CPU Architecture
Final Project
Entropy Filter and Histogram

Shir Chen 203869698
Adir Ben Azarya 203904610

Table of contents:

Part 1- MIPS

Top level block review diagram : Mips

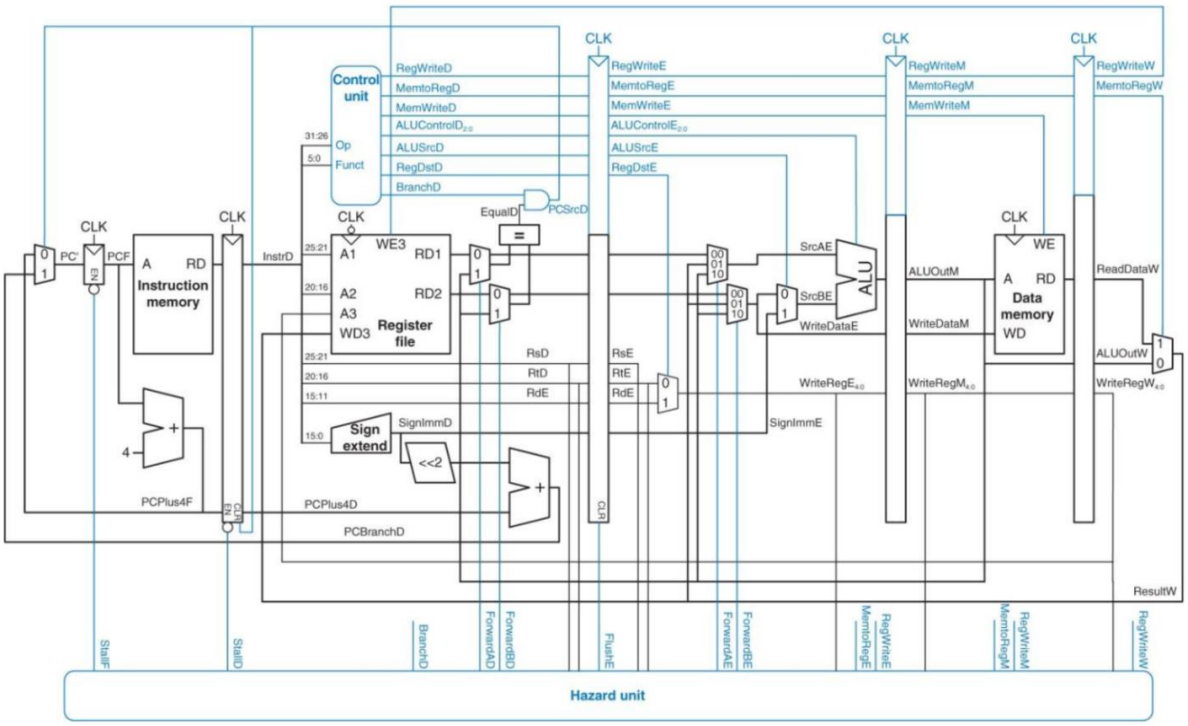


Figure 1:top level block

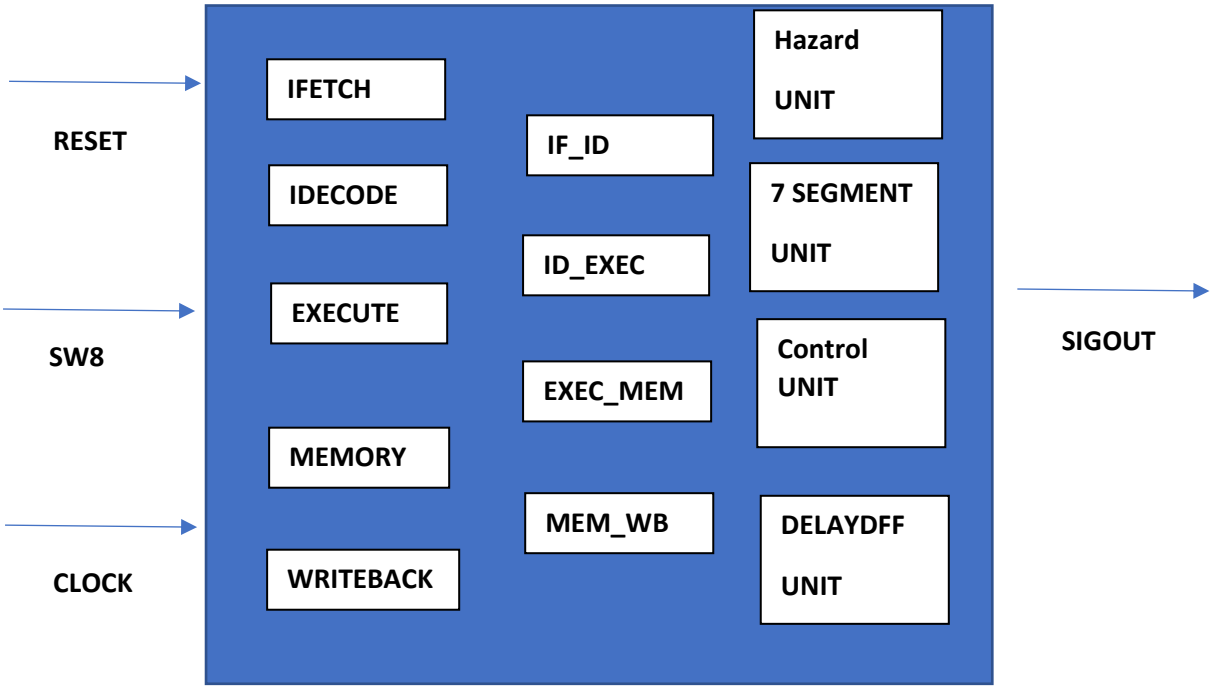


Figure 2:top level inside blocks

Port	Direction	Size	Functionality
reset	In	1 bit	Restart button
SW8	In	1 bit	LOW/HIGH output result
clock	in	1 bit	Clock
sigOut	Out	28 bits	Output of result to 7 segments

Table 1:Mips ports

IFetch Module:

