IMPERIAL COLLEGE LONDON

DEPARTMENT OF ELECTRICAL AND ELECTRONIC ENGINEERING
EXAMINATIONS 2011

EEE/ISE PART III/IV: MEng, BEng and ACGI

Corrected Copy

DIGITAL SYSTEM DESIGN

10:40 Q1 error in Fig 1.1

Wednesday, 18 May 10:00 am

Time allowed: 3:00 hours

12:25 - Q4(d) $T_0 = 9.5 \times 10^{-12}$

There are SIX questions on this paper.

Answer FOUR questions.

All questions carry equal marks

Any special instructions for invigilators and information for candidates are on page 1.

Examiners responsible First Marker(s) : C. Bouganis
Second Marker(s) : T.J.W. Clarke

Special information for invigilators:
None

Information for candidates:
In the figures showing digital circuits, all components have, unless explicitly indicated otherwise, been drawn with their inputs on the left and their outputs on the right. All signals labelled with the same name are connected together. All circuits use positive logic. The least significant bit of a bus signal is labelled as bit 0, and the most significant bit with the highest integer number. Therefore the signal X[7:0] is an eight bit bus with X7 being the MSB and X0 the LSB.

Hexadecimal numbers are prefixed with $. For example the decimal number 10 is written as $A.

In questions involving circuit design, you may use any standard digital circuits that are not explicitly forbidden by the question provided that you fully specify their operation.

Marks may be deducted for unnecessarily complex designs unless you are explicitly instructed not to simplify your solution.

The Questions

1. a) The CORDIC algorithm is based on the idea of rotating a vector $\mathbf{v} = (x, y)$ by an angle θ using a specific set of elementary angles a^i, where $i \in [0, N-1]$, for a predefined set of iterations N. Starting from the basic rotation equations:

$$x^{i+1} = x^i \cos(a^i) - y^i \sin(a^i)$$

$$y^{i+1} = y^i \cos(a^i) + x^i \sin(a^i)$$

and the fact that $z^{i+1} = z^i - a^i$, where z is the angle accumulator, derive the hardware friendly equations for each iteration of the CORDIC algorithm and describe the main ideas behind them. (Hint: use the fact that: $\cos(\theta) = 1/(1 + \tan^2(\theta))^{1/2}$)

[10]

b) Sketch the circuit of a bit-parallel iterative implementation of a CORDIC processor, and state the throughput of the system.

[4]

c) Describe the operation of the CORDIC algorithm in vectoring mode of operation.

[3]

d) State with justification how many CORDIC iterations are required to evaluate $\arctan(\beta)$, where $\beta = 0.5$, when a bit-parallel iterative implementation of a CORDIC processor is used.

Assume that the scaling factor for only 7 iterations is available in the processor. The elementary angles used in the CORDIC implementation are illustrated in Figure 1.1.

[3]

a	**atan(a) in degrees**
1	45.000
0.5	26.565
0.25	14.036
0.125	7.125
0.0625	3.576
0.03125	1.790
0.0115625	0.895

0.015625

Figure 1.1 Elementary angles and their tangents

2. a) Design a fully-unrolled architecture that implements a 3-tap FIR filter defined by the following expression

$$y(n) = 2x(n) + 0.6x(n-1) + 3x(n-2)$$

where x is the input signal and y is the output signal. The input signal takes integer values in the range $[0, 255]$. State, justifying your design choices, the number of bits used for the implementation of the coefficients and the wordlength (i.e. number of bits) of all the busses in your system. Your input x and output y signals have to be registered.

[13]

b) Comment on the throughput of your design, and describe a way to maximise the operating frequency of the system. Assume that you cannot change the internal structure of the adder and multiplication blocks of the system.

[2]

c) Derive a new bit-parallel architecture that implements the above FIR filter, when minimisation of the the arithmetic blocks (i.e. adders, multipliers) is targeted. Comment on the resulting throughput.

[5]

3. a) Describe the internals of a memory cell in SRAM and DRAM modules, and provide a comparison.

[5]

b) Design an SRAM memory module of $2^4 \times 4$ bits (i.e. 2^4 entries, 4 bits wide). Provide the circuit for the address pre-decoder module.

[5]

c) Assume that such a module is mapped onto an FPGA device that utilises 4-LUTs (Look Up Tables). Calculate the number of LUTs needed for your given pre-decoder circuit.

