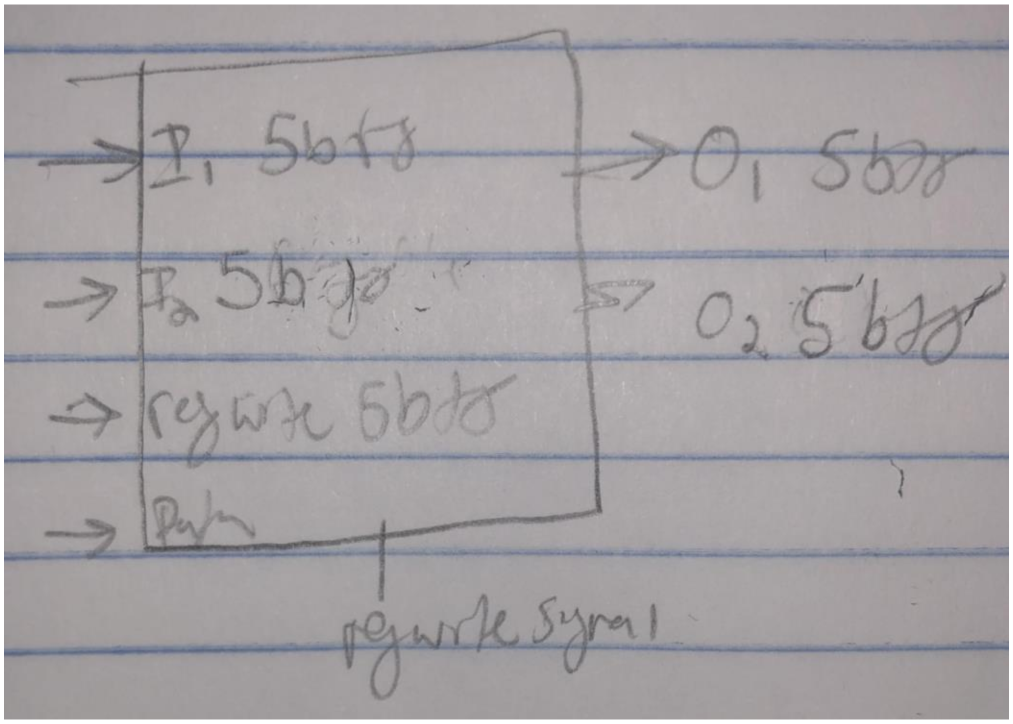
CprE 381, Computer Organization and Assembly-Level Programming

Lab 2 Report

Student Name Thriambak Giriprakash

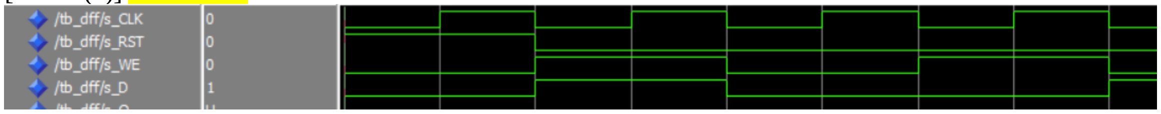
Submit a typeset pdf version of this on Canvas by the due date. Refer to the highlighted language in the lab document for the context of the following questions.

[Part 2 (a)] Draw the interface description for the MIPS register file. Which ports do you think are necessary, and how wide (in bits) do they need to be?



[Part 2 (b)] Create an N-bit register using this flip-flop as your basis.
Done

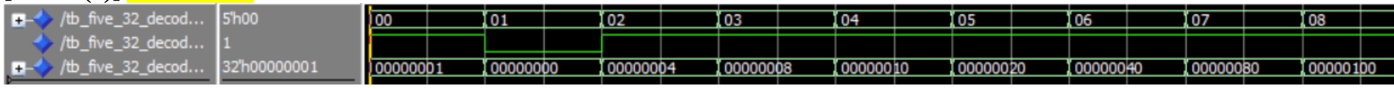
[Part 2 (c)] Waveform.



[Part 2 (d)] What type of decoder would be required by the MIPS register file and why?

5:32 bit decoder because an R type instruction takes a 5 bit address to read and write to a 32 bit register.

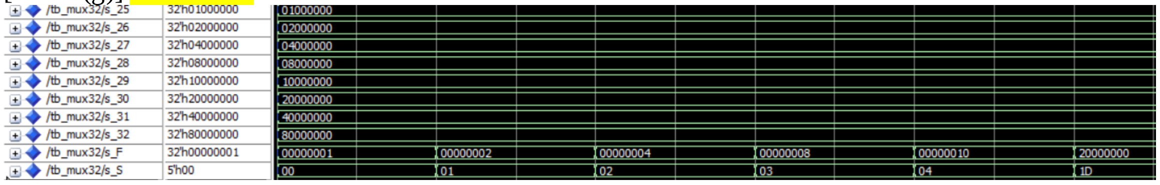
[Part 2 (e)] Waveform.



```
/tb_five_32_decod...   5'h00          00        01        02        03        04        05        06        07        08
/tb_five_32_decod...   1
/tb_five_32_decod...   32'h00000001   00000001  00000000  00000004  00000008  00000010  00000020  00000040  00000080  00000100
```

