

**lab report 6**

Andrew Nady

900184042

Question 2:  
Code

Note: all code is included in another folder

**Required code:**

Risc-v Code

|  |
| --- |
| **`timescale** **1**ns / **1**ps  **module** Full\_dataPath( **input** clk,**input** fpga, **input** rst ,**input** [**1**:**0**] LedSel ,  **input** [**3**:**0**] ssdSel , **output** **reg** [**7**:**0**] leds, **output** [**6**:**0**]SSDout, **output**[**3**:**0**] anodes);  **wire** [**31**:**0**] adder1,adder2,PCmux,PCout;  **wire** cout1\_ripple1,cout1\_ripple2;  **wire** [**31**:**0**]ints\_out;  **wire** [**31**:**0**]readdata1, readdata2 ;  **wire** [**31**:**0**] gen\_out; //should be reg || wire  **wire** branch;  **wire** memread;  **wire** memtoreg;  **wire** [**1**:**0**]aluop;  **wire** memwrite;  **wire** alusrc;  **wire** regwrite;  **wire** [**3**:**0**]aluS;  **wire** [**31**:**0**]outmux\_input\_alu;  **wire** [**31**:**0**]ALU\_result;  **wire** zero;  **wire** [**31**:**0**] dataMem\_out;  **wire** [**31**:**0**] writingData;  **wire** [**31**:**0**] shiftout;  **wire** **S**;  **reg** [**12**:**0**]num;    //instantiate 32bitreg PCin PCout  OneReg pc( clk,**1**, PCmux, rst, PCout );  ripple\_carry ripplecar0(PCout,  **32'd4**,  **0**,  adder1,  cout1\_ripple1  );  //adder tany  ripple\_carry ripplecar1(PCout,  shiftout,  **0**,  adder2,  cout1\_ripple2  );  //mux  ThirtytwoMUX mux3(adder1,adder2,branch&zero ,PCmux);  //register (PCmux, PCin) //reload the PC  //instmem  InstMem instmem(PCout[**7**:**2**],ints\_out);  //control unit  Control\_unit controlunit(ints\_out[**6**:**2**], branch,  memread, memtoreg,  aluop, memwrite, alusrc, regwrite );  //alu\_control  ALU\_Control aluControl(aluop, ints\_out[**14**:**12**],ints\_out[**30**] ,aluS);  //registerfile  RegisterFile regfile( clk, rst,  ints\_out[**19**:**15**], ints\_out[**24**:**20**], ints\_out[**11**:**7**],  writingData,  regwrite,  readdata1, readdata2 );  //immGenerator  immediate\_generator immgen( gen\_out, ints\_out );  //shifting left  shiftLeft sh(gen\_out ,shiftout);  //mux  ThirtytwoMUX mux (readdata2,gen\_out,alusrc ,outmux\_input\_alu);  //ALU  **ALU** alu(  readdata1, outmux\_input\_alu,  aluS,  ALU\_result, zero );  //dataMem  DataMem datamem( clk, memread, memwrite,  ALU\_result[**7**:**2**] , readdata2, dataMem\_out);  //mux after the datamem  ThirtytwoMUX mux\_writing (ALU\_result,dataMem\_out,memtoreg ,writingData);  //switch or if on input ledsel to generate outputs leds  **always** @(\*)  **begin**  **case**(LedSel)  **2'b00** :  **begin**  leds[**0**]= regwrite;  leds[**1**]= alusrc;  leds[**3**:**2**]= aluop;  leds[**4**]= memread;  leds[**5**]= memwrite;  leds[**6**]= memtoreg;  leds[**7**]= branch;  **end**  **2'b01**:  **begin**  leds[**3**:**0**]=aluS;  leds [**4**]=zero;  leds[**5**]=branch&zero;  leds[**7**:**6**] =**0**;    **end**  **2'b11**:  leds[**7**:**0**]= {**0**,ints\_out[**6**:**0**]}; // monotring the opcode  ////switch or if on input ssdsel to generate ssdnumber  **default** : leds[**7**:**0**]= **0**;  **endcase**  **end**  **always**@(\*) **begin**  **case** (ssdSel)  **4'b0000**:num = PCout;  **4'b0001**: num =PCout+**4**;  **4'b0010**: num = adder2;  **4'b0011**: num = PCmux;  **4'b0100**: num = readdata1;  **4'b0101**: num= readdata2;  **4'b0110**: num= writingData;  **4'b0111**: num= gen\_out;  **4'b1000**: num= shiftout;  **4'b1001**:num=outmux\_input\_alu;  **4'b1010**: num= ALU\_result;  **4'b1011**:num=dataMem\_out;  **default** : num = **0**;  **endcase**  **end**  //instantiate 7 seg  seven ssd ( fpga, num,anodes,SSDout);  **endmodule** |

