Digital Electronics II 2010/11

(http://www.ee.ic.ac.uk/hp/staff/dmb/courses/dig2/dig2.htm)

Mike Brookes (mike.brookes@imperial.ac.uk)

Lectures: Tue 16:00 Weeks 2-9 (12/10 – 30/11)
Fri 14:00 Week 1 (9/10)
Fri 17:00 Weeks 2-8 (15/10 – 26/11)

Problem Classes: Tue 15:00 Weeks 4 - 11 (26/10 – 14/12)
Fri 16:00 Week 3 (22/10)
You are strongly advised to attempt the indicated problems before attending the problem class.

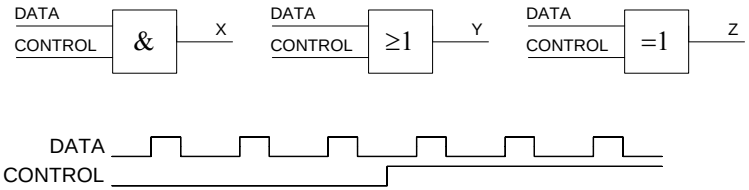
			Introduction
Week 1		L1	Notation, Cause and Effect, Flipflops, Counters
			Interfacing Digital Systems
Week 2		L2	Synchronous bit-serial interfacing
		L3	Asynchronous bit-serial interfacing
Week 3		L4	Microprocessor-to-memory interface
	Prob 1		▯ Problem Class: P1.1, P1.3, P1.5
		L5	Microprocessor-to-memory timing constraints
Week 4			▯ Problem Class: P1.8
			Synchronous State Machines
		L6	Shift register control and sequencing
		L7	Data decoding with a counter
Week 5	Prob 2		▯ Problem Class: P2.2, P2.3, P2.4
		L8	Synchronous state machine analysis
		L9	Synchronous state machine design
Week 6			▯ Problem Class: P2.7, P2.8, P2.12
			Digital ↔ Analog Conversion
		L10	Digital-to-analog conversion
		L11	Analog-to-digital conversion: Flash and dither
Week 7	Prob 3		▯ Problem Class: P3.3, P3.5, P3.7
		L12	Analog-to-digital conversion: Successive approximation
			Addition Circuits
		L13	Adders and propagation delays
Week 8	Prob 4		▯ Problem Class: P4.2, P4.5, P4.8
		L14	Fast adders: bit inversion & carry lookahead
		L15	Fast adders: carry skip and carry save
Week 9	Prob 5		▯ Problem Class: P5.5, P5.6, P5.8
Week 10	Prob 6		▯ Problem Class: P6.2, P6.8, P6.9
Week 11	Prob ?		▯ Problem Class: ??

E2.11/ISE2.22 – Digital Electronics II

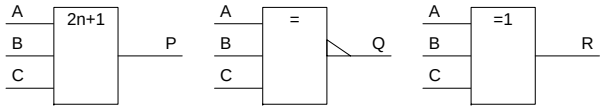
Problem Sheet 1

(Question ratings: A=Easy, …, E=Hard. All students should do questions rated A, B or C as a minimum)

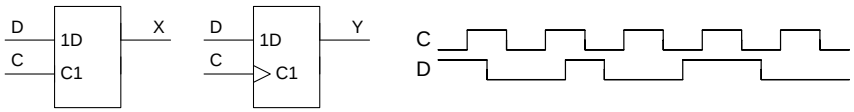
1A. The diagram shows three gates in which one input (CONTROL) is being used to modify a signal at the other input (DATA). Complete the timing diagram by drawing the waveforms of X, Y and Z. Describe in words the effect each of the gates has on DATA when CONTROL is low and when it is high.



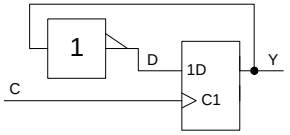
2B. The symbol in a gate generally indicates how many of the inputs need to be high to make the output high. Guess the truth tables of the following gates from their symbols. Explain why any one of them could be considered as a 3-input XOR gate.



