rpipe6), .s1(s1[7:0]), .s2(s2[7:0]), .s1s2mux(s1s2mux[7]), .newDist(newDist[7]), .Accumulate(peout [63 : 56]) , .Rpipe(Rpipe7));
pe pe8(.clock(clock), .R(Rpipe7), .s1(s1[7:0]), .s2(s2[7:0]), .s1s2mux(s1s2mux[8]), .newDist(newDist[8]), .Accumulate(peout [71 : 64]) , .Rpipe(Rpipe8));
pe pe9(.clock(clock), .R(Rpipe8), .s1(s1[7:0]), .s2(s2[7:0]), .s1s2mux(s1s2mux[9]), .newDist(newDist[9]), .Accumulate(peout [79 : 72]) , .Rpipe(Rpipe9));
pe pe10(.clock(clock), .R(Rpipe9), .s1(s1[7:0]), .s2(s2[7:0]), .s1s2mux(s1s2mux[10]), .newDist(newDist[10]), .Accumulate(peout [87 : 80]) , .Rpipe(Rpipe10));
pe pe11(.clock(clock), .R(Rpipe10), .s1(s1[7:0]), .s2(s2[7:0]), .s1s2mux(s1s2mux[11]), .newDist(newDist[11]), .Accumulate(peout [95 : 88]) , .Rpipe(Rpipe11));
pe pe12(.clock(clock), .R(Rpipe11), .s1(s1[7:0]), .s2(s2[7:0]), .s1s2mux(s1s2mux[12]), .newDist(newDist[12]), .Accumulate(peout [103 : 96]) , .Rpipe(Rpipe12));
pe pe13(.clock(clock), .R(Rpipe12), .s1(s1[7:0]), .s2(s2[7:0]), .s1s2mux(s1s2mux[13]), .newDist(newDist[13]), .Accumulate(peout [111 : 104]) , .Rpipe(Rpipe13));
pe pe14(.clock(clock), .R(Rpipe13), .s1(s1[7:0]), .s2(s2[7:0]), .s1s2mux(s1s2mux[14]), .newDist(newDist[14]), .Accumulate(peout [119 : 112]) , .Rpipe(Rpipe14));
pe pe15(.clock(clock), .R(Rpipe14), .s1(s1[7:0]), .s2(s2[7:0]), .s1s2mux(s1s2mux[15]), .newDist(newDist[15]), .Accumulate(peout [127 : 120]));