Top\_module -v Code

|  |
| --- |
| **`timescale** **1**ns / **1**ps  **module** Top\_module(**input** clk,**input** uart\_in,  **output** [**6**:**0**]SSDout,  **output** [**7**:**0**]leds,  **output** **wire** [**7**:**0**]leds2,  **output** [**3**:**0**]Anode);    // wire [7:0]leds2;  //assign sAnode=4'b1111;  // assign Anode=4'b0000;      UART\_receiver\_multiple\_Keys keyys(  clk,  uart\_in, // input receiving data line ,  leds // output  );    Full\_dataPath riscv( leds[**0**],clk , leds[**1**] ,leds [**3**:**2**] ,  leds [**7**:**4**], leds2, SSDout, Anode);  **endmodule** |

Constrain file:

|  |
| --- |
| set\_property **PACKAGE\_PIN** **E3** [get\_ports clk]  set\_property **IOSTANDARD** **LVCMOS33** [get\_ports clk]  set\_property **PACKAGE\_PIN** **C4** [get\_ports uart\_in]  set\_property **IOSTANDARD** **LVCMOS33** [get\_ports uart\_in]  set\_property **CLOCK\_DEDICATED\_ROUTE** **FALSE** [get\_nets leds[**0**]]  #set\_property -dict { **PACKAGE\_PIN** **H17** **IOSTANDARD** **LVCMOS33** } [get\_ports { **LED**[**0**] }]; #**IO\_L18P\_T2\_A24\_15** Sch=led[**0**]  #set\_property -dict { **PACKAGE\_PIN** **K15** **IOSTANDARD** **LVCMOS33** } [get\_ports { **LED**[**1**] }]; #**IO\_L24P\_T3\_RS1\_15** Sch=led[**1**]  #set\_property -dict { **PACKAGE\_PIN** **J13** **IOSTANDARD** **LVCMOS33** } [get\_ports { **LED**[**2**] }]; #**IO\_L17N\_T2\_A25\_15** Sch=led[**2**]  #set\_property -dict { **PACKAGE\_PIN** **N14** **IOSTANDARD** **LVCMOS33** } [get\_ports { **LED**[**3**] }]; #**IO\_L8P\_T1\_D11\_14** Sch=led[**3**]  #set\_property -dict { **PACKAGE\_PIN** **R18** **IOSTANDARD** **LVCMOS33** } [get\_ports { **LED**[**4**] }]; #**IO\_L7P\_T1\_D09\_14** Sch=led[**4**]  #set\_property -dict { **PACKAGE\_PIN** **V17** **IOSTANDARD** **LVCMOS33** } [get\_ports { **LED**[**5**] }]; #**IO\_L18N\_T2\_A11\_D27\_14** Sch=led[**5**]  #set\_property -dict { **PACKAGE\_PIN** **U17** **IOSTANDARD** **LVCMOS33** } [get\_ports { **LED**[**6**] }]; #**IO\_L17P\_T2\_A14\_D30\_14** Sch=led[**6**]  #set\_property -dict { **PACKAGE\_PIN** **U16** **IOSTANDARD** **LVCMOS33** } [get\_ports { **LED**[**7**] }]; #**IO\_L18P\_T2\_A12\_D28\_14** Sch=led[**7**]  #set\_property -dict { **PACKAGE\_PIN** **V16** **IOSTANDARD** **LVCMOS33** } [get\_ports { **LED**[**8**] }]; #**IO\_L16N\_T2\_A15\_D31\_14** Sch=led[**8**]  #set\_property -dict { **PACKAGE\_PIN** **T15** **IOSTANDARD** **LVCMOS33** } [get\_ports { **LED**[**9**] }]; #**IO\_L14N\_T2\_SRCC\_14** Sch=led[**9**]  #set\_property -dict { **PACKAGE\_PIN** **U14** **IOSTANDARD** **LVCMOS33** } [get\_ports { **LED**[**10**] }]; #**IO\_L22P\_T3\_A05\_D21\_14** Sch=led[**10**]  #set\_property -dict { **PACKAGE\_PIN** **T16** **IOSTANDARD** **LVCMOS33** } [get\_ports { **LED**[**11**] }]; #**IO\_L15N\_T2\_DQS\_DOUT\_CSO\_B\_14** Sch=led[**11**]  #set\_property -dict { **PACKAGE\_PIN** **V15** **IOSTANDARD** **LVCMOS33** } [get\_ports { **LED**[**12**] }]; #**IO\_L16P\_T2\_CSI\_B\_14** Sch=led[**12**]  #set\_property -dict { **PACKAGE\_PIN** **V14** **IOSTANDARD** **LVCMOS33** } [get\_ports { **LED**[**13**] }]; #**IO\_L22N\_T3\_A04\_D20\_14** Sch=led[**13**]  #set\_property -dict { **PACKAGE\_PIN** **V12** **IOSTANDARD** **LVCMOS33** } [get\_ports { **LED**[**14**] }]; #**IO\_L20N\_T3\_A07\_D23\_14** Sch=led[**14**]  #set\_property -dict { **PACKAGE\_PIN** **V11** **IOSTANDARD** **LVCMOS33** } [get\_ports { **LED**[**15**] }]; #**IO\_L21N\_T3\_DQS\_A06\_D22\_14** Sch=led[**15**]  set\_property **PACKAGE\_PIN** **U16** [get\_ports leds[**7**]]  set\_property **IOSTANDARD** **LVCMOS33** [get\_ports leds[**7**]]  set\_property **PACKAGE\_PIN** **U17** [get\_ports leds[**6**]]  set\_property **IOSTANDARD** **LVCMOS33** [get\_ports leds[**6**]]  set\_property **PACKAGE\_PIN** **V17** [get\_ports leds[**5**]]  set\_property **IOSTANDARD** **LVCMOS33** [get\_ports leds[**5**]]  set\_property **PACKAGE\_PIN** **R18** [get\_ports