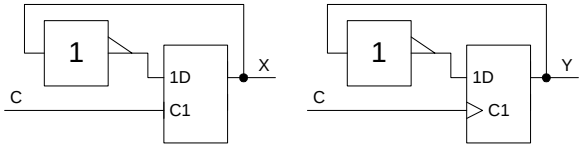
3A. The circuits below are a D-latch and a D-flipflop. Complete the timing diagram by drawing the waveforms of X and Y assuming that they are both low initially.



4B. The circuit below forms a ÷2 counter. If the inverter has a propagation delay of 5 ns and the propagation delay, setup time and hold time of the flipflop are 8 ns, 4 ns and 2 ns respectively, calculate the highest clock frequency for reliable operation.

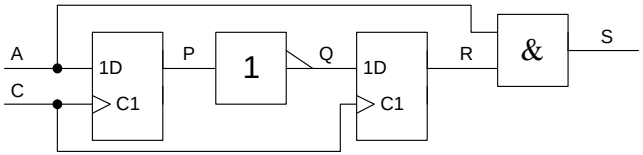


5B. The circuits below are a D-latch and a D-flipflop with their outputs connected to their inputs via an inverter. Draw the waveforms of X and Y assuming that they are both low initially and that C is a uniform square wave. (One of these circuits is a disaster and should never be used)



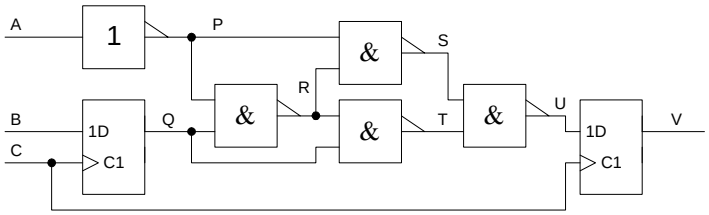
6B. In the circuit below the propagation delay of the flipflops may vary between 4 and 7 ns while the propagation delay of the gates may vary between 2 and 6 ns.

Calculate the minimum and the maximum propagation delays from each of A and C to each of P, Q and R and S.



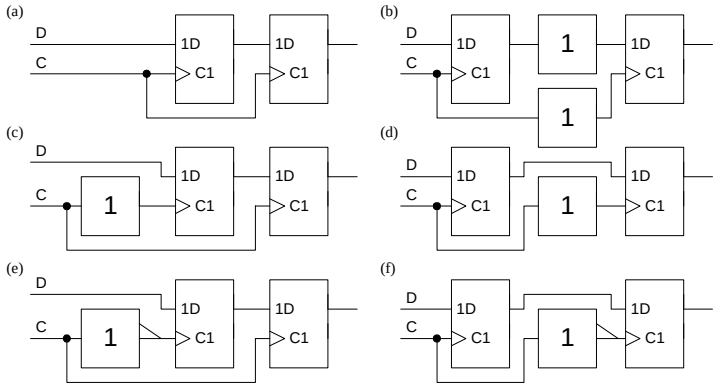
7C. In the circuit below the setup and hold times of the flipflops are 5 ns and 1 ns respectively. The propagation delay of the flipflops may vary between 4 and 7 ns while the propagation delay of the gates may vary between 2 and 6 ns.

Calculate the minimum and the maximum propagation delays between C and U. Hence calculate the maximum frequency of the clock, C.



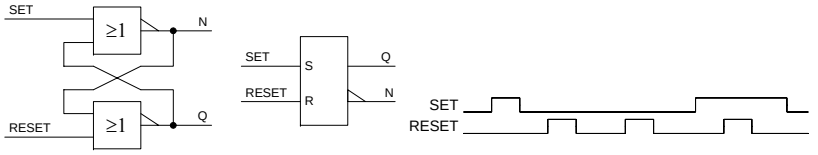
8C. In the six circuits below the setup and hold times of the flipflops are 5 ns and 1 ns respectively. The propagation delay of the flipflops may vary between 4 and 7 ns while the propagation delay of the gates may vary between 2 and 6 ns. The signal C is a symmetrical square wave.

