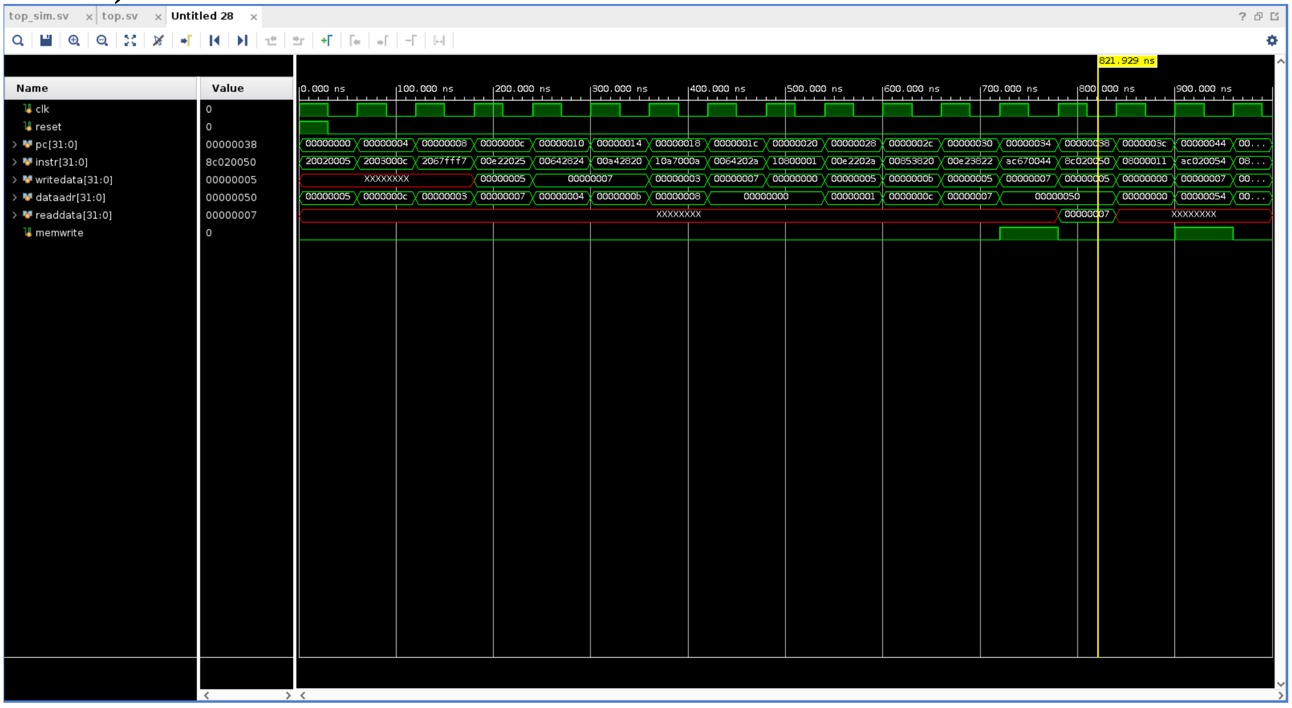
CS224
Lab No: 4
Section No: 5
Erkin Aydın
ID: 22002956

Part 1.a)

Locations(hex)	Machine Instructions(hex)	Assembly Code
0x00000000	0x20020005	addi $v0, $zero, 5
0x00000004	0x2003000c	addi $v1, $zero, 12
0x00000008	0x2067fff7	addi $a3, $v1, -9
0x0000000c	0x00e22025	or $a0, $a3, $v0
0x00000010	0xh00642824	and $a1, $v1, $a0
0x00000014	0xh00a42820	add $a1, $a1, $a0
0x00000018	0xh10a7000a	beq $a1, $a3, 0xa
0x0000001c	0xh0064202a	slt $a0, $v1, $a0
0x00000020	0xh10800001	beq $a0, $zero, 0x1
0x00000024	0xh20050000	addi $a1, $zero, 0
0x00000028	0xh00e2202a	slt $a0, $a3, $v0
0x0000002c	0xh00853820	add $a3, $a0, $a1
0x00000030	0xh00e23822	sub $a3, $a3, $v0
0x00000034	0xhac670044	sw $a3, 0x44($v1)
0x00000038	0xh8c020050	lw $v0, 0x50($zero)
0x0000003c	0xh08000011	j 0x11
0x00000040	0xh20020001	addi $v0, $zero, 0x1
0x00000044	0xhac020054	sw $v0, 0x54($zero)
0x00000048	0xh08000012	j 0x12