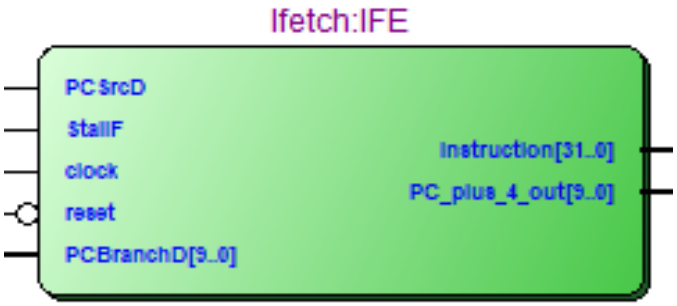


Figure 3:IFetch block

Port	Direction	Size	Functionality
reset	In	1 bit	Restart button
PCBranchD	In	10 bits	Branch Address
clock	In	1 bit	Clock
PCSrcD	In	1 bit	Switch: Branch or PC_plus4
StallF	In	1 bit	Stall
instruction	Out	32 bits	Full instruction
PC_plus4	Out	10 bits	Next address

Table 2:IFetch ports

RTL review:

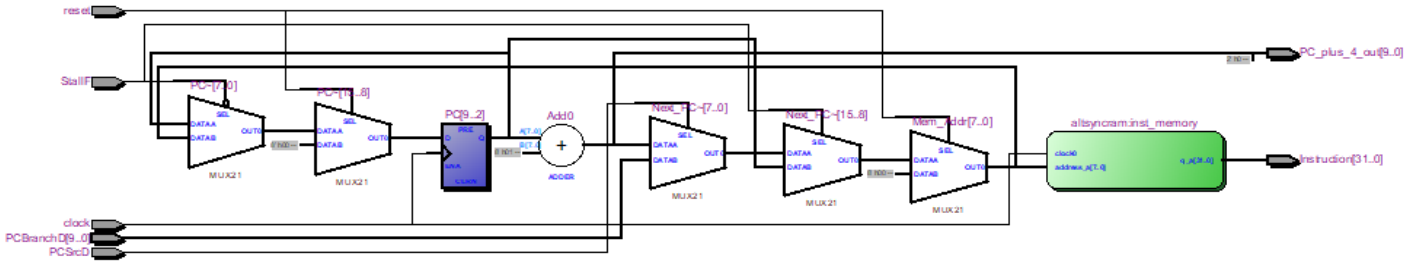


Figure 4: IFetch RTL

Description:

The unit which extracts the instruction from the script one by one. It is based on the given "Ifetch unit" with the addition to support the pipeline functionality: The ability to be stalled in hazard-situations (inserting "bubbles").

IDecode Module:

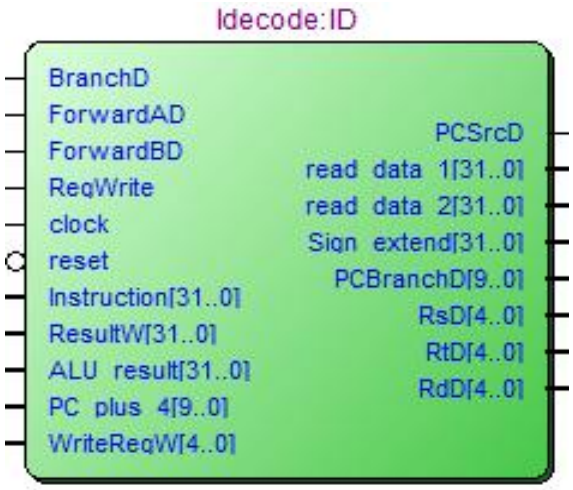


Figure 5: IDecode block

Port	Direction	Size	Functionality
BranchD	In	1 bit	Control unit branch
ForwardAD	In	1 bit	Switch: Forwarding result
ForwardBD	In	1 bit	Switch Forwarding result
RegWrite	In	1 bit	Switch: read or write to register
clock	In	1 bit	clock
instruction	In	32 bits	Full instruction
ResultW	In	32 bits	Write back result
ALU_result	In	32 bits	ALU result
PC_plus4	In	32 bits	Next address
WriteRegW	In	5 bits	Register to write in
PCSrcD	Out	1 bit	Switch: Branch or PC_plus4
Read_data1	Out	32 bits	Data of register
Read_data2	Out	32 bits	Data of register
Sign_extend	Out	32 bits	Address extended to 32 bit
PCBranchD	Out	10 bits	Branch Address
RsD	Out	5 bits	Source register
RtD	Out	5 bits	Target register
RdD	Out	5 bits	Register

Table 3: IDecode port

Description:

The unit which decodes the instructions. It is based on the given "decode unit" with a change: the branch calculation is in the decode(not in execute).

Execute Module:

Figure 6: Execute block

Port	Direction	Size	Functionality
ALUSrc	In	1 bit	Switch: Immediate sign extension or read register
RegDstE	In	1 bit	Switch: RtE or RdE
clock	In	1 bit	clock
reset	In	1 bit	reset
RsE	In	5 bits	Source register
RtE	In	5 bits	Target register
RdE	In	5 bits	Register
ALUControlE	In	3 bits	Switch between ALU ops
ResultW	In	32 bits	Write back result
Sign_extend	In	32 bits	Address extended to 32 bit
ForwardAE	In	2 bits	Switch: Forwarding result
ForwardBE	In	2 bits	Switch: Forwarding result
Read_data1	In	32 bits	Data of register
Read_data2	In	32 bits	Data of register
ALUOutM	In	32 bits	ALU result in memory unit
ALU_Result	Out	32 bits	Current Stage ALU result
WriteDataE	Out	32 bits	Data to write in memory
WriteRegE	Out	5 bits	Register to write in

Table 4: Execute ports

Description:

The unit which executes the instructions. It is based on the given "execute unit" with a change: the branch calculation is in the decode(not in execute).