Write down the setup and hold inequalities that relate to the ***second*** flipflop in each circuit. You should measure all times from the rising edge of CLOCK. Identify which of the circuits will not work reliably and determine the maximum clock frequency for each of the others.

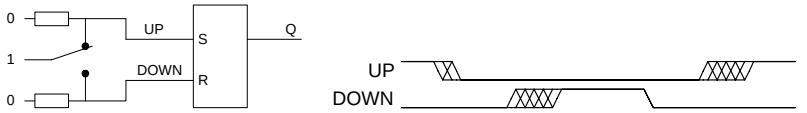


9B. The dual NOR-gate circuit shown below is called a Set-Reset latch and has the symbol shown at right. Complete the timing diagram by showing the waveforms of Q and N assuming that Q is initially low.

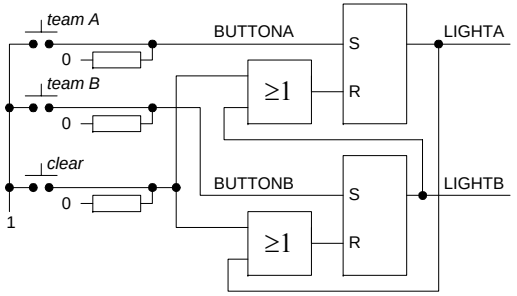
If SET and RESET are both high, say which one of these inputs dominates as far as Q is concerned and as far as N is concerned.



10A. The springing contacts in switches always bounce when they close and sometimes do so when they open as well. This contact bounce can last for several milliseconds. An SR-latch can be used to debounce switch signals in the following circuit. Complete the timing diagram by drawing the waveform of Q.



11C. The circuit shows a circuit to indicate who pressed their button first in a 2-contestant game show. Design a similar circuit for a 3-contestant game show. The SR-latches use the circuit from question 9.



E2.11/ISE2.22 – Digital Electronics II

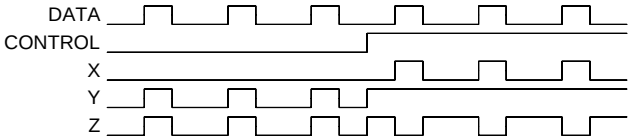
Solution Sheet 1

(Question ratings: A=Easy, …, E=Hard. All students should do questions rated A, B or C as a minimum)

1A. AND gate: 0 forces output low, 1 allows DATA through

OR gate: 0 allows DATA through, 1 forces output high

XOR gate: 0 allows DATA through, 1 inverts DATA



It is often useful to think of a gate like this: one input a signal, the others controlling it.

2B. P is high when an odd number of its inputs are high (an odd parity gate)

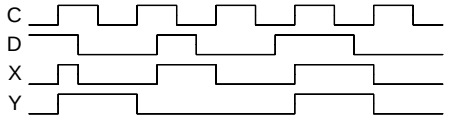
Q is low when all its inputs are the same

R is high when exactly one of its inputs is high

All of these properties are true of a 2-input XOR gate. Talking about a 3-input XOR gate (or larger) is ambiguous because no one can tell which of these three gates you mean.

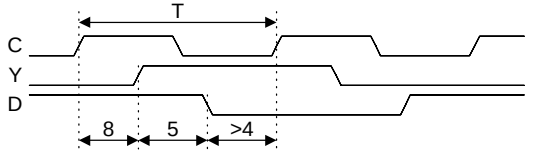
A	B	C	P	Q	R
0	0	0	0	0	0
0	0	1	1	1	1
0	1	0	1	1	1
0	1	1	0	1	0
1	0	0	1	1	1
1	0	1	0	1	0
1	1	0	0	1	0
1	1	1	1	0	0

3A. The latch output, X, follows D whenever C is high and freezes in its current state when C goes low. The flipflop output Y, only ever changes on the rising edge of C when it changes to the value that D has just prior to the edge.