leds[**4**]]  set\_property **IOSTANDARD** **LVCMOS33** [get\_ports leds[**4**]]  set\_property **PACKAGE\_PIN** **N14** [get\_ports leds[**3**]]  set\_property **IOSTANDARD** **LVCMOS33** [get\_ports leds[**3**]]  set\_property **PACKAGE\_PIN** **J13** [get\_ports leds[**2**]]  set\_property **IOSTANDARD** **LVCMOS33** [get\_ports leds[**2**]]  set\_property **PACKAGE\_PIN** **K15** [get\_ports leds[**1**]]  set\_property **IOSTANDARD** **LVCMOS33** [get\_ports leds[**1**]]  set\_property **PACKAGE\_PIN** **H17** [get\_ports leds[**0**]]  set\_property **IOSTANDARD** **LVCMOS33** [get\_ports leds[**0**]]  set\_property **PACKAGE\_PIN** **V16** [get\_ports leds2[**7**]]  set\_property **IOSTANDARD** **LVCMOS33** [get\_ports leds2[**7**]]  set\_property **PACKAGE\_PIN** **T15** [get\_ports leds2[**6**]]  set\_property **IOSTANDARD** **LVCMOS33** [get\_ports leds2[**6**]]  set\_property **PACKAGE\_PIN** **U14** [get\_ports leds2[**5**]]  set\_property **IOSTANDARD** **LVCMOS33** [get\_ports leds2[**5**]]  set\_property **PACKAGE\_PIN** **T16** [get\_ports leds2[**4**]]  set\_property **IOSTANDARD** **LVCMOS33** [get\_ports leds2[**4**]]  set\_property **PACKAGE\_PIN** **V15** [get\_ports leds2[**3**]]  set\_property **IOSTANDARD** **LVCMOS33** [get\_ports leds2[**3**]]  set\_property **PACKAGE\_PIN** **V14** [get\_ports leds2[**2**]]  set\_property **IOSTANDARD** **LVCMOS33** [get\_ports leds2[**2**]]  set\_property **PACKAGE\_PIN** **V12** [get\_ports leds2[**1**]]  set\_property **IOSTANDARD** **LVCMOS33** [get\_ports leds2[**1**]]  set\_property **PACKAGE\_PIN** **V11** [get\_ports leds2[**0**]]  set\_property **IOSTANDARD** **LVCMOS33** [get\_ports leds2[**0**]]  #seven  set\_property -dict { **PACKAGE\_PIN** **T10** **IOSTANDARD** **LVCMOS33** } [get\_ports { SSDout[**6**] }]; #**IO\_L24N\_T3\_A00\_D16\_14** Sch=ca  set\_property -dict { **PACKAGE\_PIN** **R10** **IOSTANDARD** **LVCMOS33** } [get\_ports { SSDout[**5**] }]; #**IO\_25\_14** Sch=cb  set\_property -dict { **PACKAGE\_PIN** **K16** **IOSTANDARD** **LVCMOS33** } [get\_ports { SSDout[**4**] }]; #**IO\_25\_15** Sch=cc  set\_property -dict { **PACKAGE\_PIN** **K13** **IOSTANDARD** **LVCMOS33** } [get\_ports { SSDout[**3**] }]; #**IO\_L17P\_T2\_A26\_15** Sch=cd  set\_property -dict { **PACKAGE\_PIN** **P15** **IOSTANDARD** **LVCMOS33** } [get\_ports { SSDout[**2**] }]; #**IO\_L13P\_T2\_MRCC\_14** Sch=ce  set\_property -dict { **PACKAGE\_PIN** **T11** **IOSTANDARD** **LVCMOS33** } [get\_ports { SSDout[**1**] }]; #**IO\_L19P\_T3\_A10\_D26\_14** Sch=cf  set\_property -dict { **PACKAGE\_PIN** **L18** **IOSTANDARD** **LVCMOS33** } [get\_ports { SSDout[**0**] }]; #**IO\_L4P\_T0\_D04\_14** Sch=cg  set\_property -dict { **PACKAGE\_PIN** **J17** **IOSTANDARD** **LVCMOS33** } [get\_ports { Anode[**0**] }]; #**IO\_L23P\_T3\_FOE\_B\_15** Sch=an[**0**]  set\_property -dict { **PACKAGE\_PIN** **J18** **IOSTANDARD** **LVCMOS33** } [get\_ports { Anode[**1**] }]; #**IO\_L23N\_T3\_FWE\_B\_15** Sch=an[**1**]  set\_property -dict { **PACKAGE\_PIN** **T9** **IOSTANDARD** **LVCMOS33** } [get\_ports { Anode[**2**] }]; #**IO\_L24P\_T3\_A01\_D17\_14** Sch=an[**2**]  set\_property -dict { **PACKAGE\_PIN** **J14** **IOSTANDARD** **LVCMOS33** } [get\_ports { Anode[**3**] }]; #**IO\_L19P\_T3\_A22\_15** Sch=an[**3**]  #set\_property -dict { **PACKAGE\_PIN** **P14** **IOSTANDARD** **LVCMOS33** } [get\_ports { sAnode[**0**] }]; #**IO\_L8N\_T1\_D12\_14** Sch=an[**4**]  #set\_property -dict { **PACKAGE\_PIN** **T14** **IOSTANDARD** **LVCMOS33** } [get\_ports { sAnode[**1**] }]; #**IO\_L14P\_T2\_SRCC\_14** Sch=an[**5**]  #set\_property -dict { **PACKAGE\_PIN** **K2** **IOSTANDARD** **LVCMOS33** } [get\_ports { sAnode[**2**] }]; #**IO\_L23P\_T3\_35** Sch=an[**6**]  #set\_property -dict { **PACKAGE\_PIN** **U13** **IOSTANDARD** **LVCMOS33** } [get\_ports { sAnode[**3**] }]; #**IO\_L23N\_T3\_A02\_D18\_14** Sch=an[**7**] |