Memory Module:

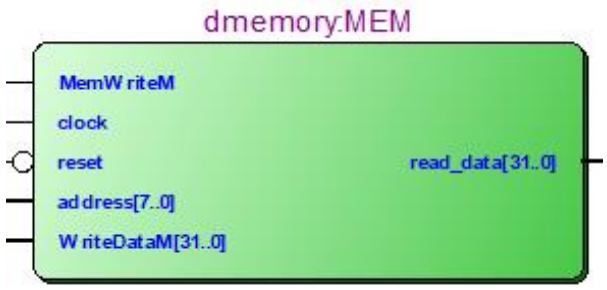


Figure 7: Memory block

Port	Direction	Size	Functionality
MemWriteM	In	1 bit	Write Enable to Memory
Address	In	7 bits	Address to write
clock	In	1 bit	clock
reset	In	1 bit	reset
WriteDataM	In	32 bits	Data to write in memory
Read_data	Out	32 bits	Data to Read from memory

Table 5: Memory ports

Description:

The unit which writes into the memory and sends out the data when there are reading ops.
It is based on the given "Memory unit".

RTL review:

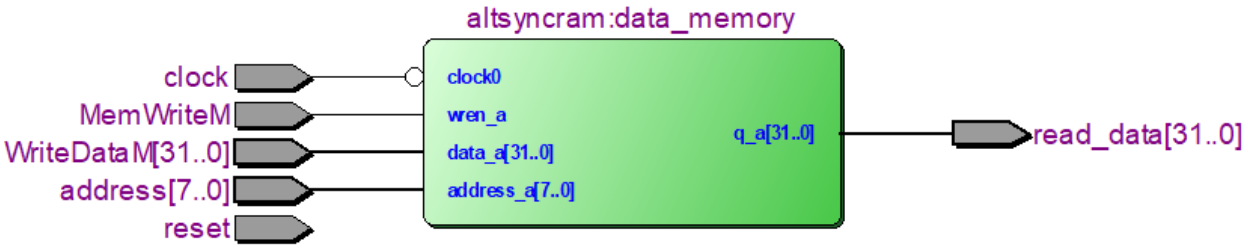


Figure 8: Memory RTL

WriteBack Module:

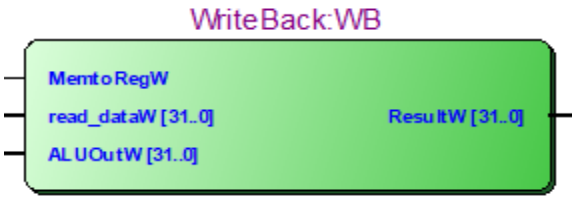


Figure 9: WriteBack block

Port	Direction	Size	Functionality
MemtoRegW	In	1 bit	Switch: data to read or ALUOut
Read_dataW	In	32 bits	Data to Read from memory
ALUOutW	In	32 bits	ALU result in writeback unit
ResultW	Out	32 bits	Write back result

Table 6: WriteBack ports

Description:

The unit which writes back into the registers or sends out the data when there are reading ops.

RTL review:

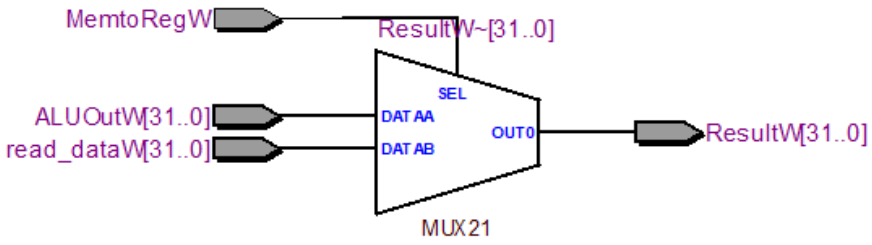


Figure 10: WriteBack RTL

Registers:

In order to implements pipeline architecture we have to use registers to store the data in every state.

IF_ID Module:

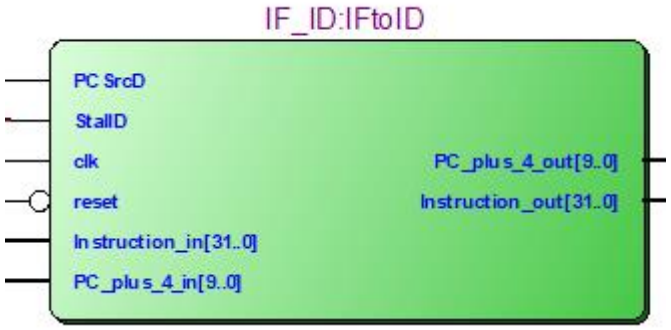


Figure 11: IF_ID block

Port	Direction	Size	Functionality
PCSrcD	In	1 bit	Switch: Branch or PC_plus4
StallD	In	1 bit	Stall
clock	In	1 bit	clock
reset	In	1 bit	reset
Instruction_in	In	32 bits	Full instruction from fetch unit
PC_plus4_in	In	10 bits	Next address from fetch unit
Instruction_out	Out	32 bits	Full instruction to decode unit
PC_plus4_out	Out	10 bits	Next address from decode unit

Table 7: IF_ID ports

Description:

The unit which stores the data from fetch unit and sends it to decode unit.

RTL review:

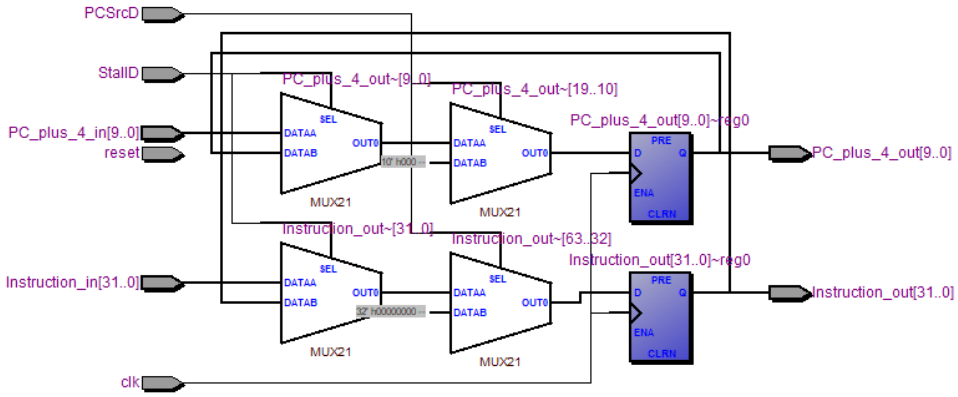


Figure 12: IF_ID RTL

ID_EX Module:

Figure 13: ID_EX block

Port	Direction	Size	Functionality
ALUSrcD	In	1 bit	Switch: Immediate sign extension or read register
Flush	In	1 bit	Set zeros
clock	In	1 bit	clock
MemWriteD	In	1 bit	Write enable to Memory
MemtoRegD	In	1 bit	Switch: data to read or ALUOut
RegDstD	In	1 bit	Switch: RtE or RdE
RegWriteD	In	1 bit	Switch: read or write to register
ALUControlD	In	3 bits	Switch between ALU ops
Sign_extendD	In	32 bits	Address extended to 32 bit
Read_data1D	In	32 bits	
Read_data2D	In	32 bits	
RsD	In	5 bits	Switch: Forwarding result
RtD	In	5 bits	Switch: Forwarding result
RdD	In	5 bits	Data of register
ALUSrcE	In	1 bit	Switch: Immediate sign extension or read register
MemWriteE	Out	1 bit	Write enable to Memory
MemtoRegE	Out	1 bit	Switch: data to read or ALUOut
RegDstE	Out	1 bit	Switch: RtE or RdE
RegWriteE	Out	1 bit	Switch: read or write to register
Read_data1E	Out	32 bits	Data of register
Read_data2E	Out	32 bits	Data of register
ALUControlE	Out	3 bits	Switch between ALU ops
Sign_extendE	Out	32 bits	Address extended to 32 bit
RsE	Out	5 bits	Switch: Forwarding result
RtE	Out	5 bits	Switch: Forwarding result
RdE	Out	5 bits	Data of register

Table 8: ID_EX ports

Description:

The unit which stores the data from decode unit and sends it to execute unit.

EX_Mem Module:

Figure 14: EX_Mem block

Port	Direction	Size	Functionality
MemWriteE	In	1 bit	Write enable to Memory
MemtoRegE	In	1 bit	Switch: data to read or ALUOut
RegWriteE	In	1 bit	Switch: read or write to register
clock	In	1 bit	clock
enable	In	1 bit	enable
reset	In	1 bit	reset
ALU_ResultE	In	32 bits	ALU result
WriteDataE	In	32 bits	Data to write in memory
WriteRegE	In	5 bits	Register to write in
MemWriteM	Out	1 bit	Write enable to Memory
MemtoRegM	Out	1 bit	Switch: data to read or ALUOut
RegWriteM	Out	1 bit	Switch: read or write to register
ALUOutM	Out	32 bits	ALU result in memory unit
WriteDataM	Out	32 bits	Data to write in memory
WriteRegM	Out	5 bits	Register to write in

Table 9: EX_Mem ports

Description:

The unit which stores the data from execute unit and sends it to memory unit.

RTL review:

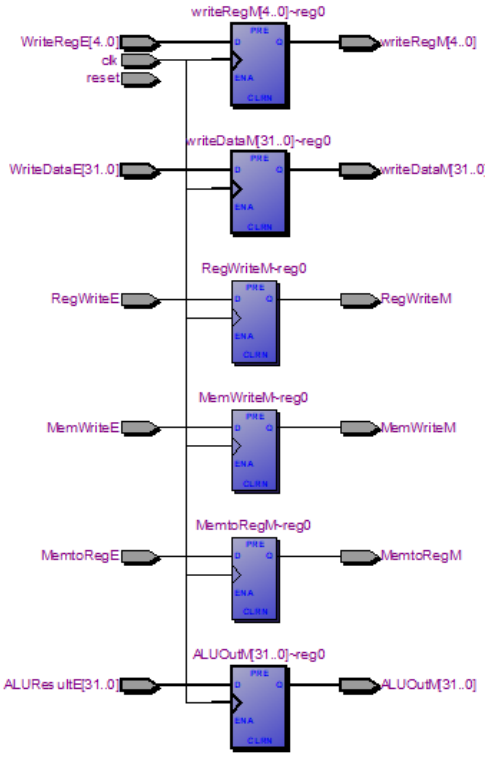


Figure 15: EX_Mem RTL

Mem_WB Module:

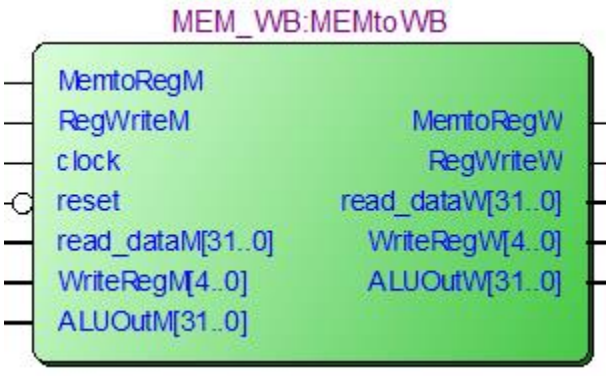


Figure 16: Mem_WB block

Port	Direction	Size	Functionality
MemtoRegM	In	1 bit	Switch: data to read or ALUOut
RegWriteM	In	1 bit	Switch: read or write to register
clock	In	1 bit	clock
reset	In	1 bit	reset
Read_dataM	In	32 bits	Data to Read from memory
WriteRegM	In	5 bits	Register to write in
ALUOutM	in	32 bits	ALU result in memory unit
MemtoRegW	Out	1 bit	Switch: data to read or ALUOut
RegWriteW	Out	1 bit	Switch: read or write to register
Read_dataW	Out	32 bits	Data to Read from memory
WriteRegW	Out	5 bits	Register to write in
ALUOutW	Out	32 bits	ALU result in memory unit

Table 10: Mem_WB ports

Description:

The unit which stores the data from memory unit and sends it to write back unit.