endmodule
```

- *Testbench for the design:*

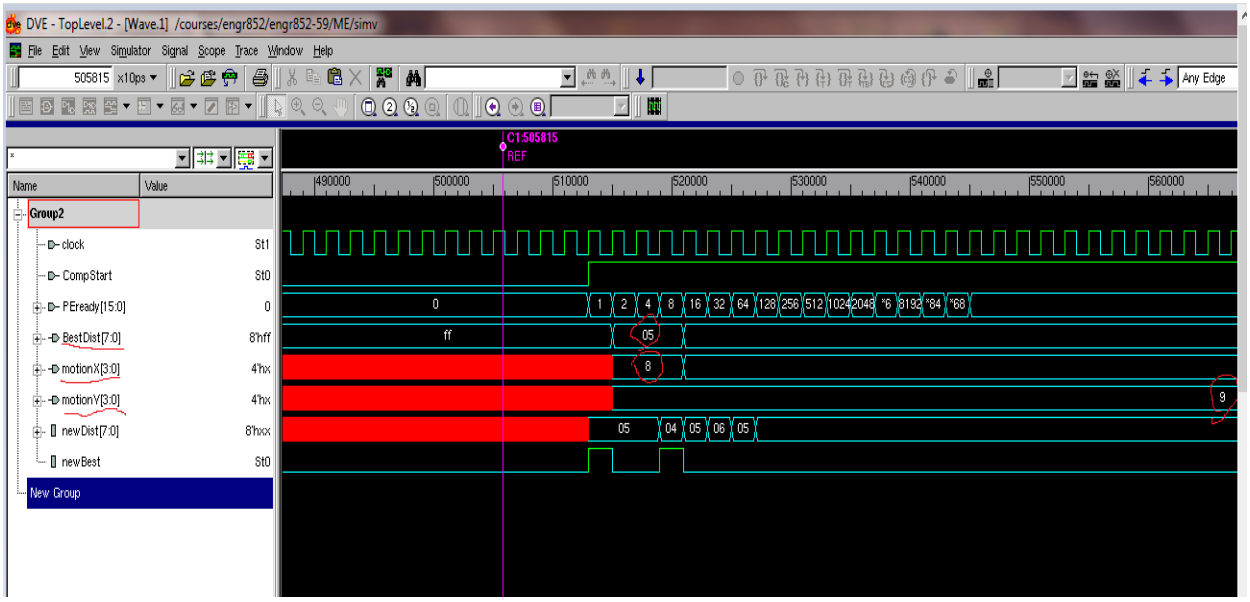
Testbench module (top_tb.v)consists of the top module along with the memory modules. Memories receive the address from the control.v module and outputs the data to the datapath module(comparator.v & PE.v). Both control.v and datapath modules (comparator.v & PE.v) are enclosed in top module.

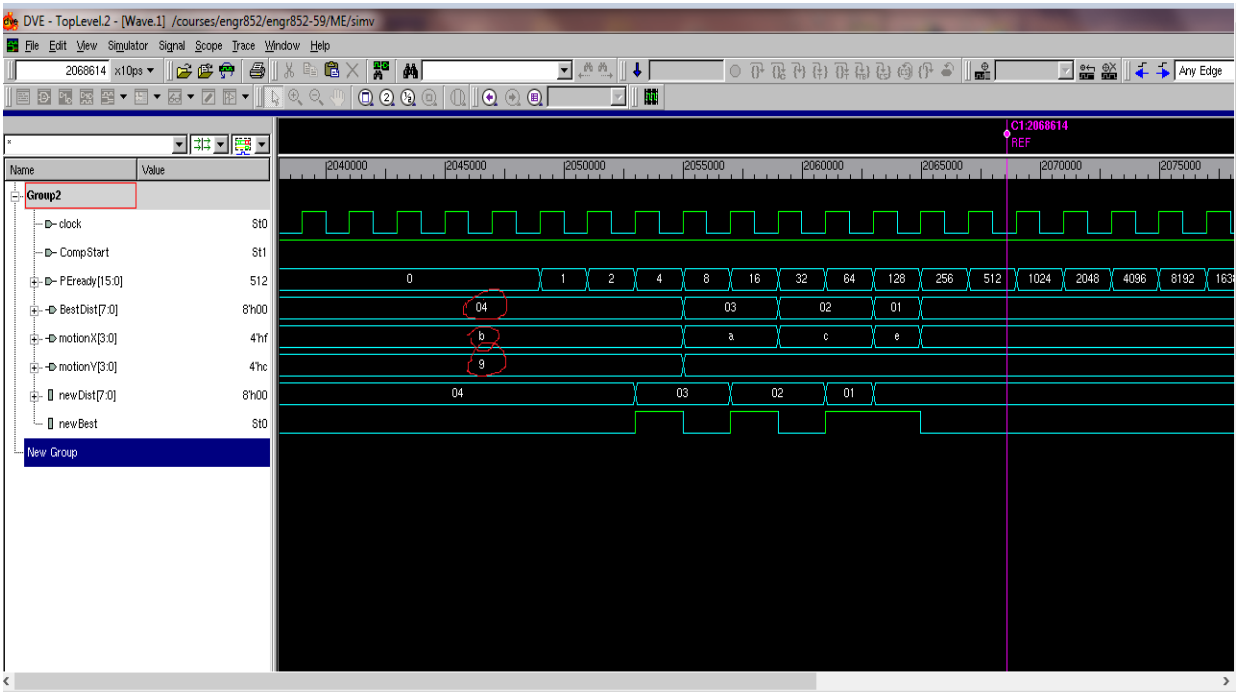
```
1   `timescale 1ns/10ps
2   module timeunit;
3      initial $timeformat(-9,1," ns",9);
4   endmodule
5
6   module toplevel_testbench_op ( ) ;
7      wire[3:0]          motionx;
8      wire[3:0]          motiony;
9      wire[7:0]          AddressR;
10     wire[9:0]          AddressS1;
11     wire[9:0]          AddressS2;
12
13     reg               clock;
14     reg               start;
15     reg[7:0]          R;
16     reg[7:0]          s1;
17     reg[7:0]          s2;
18     reg[7:0]      R_mem[255:0];
19     reg[7:0]      s1_mem[960:0];
20     reg[7:0]      s2_mem[960:0];
21     integer i,j;
22
```