Question 3:  
assembly code && binary

Assembly

|  |
| --- |
| .text  **main:**  lw t4, **0**(zero) #pc=**0**  lw t5, **4**(zero) #pc=**4**  lw t6, **8**(zero) #pc=**8**  **loop:**  beq t4,t6,exit #pc=**12**  add t4,t4,t5 #pc=**16**  beq zero,zero,loop #pc=**20**  **exit:**  sw t4, **12**(zero). #pc=**24** |

Assembly

|  |
| --- |
| 00000000000000000010111010000011  00000000010000000010111100000011  00000000100000000010111110000011  00000001111111101000011001100011  00000001111011101000111010110011  11111110000000000000110011100011  00000001110100000010011000100011 |

No unique task, we just wrote this code to test the jumbing and if it will loop or not

Question 4:

Experiment 5

LedSel

|  |  |  |  |
| --- | --- | --- | --- |
| pc | 00 | 01 | 11 |
| 0 | 11001010 | 01001000 | 11000000 |
| 4 | 11001010 | 01000000 | 11000000 |
| 8 | 11001010 | 01000000 | 11000000 |
| 12 | 00100011 | 01100000 | 11000110 |
| 16 | 10010000 | 01000000 | 11001100 |
| 20 | 00100011 | 01101100 | 11000110 |
| 24 | 01000110 | 01000000 | 11000100 |

SSDsel

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| PC | PC+4 | Bramch adder | pcMux | Readdata1 | Readdata2 | Writing data | gen\_out | shiftout | outmux\_i nput\_alu | ALU result | dataMem \_out |
| 0 | 1 | 10 | 11 | 100 | 101 | 110 | 111 | 1000 | 1001 | 1010 | 1011 |
| 0 | 4 | 0 | 4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 4 | 8 | 12 | 8 | 0 | 0 | 1 | 4 | 8 | 4 | 4 | 1 |
| 8 | 12 | 24 | 12 | 0 | 0 | 10 | 8 | 16 | 8 | 8 | 10 |
| 12 | 16 | 24 | 16 | 0 | 10 | 0 | 6 | 12 | 10 | -4 Unsigned=  (8182) | 0 |
| 16 | 20 | 74 | 20 | 0 | 1 | 1 | 29 | 58 | 1 | 1 | 0 |
| 20 | 24 | 12 | 12 | 0 | 0 | 0 | -4 Unsigned=  (8182) | -8 Unsigned=  (8182) | 0 | 0 | 0 |
| loop |  |  |  |  |  |  |  |  |  |  |  |
| . | . | . | . | . | . | . | . | . | . | . | . |
| . | . | . | . | . | . | . | . | . | . | . | . |
| . | . | . | . | . | . | . | . | . | . | . | . |
| 24 | 28 | 48 | 28 | 0 | 10 | 0 | 12 | 24 | 12 | 12 | 0 |

