IMPERIAL COLLEGE LONDON

TIMED REMOTE ASSESSMENTS 2020-2021

BEng Honours Degree in Computing Part I
MEng Honours Degrees in Computing Part I
for Internal Students of the Imperial College of Science, Technology and Medicine

This paper is also taken for the relevant assessments for the Associateship of the City and Guilds of London Institute

PAPER COMP40001

INTRODUCTION TO COMPUTER SYSTEMS

Tuesday 4 May 2021, 10:00
Duration: 95 minutes
Includes 15 minutes for access and submission

Answer ALL TWO questions
Open book assessment

Paper contains 2 questions

Introduction to Computer Systems - COMP40001

Given the present circumstances, this time-limited remote assessment is being run as an open-book examination. We have worked hard to create exams that assesses synthesis of knowledge rather than factual recall. Thus, access to the internet, notes or other sources of factual information in the time provided will not be helpful and may well limit your time to successfully synthesise the answers required.

Where individual questions rely more on factual recall and may therefore be less discriminatory in an open book context, we may compare the performance on these questions to similar style questions in previous years and we may scale or ignore the marks associated with such questions or parts of the questions. In all examinations we will analyse exam performance against previous performance and against data from previous years and use an evidence-based approach to maintain a fair and robust examination. As with all exams, the best strategy is to read the question carefully and answer as fully as possible, taking account of the time and number of marks available.

Section A (Use a separate answer book for this Section)

1 Boolean algebra, combinational circuit design and number representation.

a Using Boolean algebra, simplify the following Boolean expression to its simplest form (fewest number of operators) without any OR in its final form:

$$E = (A' \bullet (B + A)) + (B' \bullet (A + A'))$$

where $\bullet$, $+$, and $'$ represent "AND", "OR" and "NOT" operations respectively. Please show the sequence of your steps and explicitly state the rules you used.

b We want to build a circuit with a single output O, and three inputs: A, C_0 and C_1. When $C_1 = 1$, we will always set $O = 0$. Otherwise, $C_0 = 0$ sets $O = A$, and $C_0 = 1$ sets $O = A'$.

i) show the truth table and the boolean equation for this circuit in canonical minterm form.

ii) without applying further minimization, reason about the minimum number of 2-bit AND, 2-bit OR, and 1-bit NOT gates that you will need for a circuit that implements the canonical minterm form you obtained above (you do not need to write the circuit, but that might help you).

iii) given the inputs A, C_0, C_1 and the constant value 0, this circuit can be implemented with a single NOT gate and a commonly-used circuit we have already seen (which we will call "D", and itself contains multiple gates, six input signals, and one output signal). Identify which commonly-used circuit is "D", and show the circuit implementation of this problem using a single NOT gate and "D" (please carefully enumerate the inputs to "D").

c Design the circuit for a single, 3-bit two's complement arithmetic unit that can compute both additions and subtractions. The unit has two 3-bit input numbers ($A_{0..2}$ and $B_{0..2}$), a 3-bit output number ($R_{0..2}$), a carry output (C_{out}), and a function selector (F, where F=0 means compute $A + B$ and F=1 means compute $A - B$). Your design must only use 1-bit full adders, multiplexers, and NOT gates.

d Given the number $0xCAFE0000_{16}$ in hexadecimal IEEE single precision format, convert it into the following two notations. Show your working clearly.

i) $x \times 2^y$ notation using decimal numbers (one single digit before the decimal part).

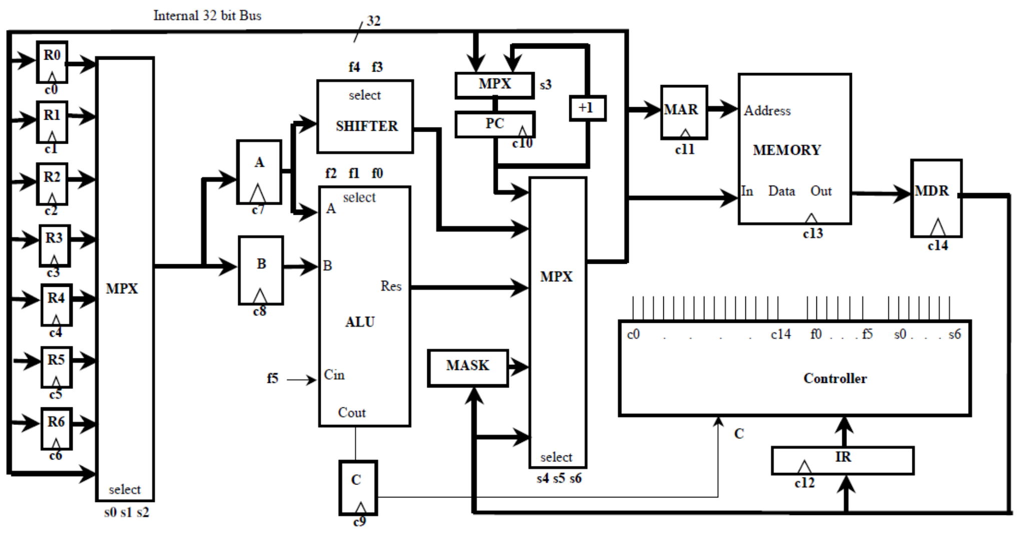
ii) $x \times 10^y$ notation using decimal numbers (one single digit before the decimal part).