```
23      always #10 clock = ~clock;
24      initial
25        begin
26      $readmemh("rmem_data.hex",R_mem);
27      $readmemh("smem_data.hex", s1_mem);
28      $readmemh("smem_data.hex", s2_mem);
29      assign R=    R_mem[AddressR] ;
30      assign s1=   s1_mem[AddressS1];
31      assign s2=   s2_mem[AddressS2];
32
33            clock = 1'b0;
34            start = 1'b0;
35            @(posedge clock);
36            #1 start= 1'b1;
37
38            #83320
39            $finish;
40        end
41
42  topmodule dut (
43            .motionx          (motionx[3:0]),
44            .motiony          (motiony[3:0]),
45
46            .start            (start),
47            .clock                (clock),
48            .R            (R[7:0]),
49            .s1           (s1[7:0]),
50            .s2                   (s2[7:0]),
51            .AddressR             (AddressR[7:0]),
52            .AddressS1        (AddressS1[9:0]),
53            .AddressS2        (AddressS2[9:0]));
54  endmodule
```

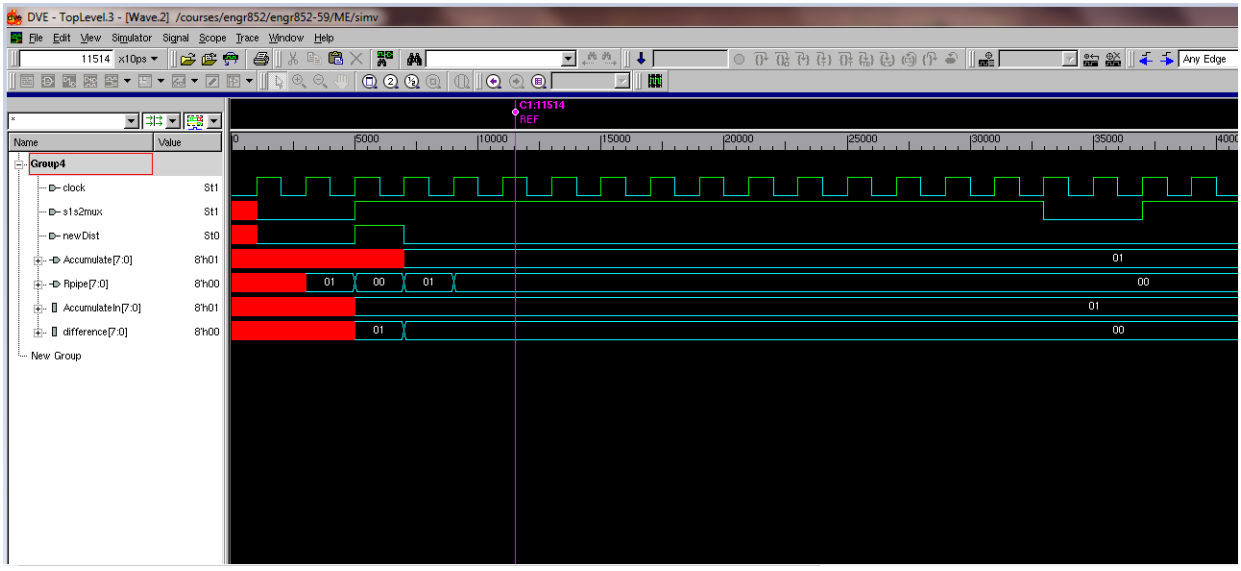
- Waveforms of VCS Simulation :

1. Comparator :

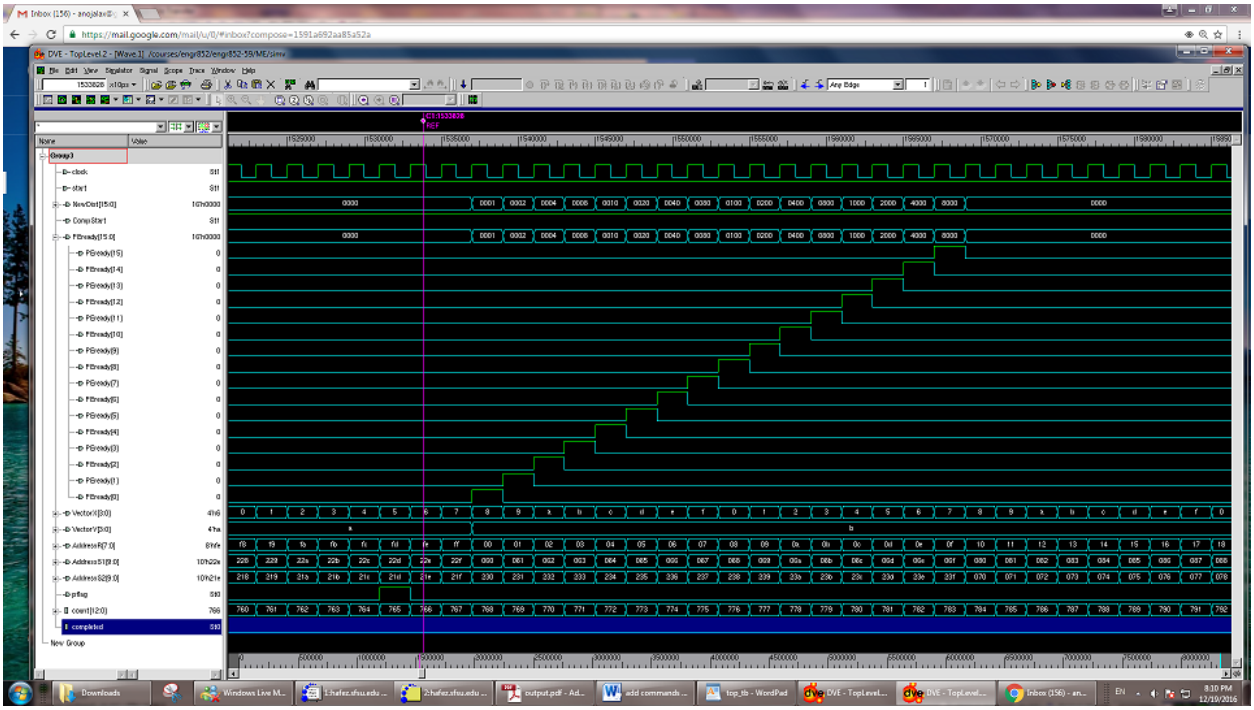



DVE - TopLevel.2 - [Wave.1] /courses/engr852/engr852-59/ME/simv
File Edit View Simulator Signal Scope Trace Window Help
2068614 x10ps
Any Edge
C1:2068614
REF
Name
Value
Group2
clock St0
CompStart St1
PEready[15:0] 512
BestDist[7:0] 8'h00
motionX[3:0] 4'hf
motionY[3:0] 4'hc
newDist[7:0] 8'h00
newBest St0
New Group
2040000
2045000
2050000
2055000
2060000
2065000
2070000
2075000
0 1 2 4 8 16 32 64 128 256 512 1024 2048 4096 8192