[Part 2 (f)] In your write-up, describe and defend the design you intend on implementing for the next part.
Going to use a dataflow design to keep neat in the code

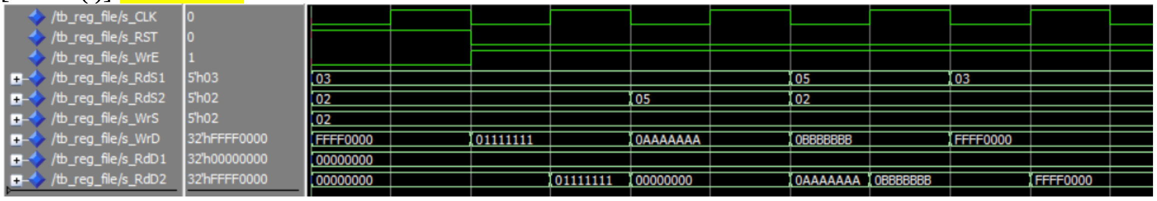
[Part 2 (g)] Waveform.



```
/tb_mux32/s_25   32'h01000000   01000000
/tb_mux32/s_26   32'h02000000   02000000
/tb_mux32/s_27   32'h04000000   04000000
/tb_mux32/s_28   32'h08000000   08000000
/tb_mux32/s_29   32'h10000000   10000000
/tb_mux32/s_30   32'h20000000   20000000
/tb_mux32/s_31   32'h40000000   40000000
/tb_mux32/s_32   32'h80000000   80000000
/tb_mux32/s_F    32'h00000001   00000001  00000002  00000004  00000008  00000010  20000000
/tb_mux32/s_S    5'h00          00        01        02        03        04        1D
```

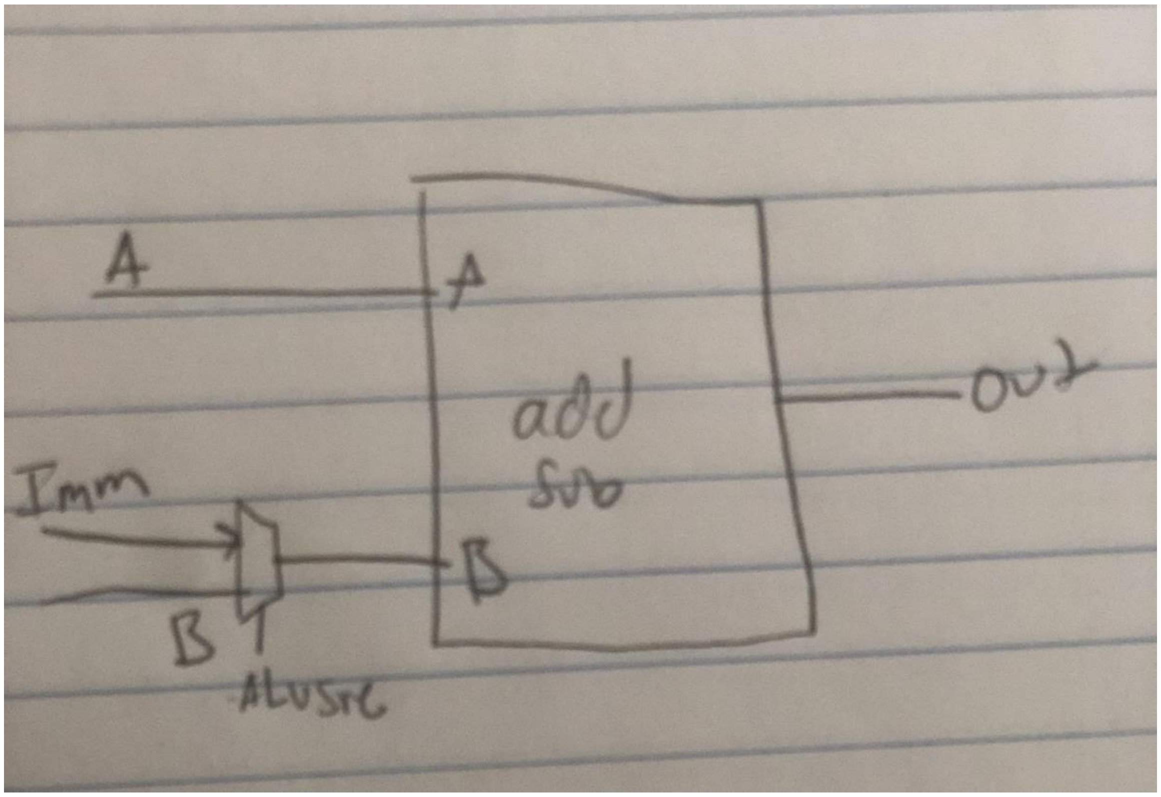
[Part 2 (h)] Draw a (simplified) schematic for the MIPS register file, using the same top-level interface ports as in your solution describe above and using only the register, decoder, and mux VHDL components you have created.

