UNIVERSITY OF MANITOBA

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING
PH.D. CANDIDACY EXAMINATION

COURSE NA
START TIME TBA
INSTRUCTOR NA

DATE December, 2013
END TIME TBA
SEATS: NA

Table 1: FOR USE OF EXAMINER

Question	Mark
1	
2	
3	
4	
5	
6	
7	
8	
TOTAL	
GRADE	

PRINT LAST NAME   PRINT FIRST NAME

STUDENT NUMBER   STUDENT SIGNATURE

EXAMINATION CENTRE   SEAT NUMBER

INSTRUCTIONS TO STUDENTS:

1. **Answer all questions in the space provided.**
2. **The value of the question is shown to the left.**
3. **Students may use the given copy of the instruction set.**
4. **Students may bring into the exam and use their copy of the lecture notes for ECE 3610 and ECE 3730.**
5. **Electronic devices are not allowed; calculators, cell phone, PDAs, etc. are not allowed.**

1. You are given: a data processor as shown in Fig. 1; the ALU function table given in Table 2; and the MUX control table given in Table 3. This question asks you to design the micro-operations for the macro-instruction: `SUB` $N_1, N_2 (N_2 - N_1)$ The binary encoding of this instruction is: `0011` N_1N_2, where `0011` is the opcode, N_1 is the 4-bit operand #1, and N_2 is the 4-bit operand #2. The result of this operation should be stored in Register B. Note: a design goal is to implement this macro-instruction with the least number of micro-operations, and your solution will be graded accordingly. Note: design all micro-operations, including fetch. Note: the program counter can be incremented by one by making PC_INC=1 and providing the clock edge, as indicated in Fig. 1.

16 Marks

[1.1] Write down each micro-operation of your design in the following table. Describe each micro-operation in clear English statements or register transfer language, and choose a short form or abbreviation to name the micro-operation.

μ-Operation Abbreviation	**μ-Operation Description**

48 Marks

[1.2] Specify the control vector for each one of your micro-operations, and fill out the following table.

μ-Operation Abbreviation	**PC_INC**	**M_OE**	**M_R/$\overline{W}$**	**A_R/$\overline{W}$**	**B_R/$\overline{W}$**	**MAR_R/$\overline{W}$**	**OCR_R/$\overline{W}$**	**S2**	**S1**	**S0**	**MS1**	**MS0**

Table 2: ALU function table for Q1.

S2	S1	S0	ALU FUNCTION
0	0	1	PASS A
0	1	0	ASIDE + BSIDE
0	1	1	ASIDE - BSIDE
1	0	0	NOT ASIDE
1	0	1	NOT BSIDE
1	1	0	ZZZZ (High Impedance)
1	1	1	1111

Table 3: MUX Control.