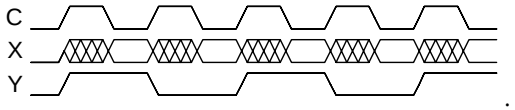
4B. From the diagram we obtain:

$$T-(5+8)>4 \;\Rightarrow\; T>17\,\text{ns} \;\Rightarrow\; f<58.8\,\text{MHz}$$



5B. Whenever C is high, the latch output, X, will follow its input: this means that we get a feedback loop containing an odd number of inverters. Such a loop will oscillate (indeed the oscillation frequency of such a loop is the standard way of measuring the propagation delay of a logic circuit). The only real use for this circuit is as a random number generator.

The flipflop circuit, Y, has no such problems because it only looks at its input for an instant and then effectively disconnects it until the next rising clock edge. It forms a ÷2 counter.



6B. This is a trick question: there is no propagation delay between A and any of P, Q or R since a transition in A does not directly cause any of these other signals to change. The min and max delays from A to S are 2 and 6 ns. Of course if R happens to be low, there is no propagation delay between A and S either.

The min and max delays from C to P, Q, R and S are 4 and 7, 6 and 13, 4 and 7, and 6 and 13 ns respectively. The important point to realise is that since a transition at Q does

not directly cause R to change, it follows that there is no delay path through both flipflops. The expression for a propagation delay **never** involves more than one flipflop delay.

7C. The shortest path from C to U passes through the flipflop and then through two gates: this gives a minimum propagation delay of 8 ns. This happens when A=1⇒P=0⇒R=S=1⇒U=!T=Q. We therefore use $2t_g$ in the hold inequality below.

The longest path from C to U passes through the flipflop and then through three gates: this gives a maximum propagation delay of 25 ns. This happens when A=0⇒P=1⇒R=!Q⇒T=1⇒U=!S=R=!Q. We therefore use $3t_g$ in the setup inequality below.

Setup: $t_p + 3t_g + t_s < T \Rightarrow T > 7 + 3\times 6 + 5 = 30\,\text{ns} \Rightarrow f < 33\,\text{MHz}$

Hold: $t_h < t_p + 2t_g \quad \Rightarrow \quad 1 < 4 + 2\times 2 = 8$ ✓

The hold inequality is always satisfied and the setup inequality gives a maximum clock frequency of 33 MHz.

8C. The setup inequality is given by

maximum delay to flipflop data input + setup time < minimum delay to flipflop clock ↑

Both delays must of course be measured from the same reference point: in this question, we are told to use the rising edge of C as our reference. We have to be a bit careful about which clock edge we are talking about. In parts (a), (b), (c) and (d) the first rising edge of C (at time 0) causes the output of the first flipflop to change and the **next** rising edge of C clocks the new data into the second flipflop. Thus, assuming the clock period to be *T*, the setup inequalities for these circuits are:

(a) $7+5 < T \Rightarrow T > 12 \Rightarrow f < 83$ MHz

(b) $7+6+5 < T+2 \Rightarrow T > 16 \Rightarrow f < 62.5$ MHz

(c) $6+7+5 < T \Rightarrow T > 18 \Rightarrow f < 55$ MHz

(d) $7+5 > T+2 \Rightarrow T > 10 \Rightarrow f < 100$ MHz

For part (e), the **falling** edge of C clocks the first flipflop and the following **rising** edge of C clocks the second one while for part (f) these rôles are reversed. This gives a ½*T* term on one side of the inequality and substantially slower clock speeds since the data must now reach the second flipflop in half a clock cycle rather than a whole one:

(e) $\frac{1}{2}T+6+7+5 < T \Rightarrow T > 36 \Rightarrow f < 28$ MHz

(f) $7+5 < \frac{1}{2}T+2 \Rightarrow T > 20 \Rightarrow f < 50$ MHz

The hold inequality is given by

maximum delay to to flipflop clock ↑ + hold time < minimum delay flipflop data input

This time though we are concerned with the clock edge that is meant to be clocking data into the second flipflop from the previous cycle. This means that for the normal shift register circuits of (a), (b), (c) and (d), the hold inequalities will not involve *T* at all:

(a) $0+1 < 4 \Rightarrow 1 < 4$ ☑

(b) $6+1 < 4+2 \Rightarrow 1 < 0$ ☒ ⇒ **Won't work**

(c) $0+1 < 2+4 \Rightarrow 1 < 6$ ☑

(d) $6+1 < 4 \Rightarrow 7 < 4$ ☒ ⇒ **Won't work**

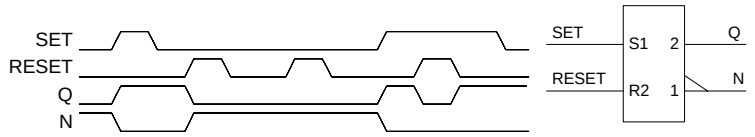
The moral is that if you clock both flipflops with the same clock edge you mustn't have any delay in the second flipflop's clock signal. Life is much easier with circuits (e) and (f) because the ½*T* that we lost from the setup equation reappears:

(e) $1 < \frac{1}{2}T+2+4 \Rightarrow \frac{1}{2}T > -5$ ☑

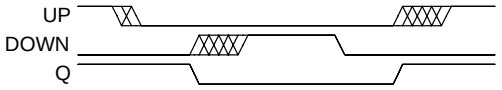
(f) $\frac{1}{2}T+6+1 < T+4 \Rightarrow \frac{1}{2}T > 3 \Rightarrow f < 167$ MHz (but also needs to be < 50 MHz above)

These circuits are used when transmitting information between circuit boards and in other situations where clock signal delays might arise.

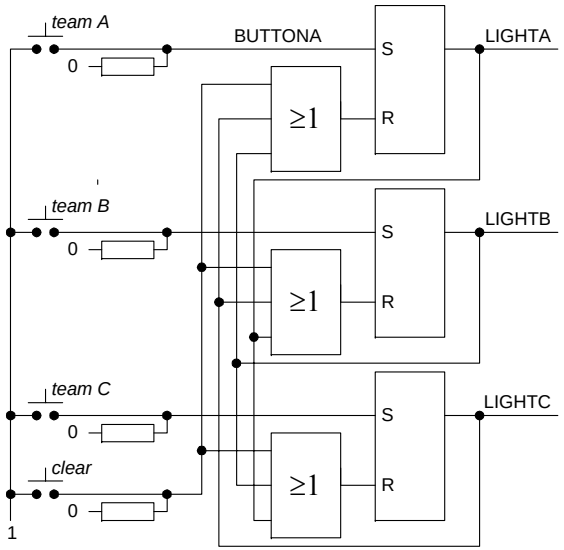
9B. If SET and RESET are both high then both outputs will be forced low. This means that SET wins as far as N is concerned and RESET wins as far as Q is concerned. This can be indicated in the logic symbol by labelling the inputs S1 and R2 and labelling each output with the identification number of the dominant input:



10A. Note that it is essential for the 2-way switch to be of the *break-before-make* variety to ensure that the latch inputs are never high simultaneously.



11C. The extension to an arbitrary number of contestants is easy: each latch must be held reset if any of the other contestants have their light on or if the CLEAR button is pressed. This circuit relies on the dominance of the RESET input referred to in question 9. In fact the OR gates that feed the reset inputs of the latches can be absorbed into the latch itself so we only need two gates per contestant.

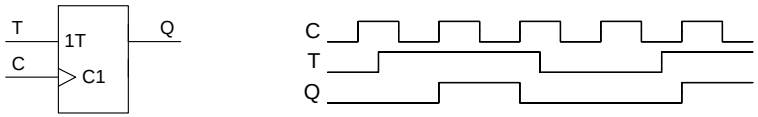


E2.11/ISE2.22 – Digital Electronics II

Problem Sheet 2

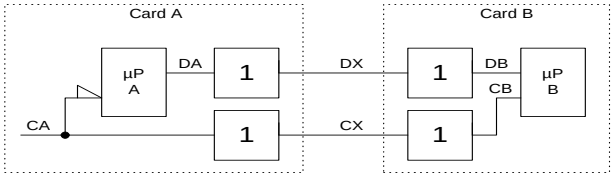
(Question ratings: A=Easy, …, E=Hard. All students should do questions rated A, B or C as a minimum)

1B. A toggle flipflop (T-flipflop) changes state whenever its T input is high on the CLOCK ↑ edge as shown in the timing diagram.

