UNIVERSITY OF DUBLIN

TRINITY COLLEGE

Faculty of Engineering and System Sciences

Department of Computer Science

B.A. (Mod) Computer Science **Trinity Term 2006**

Senior Freshman Examination

2ba4 – Computer Architecture I

Tuesday 6th June 2006 **Goldsmith Hall** **09:30 – 12:30**

Mr. M. Manzke, Dr. D.P. Mc Carthy

You are required to answer at least two questions from each section and a total of five questions.

Use separate answer books for section A and section B

Non-programmable calculators are permitted for this examination – record the make and model of your calculator on your answer book.

Mathematical tables are permitted for this examination.

You may not start this examination until you are instructed to do so by the Invigilator.

Section A – (M. Manzke)

1. a) [12 marks] Provide a detailed discussion of the IO communication mechanism that is used in the microprocessor project. Explain how the system detects that its CPU intends to communicate with one of the two IO devices. Discuss the function of the two JK-flip-flops that are defined in the following VHDL jkflipflop entity. You should provide a timing diagram that shows all the relevant signal e.g. VPA, E and complete the PORT MAPs in the architecture of the decoder entity (see question 2). Please copy the completed VHDL code into your answer book.

   b) [8 marks] Assume that the IO devices are configured to assert an interrupt on arrival of a new packet and furthermore suppose that the IRQ line is connected to one of the CPU's interrupt pins. How should the system react if this signal is asserted? What is the VPA signal's function in this context?

```
-- Michael Manzke
-- Trinity College Dublin
-- Computer Science Department
-- 3 April 2006
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity jkflipflop is
    Port (CLK, J, K, CLR_NOT : in std_logic;
          Q, Q_NOT : out std_logic);
end jkflipflop;

-- This entity implements an 73LS73A
-- J-K Flip-Flop with asynchronous clear
-- +----------------------+---------------+
-- |        Inputs        |    Outputs    |
-- +----------------------+---------------+
-- |Clear Clock   J     K |   Q      Q_not|
-- +----------------------+---------------+
-- |  L     X     X     X |   L      H    |
-- |  H    fe*    L     L |   Q      Q_not|
-- |  H    fe*    H     L |   H      L    |
-- |  H    fe*    L     H |   L      H    |
-- |  H    fe*    H     H |    TOGGEL     |
```

```
-- |  H     H     X     X  |   Q      Q_not|
-- +-----------------------+---------------+
--    * falling edge (fe)

architecture Behavioral of jkflipflop is

     signal QSIG, Q_NOTSIG: std_logic;

begin

     process (CLK, CLR_NOT)
     begin
     if CLR_NOT='0' then   --asynchronous clear active Low
          QSIG <= '0';
               Q_NOTSIG <= '1';
     elsif (CLK'event and CLK='1') then  --CLK falling edge
          if (J='1' and K='0') then
               QSIG <= '1';
                    Q_NOTSIG <= '0';
          elsif (J='0' and K='1') then
               QSIG <= '0';
                    Q_NOTSIG <= '1';
          elsif (J='1' and K='1') then
               QSIG <= not(QSIG);
                    Q_NOTSIG <= not(Q_NOTSIG);
               end if;
     end if;
          Q <= QSIG;
          Q_NOT <= Q_NOTSIG;
     end process;

end Behavioral;
```

2. a) [14 marks] Please provide a memory map for the following decoder entity. You should copy this VHDL code into your answer book and complete locations that are highlighted with question marks with the correct VHDL code or comments. Do not complete the PORT MAPs for the jkflipflop entity. This is part of question 1. You find the sizes of the individual components in the code and they are here repeated for your convenience ROM($2000), RAM1($800), RAM2($800), IO1($4) and IO2($4). Your memory map implementation should meet all project requirements.

   b) [6 marks] Discuss the function of the VAC signal and explain how the signal is generated and used in the VHDL code.

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity decoder is
    Port ( A11, A12, A13, A14, A15 : in std_logic;
           A16, A17, A18, A19 : in std_logic;
           AS : in std_logic;
           FC0, FC1, FC2 : in std_logic;
           CLOCK, E : in std_logic;
           CSROM : out std_logic;
           CSRAM1 : out std_logic;
           CSRAM2 : out std_logic;
           CSIO1 : out std_logic;
           CSIO2 : out std_logic;
           DTACK : out std_logic;
           VPA : out std_logic;
           VMA : out std_logic);
end decoder;

architecture Behavioral of decoder is

     COMPONENT jkflipflop
     PORT(CLK : IN std_logic;
          J : IN std_logic;
          K : IN std_logic;
          CLR_NOT : IN std_logic;
          Q : OUT std_logic;
          Q_NOT : OUT std_logic);
     END COMPONENT;

     signal CSROMSIG, CSRAM1SIG, CSRAM2SIG : STD_LOGIC;
     signal CSIO1SIG,   CSIO2SIG : STD_LOGIC;
     signal VAC, IOSEL, Q1J2, Q_NOT1K2 : STD_LOGIC;

begin

     -- Interrupt
     VAC <= ?;

     --   ROM ($2000)
     --   $?????   %???? ???|? ???? ???? ????
     --   $?????   %???? ???|? ???? ???? ????
     CSROMSIG <= ?;

     --   RAM1 ($800)
     --   $?????    %???? ???? ?|??? ???? ????
     --   $?????    %???? ???? ?|??? ???? ????
     CSRAM1SIG <= ?;
```