[5]

d) Describe and design a mechanism that detects single errors when a read operation takes place in the memory of part (b). State how the effective storage space of the memory module has been changed.

[5]

4. a) Figure 4.1 depicts the state diagram of a 4-state finite-state machine (FSM), with an input signal A. The input signal A is not synchronous with the system. When A is '1', the FSM moves to the next state, otherwise it stays at the same state. Using the minimum number of D flip-flops and logic, design a circuit that implements the FSM. Provide the boolean equations for the next state signals.

[10]

b) Assuming that the logic is mapped to 4-LUTs (Look Up Tables) when the target device is an FPGA, estimate the number of 4-LUTs required to implement the FSM.

[2]

c) State the problems that may be encountered due to the asynchronous input A, and describe a possible solution.

[3]

d) By applying your proposed solution, calculate the Mean Time Between Failure (MTBF) when the whole design is clocked with a 20MHz clock. The rate of asynchronous transitions per second is 40, and $\tau = 1ns$ and $T_0 = 9.5 \times 10^{12} sec.$ The propagation delays of the components of the system are given in Figure 4.2. Assume that the wires do not exhibit any delay, and all registers are clocked with the same clock signal.

MINUS
9.5×10^{-12}

[5]

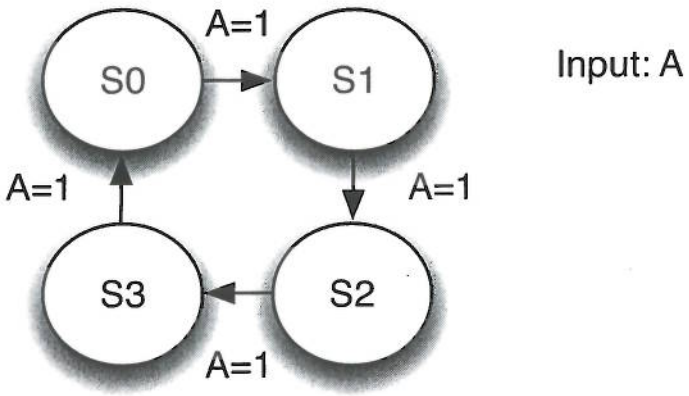


Figure 4.1 State diagram

symbol	value	description
T_p^{reg}	5ns	register propagation delay
T_{setup}^{reg}	2ns	register setup time
T_{hold}^{reg}	3ns	register hold time
T_p^{LUT}	10ns	4-LUT propagation time

Figure 4.2 Propagation delay of the components

5. A block diagram of a 1 bit full adder (FA) is given in Figure 5.1. The equations that describe its outputs are $s = x \oplus y \oplus c_{in}$, and $c = x \cdot y + c_{in} \cdot (x \oplus y)$.

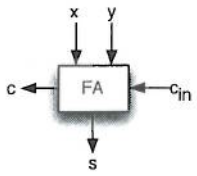


Figure 5.1 Full Adder

a) Using the above FA as a building block, design a 2-bit serial adder that produces a 2 bit output (i.e. ignore the carry from the MSB stage).

[4]

b) Assuming that only 4-LUTs (Look Up Tables) are used when such a design is mapped on an FPGA, estimate the number of LUTs required for the design of part (a), without taking into account the logic required for the registers.

[4]

c) State the critical path, and derive the maximum frequency of the design in part (b). The propagation delays of the components are shown in Figure 5.2.

[4]

symbol	value	description
T_p^{reg}	5ns	register propagation delay
T_{setup}^{reg}	2ns	register setup time
T_{hold}^{reg}	3ns	register hold time
T_p^{LUT}	10ns	4-LUT propagation time

Figure 5.2 Propagation delay of the components

d) An alternative architecture would be a 2-bit full carry look-ahead adder (ignore the carry from the MSB stage). State the total number of 4-LUTs required for the above 2-bit full carry look-ahead adder design.

[4]

e) State the critical path, and derive the maximum frequency of the design in part (d). The propagation delays of the components are shown in Figure 5.2. Assume that the input and output signals are registered.