[Part 2 (i)] Waveform.

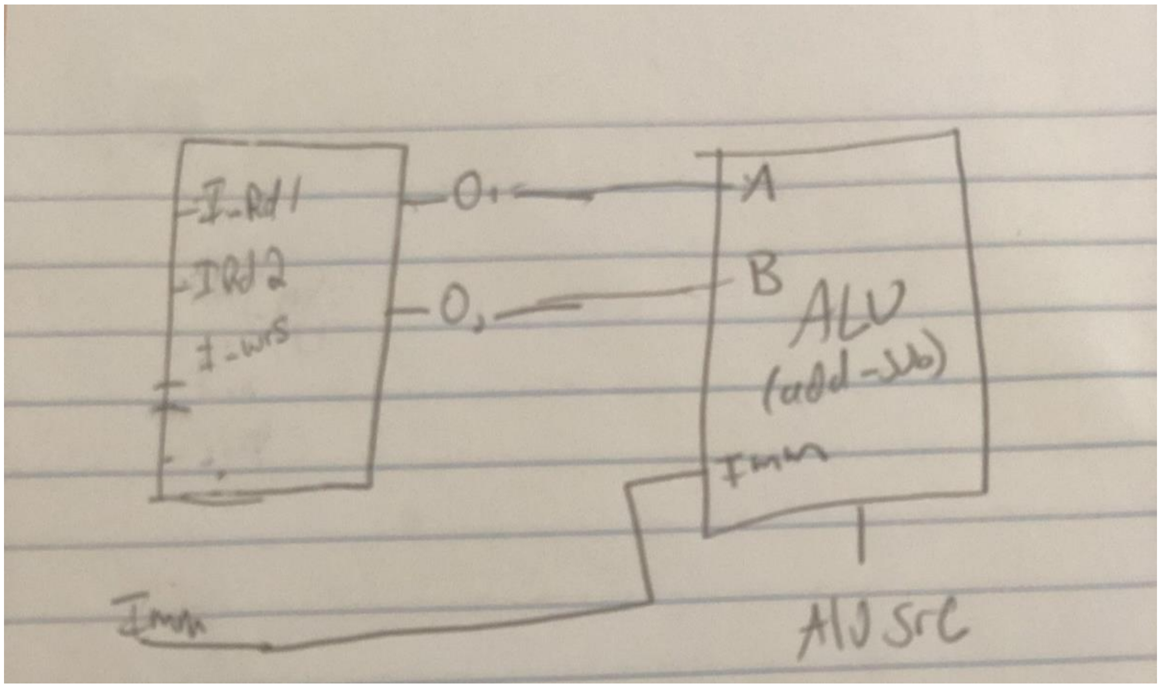


```
/tb_reg_file/s_CLK    0
/tb_reg_file/s_RST    0
/tb_reg_file/s_WrE    1
/tb_reg_file/s_RdS1   5'h03          03                                 05                  03
/tb_reg_file/s_RdS2   5'h02          02                        05       02
/tb_reg_file/s_WrS    5'h02          02
/tb_reg_file/s_WrD    32'hFFFF0000   FFFF0000  01111111        0AAAAAAA  0BBBBBBB  FFFF0000
/tb_reg_file/s_RdD1   32'h00000000   00000000
/tb_reg_file/s_RdD2   32'hFFFF0000   00000000  01111111  00000000  0AAAAAAA  0BBBBBBB  FFFF0000
```

[Part 3 (b)] Draw a symbol for this MIPS-like datapath.

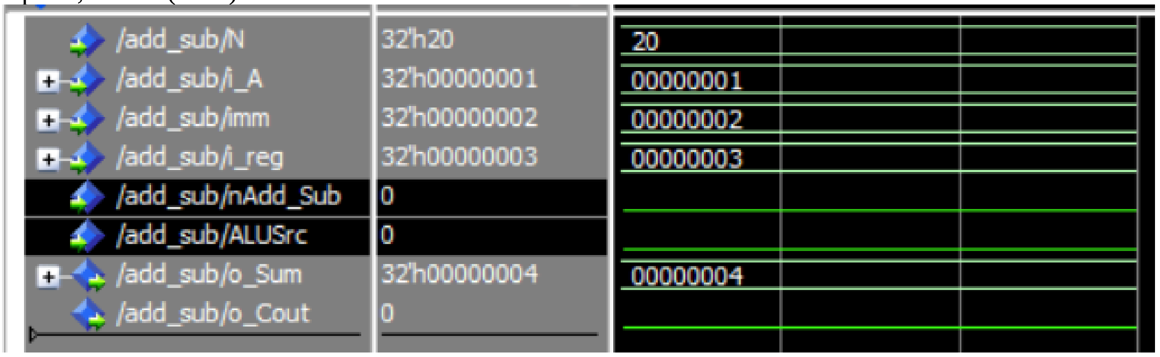


[Part 3 (c)] Draw a schematic of the simplified MIPS processor datapath consisting only of the component described in part (a) and the register file from problem (1).