04 03 02 01
b a c e
9
04 03 02 01

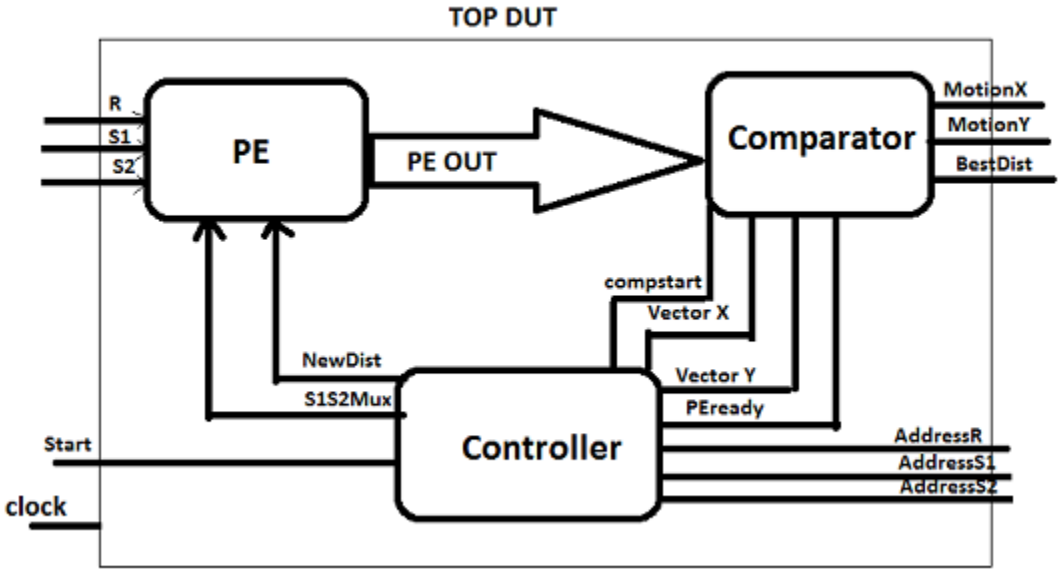
2. PE:


DVE - TopLevel.3 - [Wave.2] /courses/engr852/engr852-59/ME/simv
File Edit View Simulator Signal Scope Trace Window Help
11514 x10ps
Any Edge
C1:11514
REF
Name
Value
Group4
clock St1
s1s2mux St1
newDist St0
Accumulate[7:0] 8'h01
Rpipe[7:0] 8'h00
AccumulateIn[7:0] 8'h01
difference[7:0] 8'h00
New Group
0
5000
10000
15000
20000
25000
30000
35000
01
01 00 01
00
01
01
00

3. Controller:



- *Top level Block Diagram:*



- **Verification:**

The functionality of the whole design is verified as follows.

- We have written a Verilog code for test bench for the design module to ensure the correctness of its functionality. We have tested and verified functional correctness of the design, by passing the random series of 1's and 0's from two hex files, rmem.hex & smem.hex respectively through the design topmodule, to simulate the reference memory and search memory. We have simulated test bench, checked the respective waveforms in the Synopsys tool. We have inspected the waveform of the test bench, by including all the signals in the design during simulation and then checking for the value for Motion vector in the output memory which file made the design robust and functionally correct. After verification synthesis is done using the Verilog code.

- **Physical Design Reports:**

- **Synthesis:**

- Target Clock: 260 MHz: Period: 3.846 ns
- Target Library: saed32_max.db
- Input and Output Delay: 0 ns

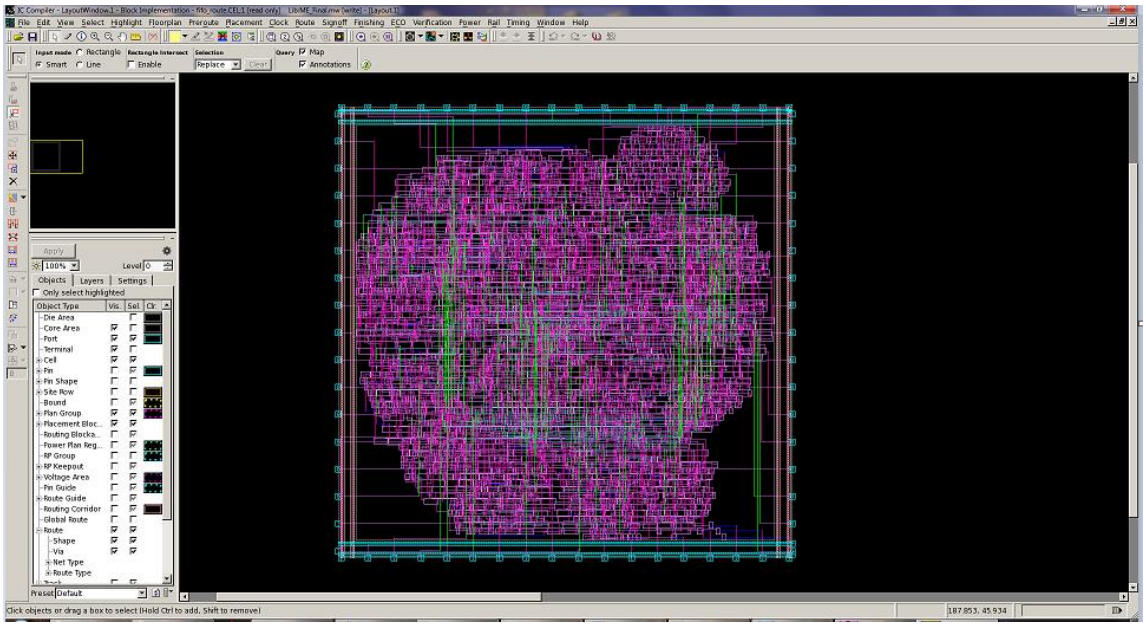
Slack (MET) : 1.94