[4]

6. a) Assume an FPGA device with embedded RAM blocks that can be configured in the following configurations (entries $\times$ width): $16K \times 1$, $8K \times 2$, $4K \times 4$, $2K \times 9$, $1K \times 18$, 512×36. Design a module for the calculation of $f(x) = round(e^x)$ based on a memory look-up table, where $round()$ is the rounding function. x takes values in the range $[0, 15.75]$ with resolution 0.25. That is, x takes values such as: $0, 0.25, 0.5, 10.25, 11.75, \ldots$. Provide the number of bits required for representing x and $f(x)$.

[10]

b) What is the most appropriate configuration for the embedded RAM block?

[2]

c) Comment on the scalability of the design with regards to the required memory size with respect to the resolution of the input and the resolution of the output.

[3]

d) An alternative design would be to use a polynomial approximation for the calculation of $f(x) = e^x$ such as:

$$e^x \approx 1 + \frac{x}{1} + \frac{x^2}{2}$$

Derive an architecture that is based on the above polynomial approximation that exhibits maximum throughput. State the word-length (i.e. number of bits) of the busses in your design, which satisfy only the input specification of part (a). Assume that you cannot change the internal structure of the arithmetic components of the design.

[5]

DSD solutions 2011

EE3-65

[Bookwork]

Note: In round brackets is the break down of the marks for each question.

1. a)

$$x^{i+1} = x^i \cdot \cos\alpha^i - y^i \cdot \sin\alpha^i = \left(x^i - y^i \cdot \tan\alpha^i\right)/(1+\tan^2$$

$$y^{i+1} = y^i \cdot \cos\alpha^i + x^i \cdot \sin\alpha^i = \left(y^i + x^i \cdot \tan\alpha^i\right)/(1+\tan^2$$

$$z^{i+1} = z^i - \alpha^i$$

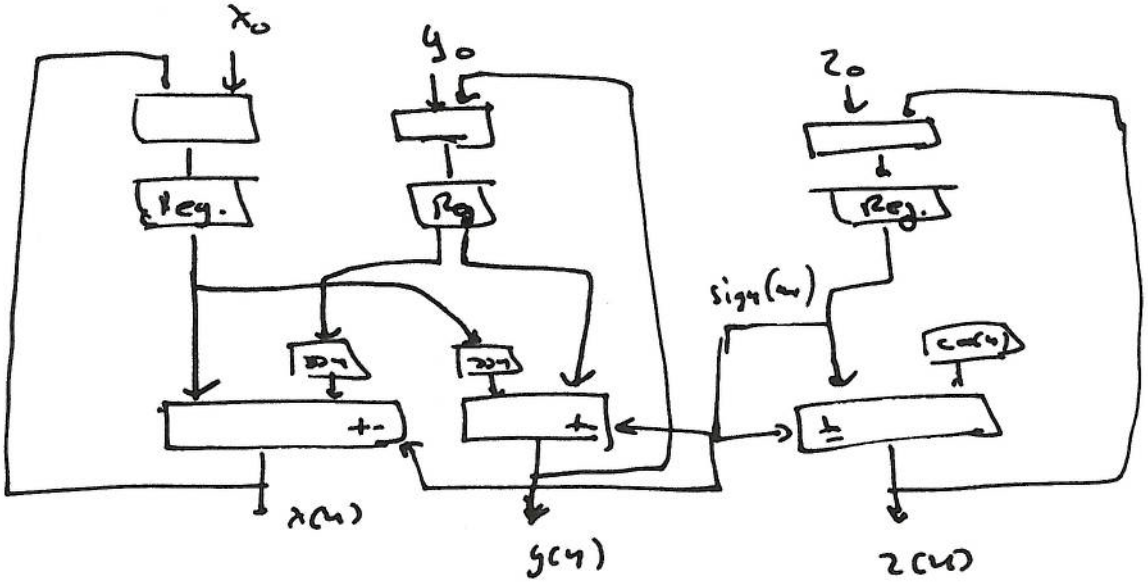
$$\Rightarrow \quad x^{i+1} = x^i - y^i \cdot \tan\alpha^i$$

$$y^{i+1} = y^i + x^i \cdot \tan\alpha^i \qquad (4)$$

There are pseudorotations, and the final results should be scaled by: $K = \prod_{i=0}^{N-1} \frac{1}{(1+\tan^2\alpha^i)^{1/2}}$ (3)

Also, the set of angles α^i should have $\tan\alpha^i = 2^{-i}$ (3) in order to avoid multiplication.

b) [10]



[4]

Throughput: 1 result per K clock cycles.

c) In vectoring mode, we initialize $x = x$, $y = y$, $z = z$,

and iterate with $d_i = -\text{sign}(x_i y_i)$, which forces y towards 0

At the end $z = z_{init} + \tan^{-1}(y/x)$

[3]

d) Possible solution

Since the scaling factor is only for 7 iterations $\Rightarrow$ 7 iterations will be performed.

