UNIVERSITY OF DUBLIN

TRINITY COLLEGE

Faculty of Engineering, Mathematics and Science

School of Computer Science and Statistics

B.A.(Mod.) Computer Science **Trinity Term 2010**
Senior Freshman Examination

CS2021 – Computer Architecture I
CS2022 – Computer Architecture II

Friday 7th May, 2010 **Regent House** **14:00-17:00**

Dr. Ross Brennan, Dr. Michael Manzke

Answer **FIVE** questions, at least **TWO** from each section.

Use separate answer books for section A and section B.

The use of non-programmable calculators is permitted.

Please note the make and model of your calculator on your answer book.

SECTION A

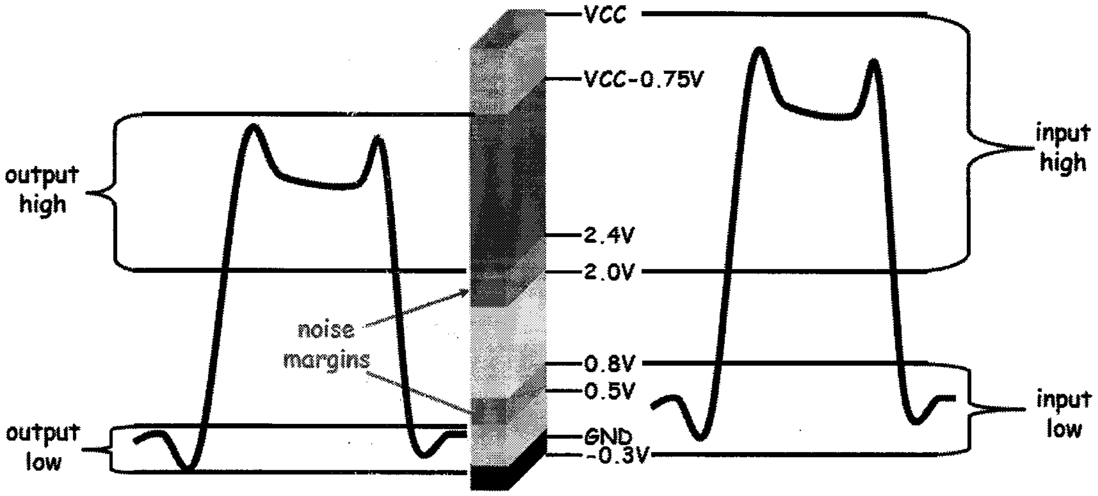
(CS2021 – Dr. Ross Brennan)

1.

a) Describe the basic operation and purpose of a Logic State Analyser and an Oscilloscope. Discuss how these instruments aid in the debugging and verification of the CS2021 microprocessor project boards.

[6 marks]

b) The diagram below outlines the various switching levels for input and output signalling as defined by the TTL standard (VCC = 5.0V). Briefly describe what is meant by the terms: “input high”, “input low”, “output high”, “output low”, “noise margins” and “signal rise time”.



[6 marks]

c) Describe each of the following types of output pin: “Tri-State”, “Fully-Driven” and “Partially-Driven”. Explain why Tri-State and Partially-Driven output pins should not be connected together. Discuss how a “wired-and” and a “wired-or” circuit could be created using pull-up and pull-down resistors in conjunction with pull-up and pull-down pin types.

[8 marks]

2.

a) Discuss the purpose of bus-cycle wait-states and briefly describe the effect they have on MC68008 memory read and write cycles.

[4 marks]

b) Give a detailed description of the changes that would be required to the CS2021 microprocessor project hardware and VHDL code in the CPLD in order to implement configurable wait-states for the RAM devices. The CPU should be capable of reading and writing configuration registers in the CPLD that allow it to modify the timing of the wait-states for each individual RAM device. You can assume that the ROM device and the CPLD itself do not require any wait-states and that the RAM devices have already been memory mapped into the system using the following active-high decoder signals: CSRAM1int and CSRAM2int.

[10 marks]

c) Discuss how the CPU could automatically determine and set the correct wait-state values for the RAM devices assuming the changes to the CS2021 microprocessor project system described in part (b) have been implemented.

[6 marks]

3.

a) What are emulator (line-A and line-F) instructions? Briefly describe how they can be used to implement new instructions.

[4 marks]

b) Describe the sequence of events that occur when the MC68008 detects an illegal instruction. How does this sequence differ when bits [15:12] of the instruction are equal to “1010” or “1111”.