```
c/rem_49_I16/U42/Y (AND2X1_HVT)                          0.03       1.54 f
c/rem_49_I16/U54/Y (AND2X1_HVT)                          0.03       1.57 f
c/rem_49_I16/U66/Y (AND2X1_HVT)                          0.03       1.59 f
c/rem_49_I16/U4/Y (AO21X1_HVT)                           0.06       1.65 f
c/rem_49_I16/U101/Y (MUX21X1_HVT)                        0.04       1.69 r
c/rem_49_I16/remainder[7] (control_DW_div_uns_0)         0.00       1.69 r
c/U5/Y (NOR4X1_HVT)                                      0.06       1.74 f
c/U29/Y (AND2X1_HVT)                                     0.05       1.79 f
c/pflag (control)                                        0.00       1.79 f
comparator1/pflag (Comparator)                           0.00       1.79 f
comparator1/U128/Y (INVX1_HVT)                           0.03       1.83 r
comparator1/U111/Y (INVX1_HVT)                           0.04       1.87 f
comparator1/U206/Y (AO22X1_HVT)                          0.03       1.89 f
comparator1/tempPEout_reg[101]/D (DFFX1_HVT)             0.00       1.89 f
data arrival time                                                   1.89

clock ideal_clock1 (rise edge)                           3.85       3.85
clock network delay (ideal)                              0.00       3.85
comparator1/tempPEout_reg[101]/CLK (DFFX1_HVT)           0.00       3.85 r
library setup time                                      -0.01       3.83
data required time                                                  3.83
--------------------------------------------------------------------------
data required time                                                  3.83
data arrival time                                                  -1.89
--------------------------------------------------------------------------
slack (MET)                                                         1.94
```

➢ **Place and Route**

- Target Library: saed32_max.db
- Clock: 3.864ns



Chip Area: 52854 sites