Init values: $z = 0$

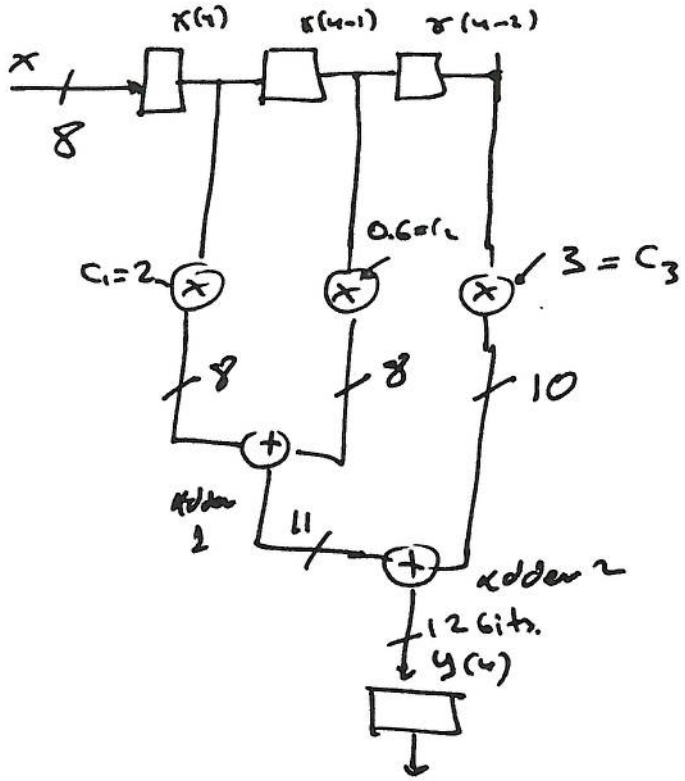
$x = 1$

$y = 4$ [redacted] 0.5.

However, a better solution would be that only 7 iterations will be performed, regardless of the availability of the scaling factor for 7 iterations.

[Calculation for a new example]

2) a)



$C_1 = 2$ → no bits. just rewiring.

$C_2 = 2 \cdot 0.3$, 0.3 needs infinite number of bits under a finite arithmetic. Selection of 3 bits : $0.3 \cong 0.01_{\Delta} = 0.25$, and I need to check the impact on the transfer function of the FIR.

$C_3 = 3$ ⇒ 11_{Δ}, 2 bits.

Result of c_1 would. input xxxxxxxx$_{\Delta}$ Δ: decimal po

output xxxxxxxxx$_{\Delta}$

Result of c_2 would input xxxxxxxx$_{\Delta}$

output xxxxxxxxx$_{\Delta}$

adder 1: xxxxxxxx$_{\Delta}$

xxxxxxxx$_{\Delta}$

adder 2:

$$\begin{array}{rl} & \times\times\times\times\times\times\times\times\times\times\times_0{}^{\times} \\ + & \times\times\times\times\times\times\times\times\times\times\times\times_{\Delta} \end{array}$$

Result : 12 bits, one of them decimal.

[13]

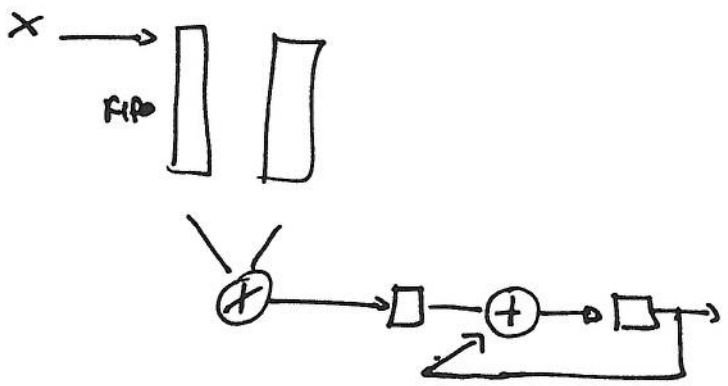
b) The current throughput is 1 result/ clock cycle.(1) However the critical path is long. Addition of pipline stages ~~reg~~ increases the frequency ⇒ and the throughput.

add reg. after the muls and adder 1. Note, after each c_3, you need to add two reg. (1)

However, for this specific choice of coeffs, the first stage of the pipeling can be avoided, due to simple ~~re-delay~~ reordering of the muls.