The four parts carry equal marks.

Section B (Use a separate answer book for this Section)

2 A Manual Processor with Memory

A 32-bit manual processor with memory has the following block diagram:

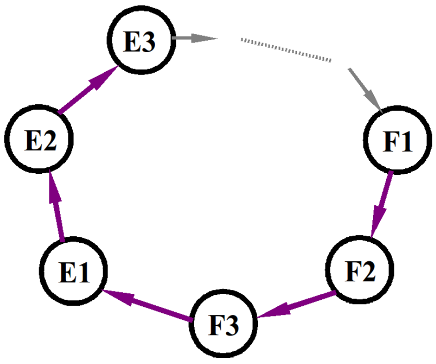


The function of the ALU is defined by the following table:

Selection Bits	000	001	010	011	100	101	110	111
Function	0	B-A	A-B	A plus B	A xor B	A+B	A·B	-1

The function of the shifter (by selection bits) is (00) hold, (01) shift left, (10) shift right arithmetic (duplicating the top bit), and (11) rotate right (setting the top bit to bit zero).

The controller is a finite state machine that first fetches a program instruction from the memory and then provides the correct operation sequences to execute that particular instruction. Hence, there are some fetch states (F) followed by some execute states (E):



a You must now define a part of the instruction set. The top eight bits (bits 31-24, if the lowest bit is bit 0) of the 32 bit instruction word will define the instruction ("opcode"). The next four bits (23-20) store the index of a destination register (Rdest). The data (such as an address) carried in the instructions (if any) are stored in bits 19-0. This enables us to separate out any data from the opcode with a very simple MASK circuit shown on the data path diagram in the first figure.

The two-register instructions are now to be defined. In the following table, there is an example given for MOVE. Complete for ADD, and COMPARE. Rsrc thereby stands for the source register. ADD adds two registers. COMPARE is just a subtract with a check to see that the result is zero. This is achieved checking the carry. This will be zero if Rsrc≤Rdest, and 1, otherwise.

Instruction	Cycle	Transfers	Path

MOVE Rdest, Rsrc	E1
	E2
		A←Rsrc
		Rdest←Shifter
			
			Shifter set to no change

ADD Rdest, Rsrc	E1 E2 E3		
COMPARE Rdest, Rsrc	E1 E2 E3		

b Which further two-register instructions exist from the specification above? Name these, and describe, which from the commands in the table from the last question they resemble most, and how they differ from it.

c We now design the skip instructions. They either skip the next instruction in memory (“SKIP”) or do nothing (“NOP”). Use the table below to specify SKIP. How is SKIP used in connection with COMPARE?

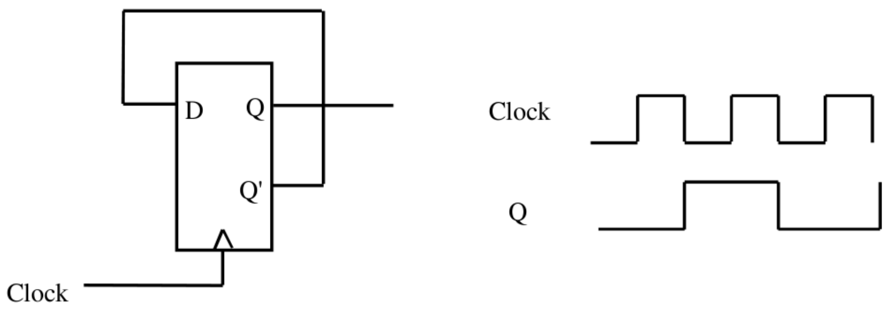
How many cycles does NOP require, and which registers are transferred? Name an example, how NOP can be usefully applied.

Instruction	Cycle	Transfers	Path

d The architecture shown above can be run as fast as 100 kHz.

Suppose you have a quartz crystal as a regulator which has a characteristic property that allows a very accurate square wave of 1 MHz to be produced. Hence, you need to design a circuit which divides the clock by 10.

It is possible to divide a clock by two simply using the one bit counter shown in the figure below. Here, the value of D is taken directly from Q’, and therefore, at each clock pulse, the flip flop will simply change state. If we look at the timing diagrams, we see that Q will output a square wave which will be exactly half the frequency of the clock as shown in the below figure’s right hand side. We can therefore consider the circuit a clock divider.



Show how you can connect according flip flops with an additional clear input on top to build the needed divide by ten counter. You may use additional AND and inverter gates.

e We now assume you need to design a circuit which divides the clock by 512. Show how you can solve this concatenating synchronous divide by eight circuits. Is the resulting circuit synchronous?

The five parts carry equal marks.