IMPERIAL COLLEGE LONDON

TIMED REMOTE ASSESSMENTS 2021-2022

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

Friday 20 May 2022, 10:00
Writing time: 80 minutes
Upload time: 25 minutes

Answer ALL TWO questions
Open book assessment

This time-limited remote assessment has been designed to be open book. You may use resources which have been identified by the examiner to complete the assessment and are included in the instructions for the examination. You must not use any additional resources when completing this assessment.

The use of the work of another student, past or present, constitutes plagiarism. Giving your work to another student to use constitutes an offence. Collusion is a form of plagiarism and will be treated in a similar manner. This is an individual assessment and thus should be completed solely by you. The College will investigate all instances where an examination or assessment offence is reported or suspected, using plagiarism software, vivas and other tools, and apply appropriate penalties to students. 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.

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.

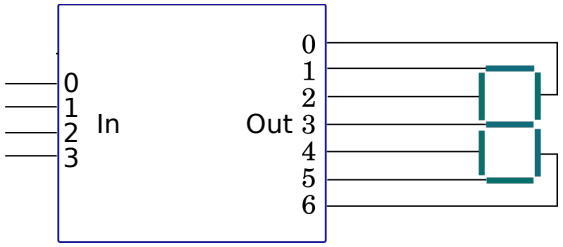
1 Boolean algebra, combinational circuit design and number representation.

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

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

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 have a digital controller that takes an unsigned binary representation of a decimal number (4 bits, with input lines $In_{0..3}$) and lights up the corresponding segments of an LCD display to show that number (with output lines $Out_{0..6}$).



Note: the LCD should only show decimal digits (and stay blank otherwise).

Build the boolean equations for Out_4 and Out_6, expressed using:

i) the canonical minterm form.

ii) their minimized version using a Karnaugh map.

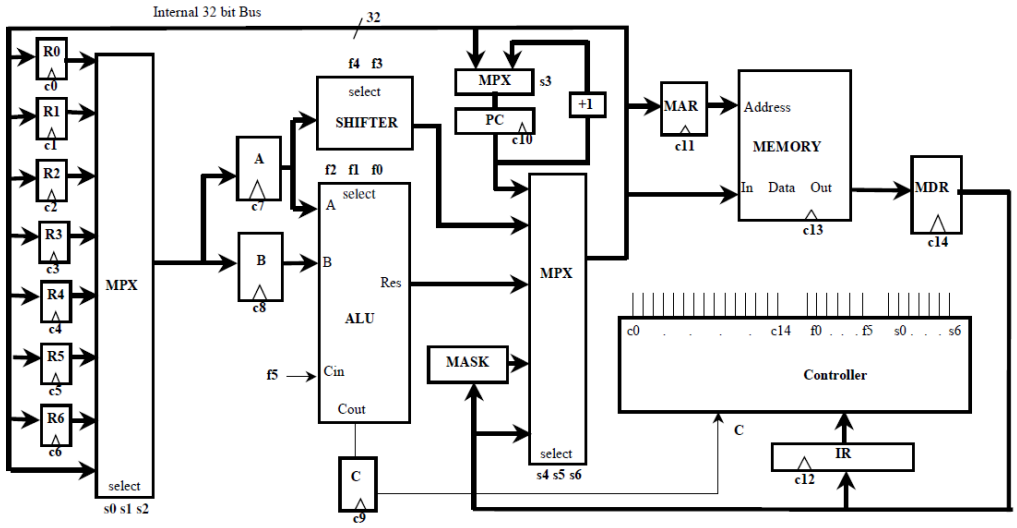
c Design a unit that has one 2-bit input number ($A_{0..1}$), a 3-bit output number ($R_{0..2}$), and a 2-bit function selector ($F_{0..1}$) where F=0 means compute A, F=1 means compute $A + 1$, and F=2 means compute $A - 1$). Your design can only use 1-bit full adders, and an assortment of 2-way 1-bit multiplexers and logic gates, using the minimum amount of each of them.

d Given the decimal number -123.45, express it in binary and hexadecimal using the 32-bit IEEE single precision format. Show your working clearly. If you use extra bits in your calculation, use rounding rather than truncation when calculating the result.

The four parts carry equal marks.

2 A Manual Processor with Memory

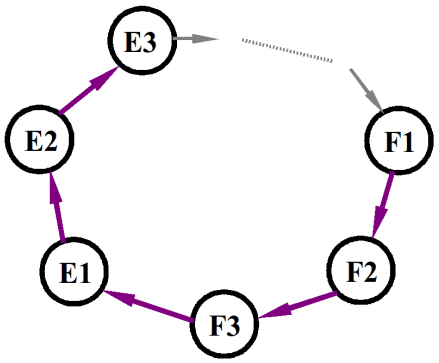
A 32-bit manual processor with memory has the following block diagram:



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 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 "One Register Instructions" are now to be defined. In the following table, there is an example given for CLEAR. Complete for INC (increment register by one), DEC (decrement register by one), and RETURN (return after a CALL; CALL jumps to an address calling a sub-routine, but preserves the Program Counter (PC) incremented properly for continuation after returning).

Instruction	Cycle	Transfers	Path
CLEAR Rdest	E1	Rdest←ALUres	ALU = zero out
INC Rdest	E1		
DEC Rdest	E1		
RETURN Rdest	E1		

b The one register instructions include further instructions such as compare or shifts. We now design arithmetic shift left (ASL) and arithmetic shift right (ASR) instructions. Use the tables below to specify ASL and ASR and to define the usage of the function bits f4 and f3 in the first figure in this exam.

Instruction	Cycle	Transfers	Path

Instruction	Function	f4	f3
Default	No Action	0	0
		0	1
		1	0
		1	1

In addition to the realised shift operations, one could design commands for logic shift left (LSL), logic shift right (LSR). How would these differ from their respective counter-parts ASL and ASR? You can use MSB for most significant bit and LSB for least significant in your answer.

c Further, we could add a selection of rotate instructions. These can be divided based on carry treatment and direction. Name the resulting different types for rotating in this respect.

How would you need to change the shifter inputs in the first figure in this exam to cater additionally for the new LSL, LSR, and four further rotate commands?

d The architecture shown above can be run as fast as 100 kHz. How could you make it go faster? Name two suggestions other than speeding up the clock.

e Can you divide a clock by two with a single D-Q Flip Flop? How would you need to wire it up, if so? And how could you divide a clock by four with D-Q Flip Flops in an asynchronous manner?

The five parts carry equal marks.