IMPERIAL COLLEGE OF SCIENCE TECHNOLOGY AND MEDICINE
UNIVERSITY OF LONDON

DEPARTMENT OF ELECTRICAL AND ELECTRONIC ENGINEERING
M.Eng., B.Eng., B.Sc(Eng.) and A.C.G.I. EXAMINATIONS 2008

PART III and PART IV

DIGITAL SYSTEM DESIGN

SOLUTIONS

First Marker: *Peter Y. K. Cheung*

Second Marker: *Thomas Clarke*

Answer to Question 1

This question tests students' understanding of FSM and how it may be designed and implemented on an FPGA.

(a)

Current State					Inputs		Next State					
D4	D3	D2	D1	D0	in_A	in_B	Q4	Q3	Q2	Q1	Q0	FSM_out
0	0	0	0	1	0	X	0	0	0	0	1	0
0	0	0	0	1	1	0	0	0	0	1	0	0
0	0	0	0	1	1	1	0	0	0	0	1	0
0	0	0	1	0	0	X	0	0	0	0	1	0
0	0	0	1	0	1	0	0	0	0	1	0	0
0	0	0	1	0	1	1	0	0	1	0	0	0
0	0	1	0	0	0	0	1	0	0	0	0	0
0	0	1	0	0	0	1	0	1	0	0	0	0
0	0	1	0	0	1	0	0	0	0	1	0	0
0	0	1	0	0	1	1	0	0	1	0	0	0
0	1	0	0	0	0	0	1	0	0	0	0	0
0	1	0	0	0	0	1	0	1	0	0	0	0
0	1	0	0	0	1	0	0	0	0	1	0	0
0	1	0	0	0	1	1	0	0	1	0	0	0
1	0	0	0	0	0	X	0	0	0	0	1	1
1	0	0	0	0	1	0	0	0	0	1	0	1
1	0	0	0	0	1	1	0	0	0	0	1	1

$$Q0 = D0 \bullet \overline{in\_A} + D0 \bullet in\_A \bullet in\_B + D1 \bullet \overline{in\_A} + D4 \bullet \overline{in\_A} + D4 \bullet in\_A \bullet in\_B$$

$$Q1 = D0 \bullet in\_A \bullet \overline{in\_B} + D1 \bullet in\_A \bullet \overline{in\_B} + D2 \bullet in\_A \bullet \overline{in\_B} + D3 \bullet in\_A \bullet \overline{in\_B} + D4 \bullet in\_A \bullet \overline{in\_B}$$

$$Q2 = D1 \bullet in\_A \bullet in\_B + D2 \bullet in\_A \bullet in\_B + D3 \bullet in\_A \bullet in\_B$$

$$Q3 = D2 \bullet \overline{in\_A} \bullet in\_B + D3 \bullet \overline{in\_A} \bullet in\_B$$

$$Q4 = D2 \bullet \overline{in\_A} \bullet \overline{in\_B} + D3 \bullet \overline{in\_A} \bullet \overline{in\_B}$$