[6 marks]

c) The following segment of code is taken from the monitor for the CS2021 microprocessor project. Give a detailed description of the purpose and functionality of the code and discuss any limitations that it might impose.

```
**
** Line 1111 Exception Handler
**
HL1111      CLR.L       -(SP)                    *  1
            MOVEM.L     D0/A0,-(SP)              *  2
            MOVE.L      14(SP),A0                *  3
            ADDQ.L      #2,14(SP)                *  4
            MOVE.W      (A0),D0                  *  5
            ANDI.L      #$0F,D0                  *  6
            ADD.L       D0,D0                    *  7
            MOVE.L      D0,A0                    *  8
            MOVE.W      CALLT(A0),10(SP)         *  9
            MOVEM.L     (SP)+,D0/A0              * 10
            RTS                                  * 11
```

[10 marks]

4.

a) Describe how the MC68008 can determine if a memory read or write operation is targeting a M6800 device or not.

[4 marks]

b) Design a detailed memory map that will allow the following components to be interconnected in a microprocessor system, through a shared bus.

- 1 x MC68008 CPU [20 address pins, reset @ $00000]
- 2 x 8k EEPROM
- 1 x 4k RAM
- 2 x 2k RAM
- 2 x 4-byte R6551 ACIAs

Your solution should comply with the design rules specified as part of the microprocessor project and should specify the first and last address of every device as well as the address-bus lines that are connected to each device. The first RAM device should be located at a base address of $08000.

[8 marks]

c) Given the entity port declaration on the following page, implement your decoder design from part (b), using VHDL code.

Question continued on the next page.

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;