RTL review:

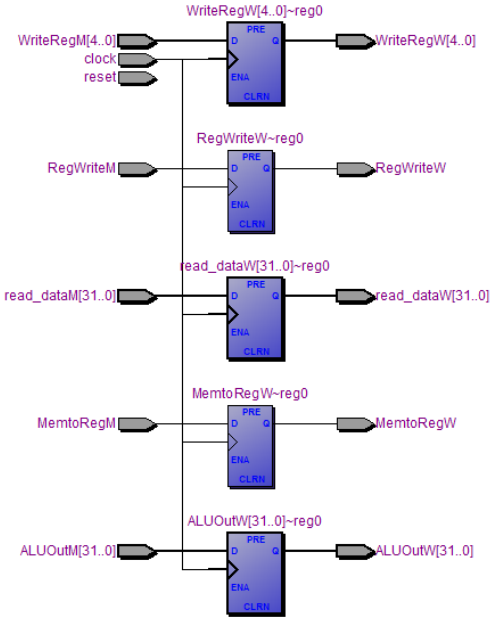


Figure 17: Mem_WB RTL

Hazard Unit module:

Figure 18: Hazard unit block

Port	Direction	Size	Functionality
BranchD	In	1 bit	Control unit branch
MemtoRegE	In	1 bit	Switch: data to read or ALUOut
MemtoRegM	In	1 bit	Switch: data to read or ALUOut
RegWriteE	In	1 bit	Switch: read or write to register
RegWriteM	In	1 bit	Switch: read or write to register
RegWriteW	In	1 bit	Switch: read or write to register
clk	In	1 bit	clock
RsE	In	5 bits	Source register
RtE	In	5 bits	Target register
RdE	In	5 bits	Register
WriteRegW	In	5 bits	Register to write in
WriteRegE	In	5 bits	Register to write in
WriteRegM	In	5 bits	Register to write in
FlushE	Out	1 bit	Set zeros
ForwardAD	Out	1 bit	Switch: Forwarding result
ForwardBD	Out	1 bit	Switch: Forwarding result
StallD	Out	1 bit	Stall
StallF	Out	1 bit	Stall
ForwardAE	Out	1 bit	Switch: Forwarding result
ForwardBE	Out	1 bit	Switch: Forwarding result

Table 11: Hazard unit ports

Description:

The unit which detects hazard in the program. When detects, sends the appropriate signals to handle it.

Control Unit module:

Figure 19: Control unit block

Port	Direction	Size	Functionality
clk	In	1 bit	clock
reset	In	1 bit	reset
Opcode	In	6 bits	
Funct	In	6 bits	Switch: read or write to register
ALUSrc	Out	1 bit	Switch: Immediate sign extension or read register
Branch	Out	1 bit	Control unit branch
MemWrite	Out	1 bit	Write Enable to Memory
MemtoReg	Out	1 bit	Switch: data to read or ALUOut
RegDstE	Out	1 bit	Switch: RtE or RdE
RegWrite	Out	1 bit	Switch: read or write to register
ALUControlE	Out	3 bits	Switch between ALU ops

Table 12: Control unit ports

Description:

The Control Unit receives the Op-code from the instruction and dispatches several signals to units in different stages.

RTL review:

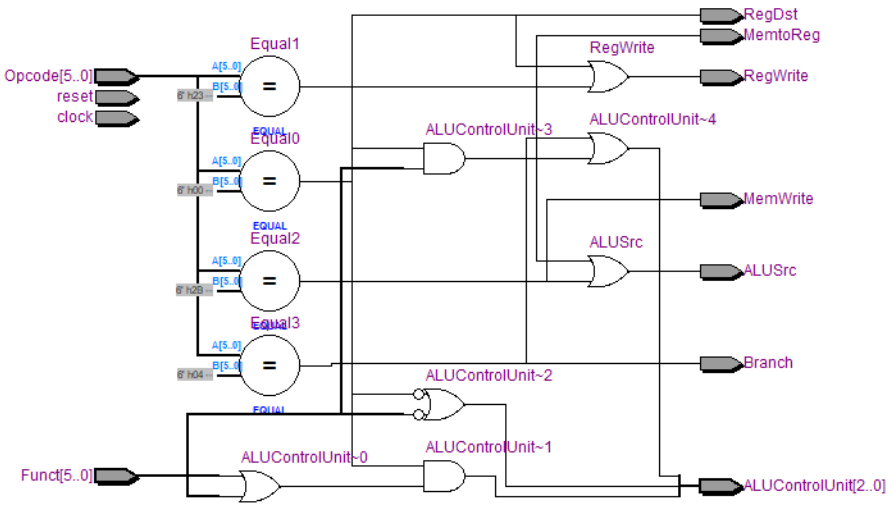


Figure 20: Control unit RTL

DelayDFF module:

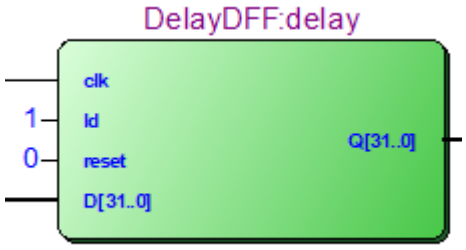


Figure 21: DelayDFF block

Port	Direction	Size	Functionality
clk	In	1 bit	clock
ld	In	1 bit	load
reset	In	1 bit	reset
D	In	32 bits	Data to store
Q	Out	32 bits	Next data

Table 13: DelayDFF ports

Description:

The DelayDFF Unit does exactly like a regular DFF with a small change: it stores the data after specific time. Therefore the delay.

