UNIVERSITY OF DUBLIN

TRINITY COLLEGE

Faculty of Engineering, Mathematics and Science
School of Computer Science and Statistics

Integrated Computer Science Programme
Integrated Computer Engineering
Integrated Electronic Computer Engineering
Senior Freshman Examination

Trinity Term 2012

CS2022 – Computer Architecture II

11th May 2012

Main Hall
RDS

14:00-16:00

Dr. Michael Manzke

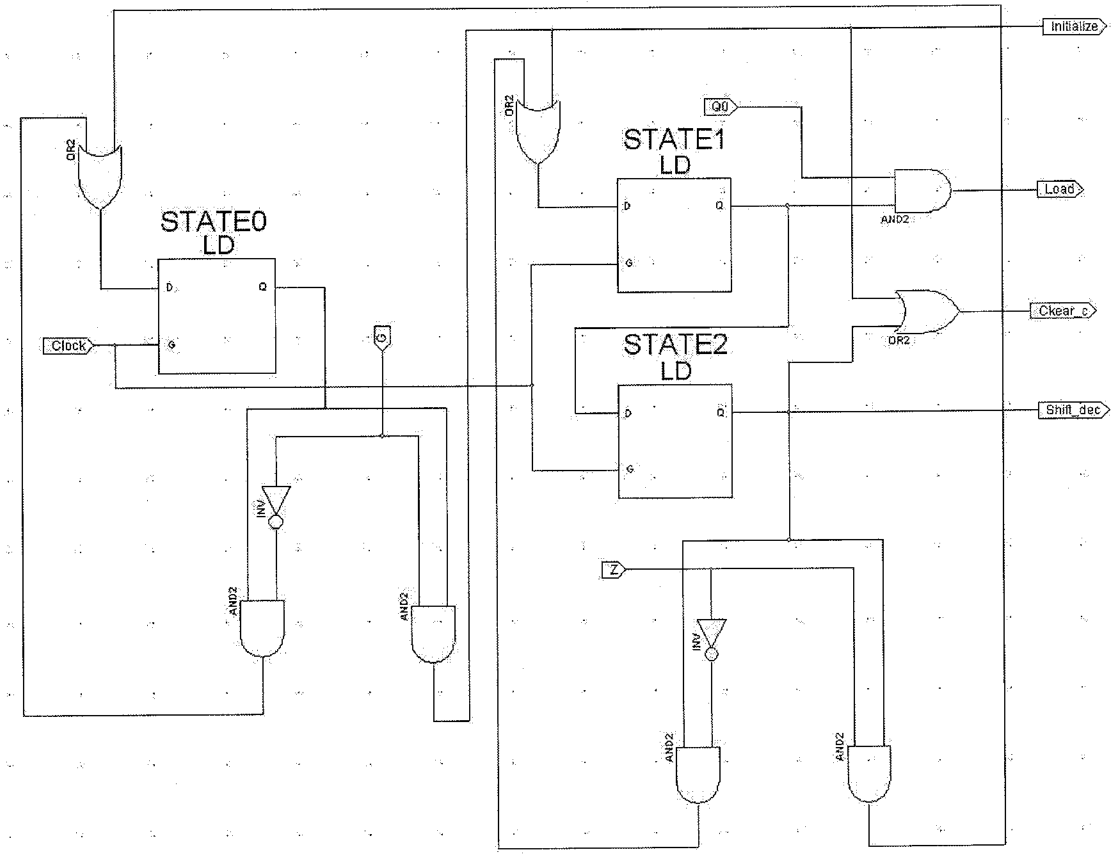
Answer **three** questions.

The use of non-programmable calculators is permitted.

1.

a) Provide an *Algorithmic State Machine* (ASM) chart for the following schematic. Explain how the logic is derived from the ASM.