```
****************************************
  Report : Chip Summary
  Design : topmodule
  Version: I-2013.12-ICC-SP3
  Date   : Mon Dec 19 21:27:30 2016
****************************************
Std cell utilization: 60.30%  (31872/(52854-0))
(Non-fixed + Fixed)
Std cell utilization: 60.30%  (31872/(52854-0))
(Non-fixed only)
Chip area:            52854    sites, bbox (5.00 5.00 121.43 120.37) um
Std cell area:        31872    sites, (non-fixed:31872  fixed:0)
                      2357     cells, (non-fixed:2357   fixed:0)
Macro cell area:      0        sites
                      0        cells
Placement blockages:  0        sites, (excluding fixed std cells)
                      0        sites, (include fixed std cells & chimney area)
                      0        sites, (complete p/g net blockages)
Routing blockages:    0        sites, (partial p/g net blockages)
                      0        sites, (routing blockages and signal pre-route)
Lib cell count:       28
Avg. std cell width:  1.89 um
Site array:           unit     (width: 0.152 um, height: 1.672 um, rows: 69)
Physical DB scale:    1000 db_unit = 1 um
```

Power: 357.0981uW

```
Global Operating Voltage = 1.16
Power-specific unit information :
    Voltage Units = 1V
    Capacitance Units = 1.000000ff
    Time Units = 1ns
    Dynamic Power Units = 1uW    (derived from V,C,T units)
    Leakage Power Units = 1pW


  Cell Internal Power  =   0.0000 uW    (0%)
  Net Switching Power  = 140.8269 uW  (100%)
                         ---------
Total Dynamic Power    = 140.8269 uW  (100%)

Cell Leakage Power     = 216.2712 uW


                 Internal         Switching           Leakage            Total
Power Group      Power            Power               Power              Power   (   %    )  Attrs
--------------------------------------------------------------------------------------------------
io_pad             0.0000           0.0000              0.0000             0.0000  (   0.00%)
memory             0.0000           0.0000              0.0000             0.0000  (   0.00%)
black_box          0.0000           0.0000              0.0000             0.0000  (   0.00%)
clock_network      0.0000           0.0000              0.0000             0.0000  (   0.00%)
register           0.0000          22.3771          5.7247e+07            79.6245  (  22.30%)
sequential         0.0000           0.0000          1.1548e+06             1.1548  (   0.32%)
combinational      0.0000         118.4498          1.5787e+08           276.3188  (  77.38%)
--------------------------------------------------------------------------------------------------
Total              0.0000 uW      140.8269 uW      2.1627e+08 pW       357.0981 uW
1
```

Hold Time: 0.06

```
Startpoint: comparator1/motionX_reg[1]
            (rising edge-triggered flip-flop clocked by ideal_clock1)
Endpoint: motionx[1] (output port clocked by ideal_clock1)
Path Group: ideal_clock1
Path Type: min