RTL review:

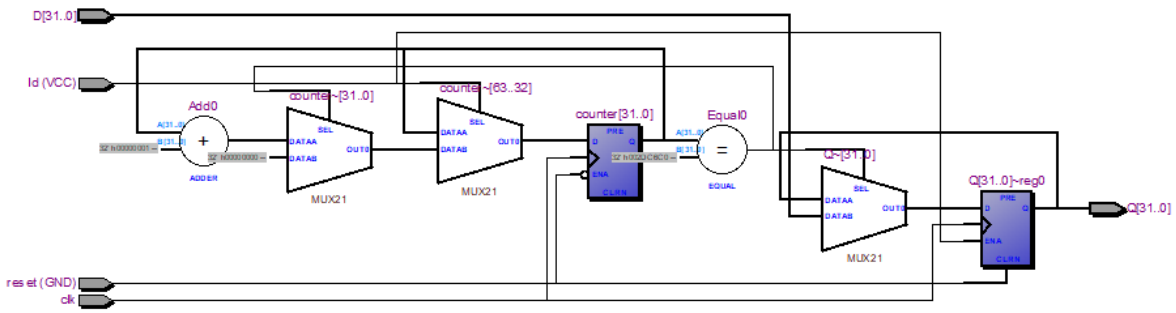


Figure 22: Control unit RTL

Part 2- Camera

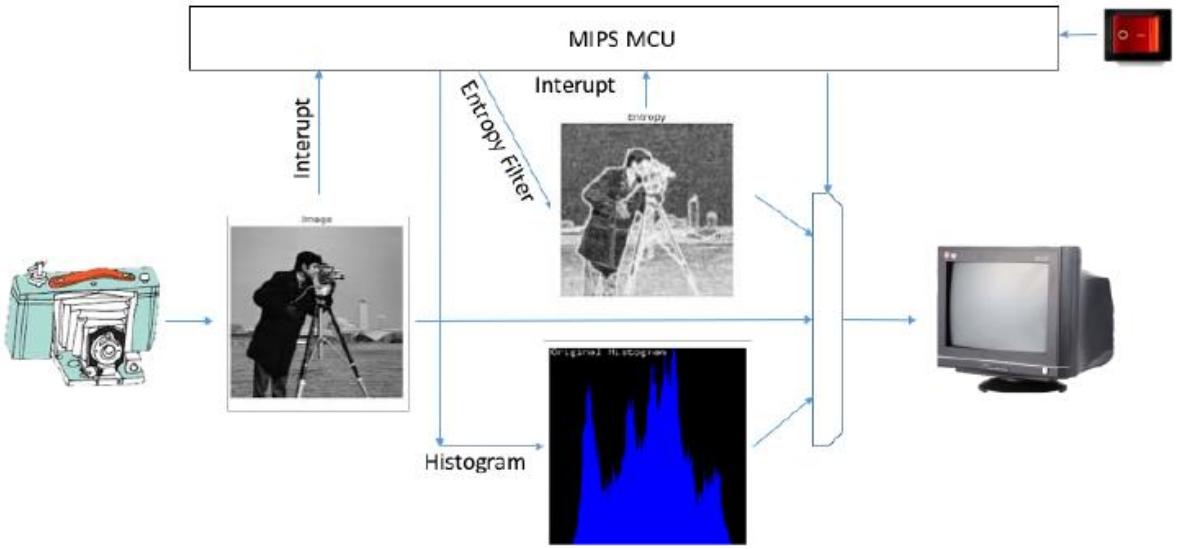


Figure 23: Processing Diagram

Top level overview:

The top level entity of the project includes components that process a captured image in several different ways. First, the component of CCD captured a colored image. Next, according to the switch we select, colored image is passed through a greyscale filter or histogram or Entropy filter. The processed imaged is then finally passed through the SD memory and VGA controller and is displayed the LCD screen.

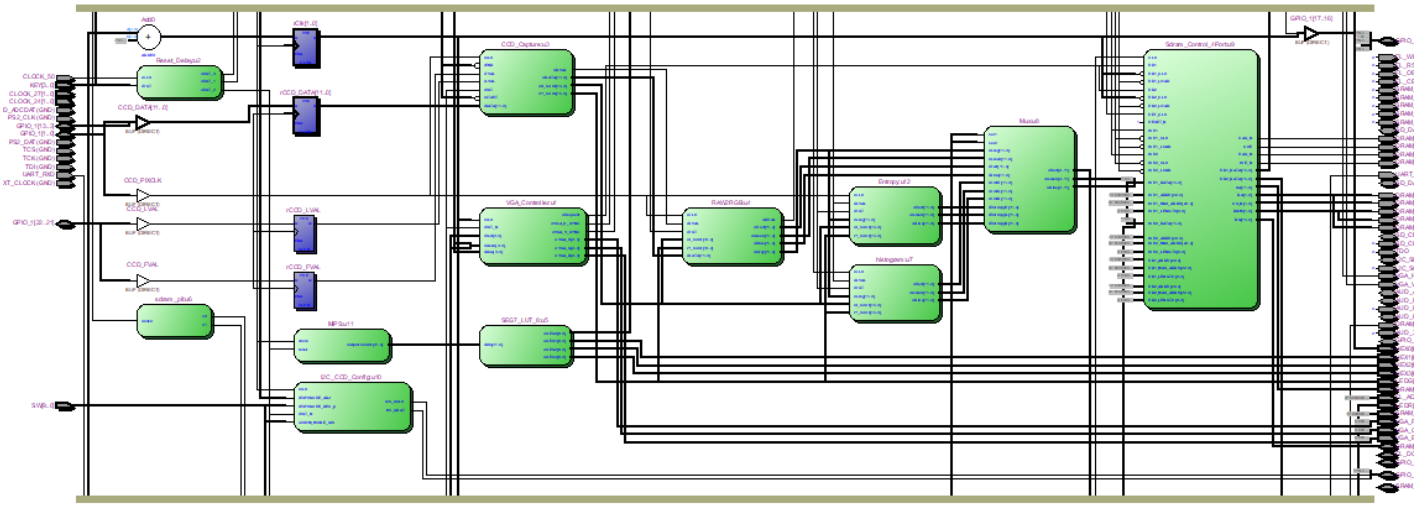


Figure 24: camera RTL

Mips and Camera integration:

When a new frame is captured in the CCD_Capture component, an interrupt is sent from the camera module to the MIPS fetch component.

This interrupt triggers a jump in the program to an instruction address of a counter -ISR_Count. The counter is displayed on the 7-Segment display.

Meanwhile in the background, the MIPS is running a simple program another endless loop-Loop.

