Score:_____ Name:_____

ECE 3055 Test 1

1. **(20 *points*)** Compare the execution time of the program below on the three MIPS hardware models studied in class.

sw	**$2,100($0)**	4
lw	**$3,200($0)**	5
or	**$4,$5,$7**	4
add	**$8,$7,$5**	4
		17 For Mult. cycle

Part 1: A single clock cycle MIPS with a clock cycle time of 40ns would take

160 ns to execute the program. 4 x 40

Part 2: The multi clock cycle model of the MIPS with a clock cycle time of 30ns would take

510 ns to execute the program.

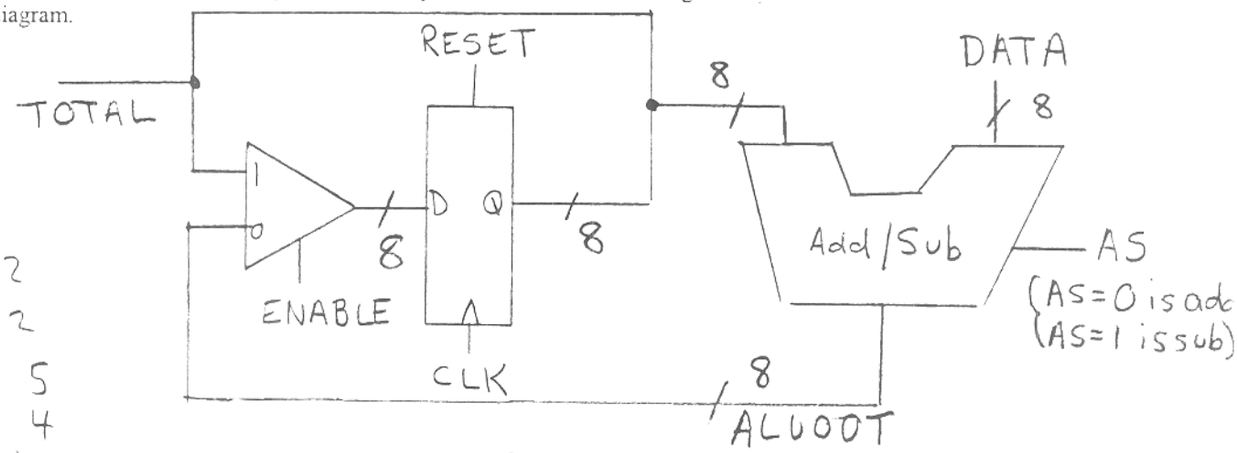
Part 4: The pipelined MIPS model with data forwarding and hazard detection and a clock cycle time of 10ns would require 40 ns to execute the program. Assume a register will write and read correctly during the same clock cycle. Do not include the time required to fill and flush the pipeline.

Part 5: If the pipelined MIPS model started this program immediately after powering up, it would take 80 ns. This includes the time required to fill and flush the pipeline.

2. ***(5 Points)*** The Pentium 4 has a faster clock cycle time than the PIII. What main change in the architecture (i.e. not just a faster transistor technology) increased the clock cycle time. What is the potential performance loss in this new architecture that must be addressed?

deeper pipelines (20 vs. 10), Needs better branch prediction or it will have more stalls.

3. *(20 points)* Write a complete VHDL synthesis model for the digital hardware shown in the block diagram.



lib 2
ent 2
FF 5
mux 4
add/sub 5
signals 2

Inputs DATA, AS, CLK, RESET, ENABLE
Outputs TOTAL

```
library IEEE;
use  IEEE.STD_LOGIC_1164.all;
use  IEEE.STD_LOGIC_ARITH.all;
use  IEEE.STD_LOGIC_UNSIGNED.all;
entity Problem3 is
   port(    AS, Clk, Reset, Enable: in std_logic;
            Data: in std_logic_vector(7 downto 0);
            Total : out std_logic_vector(7 downto 0));
end Problem3;
architecture behavior of Problem3 is
signal total_int, ALUout: std_logic_vector(7 downto 0);
begin
    Total <= total_int;
    ALUout <= Total_int + Data WHEN AS='0' ELSE Total_int - Data;
process
    begin
    WAIT UNTIL (CLK'EVENT and CLK='1');
    IF reset = '1' THEN
        total_int <= "00000000";
    ELSE
        IF (Enable ='1') THEN
            total_int <= total_int;
        ELSE
            total_int <= ALUout;
        END IF;
    END IF;
end process;
end behavior;
```

4. (*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. Memory location 1 contains 0x55555555. Assume that each register contains a value equal to the register number prior to execution of this code.

```
add   $2,$5,$6
lw    $2,1
sub   $3,$1,$4
andi  $6,$7,8
add   $4,$3,$0
```

Instruction = 30E60008

Read Data 1 = 00000007

Read Data 2 = 00000006

ALU Result = FFFFFFFD

(Data Memory) Read Data = 55555555

Write Register (Address) = 2

Write Data (input at register file after mux) = 0000000B

Memread = 1

RegWrite = 1

Memtoreg = 0

5. (*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)
       lw    $5,200($2)
       sub   $6,$5,$2
       and   $8,$3,$6
       add   $8,$5,$8
       or    $8,$3,$8
       sw    $5,100($6)
       beq   $6,$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. 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 branch in the original code sequence above.

```
sw $2,100($0)
lw $5,200($2)
nop
nop
nop
sub $6,$5,$2
nop
nop
nop
and $8,$3,$6
nop
nop
nop
add $8,$5,$8
nop
nop
nop
or $8,$3,$8
sw $5,100($6)
beq $6,$0,loop
nop
nop3 no new branch compare!
nop
```

Part II (*15 points*) Assume the control unit is improved by adding the hazard and forwarding unit as outlined in the text, and adding a branch compare unit to the decode stage. 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. (8 without Fill)

If there were no hazards or branch flushing, the original program would require 12 clock cycles for execution. 5+7

But the program stalls and/or flushes the pipeline 2 clock cycles so a total of 14 clock cycles is required for execution.