Score:_____ Name:_____ 8AM_____ 9AM_____

ECE 3055 Test 1

Wednesday, February 13

Open Book & Notes - Copy All Short Answers (1-4) to first page!

1. Part 1. 16 ns. Part 2. 16 ns.

17pts.

Part 3. 5 ns. Part 4. 9 ns.

2. Complexity of Control Unit	Total Hardware Used	CPI	Clock Period
1 Single Cycle	2 Single Cycle	1 Single Cycle	3 Single Cycle
8pts. 3 Multi Cycle	1 Multi Cycle	3 Multi Cycle	1 Multi Cycle or 2
2 Pipelined	3 Pipelined	2 Pipelined	2 Pipelined or 1

30pts.

3. Instruction = 8C02 0001 Read Data 1 = 00000000

Read Data 2 = 00000002 ALU Result = 00000003

(Data Memory) Read Data = 0 or X Write Register (Address) = 4

Write Data (input at register file after mux) = 00000003

010

ALU control input = 2 RegWrite = 1 ALUSrc = 0

4. **Part I:** Total number of NOPs required 16 **Part II:** A total of 11 clock cycles is required for execution.

10pts. 15pts.

1. (*17 points*) Compare the execution time of the program segment below on the three MIPS hardware models studied in class. Assume the branch is taken.

```
lw    $3,200($0)   5
sw    $2,100($0)   4
or    $4,$3,$7     4
beq   $8,$7,Label1 3
                   16
```

Part 1: The single clock cycle MIPS with a clock frequency of 250 Mhz. would take

16 ns. to execute the program. 4x4ns=16ns

Part 2: The multi clock cycle model of the MIPS with a clock frequency of 1Ghz. would take 16 ns. to execute the program. 16x1ns=16ns.

Part 3: The pipelined MIPS model with data forwarding and hazard detection and a clock frequency of 1Ghz, would require 5 ns. to execute the program. 5x1ns. →1 beq stall! Assume a register will write and read correctly during the same clock cycle and the branch decision is made in decode. Include any stalls or flushes, but do not include the time required to initially fill the pipeline.

Part 4: If the pipelined MIPS model started this program immediately after powering up, it would take 9 ns. before the program finished execution. 5+(4-1)+(stall) This includes the time required to fill and flush the pipeline.

2. *(8 Points)* Rank the three MIPS models 1,2,3 in the following categories.1 is least or smallest and 3 is most or largest. For control unit complexity only consider what is included in the one hardware block labeled "control unit" in the text. (don't include data pipeline registers and forwarding hardware)

Complexity of Control Unit	Total Hardware Used	CPI	Clock Period
1 Single Cycle	2 Single Cycle	1 Single Cycle	3 Single Cycle
3 Multi Cycle	1 Multi Cycle	3 Multi Cycle	1 Multi Cycle
2 Pipelined	3 Pipelined	2 Pipelined	2 Pipelined or 2 or 1

3. (*30 points*) The following sequence of MIPS instructions is clocked into the pipeline shown on page 472-476. Examine this figure carefully to see exactly where each signal is located (i.e. before or after pipeline registers). After Clock cycle 5, Indicate the resulting register values in the spaces provided below. All numbers are in hex. Assume all data memory locations contain the word address of the location. Assume that each register contains a value equal to the register number prior to execution of this code.

WB
DM
E
D
F

add	**$4,$3,$0**	
andi	**$6,$7,8**	
add	**$2,$5,$6**	
lw	**$2,1**	→ op=35
sub	**$3,$1,$4**	

↓

1000 | 11 | 00000 | 00010 | 0_x 0001

8 C 0 2 0001

Instruction = 8C020001

Read Data 1 = 00000006

Read Data 2 = 00000002

ALU Result = 00000008

(Data Memory) Read Data = 00000006

Write Register (Address) = 4

Write Data (input at register file after mux) = 0000 0003

p355 add=010

ALU control input = 2

RegWrite = 1

ALUSrc = 0

4. (*25 points*) The program below is executed on the 5 stage pipelined MIPS described in chapter 6. Answer the following questions about this program.

```
loop:   sw      $2,100($0)
        sub     $2,$5,$3
        lw      $7,200($2)
        and     $8,$3,$4
        andi    $6,$7,8
        beq     $6,$8,then
        add     $5,$5,$8
then:   or      $8,$3,$8
        sw      $5,100($6)
        beq     $8,$0,loop
```

Part I (*10 points*) Assume the control unit **does not have** any hazard detection, forwarding, a new branch compare circuit, or automatic branch flushing. That register file will not write and then read a new register value in one clock cycle. Rewrite the code sequence by adding the minimum number of NOP instructions to eliminate all potential data and branch hazards – do not change the order of the instructions. Assume other non-NOP instructions follow the last branch in the original code sequence above.