```
    --   RAM2 ($800)
    --   $?????    %???? ???? ?|??? ???? ????
    --   $?????    %???? ???? ?|??? ???? ????
    CSRAM2SIG <=?;

    --   IO1 ($4)
    --   $?????    %???? ???? ?|??? ???? ????
    --   $?????    %???? ???? ?|??? ???? ????
    CSIO1SIG <=?;

    --   IO2 ($4)
    --   $?????    %???? ???? ?|??? ???? ????
    --   $?????    %???? ???? ?|??? ???? ????
    CSIO2SIG <=?;

    CSROM <= not(CSROMSIG);
    CSRAM1 <= not(CSRAM1SIG);
    CSRAM2 <= not(CSRAM2SIG);
    CSIO1 <= CSIO1SIG;
    CSIO2 <= CSIO2SIG;

    DTACK <= ?

    IOSEL <= ?

--
-- This must be completed for Question 1
--

    JK1: jkflipflop PORT MAP(
         CLK => ?,
         J => ?,
         K => '0',
         CLR_NOT => ?,
         Q => ?,
                                                Q_NOT => ?);

    JK2: jkflipflop PORT MAP(
         CLK => ?,
         J => ?,
         K => ?,
         CLR_NOT => '1',
         --Q => , we do not need this
         Q_NOT => ?);

    VPA <= Q_NOT1K2 or VAC;

end Behavioral;
```

3. a) [marks 6] Provide a detailed discussion of *Logic State Analysers.*

b) [marks 14] Assume you use a *Logic State Analyser* to observe bus transaction on a microprocessor board that executes the following program. This code shows only a very small subset of the entire program but it provides the information necessary to explain the operations that you would observe with a *Logic State Analyser* if you assume the following:

You monitor the *Interrupt line IPL0/2*, the *Databus lines* and the *Addressbus lines*. Provide a *Waveform drawing* that shows 7 consecutive address bus signals (addresses) and associated Databus signals (data) and the interrupt signal that you would observe if the first address bus signals (address) is $00480 and the CPU accepts an interrupt level 5 while the instruction is fetch from memory.

```
           00040BE6       9  INITSP   equ   $40BE6
           00000400       10 PROGR    equ   $00400
           00000400       11 INITPC   equ   PROGR

           00040000       15  RAM1F   equ   $40000 * Base Address RAM 1
           000407FF       16  RAM1L   equ   $407FF *
           00040800       17  RAM2F   equ   $40800 * Base Address RAM 2
           00040FFF       18  RAM2L   equ   $40FFF *

00000000                  31          org   $00000
00000000   00040BE6       32          dc.l  INITSP
00000004   00000400       33          dc.l  INITPC
00000074                  34          org   $00074  * Level 5 Interrupt
00000074   00000500       35          dc.l  AUTO5

0000046C   0C19 0000      82          cmpi.b #0,(a1)+
00000470   6700 000A      83          beq    OUT2
00000474   13D1 000C0000  84          move.b (a1),IODEV2+IODATA
0000047A   60E6           85          bra    WDEV2
0000047A   60E6           86  OUT2

0000047C   027C 2300      98          andi.w #%0010001100000000,SR

00000480   60FE           104 CLOOP   bra.s CLOOP
```

4. a) [10 marks] Explain how a system service is implemented in the processor's monitor. How many system services can be implemented and how would we calculate this number?

b) [10 marks] You should also discuss the implementation of an RAM vector table and provide arguments for this solution.

Section B – (D.P. Mc Carthy)

5. a) [15 marks] Design a circuit which will efficiently implement the following micro-operations under the control of a 3-bit select control S:

S	Micro-operation
000	A + 1
001	A - B
010	A + B
011	A - 1
100	sr A
101	sl A
110	A NAND B
111	A

b) [5 marks] Explain why shifters in datapath functional units are implemented using combinatorial circuits rather than a shift register.

6. a) [10 marks] Show how tristate buffers may be used to interconnect n registers R_i, i=0, n-1 to implement just one $R_i \leftarrow R_j$ i,j = 0, n-1 per clock cycle, and state the principal advantage of this system.

b) [10 marks] Design an efficient implementation of the following set of register transfers assuming that all control signals are mutually exclusive:

C_0: R1 ← R0
C_1: R4 ← R1, R2 ← R0, R0 ← R1
C_2: R3 ← R4, R1 ← R4
C_3: R3 ← R0, R0 ← R3

7. a) [12 marks] Draw a schematic of a micro-programmed control unit for a datapath, which will allow the implementation of the following instructions:

   i) Single-cycle datapath micro-instructions.

   ii) Unconditional branches.

   iii) Conditional branches based on the carry bit C being set or reset, i.e. BC or BNC.

   b) [8 marks] Write a symbolic micro-code sequence for your schematic of part a), specifying the instruction fetch sequence.

8. a) [10 marks] Describe how the status bits of Zero, Negative, Carry and oVerflow are generated in the functional unit of a datapath and explain when it is appropriate to load their values into the status register Z, N, C, V.

   b) [10 marks] For the following sequence of register transfers draw waveforms of R0 and the status bits Z, N, C, V against the CLOCK, assuming that all registers load on the rising edge of CLOCK:

CLOCK	Register transfer
1	R0 ← R2 – R2
2	R0 ← R0 + 1
3	R0 ← rol R0
4	R0 ← R0 - 1