Point                                                   Incr       Path
-----------------------------------------------------------------------------
clock ideal_clock1 (rise edge)                          0.00       0.00
clock network delay (ideal)                             0.00       0.00
comparator1/motionX_reg[1]/CLK (DFFX1_HVT)              0.00       0.00 r
comparator1/motionX_reg[1]/Q (DFFX1_HVT)                0.06       0.06 r
comparator1/motionX[1] (Comparator)                     0.00       0.06 r
motionx[1] (out)                                        0.00 *     0.06 r
data arrival time                                                  0.06

clock ideal_clock1 (rise edge)                          0.00       0.00
clock network delay (ideal)                             0.00       0.00
output external delay                                   0.00       0.00
data required time                                                 0.00
-----------------------------------------------------------------------------
data required time                                                 0.00
data arrival time                                                 -0.06
-----------------------------------------------------------------------------
slack (MET)                                                        0.06
```

Set Up Time: 1.37

```
c/rem_49_I16/U101/Y (MUX21X1_HVT)                       0.04 *     1.94 r
c/rem_49_I16/remainder[7] (control_DW_div_uns_0)        0.00       1.94 r
c/U5/Y (NOR4X1_HVT)                                     0.06 *     2.00 f
c/U29/Y (AND2X1_HVT)                                    0.22 *     2.22 f
c/pflag (control)                                       0.00       2.22 f
comparator1/pflag (Comparator)                          0.00       2.22 f
comparator1/U119/Y (NBUFFX2_HVT)                        0.09 *     2.31 f
comparator1/U120/Y (INVX1_HVT)                          0.09 *     2.40 r
comparator1/U35/Y (AO22X1_HVT)                          0.05 *     2.45 r
comparator1/tempPEout_reg[62]/D (DFFX1_HVT)             0.00 *     2.45 r
data arrival time                                                  2.45

clock ideal_clock1 (rise edge)                          3.85       3.85
clock network delay (ideal)                             0.00       3.85
comparator1/tempPEout_reg[62]/CLK (DFFX1_HVT)           0.00       3.85 r
library setup time                                     -0.02       3.82
data required time                                                 3.82
-----------------------------------------------------------------------------
data required time                                                 3.82
data arrival time                                                 -2.45
-----------------------------------------------------------------------------
slack (MET)                                                        1.37
```

- **Summary:**

✓ **Design & Verification**
- **RTL design – Done**
- **Verification - Done**

✓ **Synthesis - Done**
- **Clock: 260MHz (3.846ns)**
- **Input and Output Delays: Set at Zero,**
- **Min Area, Min Power**
- **Slack(MET): 1.94**

✓ **Place and Route**
- **Chip Layout - Done**
- **Chip Area: 52854 sites**
- **Chip Power: 357.0981uW**

✓ **Post-Layout Primetime**
- **Positive Slack on: Hold and Setup timings: MET (Hold: 0.06, Setup: 1.37)**

- **CONLUSION:**

Thus, specified functionality of motion estimator is implemented in Verilog design and verified using Verilog testbench. Further, the design is successfully synthesized and placement and routing is applied to it so as to find out its post layout chip area required and power consumption. We also checked that the design fulfills the timing requirements.
In all, throughout the process of implementation, we demonstrated the skills in designing a substantial digital system using the Verilog/Synthesis techniques taught in class.
Also, the project helped us to gain skills of ASIC implementation using Synopsys EDA tools. we saw the global scenario in video compression and came across with a real time problems and found a way to solve this challenge.

- **Reference:**

[1] M. E. Rizkalla, et. al. "Hardware Implementaion of Block-based Motion Estimation for Real Time Applications," Journal of VLSI Signal Processing, pp. 139-159, 2007
[2] R. Meagher, et. al, "VHDL Design for Real Time Motion Estimation Video Applications," Journal of Signal Processing Systems, Oct. 2008
[3] S. Lee, J. Kim, and S. Chae, "New motion estimation algorithm using adaptively quantized low bit-resolution image and its VLSI architecture for MPEG2 video encoding," IEEE Transactions on Circuits and Systems for Video Technology, vol. 8, no. 6, pp. 734-744, 1998

Advanced digital Design with Verilog HDL, Michael D. Ciletti.

Verilog HDL A Guide to Digital Design and Synthesis, IEEE 1364-2001 Compliant, Samir Palnitkar.