[2]

c)

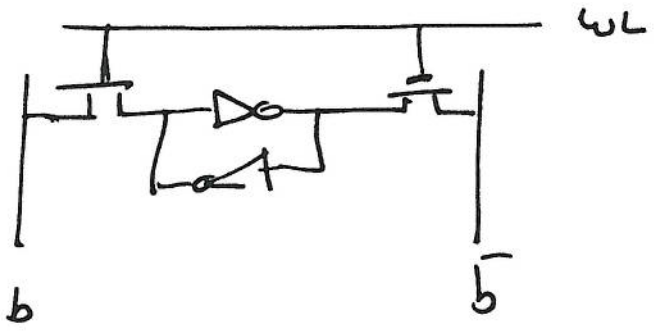


[3]

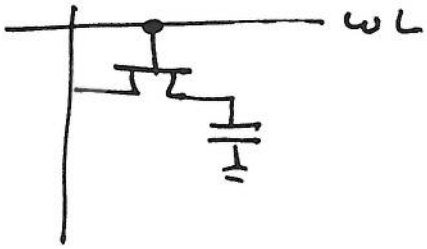
Throughput: 3 clock cycles per result.

[Goodwords + calculation for a new example]

3) a)



SRAM



DRAM

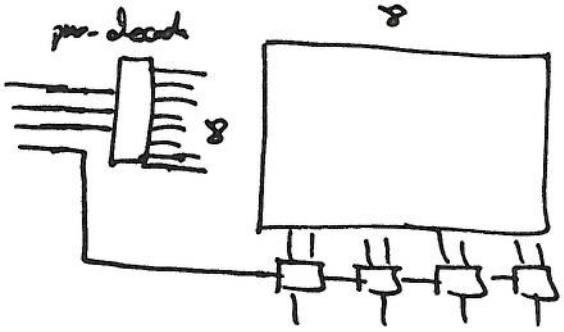
SRAM
- faster
- no refreshing
- uses 6T

DRAM
- refreshing is needed
- more dense
- uses 1T

[5]

b) The memory array cell should be as square as possible

I need a memory 16 x 4 → 8 x (2x4)



[7]

c) The predecoder has 8 output, each one is a function of 3 inputs. ⇒ 8 4-LUTs.

[5]

d) To detect a single error is to use a parity bit.

I will use an even parity, so need to check whether the received number of 1's is an even number. If no then a single error has occured.

Initial memory: $\underbrace{16 \times 4}_{64}$ bits. Current memory is $\underbrace{16 \times 3}_{48}$ bits.

[5]

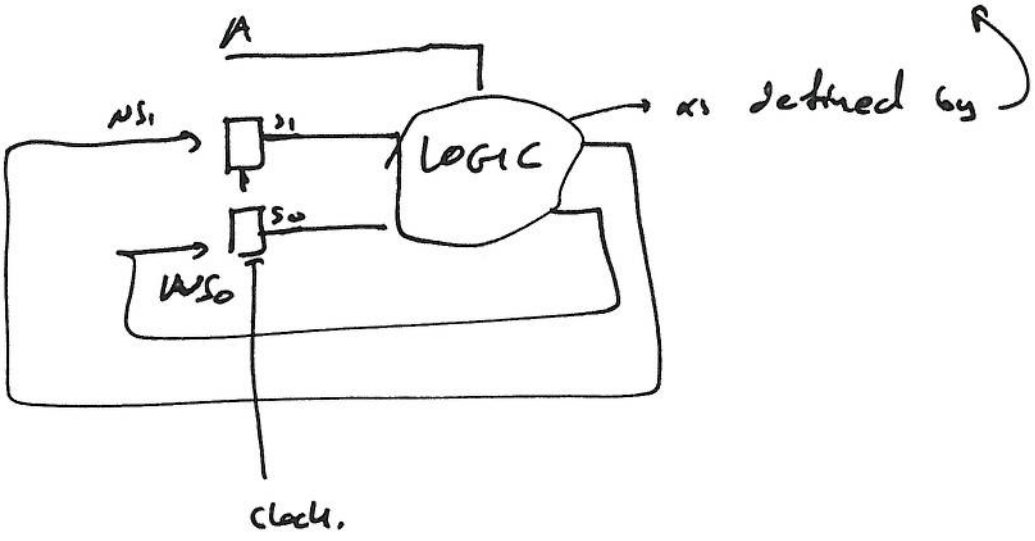
[calculate & new couple]