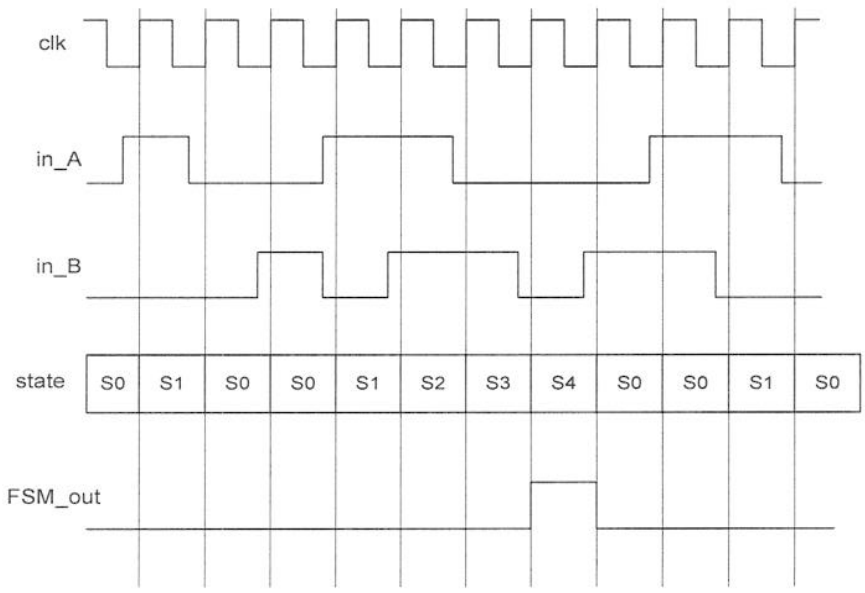
$$FSM\_out = D4$$

[10 marks]

b) FPGA is register rich, therefore one-hot encoding, which requires one register per state, matches the FPGA fabric well. Furthermore, there is no need for state decoding, reducing the demand on the combinatorial logic (i.e. LUT resources).

Q0, Q1, Q2 – 2 LUTs each; Q3, Q4 – 1 LUT each, FSM_out is free. Total: 8 LUTs.

[5 marks]



[5 marks]

Answer to Question 2

This question is mostly bookwork with some application to a new problem (i.e new angle to compute sin and cos values).

a) Given: $x_{i+1} = x_i - y_i \tan\alpha_i$

$y_{i+1} = y_i + x_i \tan\alpha_i$

$z_{i+1} = z_i - \alpha_i$

Let: $\alpha_i = \tan^{-1} 2^{-i}$

$\tan\alpha_i = 2^{-i}$

This gives: $x_{i+1} = x_i - d_i\, y_i 2^{-i}$

$y_{i+1} = y_i + d_i\, x_i 2^{-i}$

$z_{i+1} = z_i - d_i\, \alpha_i$

where $d_i \in \{-1, 1\}$

as determined by some criterion

For rotation mode to compute sin α and cos α, we need to perform the following:

1. Set z = α
2. x = 1/K = 0.607252935 (or some scaling constant)
3. y = 0
4. Iterate with d_i = sign (z_i)

After m rotations, we get

$$x_m \approx \cos(z)$$
$$y_m \approx \sin(z)$$
$$z_m \approx 0$$
$$y / x \approx \tan(z)$$

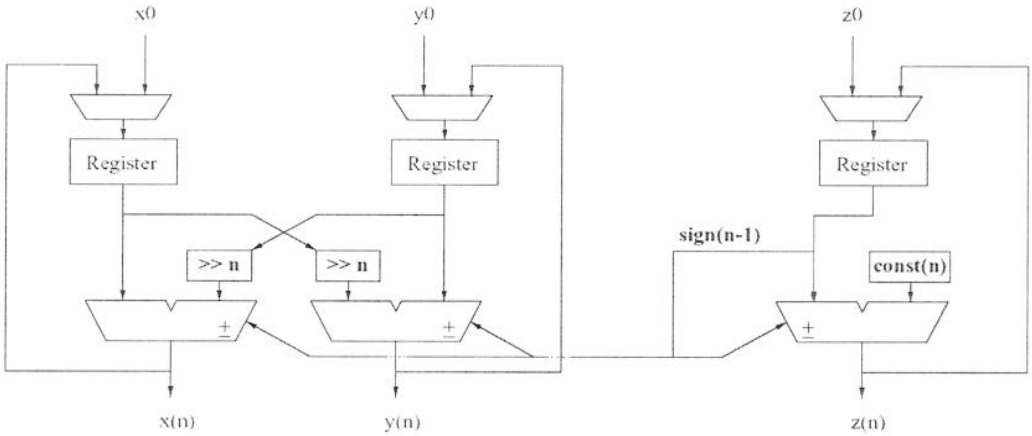
[6 marks]

For α = 47.163°, the following angles of rotation are used:

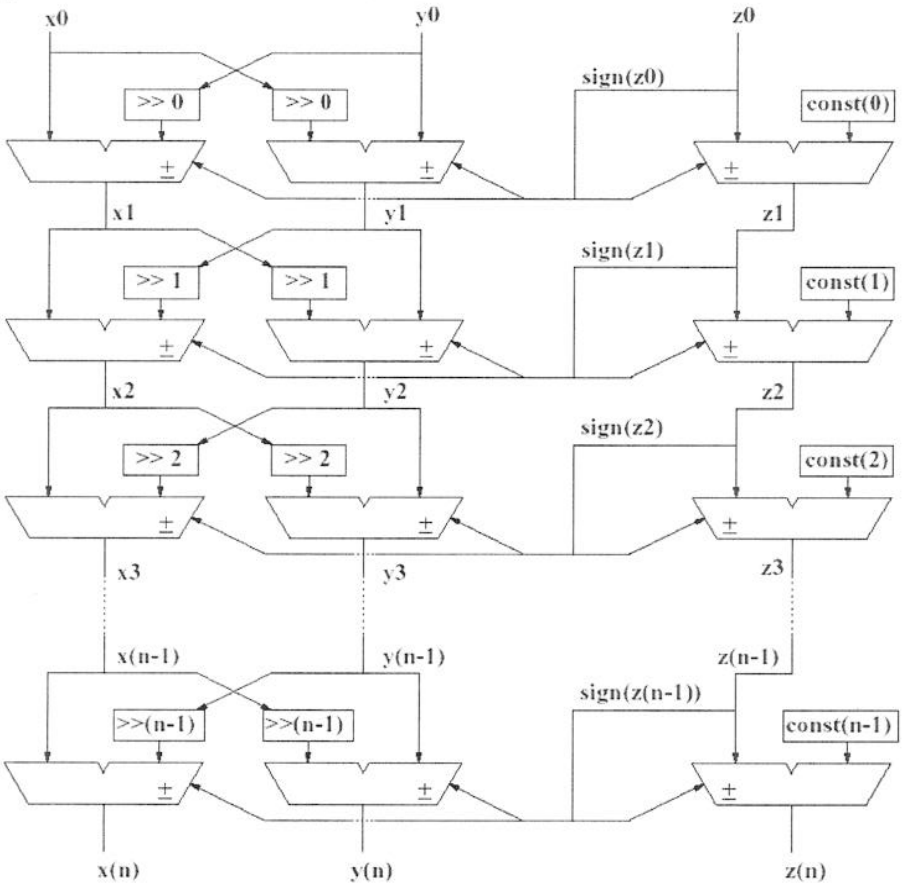
Iteration	d_i	Operation	z value
0			47.163
1	-	47.163 – 45	2.163
2	-	2.163 – 26.565	-24.402
3	+	-24.402 + 14.036	-10.366
4	+	-10.366 + 7.125	-3.241
5	+	-3.241 + 3.576	0.335
6	-	0.335 – 1.790	-1.455
7	+	-1.455 + 0.895	-0.560
8	+	-0.560 + 0.448	-0.112
9	+	-0.112 + 0.224	0.112
10	-	0.112 – 0.112	0.000

[6 marks]

b) Parallel implementation without unrolling:



Parallel implementation with iteration unrolled:



[8 marks]

Answer to Question 3

a)

Parity bit is an extra bit that can be added to a group of data bits to make the parity of the group odd or even. Odd and even parity means that there are odd or even number of "1" s in the group including the parity bit. Parity bit can be used to detect single bit error.

[2 marks]

b)

Book work – show this using a 4-bit example adding 3 check bits. If there are N check bit, can correct single bit error in $2^N - N - 1$ bit wide data.