entity decoder is
    port(
        CLK,E        : in  std_logic;
        FC0,FC1,FC2  : in  std_logic;
        A19,A18,A17  : in  std_logic;
        A16,A15,A14  : in  std_logic;
        A13,A12,A11  : in  std_logic;
        ASn,DSn      : in  std_logic;
        WEn,OEn      : in  std_logic;
        CSROM1n      : out std_logic;
        CSROM2n      : out std_logic;
        CSRAM1n      : out std_logic;
        CSRAM2n      : out std_logic;
        CSRAM3n      : out std_logic;
        CSIO1        : out std_logic;
        CSIO2        : out std_logic;
        BERRn        : out std_logic;
        DTACKn       : out std_logic;
        VPAn         : out std_logic;
        VMAn         : out std_logic
    );
end decoder;
```

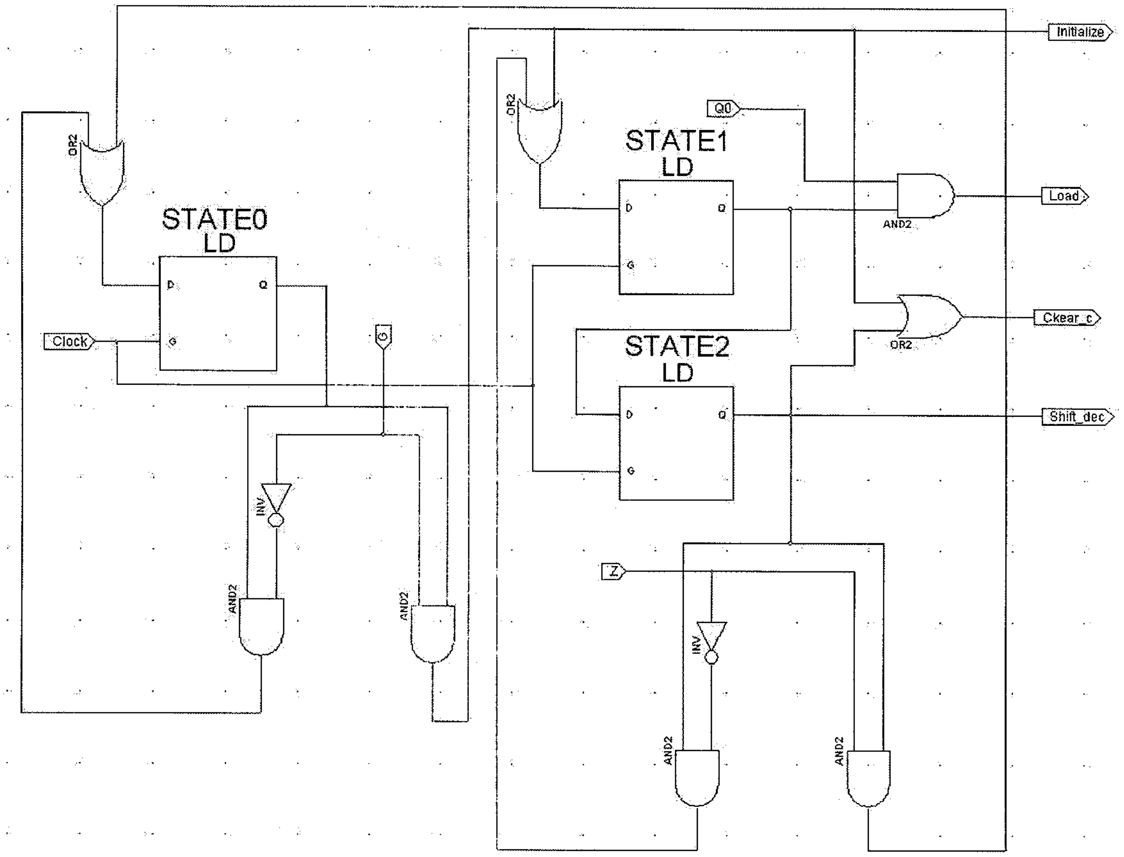
[8 marks]

SECTION B
(CS2022 – Dr. Michael Manzke)

5.

a) Provide an *Algorithmic State Machine* (ASM) chart for the following schematic.

[8 marks]



b) Discuss the following two alternative designs: *Sequence Register and Decoder* method and *One Flip-Flop per State* method. Draw a schematic that transforms the schematic from *Question 1.a)* into the alternative design.

[4 marks]

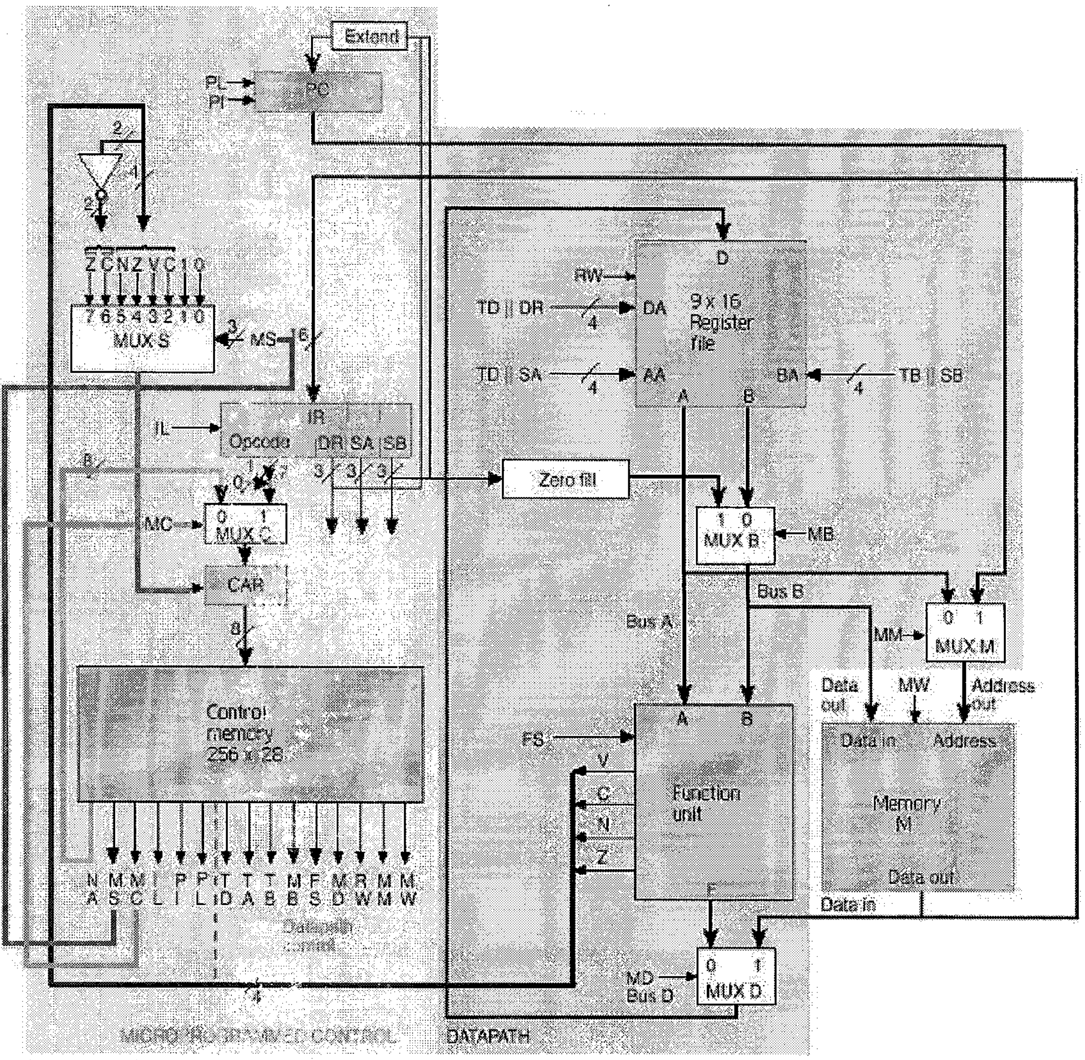
c) Write VHDL code that implements the part of the *Algorithmic State Machine* (ASM) that is shown in *Question 1.a)*.

[8 marks]

6.

a) Explain in detail the operations that take place when the following multiple-cycle microprogrammed instruction set processor executes machine instructions.

[10 marks]



Expand on your discussion from Question 6. a) by explaining how the following instruction is executed:

b) Binary multiplication of two registers in the register file (8 least significant bits only). Result is written into a third register in the register file.

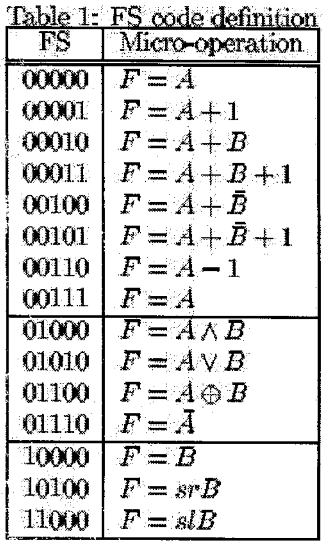
[10 marks]

7.

a) Provide a detailed schematic for a *Function Unit* that implements the following *micro-operations*:

[10 marks]

Table 1: FS code definition



FS	Micro-operation
00000	$F = A$
00001	$F = A + 1$
00010	$F = A + B$
00011	$F = A + B + 1$
00100	$F = A + \bar{B}$
00101	$F = A + \bar{B} + 1$
00110	$F = A - 1$
00111	$F = A$
01000	$F = A \wedge B$
01010	$F = A \vee B$
01100	$F = A \oplus B$
01110	$F = \bar{A}$
10000	$F = B$
10100	$F = srB$
11000	$F = slB$

b) Discuss design options for the adder of the Function Unit. Your discussion should include a comparison of *carry look-ahead adder* and *ripple carry adder.*

[5 marks]

c) Provide a schematic that shows a *Pipelined Datapath* and discuss the differences between a *pipelined* and an *un-pipelined Datapath*.

[5 marks]

8.

a) Provide a ***Block Diagram*** for the following VHDL code:

[10 marks]