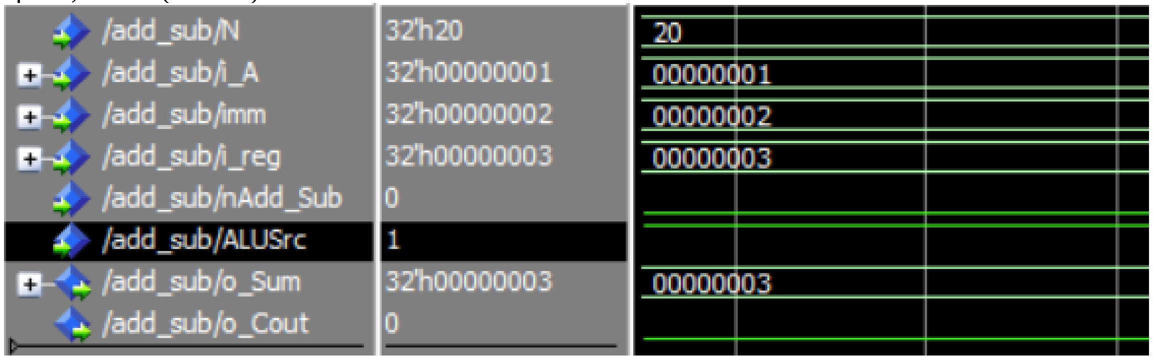
[Part 3 (d)] Include in your report waveform screenshots that demonstrate your properly functioning design. Annotate what the final register file state should be.

Op = 0,Src = 0 (A+B)



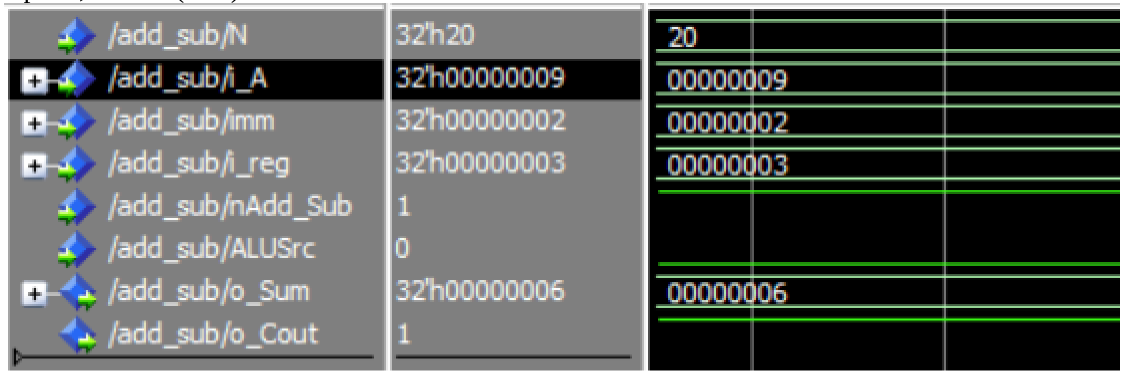
```
/add_sub/N          32'h20           20
/add_sub/i_A        32'h00000001     00000001
/add_sub/imm        32'h00000002     00000002
/add_sub/i_reg      32'h00000003     00000003
/add_sub/nAdd_Sub   0
/add_sub/ALUSrc     0
/add_sub/o_Sum      32'h00000004     00000004
/add_sub/o_Cout     0
```

Op = 0,Src = 1 (A+imm)



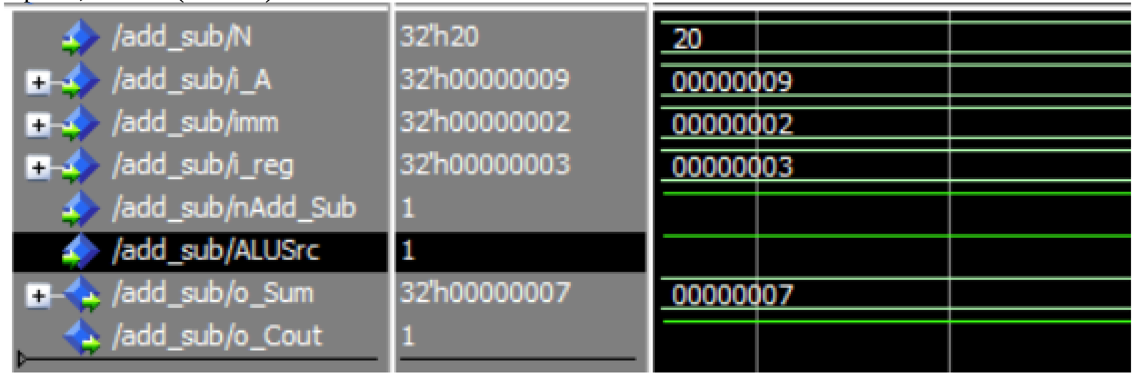
```
/add_sub/N          32'h20           20
/add_sub/i_A        32'h00000001     00000001
/add_sub/imm        32'h00000002     00000002
/add_sub/i_reg      32'h00000003     00000003
/add_sub/nAdd_Sub   0
/add_sub/ALUSrc     1
/add_sub/o_Sum      32'h00000003     00000003
/add_sub/o_Cout     0
```