[3 marks]

c) i)

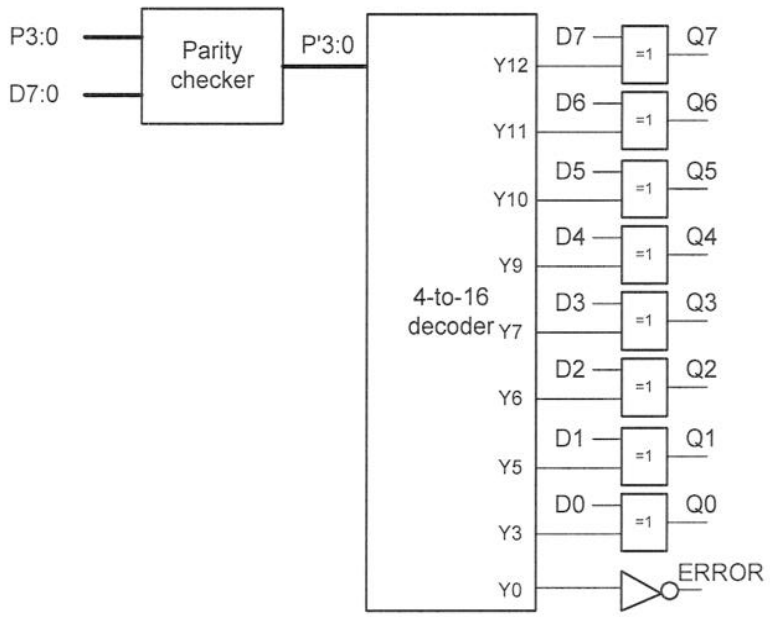
4 parity check bits are required to correct single bit error in 8-bit data. The 4 parity bits could be assigned according to this table:

	d7	d6	d5	d4	d3	d2	d1	d0
P3	x	x	x	x				
P2	x				x	x	x	
P1		x	x		x	x		x
P0		x		x	x		x	x
Code	12	11	10	9	7	6	5	3

$P3 = d7 \oplus d6 \oplus d5 \oplus d4$
$P2 = d7 \oplus d3 \oplus d2 \oplus d1$
$P1 = d6 \oplus d5 \oplus d3 \oplus d2 \oplus d0$
$P0 = d6 \oplus d4 \oplus d3 \oplus d1 \oplus d0$

[5 marks]

ii)

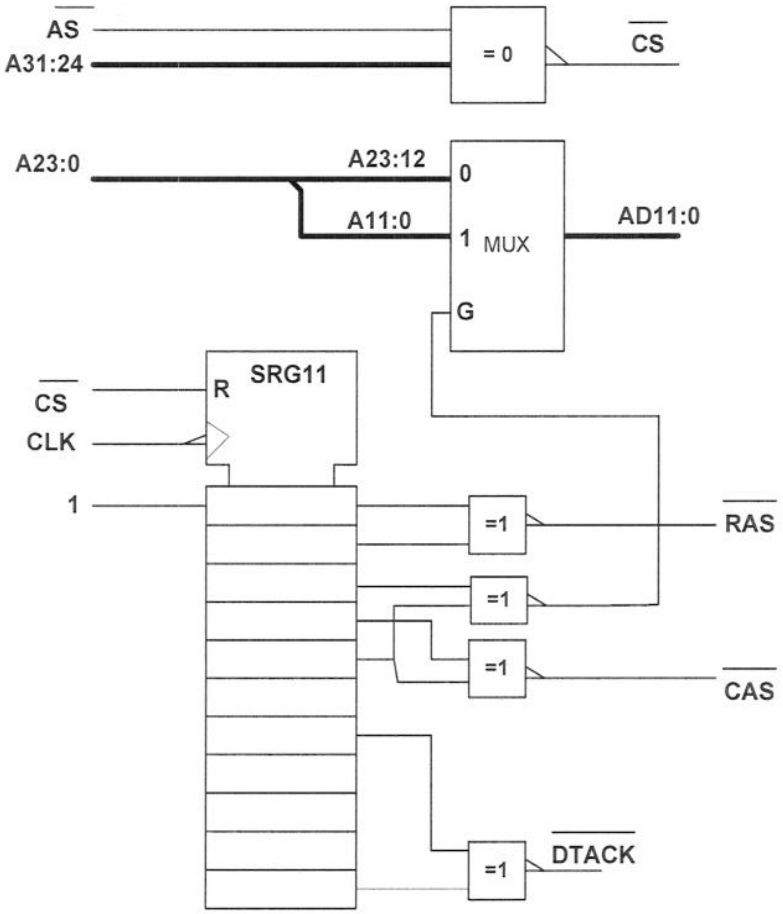


[10 marks]

Answer to Question 4

This question tests student's understanding and the design of a memory interface circuit.