S1	S0	Input	Output
0	0	don't care	Hi-Z
0	1	RBUS	DBUS
1	0	DBUS	RBUS

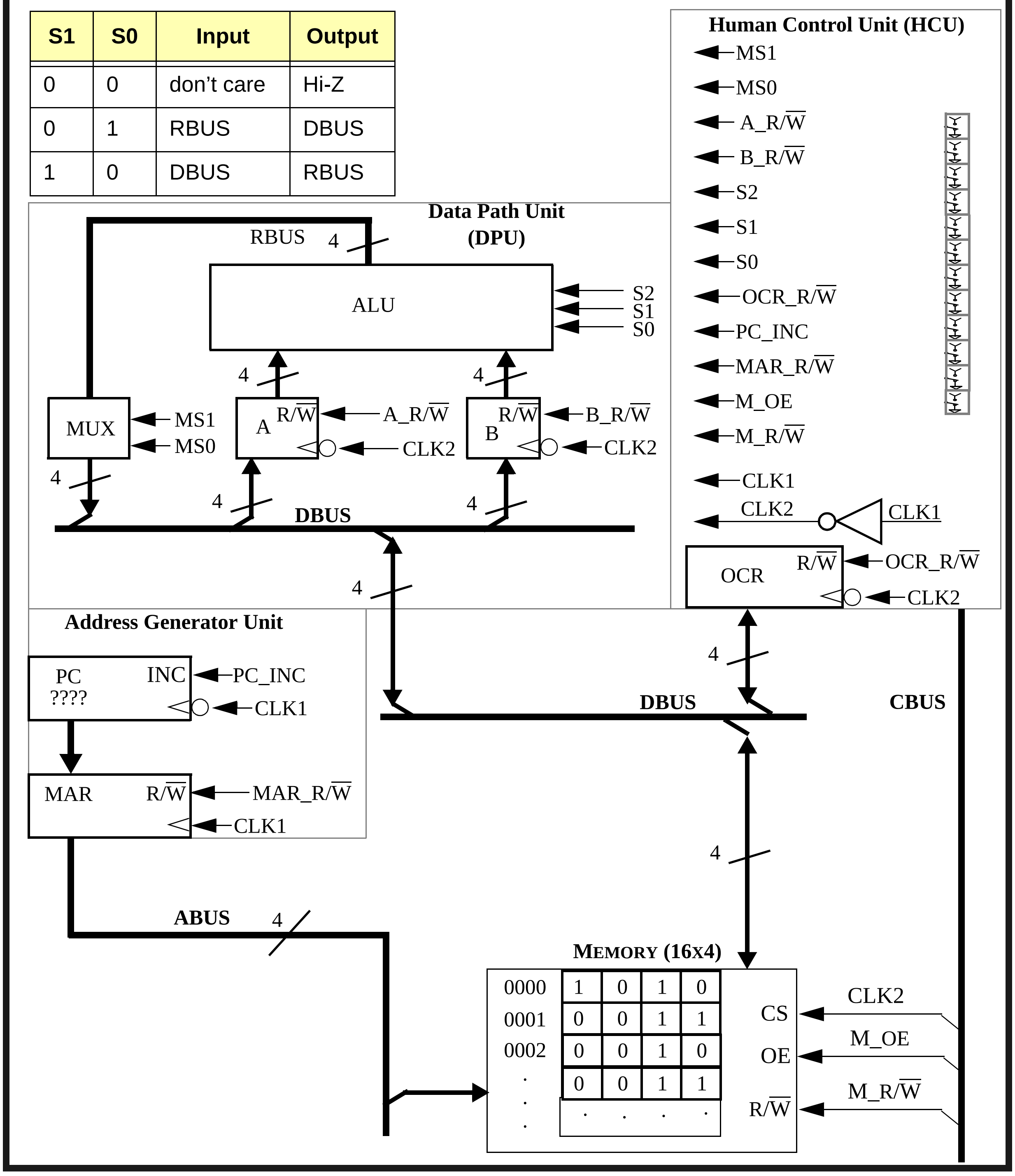

Fig. 1. A tiny data processor.

1 Mark

[1.3] An example instruction SUB 0010,0011 has been placed in the memory of Fig. 1. By analyzing the contents of the memory, determine the initial value of the PC so that the first instruction the machine executes is "SUB 0010,0011" and write that value down here: _____________

20 Marks

2. Write Verilog code to describe the following circuit diagram. Use the opposite page to develop your solution. Write your final solution in the space provided on this page.

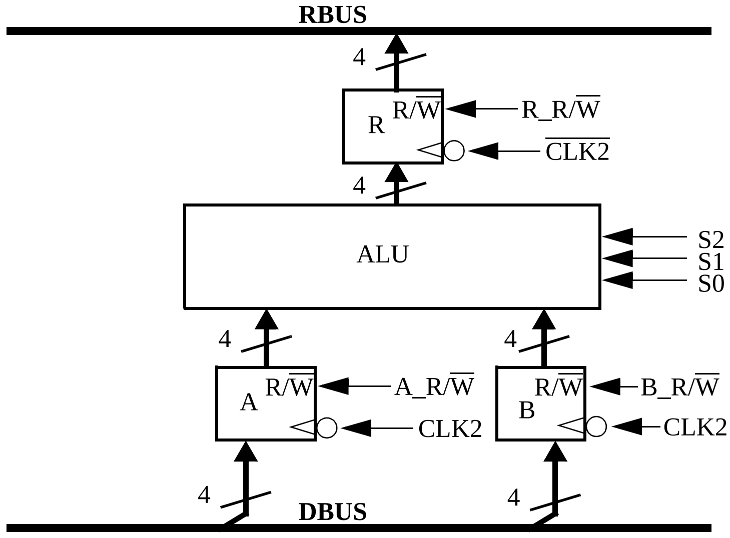

Fig. 2. Circuit diagram for Q2.

Example 4-bit Register:

```
module reg4 (D, Wn, Clock, Q);
    input [3:0] D;
    input Clock, Wn;
    output reg [3:0] Q;

    always @ (posedge Clock)
        if (Wn==0)
          Q <= D;
endmodule
```

Example ALU:

```
module alu4 (ASIDE, BSIDE, S,  F);
      input [3:0] ASIDE, BSIDE, S;
      output reg [3:0] F;

      always @ (ASIDE, BSIDE, S)
            case (S)
               0: F = 4'b0000;
               1: F = BSIDE - ASIDE;
               2: F = ASIDE - BSIDE;
               3: F = ASIDE + BSIDE;
               4: F = ASIDE ^ BSIDE;
               5: F = ASIDE | BSIDE;
               6: F = ASIDE & BSIDE;
               7: F = 4'b1111;
               default: F = 4'bzzzz;
            endcase

endmodule
```