Total number of NOPs required 16

```
sw          beq
sub         nop
nop         nop
nop         nop
nop         add
lw          or
and         sw
nop         nop
nop         nop
andi        beq
nop         nop
nop         nop
nop
```

Part II (*15 points*) Assume the control unit is improved by adding the hazard and forwarding unit as outlined in the text, adding a branch compare unit to the decode stage, and the register file writes then reads a new value in a single clock cycle. Determine the number of clock cycles required to complete the first loop execution (i.e. executes code in loop and branches back to top of loop and is just ready to fetch sw again) of the original code sequence. Assume the inner branch is taken. → skips add

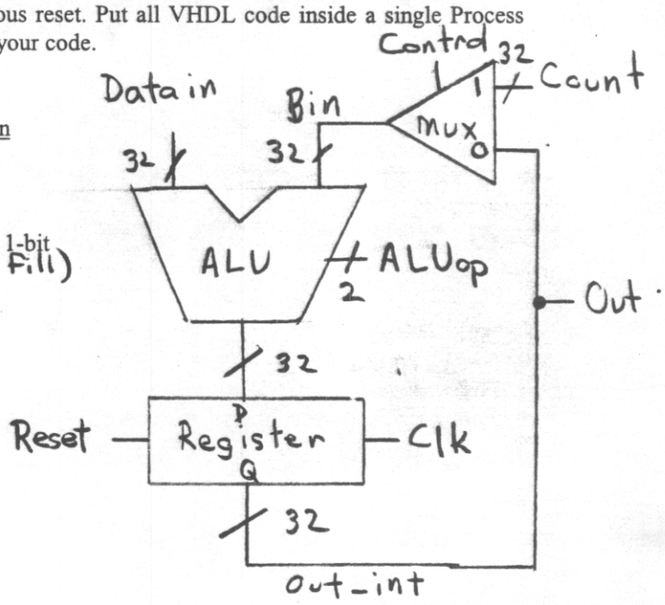
If there were no hazards or branch flushing, the original program would require 9 clock cycles for execution.

beq+beq stall no lw stall!

But the program stalls and/or flushes the pipeline 2 clock cycles so **a total of** 11 **clock cycles** is required for execution (do not include time to fill pipeline).

5. *(20 points)* Write a complete VHDL synthesis model for the digital hardware shown in the block diagram. Use a positive edge clock with a synchronous reset. Put all VHDL code inside a single Process block. The signal, Bin may or may not be required in your code.

ALUop	Operation
00	add
01	subtract
02	OR
03	Shift left 1-bit (zero Fill)



library 2
entity 2
arch
  bin mux 2
  out 2
  Process
    ALU Case 3
    ALU Ops 3
    Register 3
    Reset 3

Extra registers -2 each
Not in Process -5

```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
USE IEEE.STD_LOGIC_ARITH.ALL;
USE IEEE.STD_LOGIC_SIGNED.ALL;
ENTITY  test1c IS
 PORT( Datain,Count          : IN  STD_LOGIC_VECTOR( 31 DOWNTO 0 );
       ALU_Op                : IN  STD_LOGIC_VECTOR( 1 DOWNTO 0 );
       Control,clk,reset     : IN  STD_LOGIC;
       OUT                   : OUT STD_LOGIC_VECTOR( 31 DOWNTO 0 ));
END test1c;
ARCHITECTURE behavior OF test1c IS
SIGNAL  Bin, OUT_int   : STD_LOGIC_VECTOR( 31 DOWNTO 0 );
BEGIN
 Bin <= OUT_int WHEN Control='0' ELSE Count;
 OUT <= OUT_int;
PROCESS
 BEGIN
  WAIT UNTIL clk'EVENT AND clk='1';
   IF reset='1' THEN Outp_int <= "00000000000000000000000000000000";
   ELSE
     CASE ALU_op IS
      WHEN "00"  => Out_int <= Datain + Bin;
      WHEN "01"  => Out_int <= Datain - Bin;
      WHEN "10"  => Out_int <= Datain OR Bin;
      WHEN "11"  => Out_int <= Datain(30 Downto 0) & "0";
      WHEN OTHERS => Out_int <= "00000000000000000000000000000000";
     END CASE;
   END IF;
 END PROCESS;
END behavior;
```