[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]

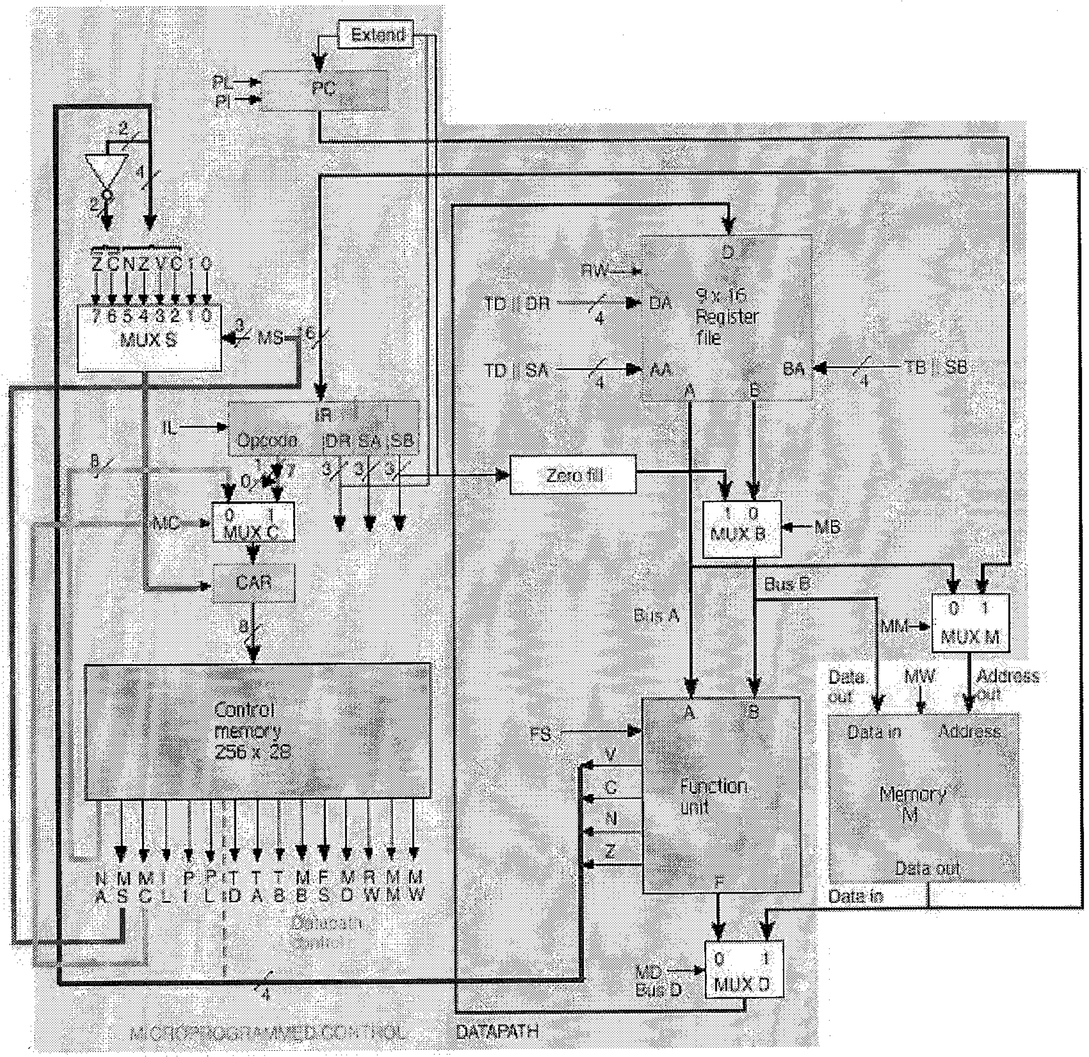
c) Write VHDL code that implements the part of the *Algorithmic State Machine* (ASM) that is shown in *Question 1.a)* with gates and flip-flop.

[8 marks]

2.

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

[10 marks]



b) Expand on your discussion from Question 2.a) by explaining how the branch instruction is executed.

[10 marks]

3.

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

[10 marks]

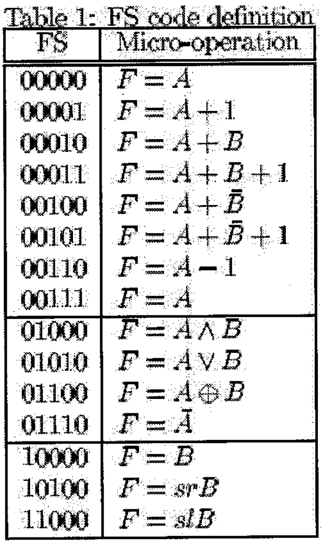


Table 1: FS code definition

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 barrel shifter vs. single bit shifter.

[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]

4.

a) Provide a ***Block Diagram*** for the following VHDL code. Your solution should show all registers and their connections. The control may be shown as a box.

[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 for a microcoded control solution. Discuss the operations performed by the VHDL code.

[10 marks]