4) a) I have 4 states, ∴ I need 2-D-ff.

A	S_1	S_0	NS_1	NS_0
0	0	0	0	0
0	0	1	0	1
0	1	0	1	0
0	1	1	1	1
1	0	0	0	1
1	0	1	1	0
1	1	0	1	1
1	1	1	0	0

$$NS_1 = \bar{A}\, S_1 \bar{S}_0 + \bar{A} S_1 S_0 + A \bar{S}_1 S_0 + A S_1 S_0$$

$$NS_2 = \bar{A}\, \bar{S}_1 S_0 + \bar{A} S_1 S_0 + A \bar{S}_1 \bar{S}_0 + A S_1 \bar{S}_0$$



[10]

b)

each NS_i is a function of 3 inputs. So in total I need 2 - 4-LUTs.

c) The problem is that A can cause a change during the setup-hold window of the 2 ff. This will cause the system to become metastable, and may go not to the proper state. A possible solution is to add a synchronizer.

Multiple clock domains are not needed, as the system changes with the transition of A.

Two possible solutions: a) using 1 FF

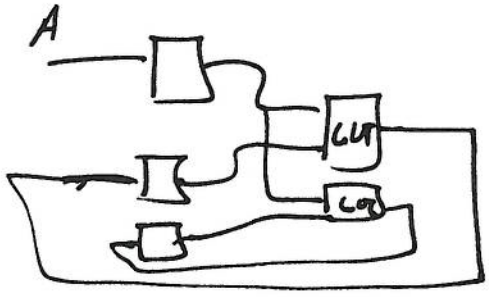
b) using 2 FF.

[3]

d)

The solution depends on the solution given in point c).

a) for 1 D-ff



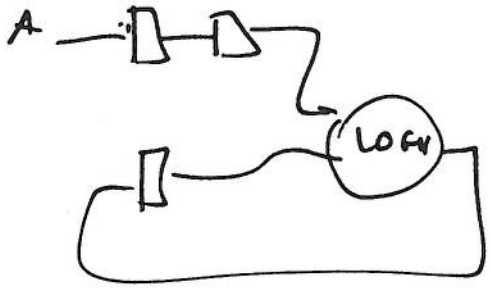
Assuming the two LUTs have the same delay, consider only one path.

~~$t_r = t_p^{LUT} + t_p^{LUT}$~~ $t_r + t_p^{LUT_1} + t_s = T \Rightarrow t_r = T - t_p^{LUT} - t_s \Rightarrow$

$t_r = 5 \cdot 10^{-8} - 10 \cdot 10^{-9} - 2 \cdot 10^{-9} = 38 \cdot 10^{-9}$

$MTBF = \frac{e^{\frac{38 \cdot 10^{-9}}{1 \cdot 10^{-9}}}}{\quad}$

d) il a R4.