Assembly code:

```
.data
one: .word 1
counter: .word 0
.text
# Before running this code make sure that
# Settings -> Memory Configuration -> Compact, Data at Address 0
    add  $t0, $zero,$zero
    lw  $t1, counter
    lw  $t2, one

Loop:
   add $t0,$t0,$t2
   nop
   nop
   beq $zero,$zero,Loop
   nop
   nop
   nop
   nop
   nop
   nop
ISR_Count:
   add $t1,$t1,$t2
   sw $t1,counter
   break
```

Raw2RGB module:

Figure 25: Raw2RGB Module

Port	Direction	Size	Functionality
iCLK	In	1 bit	clock
iDVAL	In	1 bit	Checks active frame
iRST	In	1 bit	reset
iX_Cont	In	11 bits	X cordinate
iY_Cont	In	11 bits	Y cordinate
IDATA	In	12 bits	Pixle in
oRed	Out	12 bits	Red out
oGreen	Out	12 bits	Green out
oBlue	Out	12 bits	Blue out
oGrey	Out	12 bits	Grey out

Table 14: RAW2RGB ports

Description:

The RAW2RGB Unit receives raw data from the capture component of the Camera unit(CCD) and translates it into red, green, and blue tonal values for each pixel. The unit is almost the same as we receive it except for changing the red, green and blue according to formula:

$$Grey = .25 \times Red + .03125 \times Red + .50 \times Green + 0.125 \times Green + 0.125 \times Blue + 0.03125 \times Blue$$

$$Grey = .28125 \times Red + .6125 \times Green + 0.15625 \times Blue$$

We changed the values according to the hardware.

Histogram module:

Figure 26: Histogram Module

Port	Direction	Size	Functionality
iCLK	In	1 bit	clock
iDVAL	In	1 bit	Checks active frame
iRST	In	1 bit	reset
iGrey	In	12 bits	Pixle in
iX_Cont	In	16 bits	X cordinate
iY_Cont	In	16 bits	Y cordinate
oRed	Out	12 bits	Red out
oGreen	Out	12 bits	Green out
oBlue	Out	12 bits	Blue out

Table 14: Histogram ports

Description:

The Histogram Unit receives a digital image from the RAW2RGB unit(which gives pixels in greyscale) processes the data, and gives a graphical representation of the image. It plots the number of pixels for each tonal value into 16 levels of intensities.

Entropy filter module:

Figure 27: Entropy filter Module

Port	Direction	Size	Functionality
iCLK	In	1 bit	clock
iDVAL	In	1 bit	Checks active frame
iRST	In	1 bit	reset
iGrey	In	12 bits	Pixle in
iX_Cont	In	16 bits	X cordinate
iY_Cont	In	16 bits	Y cordinate
oRed	Out	12 bits	Red out
oGreen	Out	12 bits	Green out
oBlue	Out	12 bits	Blue out

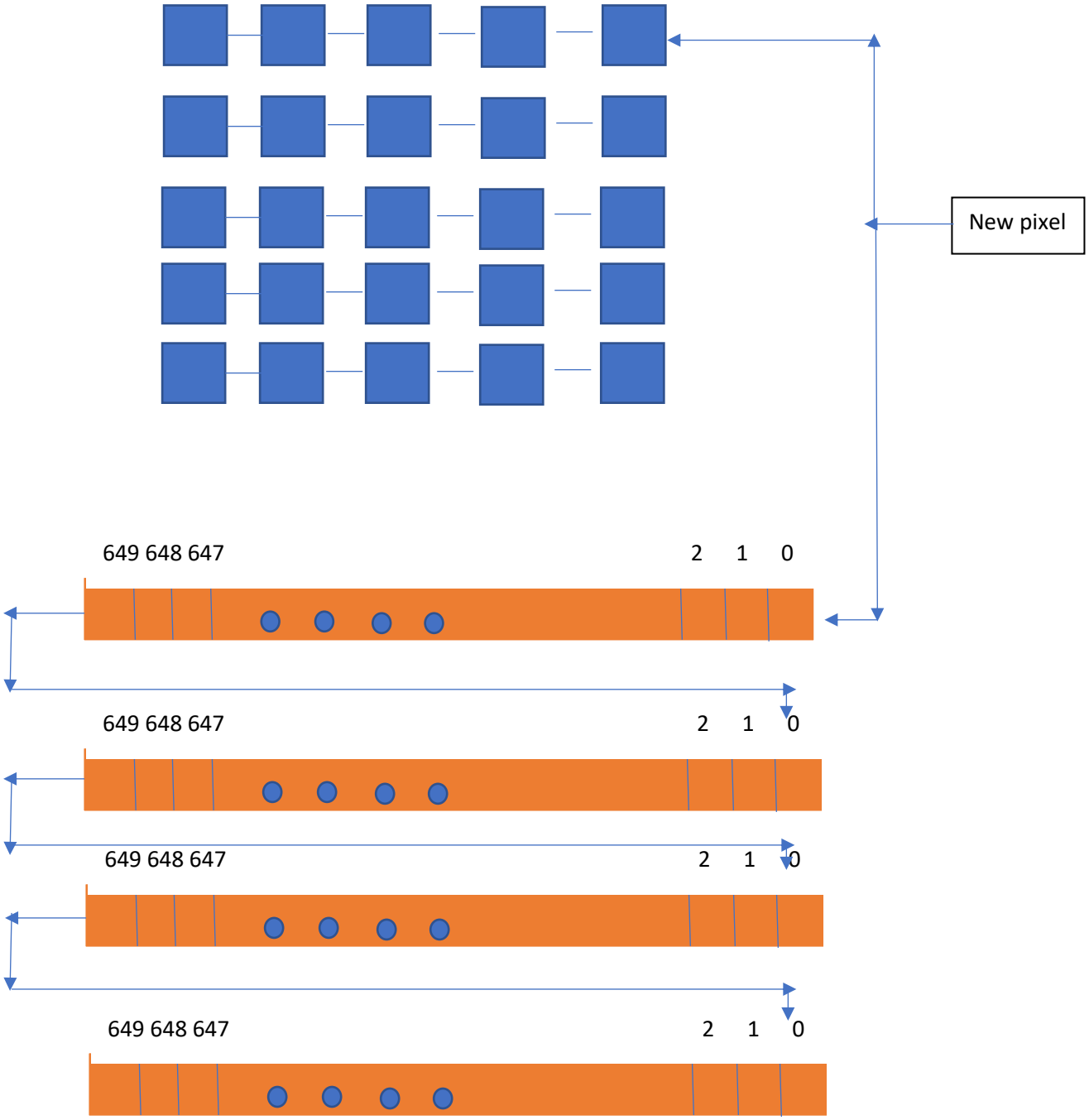
Table 15: Entropy filter ports

Description:

The Entropy filter Unit receives a digital image from the RAW2RGB unit, and passes the image through a Entropy detection filter. The component is made up of 4 line buffers that, at any given point in time, contain 4 rows of pixels of a certain frame. The 4 buffers contain 640 pixels . These 4 line buffers are used for the convolution operation that is implemented on a frame in order to give the desired filter.