Op = 1, Src = 0 (A-B)



/add_sub/N	32'h20	20
/add_sub/i_A	32'h00000009	00000009
/add_sub/imm	32'h00000002	00000002
/add_sub/i_reg	32'h00000003	00000003
/add_sub/nAdd_Sub	1	
/add_sub/ALUSrc	0	
/add_sub/o_Sum	32'h00000006	00000006
/add_sub/o_Cout	1	

Op = 1, Src = 1 (A- imm)



/add_sub/N	32'h20	20
/add_sub/i_A	32'h00000009	00000009
/add_sub/imm	32'h00000002	00000002
/add_sub/i_reg	32'h00000003	00000003
/add_sub/nAdd_Sub	1	
/add_sub/ALUSrc	1	
/add_sub/o_Sum	32'h00000007	00000007
/add_sub/o_Cout	1	

[Part 4 (a)] Read through the mem.vhd file, and based on your understanding of the VHDL implementation, provide a 2-3 sentence description of each of the individual ports (both generic and regular).

Generic Ports:
DATA_WIDTH: Sets up the number of bits of incoming data stream. Incoming data stream will be the data that the code works with.
ADDR_WIDTH: Sets up the number of bits of an incoming data stream. Incoming data stream will be the address that the data will get written to.

Regular Ports:
Clk: clock signal, used for timing operations on a realistic setting.

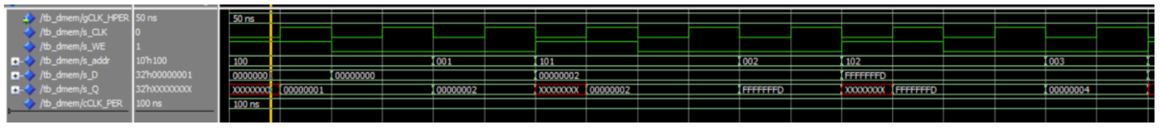
Addr. Address that the data will get written to. Will be using the size indicated from ADDR_WIDTH.

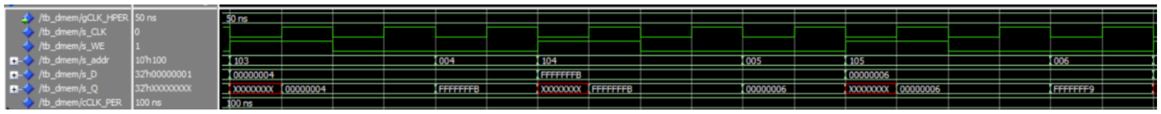
Data: Data that will get written to an address. Will be using the size indicated from DATA_WIDTH.

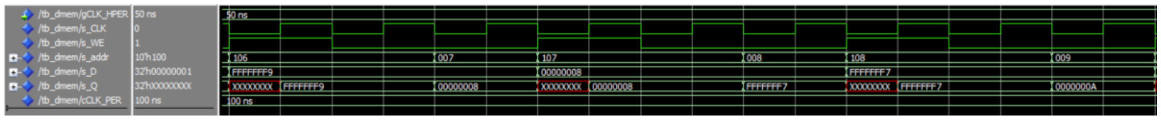
We: Value always stays as '1' used for adding data to the address in a loop later on.

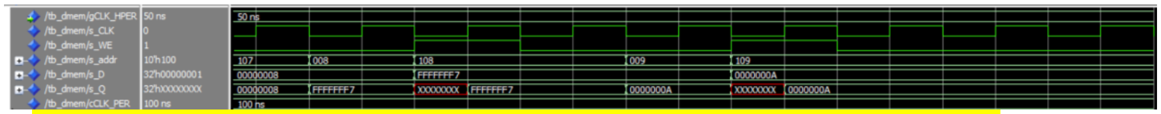
Q: output which shows the address successfully holds the data .

[Part 4 (c)] Waveforms.


/tb_dmem/gCLK_HPER 50 ns
/tb_dmem/s_CLK 0
/tb_dmem/s_WE 1
/tb_dmem/s_addr 10'h100
/tb_dmem/s_D 32'h00000001
/tb_dmem/s_Q 32'hXXXXXXXX
/tb_dmem/cCLK_PER 100 ns
50 ns
100 001 101 002 102 003
00000001 00000000 00000002 FFFFFFFD
XXXXXXXX 00000001 00000002 XXXXXXXX 00000002 FFFFFFFD XXXXXXXX FFFFFFFD 00000004
100 ns


/tb_dmem/gCLK_HPER 50 ns
/tb_dmem/s_CLK 0
/tb_dmem/s_WE 1
/tb_dmem/s_addr 10'h100
/tb_dmem/s_D 32'h00000001
/tb_dmem/s_Q 32'hXXXXXXXX
/tb_dmem/cCLK_PER 100 ns
50 ns
103 004 104 005 105 006
00000004 FFFFFFFB 00000006
XXXXXXXX 00000004 FFFFFFFB XXXXXXXX FFFFFFFB 00000006 XXXXXXXX 00000006 FFFFFFF9
100 ns