```
-- VHDL code
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
entity binary_multiplier is
     port(CLK, RESET, G, LOADB, LOADQ: in std_logic;
       MULT_IN: in std_logic_vector(3 downto 0);
       MULT_OUT: out std_logic_vector(7 downto 0));
end binary_multiplier;

architecture behavior_4 of binary_multiplier is
     type state_type is (IDLE, MUL0, MUL1);
     signal state, next_state : state_type;
     signal A, B, Q: std_logic_vector(3 downto 0);
     signal P: std_logic_vector(1 downto 0);
     signal C, Z: std_logic;
begin
     Z <= P(1) NOR P(0);
     MULT_OUT <= A & Q;

     state_register: process (CLK, RESET)
     begin
       if (RESET = '1') then
         state <= IDLE;
       elsif (CLK'event and CLK = '1') then
         state <= next_state;
       end if;
     end process;

     next_state_func: process (G, Z, state)
     begin
       case state is
         when IDLE =>
           if G = '1' then
             next_state <= MUL0;
           else
             next_state <= IDLE;
           end if;
         when MUL0 =>
           next_state <= MUL1;
         when MUL1 =>
           if Z = '1' then
             next_state <= IDLE;
           else
```

```
            next_state <= MUL0;
          end if;
      end case;
    end process;

    datapath_func: process (CLK)
    variable CA: std_logic_vector(4 downto 0);
    begin
      if (CLK'event and CLK = '1') then
        if LOADB = '1' then
        B <= MULT_IN;
        end if;
        if LOADQ = '1' then
        Q <= MULT_IN;
        end if;
        case state is
          when IDLE =>
            if G = '1' then
              C <= '0';
              A <= "0000";
              P <= "11";
            end if;
          when MUL0 =>
            if Q(0) = '1' then
              CA := ('0' & A) + ('0' & B);
            else
              CA := C & A;
            end if;
            C <= CA(4);
            A <= CA(3 downto 0);
          when MUL1 =>
            C <= '0';
            A <= C & A(3 downto 1);
            Q <= A(0) & Q(3 downto 1);
            P <= P - "01";
        end case;
      end if;
    end process;
end behavior_4;
```

b) Provide an ***ASM chart*** for the above VHDL code:

[10 marks]