Testing on FPGA:  
we tested this program on the FPGA so with the same sequence of the SSDsel as the number is displayed on the FPGA and the leds will be the same as the above table of ledSel, so I will include some screenshots that I took from the data above.

**instruction -> lw t4, 0(zero) #pc=0**



     begin

      leds[0]= regwrite;

      leds[1]= alusrc;

      leds[3:2]=  aluop;

      leds[4]=  memread;

      leds[5]= memwrite;

      leds[6]=  memtoreg;

      leds[7]=  branch;

     end

As shown above that when the LEDsel =00, then 11001010.

Where

leds[0]= 1;

      leds[1]= 1;

      leds[3:2]=  00;

      leds[4]=  1;

      leds[5]= 0;

      leds[6]=  x;

      leds[7]=  0;



As shown above that when the LEDsel =01, then 01001000.

begin

     leds[3:0]=aluS;

     leds [4]=zero;

     leds[5]=branch&zero;

     leds[7:6] =0;

     End

Where

leds[0]= 0;

      leds[1]= 1;

      leds[3:2]=  00; //where alus 0010

      leds[4]=  1; //zero flag =1

      leds[5]= 0; //branch =0

      leds[6]=  0;

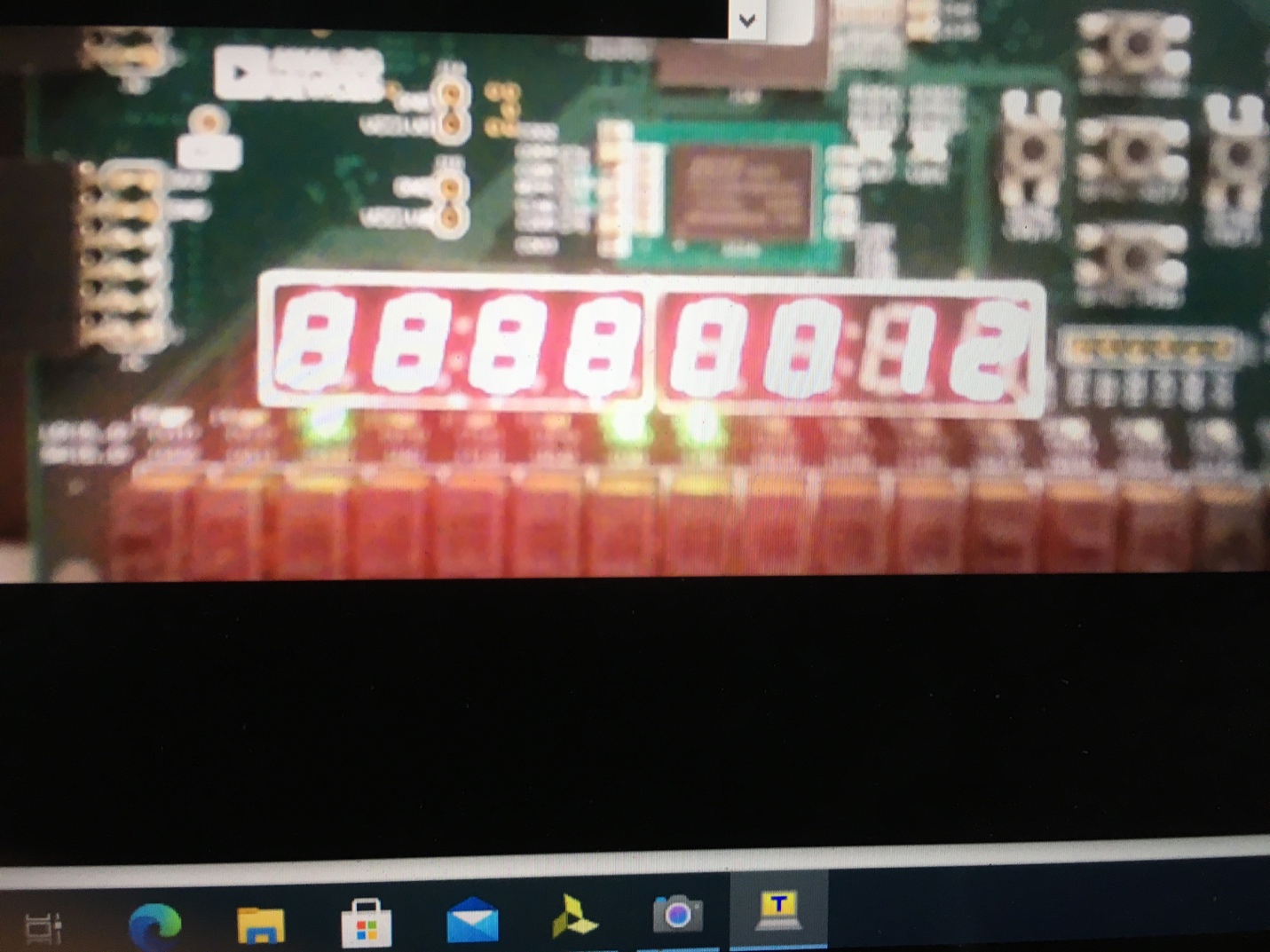
      leds[7]=  0;

