Alex Oswald

EECE 2323

Due: 20 Nov 2020

Pre-Lab Assignment 8

**§1 Design a PC logic with branching capability**

1 `timescale 1ns / 1ps

2

3 **module** pc\_logic(

4 **input** clk, rst,

5 **input** take\_branch,

6 **input** [7:0] offset,

7 **output** reg [7:0] pc);

8

9 reg condition;

10

11 **always**@ (**posedge** clk)

12 **begin**

13 condition <= (pc + offset > 8'd127) ^ (pc + offset < -8'd128);

14 pc <= ((rst | condition) ? 8'd0 : (pc + (take\_branch ? 1 : offset)));

15

16 */\* MUXES PREFERRED*

17  *// reset or too high*

18  *if (rst | condition) begin*

19  *pc <= 8'd0;*

20  *end*

21  *// regulaar or branch*

22  *if (~condition) begin*

23  *pc <= pc + (take\_branch ? 1 : offset );*

24  *end*

25  *\*/*

26

27 */\* IF-STATEMENT HEAVY*

28  *// reset*

29  *if (rst) begin*

30  *pc <= 0;*

31  *end*

32  *// branching*

33  *else if (take\_branch) begin*

34  *pc <= pc + offset;*

35  *end*

36  *// regular*

37  *else begin*

38  *pc <= pc + 1;*

39  *end*

40  *\*/*

41 **end**

42

43 **endmodule**

**§2 A Signed Multiplier in Assembly**

1 ############################# PRELAB 7 #############################

2 # EECE 2323 – Lab for EECE 2322 #

3 # By: Alex Oswald #

4 # #

5 # Description: #

6 # Program that multiplies two numbers. #

7 #####################################################################

8

9 clr **$**0 # count reg

10 clr **$**1 # num1

11 clr **$**2 # num2

12 clr **$**3 # product (working sum)

13

14 addi **$**0, **$**0, 0x00 # counter = 0

15 addi **$**1, **$**1, 0x07 # num1 = 7

16 addi **$**2, **$**2, 0xFB # num2 = -5

17

18 label:

19 add **$**3, **$**3, **$**2 # add num2 for every occurence of num1

20 addi **$**0, **$**0, 0x01

21 bne **$**0, **$**1, label # go back if (counter != num1)

22

23 sw **$**3, 0x10(**$**0)

24 lw **$**0, 0x10(**$**0)

**§3 Generate the Machine Codes using the Assembler**

1 memory\_initialization\_radix=16;

2 memory\_initialization\_vector=d000,d140,d280,d3c0,3000,3507,3afb,2ec0,3001,c1fe,1310,0010;