/tb_dmem/gCLK_HPER 50 ns
/tb_dmem/s_CLK 0
/tb_dmem/s_WE 1
/tb_dmem/s_addr 10'h100
/tb_dmem/s_D 32'h00000001
/tb_dmem/s_Q 32'hXXXXXXXX
/tb_dmem/cCLK_PER 100 ns
50 ns
106 007 107 008 108 009
FFFFFFF9 00000008 FFFFFFF7
XXXXXXXX FFFFFFF9 00000008 XXXXXXXX 00000008 FFFFFFF7 XXXXXXXX FFFFFFF7 0000000A
100 ns


/tb_dmem/gCLK_HPER 50 ns
/tb_dmem/s_CLK 0
/tb_dmem/s_WE 1
/tb_dmem/s_addr 10'h100
/tb_dmem/s_D 32'h00000001
/tb_dmem/s_Q 32'hXXXXXXXX
/tb_dmem/cCLK_PER 100 ns
50 ns
107 008 108 009 109
00000008 FFFFFFF7 0000000A
00000008 FFFFFFF7 XXXXXXXX FFFFFFF7 0000000A XXXXXXXX 0000000A
100 ns

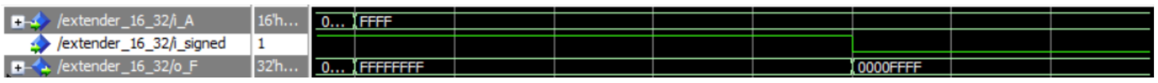
[Part 5 (a)] What are the MIPS instructions that require some value to be sign extended? What are the MIPS instructions that require some value to be zero extended?

Loads need to be sign extended and unsigned loads need to be zero extended

[Part 5 (b)] what are the different 16-bit to 32-bit "extender" components that would be required by a MIPS processor implementation?

Would need a zero and sign extender. An extender component could contain a control signal to switch between the modes

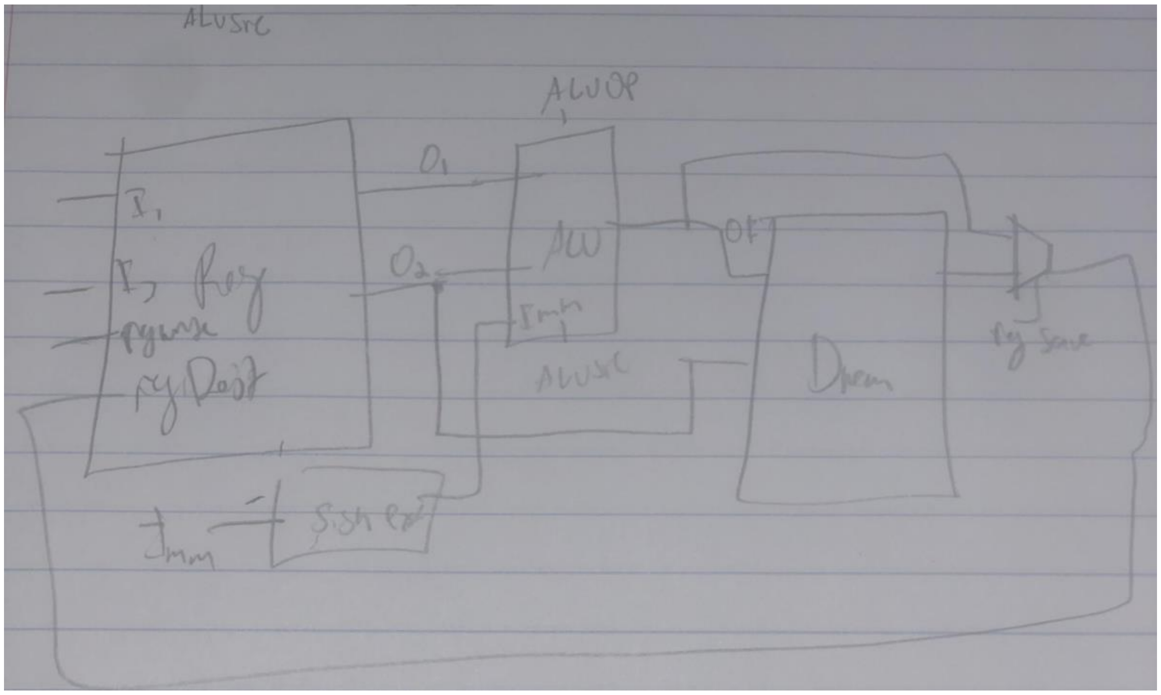
[Part 5 (d)] Waveform.