****

leds[7:0]= {0,ints\_out[6:0]};   //  monitoring the opcode

**The output should be 1100 0000 where ledSel=11**

**instruction -> beq t4,t6,exit #pc=12**



As shown, SSDsel=0000,PC=12,

Ledsel=00, then leds =00100011

leds[0]= 0;

      leds[1]= 0;

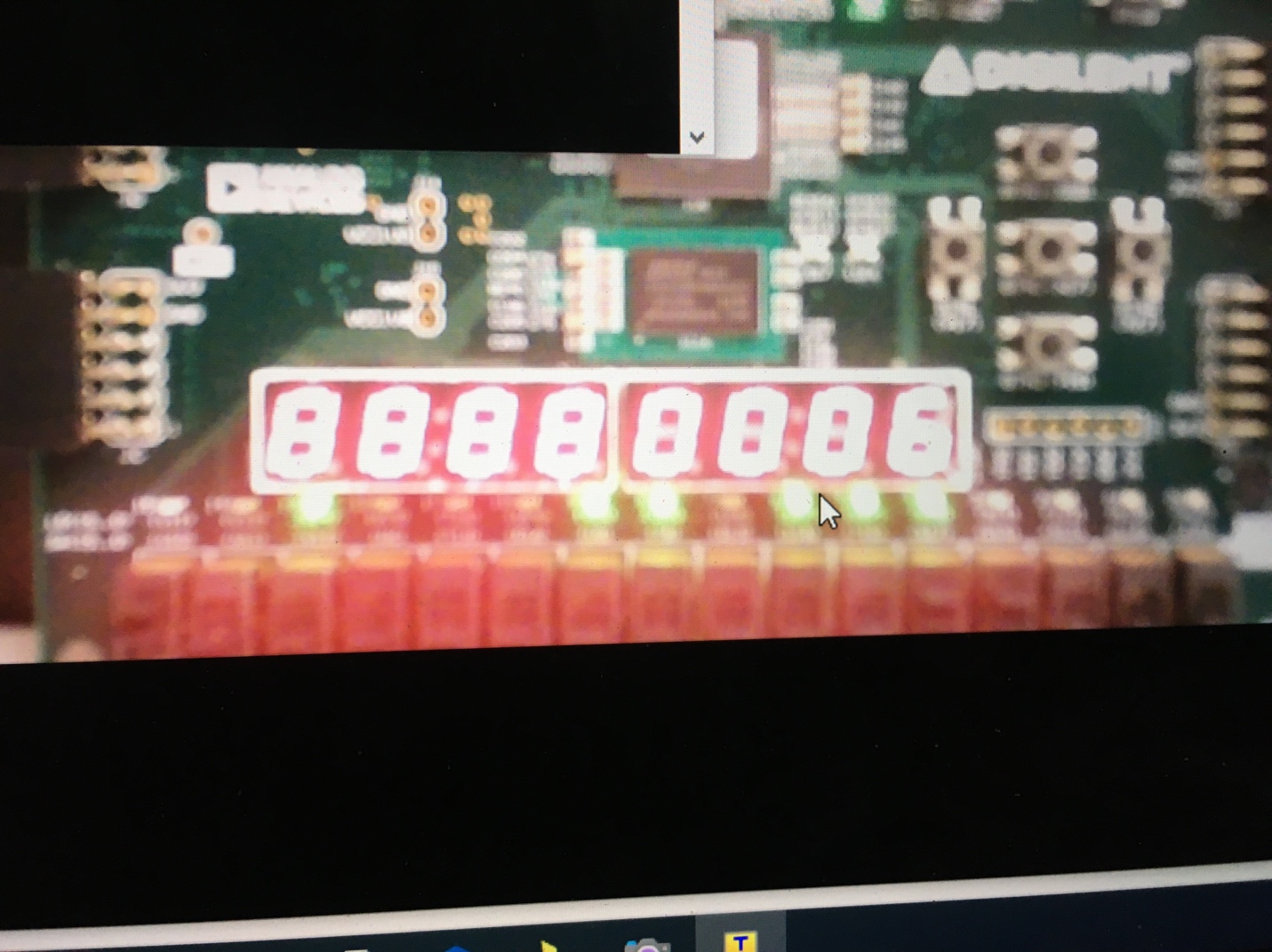
      leds[3:2]=  01;