12 Marks

3. Consider using the 4-bit ALU with most significant carry detector and indicator, and an overflow detector and indicator. This ALU is configured to perform an addition operation (ASIDE + BSIDE). Let the numbers input to the ASIDE be represented by $A_3A_2A_1A_0$ and BSIDE be represented by $B_3B_2B_1B_0$. Let the result be represented by $F_3F_2F_1F_0$.	Place answer in this column
3.1 If $A_3 = 0$ and $B_3 = 0$, then what is the state of the carry indicator?	
3.2 If $A_3 = 0$ and $B_3 = 1$, then what is the state of the carry indicator?	
3.3 If $A_3 = 1$ and $B_3 = 0$, then what is the state of the carry indicator?	
3.4 If $A_3 = 1$ and $B_3 = 1$, then what is the state of the carry indicator?	
3.5 If $A_3 = 0$, $B_3 = 0$, and $F_3 = 0$, then what is the state of the overflow indicator?	
3.6 If $A_3 = 0$, $B_3 = 0$, and $F_3 = 1$, then what is the state of the overflow indicator?	
3.7 If $A_3 = 0$, $B_3 = 1$, and $F_3 = 0$, then what is the state of the overflow indicator?	
3.8 If $A_3 = 0$, $B_3 = 1$, and $F_3 = 1$, then what is the state of the overflow indicator?	
3.9 If $A_3 = 1$, $B_3 = 0$, and $F_3 = 0$, then what is the state of the overflow indicator?	
3.10 If $A_3 = 1$, $B_3 = 0$, and $F_3 = 1$, then what is the state of the overflow indicator?	
3.11 If $A_3 = 1$, $B_3 = 1$, and $F_3 = 0$, then what is the state of the overflow indicator?	
3.12 If $A_3 = 1$, $B_3 = 1$, and $F_3 = 1$, then what is the state of the overflow indicator?	

30 Marks

4. Overflow circuit design:

4.1 Give an English language definition of overflow for the addition of two 4-bit numbers (A + B).

4.2 Give an English language definition of overflow for the subtraction of two 4-bit numbers (A - B).

4.3 Design a digital circuit to implement an overflow indicator for the circuit shown in Fig. 2. Store the overflow state in a flip-flop called V.

5. Design an assembly language program that replaces each byte in a 256 Byte block of data with the *modulo 256 sum* of the previous byte and the next byte location indexed by the value of the byte. The block of data starts at $1000. Do not consider the first or last byte in the block. Table 4 shows an example for a 4-byte block starting at $1000.

Table 4: Example for Q5.

Address	Before	After
1000	FF	FF
1001	43	01
1002	02	66
1003	23	23

30 Marks

[5.1] Draw a flow chart that models your solution. Note: if your flow chart is incorrect, then your assembly language program will not be marked.

20 Marks

5.2 Translate your flow chart to 68HC11 assembly language.

Table 5: Assembly Language for Q 5.

Label	Instruction	Comments

6. Write an assembly language program that estimates the period of a square wave.
Known information: the square wave will have a period between 5s and 10s.
Given: A square wave generator device outputs a periodic square wave voltage signal on a wire, which is connected to Flip-Flop-0 (FF-0) of memory location $8000. When the square wave is high, 5V is written to FF-0 of location $8000. When the square wave is low, 0V is written to the FF-0 of location $8000.
Given: An interrupt routine sets bit-1 of memory location $8000 each second.

20 Marks

6.1 Draw a flow chart that models your solution. Note: if your flow chart is incorrect, then your assembly language program will not be marked.

10 Marks

6.2 Translate your flow chart to 68HC11 assembly language.

Table 6: Assembly Language for Q 6.

Label	Instruction	Comments

7. Design an General Purpose Input Output (GPIO) module for a microcontroller. The GPIO module should have one 8-bit port, called Port A. The module should have the ability to either write to or read from any of the 8 port pins. For example, after the microcontroller writes a ‘1’ to Port A Bit-0, the port pin that is connected to Port-A Bit-0 should output +5 V, and remain that way, until the microcontroller changes the logic value of Port A Bit-0. Furthermore, if an external device applies +5 V to pin-7 of Port A, then the microcontroller should read logic ‘1’ on Port A Bit-7, for as long as the external device maintains +5 V on that pin. The module should have the ability to interrupt the processor whenever there is an active edge on any of the 8 port pins.

20 Marks

7.1 State the number of data/control/status registers in your module, and explain the purpose of each of the registers. State your assumptions.

20 Marks

7.2 Draw a circuit that shows how your design establishes a port pin (e.g., Port A Bit-0) as either an input or output pin.