Shift register based design is probably the simplest:



[18 marks]

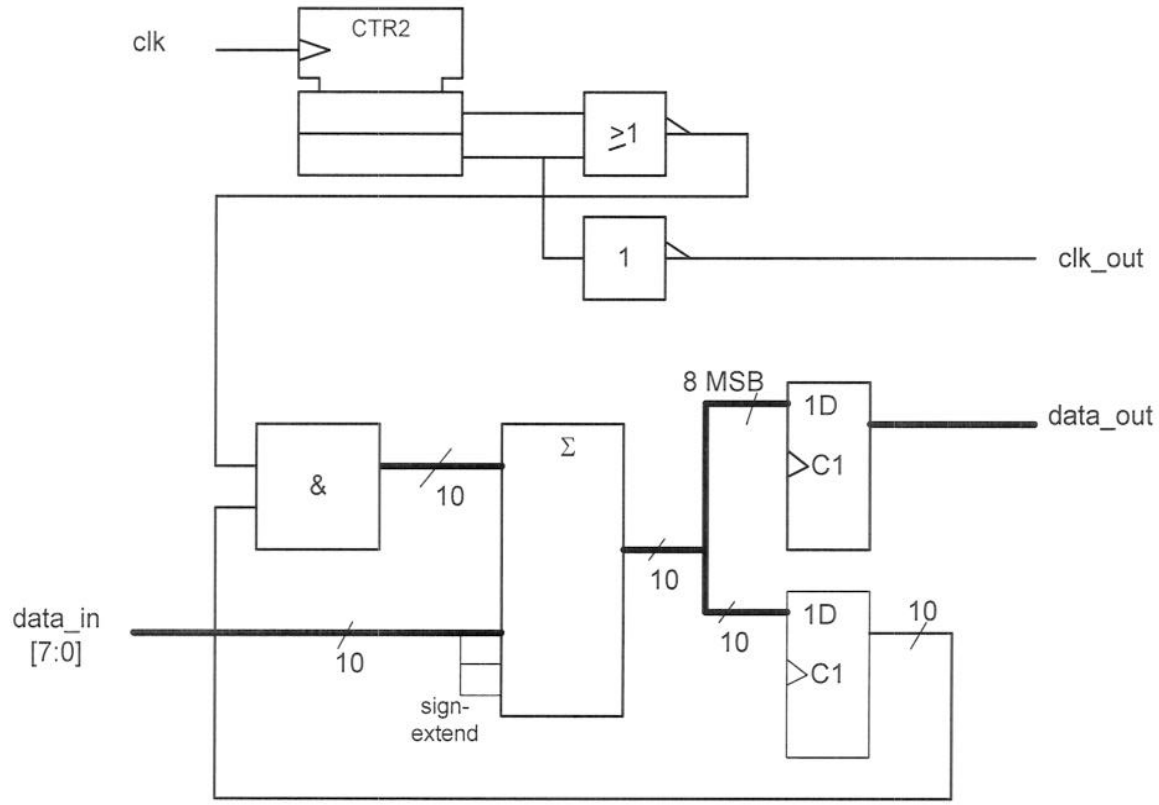
Timing assumption: A >= B + C + setup of SR

[2 marks]

Answer to Question 5

This question test student's ability to design a relatively straight forward circuit with simple control (over 4 clock cycles).