/extender_16_32/i_A 16'h...
/extender_16_32/i_signed 1
/extender_16_32/o_F 32'h...
0... FFFF
0... FFFFFFFF 0000FFFF

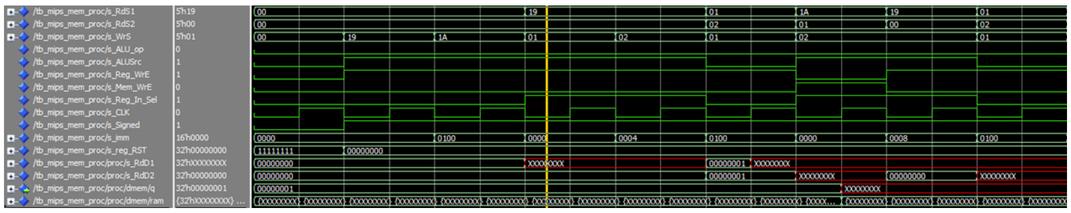
[Part 6 (a)] what control signals will need to be added to the simple processor from part 2? How do these control signals correspond to the ports on the mem.vhd component analyzed in part 3?

Signal to determine if a value should be written to a register file again after it exits dmem

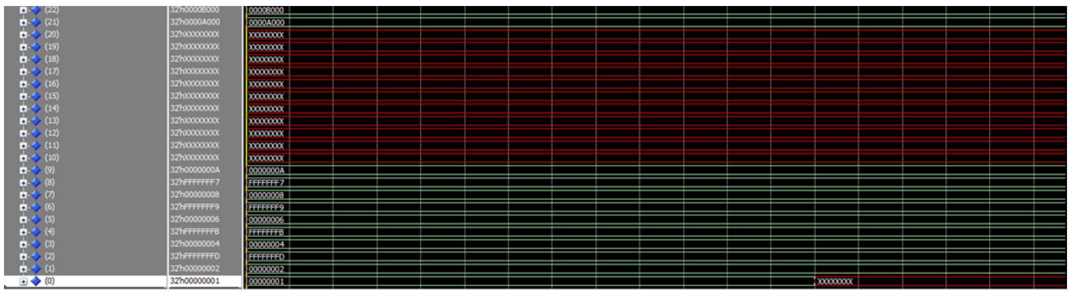
[Part 6 (b)] Draw a schematic of a simplified MIPS processor consisting only of the base components used in part 2, the extender component described in part 4, and the data memory from part 3.



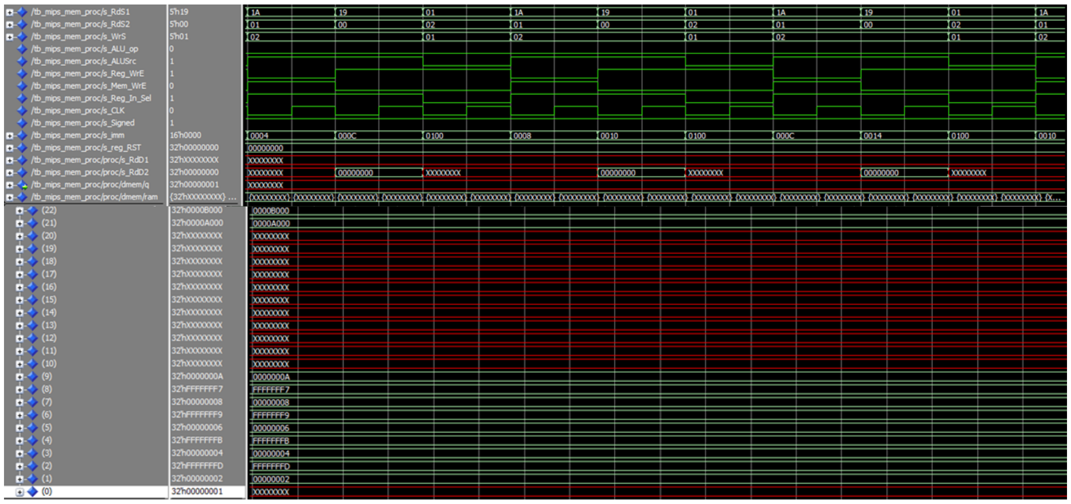
[Part 6 (c)] Waveform.