This filter detects the differnces between colors in the frame. When there is difference it paints with white else it paints with black.

The process:



In addition there connections: windowLine5(0) := to_integer(unsigned(line5(0)))

windowLine4(0) := to_integer(unsigned(line4(0)))

windowLine3(0) := to_integer(unsigned(line3(0)))

windowLine2(0) := to_integer(unsigned(line2(0)))

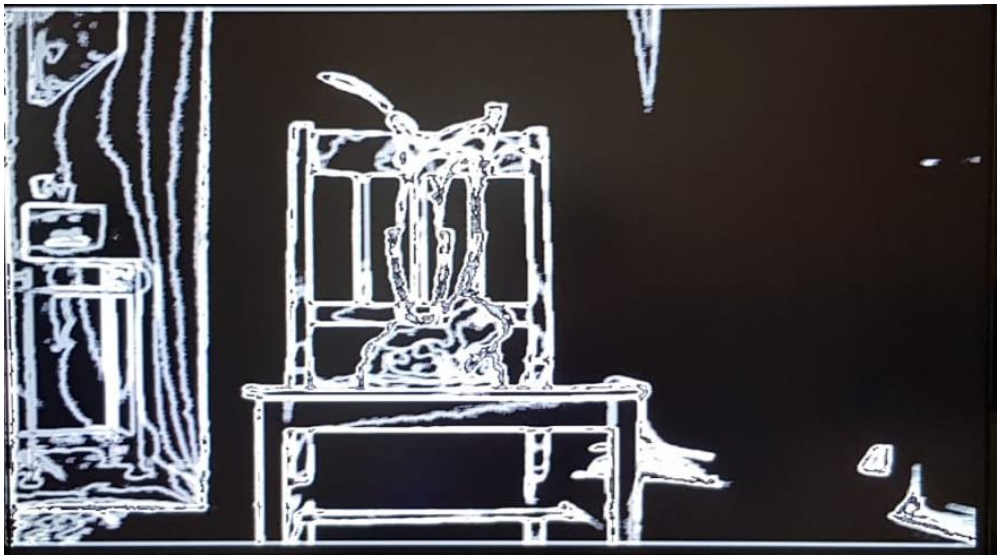
windowLine1(0) := to_integer(unsigned(line1(0)))

Switches:

“00” original frame:

“01” greyscale:

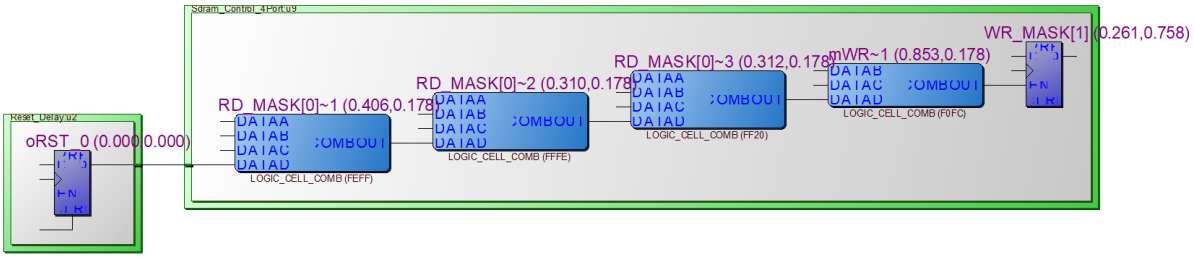
“10” entropy filter:

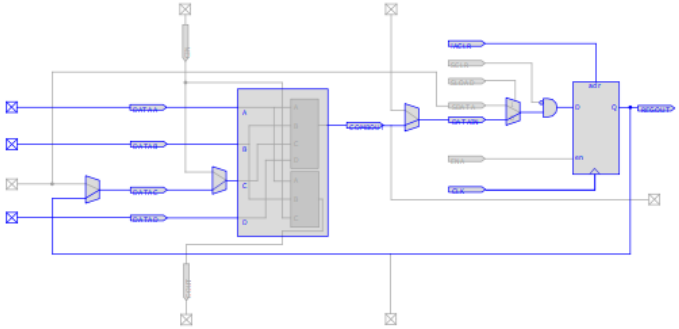


“11” histogram:

Part 3- Analysis and Simulation

Critical Path:





Logic usage and Fmax:

Flow Summary	
Flow Status	Successful - Sun Jun 17 10:56:53 2018
Quartus II 32-bit Version	12.1 Build 177 11/07/2012 SJ Web Edition
Revision Name	DE1_D5M
Top-level Entity Name	DE1_D5M
Family	Cyclone II
Device	EP2C20F484C7
Timing Models	Final
Total logic elements	8,371 / 18,752 (45 %)
Total combinational functions	6,967 / 18,752 (37 %)
Dedicated logic registers	4,335 / 18,752 (23 %)
Total registers	4349
Total pins	302 / 315 (96 %)
Total virtual pins	0
Total memory bits	83,752 / 239,616 (35 %)
Embedded Multiplier 9-bit elements	0 / 52 (0 %)
Total PLLs	1 / 4 (25 %)

Slow Model Fmax Summary

	Fmax	Restricted Fmax	Clock Name	Note
1	41.44 MHz	41.44 MHz	clk	
2	143.08 MHz	143.08 MHz	u6|altpll_component|pll|clk[0]	