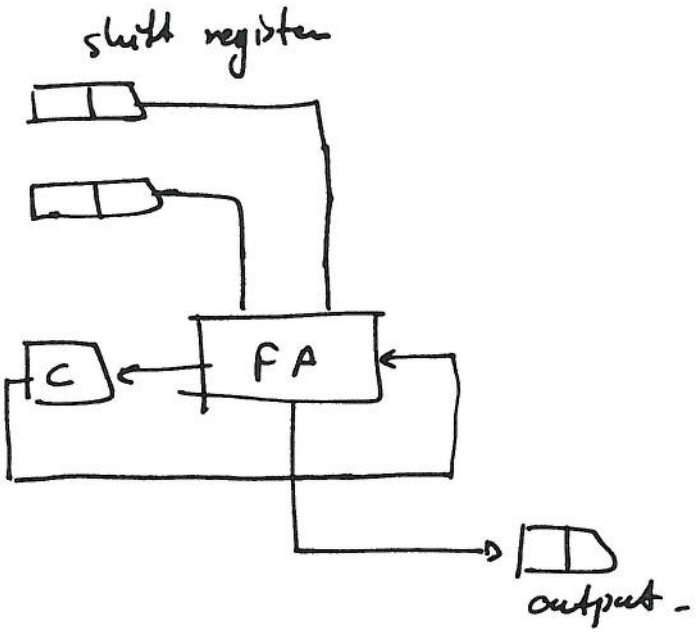
Then $t_r + t_s = T \rightarrow t_r = \ldots$

MTBF. = ...

[5]

[Goodwork + new example]

5) a)



[4]

b) There are 2 output signals out of FA, which are ~~both~~ functions of 3 Boolean variables. Thus, total number of 4-LUTs is 2.

[4]

c) Both outputs will have the same propagation delays.

The ~~critical~~ critical path is from the ff of the input reg to the $\overline{C}$ or the output registers.

For the maximum freq only the setup eq. needs to be considered.

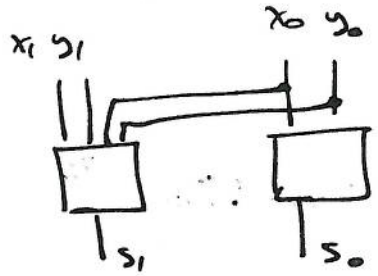
$$t_{cp}^{ff} + t_p^{reg} + t_s^{ff} < T \Rightarrow T > 5 + 2 + 10 \Rightarrow$$

$T > 17$ nsec.

$$f < \frac{1}{17} \cdot 10^{-9}$$

[4]

d) The full carry look ahead looks at all the bits of the input.



Each output is a function of 4 boolean variables, so 2 4-LUTs are needed.

[4]

e) The critical path is from any of the inputs, to any of the outputs.

$$t_p^d + t_p^{LUT} + t_p^d < T \Rightarrow T > 17\text{ns}.$$

[4]

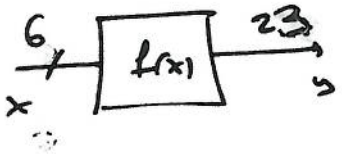
6) a) [new theory]

since x takes values in that range, it needs 4 bits for the integer part and 2 bits for the decimal part.

The minimum value of f(x) is 1, when the maximum is

round $(e^{15.75}) = 6990510 = A$

$\log_2 A = 22.72 \Longrightarrow$ So we need 23 bits to represent f(x).



[10]

b) The number of entries are $2^6 = 64$, so I need 64×23 memory. Out of the available configurations I need 512×36, using a part of the memory.

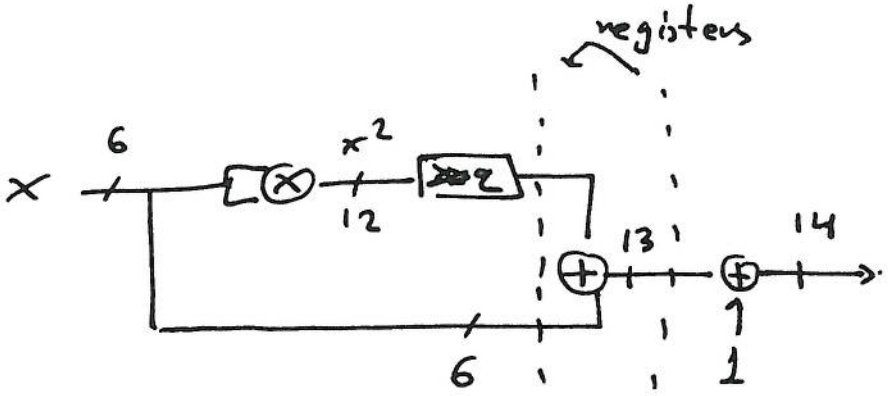
[2]

c) Increase by one bit at the input doubles the number of entries. ⇒ and the memory.

Increase by one bit at the output, increases the memory size by the number of entries

[3]

d) for maximum throughput, it should be pipelined.



[5]