Part 1.d)



```
Name             Value
clk              0
reset            0
pc[31:0]         00000038
instr[31:0]      8c020050
writedata[31:0]  00000005
dataadr[31:0]    00000050
readdata[31:0]   00000007
memwrite         0
```

```
0.000 ns  100.000 ns  200.000 ns  300.000 ns  400.000 ns  500.000 ns  600.000 ns  700.000 ns  800.000 ns  900.000 ns
pc:        00000000 00000004 00000008 0000000c 00000010 00000014 00000018 0000001c 00000020 00000028 0000002c 00000030 00000034 00000038 0000003c 00000044 00...
instr:     20020005 2003000c 2067fff7 00e22025 00642824 00a42820 10a7000a 0064202a 10800001 00e2202a 00853820 00e23822 ac670044 8c020050 08000011 ac020054 08...
writedata: XXXXXXXX 00000005 00000007 00000003 00000007 00000000 00000005 0000000b 00000005 00000007 00000005 00000000 00000007 00...
dataadr:   00000005 0000000c 00000003 00000007 00000004 0000000b 00000008 00000000 00000001 0000000c 00000007 00000050 00000000 00000054 00...
readdata:  XXXXXXXX 00000007 XXXXXXXX
```

Part 1.e)
I) writedata is the data that is going to be written back to a register in the register file in R-Type instructions.
ii) It is because first three instructions are “addi” instructions, thus they are I-Type instructions. In these instructions writedata is not used.
iii) It is because, most of the time, we do not read data from the data memory. For example, as it can be seen on the screenshot above, where pc is 0x00000038, we load a word to a register, thus we need to read from the memory, thus there readdata is defined. It other places, we do not do that.
iv) It is the data that will be written back to the register file in R-Type instructions.
v) In instructions that data is written to the data memory, sw is an example.

Part 1.f)

```
module alu(input  logic [31:0] a, b,
           input  logic [2:0]  alucont,
           output logic [31:0] result,
           output logic zero);

   always_comb
      case(alucont)
         3'b010: result = a + b;
         3'b110: result = a - b;
         3'b000: result = a & b;
         3'b001: result = a | b;
         3'b111: result = (a < b) ? 1 : 0;
         3'b011: result = a << b;
         default: result = {32{1'bx}};
      endcase

   assign zero = (result == 0) ? 1'b1 : 1'b0;
endmodule
```