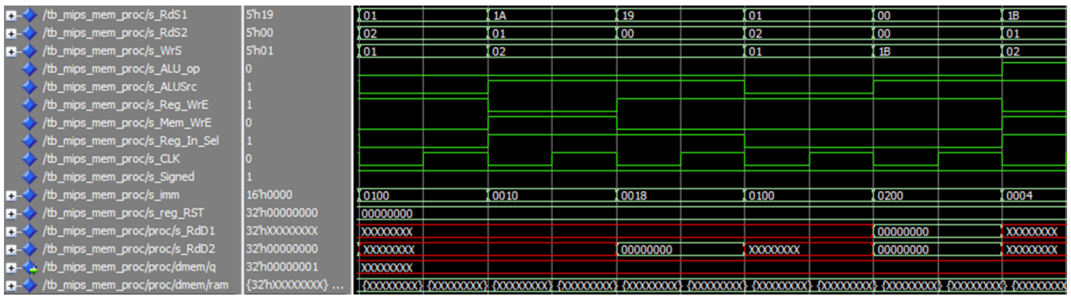
```
/tb_mips_mem_proc/s_RdS1            5'h19
/tb_mips_mem_proc/s_RdS2            5'h00
/tb_mips_mem_proc/s_WrS             5'h01
/tb_mips_mem_proc/s_ALU_op          0
/tb_mips_mem_proc/s_ALUSrc          1
/tb_mips_mem_proc/s_Reg_WrE         1
/tb_mips_mem_proc/s_Mem_WrE         0
/tb_mips_mem_proc/s_Reg_In_Sel      1
/tb_mips_mem_proc/s_CLK             0
/tb_mips_mem_proc/s_Signed          1
/tb_mips_mem_proc/s_imm             16'h0000
/tb_mips_mem_proc/s_reg_RST         32'h00000000
/tb_mips_mem_proc/proc/s_RdD1       32'hXXXXXXXX
/tb_mips_mem_proc/proc/s_RdD2       32'h00000000
/tb_mips_mem_proc/proc/dmem/q       32'h00000001
/tb_mips_mem_proc/proc/dmem/ram     {32'hXXXXXXXX} ...
```



```
(22)   32'h0000B000
(21)   32'h0000A000
(20)   32'hXXXXXXXX
(19)   32'hXXXXXXXX
(18)   32'hXXXXXXXX
(17)   32'hXXXXXXXX
(16)   32'hXXXXXXXX
(15)   32'hXXXXXXXX
(14)   32'hXXXXXXXX
(13)   32'hXXXXXXXX
(12)   32'hXXXXXXXX
(11)   32'hXXXXXXXX
(10)   32'hXXXXXXXX
(9)    32'h0000000A
(8)    32'hFFFFFFF7
(7)    32'h00000008
(6)    32'hFFFFFFF9
(5)    32'h00000006
(4)    32'hFFFFFFFB
(3)    32'h00000004
(2)    32'hFFFFFFFD
(1)    32'h00000002
(0)    32'h00000001
```



```
/tb_mips_mem_proc/s_RdS1            5'h19
/tb_mips_mem_proc/s_RdS2            5'h00
/tb_mips_mem_proc/s_WrS             5'h01
/tb_mips_mem_proc/s_ALU_op          0
/tb_mips_mem_proc/s_ALUSrc          1
/tb_mips_mem_proc/s_Reg_WrE         1
/tb_mips_mem_proc/s_Mem_WrE         0
/tb_mips_mem_proc/s_Reg_In_Sel      1
/tb_mips_mem_proc/s_CLK             0
/tb_mips_mem_proc/s_Signed          1
/tb_mips_mem_proc/s_imm             16'h0000
/tb_mips_mem_proc/s_reg_RST         32'h00000000
/tb_mips_mem_proc/proc/s_RdD1       32'hXXXXXXXX
/tb_mips_mem_proc/proc/s_RdD2       32'h00000000
/tb_mips_mem_proc/proc/dmem/q       32'h00000001
/tb_mips_mem_proc/proc/dmem/ram     {32'hXXXXXXXX} ...
(22)   32'h0000B000
(21)   32'h0000A000
(20)   32'hXXXXXXXX
(19)   32'hXXXXXXXX
(18)   32'hXXXXXXXX
(17)   32'hXXXXXXXX
(16)   32'hXXXXXXXX
(15)   32'hXXXXXXXX
(14)   32'hXXXXXXXX
(13)   32'hXXXXXXXX
(12)   32'hXXXXXXXX
(11)   32'hXXXXXXXX
(10)   32'hXXXXXXXX
(9)    32'h0000000A
(8)    32'hFFFFFFF7
(7)    32'h00000008
(6)    32'hFFFFFFF9
(5)    32'h00000006
(4)    32'hFFFFFFFB
(3)    32'h00000004
(2)    32'hFFFFFFFD
(1)    32'h00000002
(0)    32'h00000001
```



```
/tb_mips_mem_proc/s_RdS1            5'h19
/tb_mips_mem_proc/s_RdS2            5'h00
/tb_mips_mem_proc/s_WrS             5'h01
/tb_mips_mem_proc/s_ALU_op          0
/tb_mips_mem_proc/s_ALUSrc          1
/tb_mips_mem_proc/s_Reg_WrE         1
/tb_mips_mem_proc/s_Mem_WrE         0
/tb_mips_mem_proc/s_Reg_In_Sel      1
/tb_mips_mem_proc/s_CLK             0
/tb_mips_mem_proc/s_Signed          1
/tb_mips_mem_proc/s_imm             16'h0000
/tb_mips_mem_proc/s_reg_RST         32'h00000000
/tb_mips_mem_proc/proc/s_RdD1       32'hXXXXXXXX
/tb_mips_mem_proc/proc/s_RdD2       32'h00000000
/tb_mips_mem_proc/proc/dmem/q       32'h00000001
/tb_mips_mem_proc/proc/dmem/ram     {32'hXXXXXXXX} ...
```