      leds[4]=  0; //memread=0

      leds[5]= 0; //memwrite=0

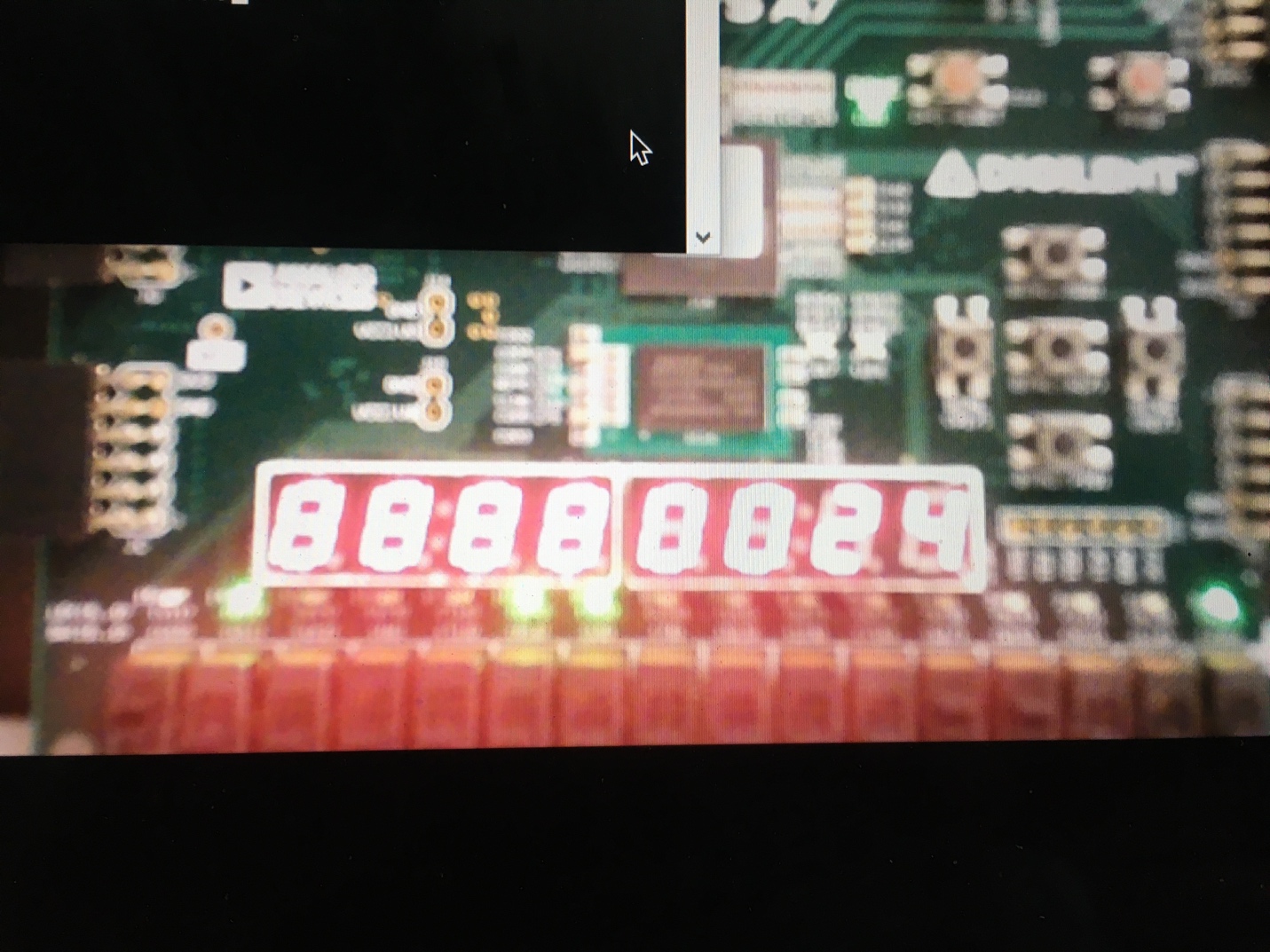
      leds[6]=  1 //memreg=x

      leds[7]=  1;  //beanch =1



SSD=111, Imm gen =6

**instruction -> sw t4, 12(zero). #pc=24**



As shown, SSDsel=0000,PC=24,

Ledsel=11, then leds =0100 0110

Leds[0]= 0;

      leds[1]= 1;