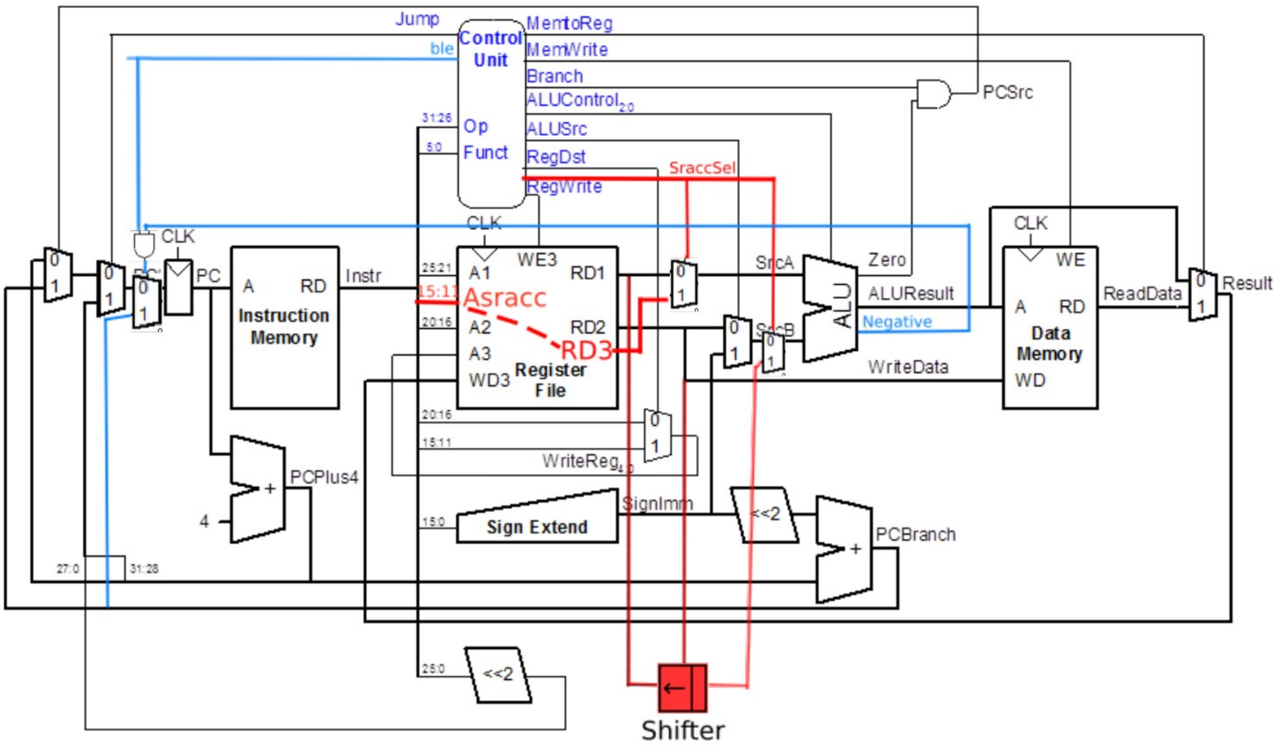
Part 2.a)

```
sracc:
IM[PC]
RD[rd] =  RF[rd] + (RF[rs] >> RF[rt])
PC = PC + 4

ble:
IM[PC]
cond ← RF[rs] – RF[rt]
if(cond less 0)
        PC ←  PC + 4 + (SignExt(imm)x4)
else
        PC ← PC + 4
```

Part 2.b)

Due to my inability, newly added multiplexers, and an and gate, are not colored. Since these new components are only connected to new signals, this wouldn't create confusion. The red color is for sracc, and the blue color is for ble for section 5. For sracc, two 2:1 multiplexers and one shifter, and a new signal "sraccSel" to the control unit, is added. For ble of section 5, an and gate and a 2:1 multiplexer added for PC selection, a new "ble" signal added to control unit, and a new "negative" signal added to the ALU. This new signal will be 1 when RF[rs] is less than RF[rt], and 0 when it is not.



Part 2.c)

Instruction	Opcode	RegWrite	RegDst	ALU Src	Branch	MemWrite	MemToReg	ALUOp	Jump	sraccSel	ble
R-type	000000	1	1	0	0	0	0	10	0	0	0
lw	100011	1	0	1	0	0	1	00	0	0	0
sw	101011	0	x	1	0	1	x	00	0	0	0
beq	000100	0	x	0	1	0	x	01	0	0	0
addi	001000	1	0	1	0	0	0	00	0	0	0
j	000010	0	x	x	x	0	x	xx	1	0	0
sracc	101111	1	1	0	0	0	0	10	0	1	0
ble	111111	0	0	x	0	0	x	11	0	0	1