Possible design:



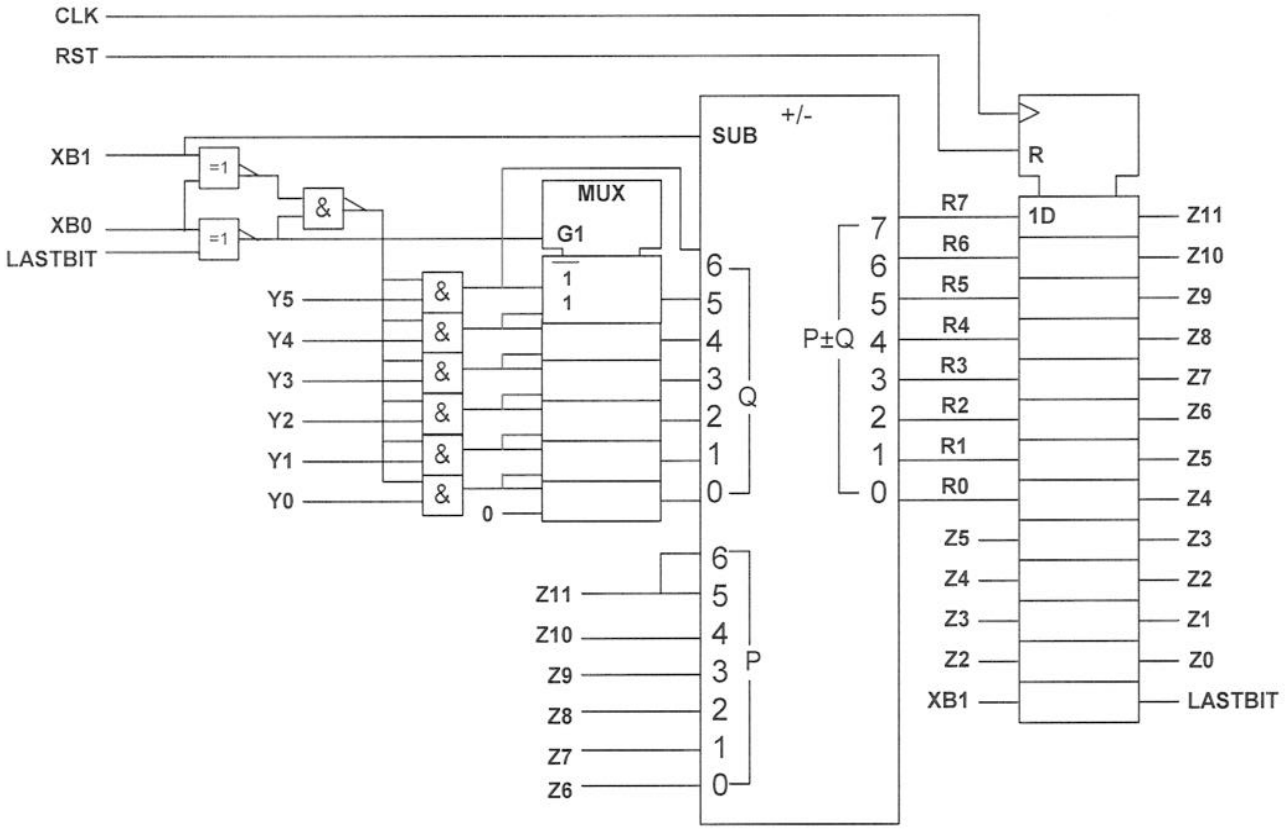
[15 marks]

Students should check that their circuit meets the timing requirement at 40MHz clock.

[5 marks]

Answer to Question 6

This is a standard modified booth multiplier that takes three cycles to do 6 bits.



[15 marks]

The +/- block consists of 8-bit adder with XOR gates on Q-input controlled by SUB. Therefore worst case delay is 7+2 = 9ns.

Worst case path relative to CLK:

CLK → LASTBIT → XNOR → 2 gates →MUX →add/sub →setup time

1 2 2 1 7 1.5 = 14.5ns

(This will vary depending on student's circuit.)

Therefore max frequency = 69 MHz.

[5 marks]