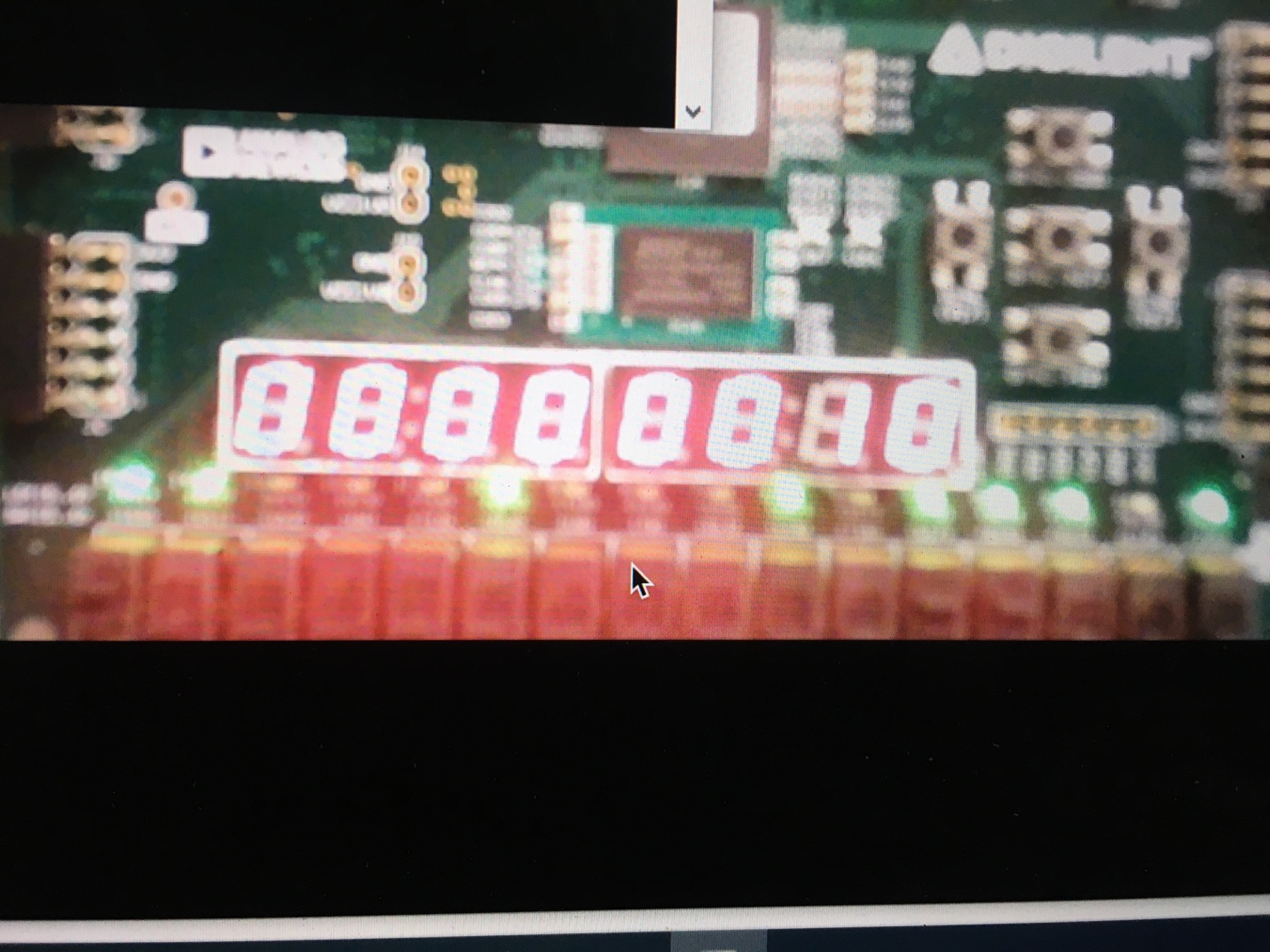
      leds[3:2]=  00;

      leds[4]=  0; //memread=0

      leds[5]= 1; //memwrite=0

      leds[6]=  1 //memreg=1

      leds[7]=  0;  //beanch =1



As shown, SSDsel=0101,read data 2=10,

Ledsel=11,

leds[7:0]= {0,ints\_out[6:0]};

Opcode =11000100