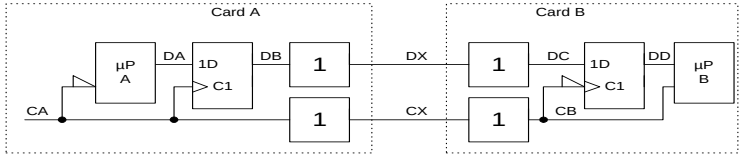


Show how a T-flipflop can be made by combining an XOR gate with a D-flipflop.

2C. A multi-processor system contains two microprocessors which are mounted on separate printed circuit cards. The clock and data signals pass through a line driver when they leave one card and a line receiver when they pass onto the next. The combined delay of the driver+receiver may vary between 13 ns and 22 ns. New data values appear at DA on the falling edge of CA with a propagation delay of 5 to 50 ns. Data is clocked into µP B on the rising edge of CB with a setup time of 12 ns and a hold time of 27 ns. If the clock, CA, is a symmetrical squarewave, calculate its maximum frequency.

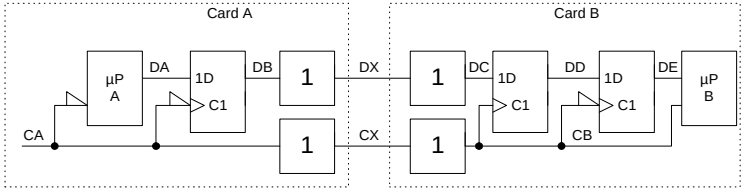


3C. We can speed up the circuit from the previous question by using high-speed flipflops with shorter propagation delays and setup times. The flipflops in the revised circuit have setup and hold times of 5 ns and 3 ns and propagation delays in the range 2 to 10 ns. Note that the second flipflop has an inverted clock. Calculate the new maximum clock frequency by considering its minimum period for each of µPA→flipflop, flipflop→flipflop and flipflop→µPB.



4C. By considering the Hold requirements, explain why the circuit in question 3 would not work if the two flipflops were interchanged.

5D. If we add a third flipflop, we can improve the speed further. Calculate the maximum clock frequency for the following circuit and explain in words how it has achieved the performance increase when compared with the original circuit of question 2.



6A. Explain why most memory integrated circuits have “tri-state” data output pins.

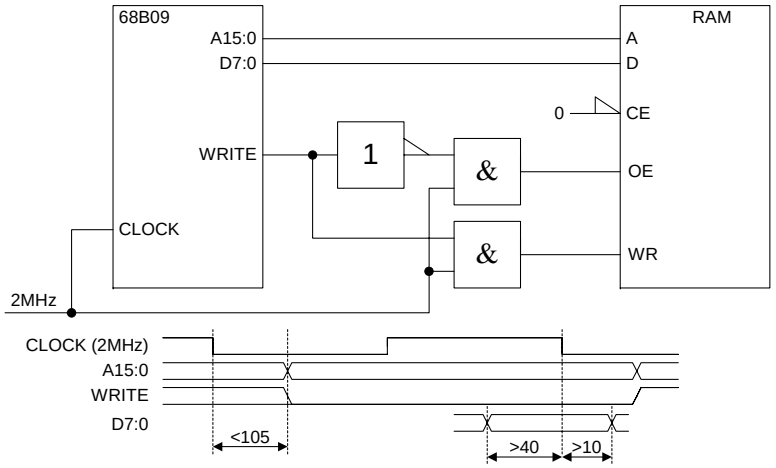
7B. In an 8-bit microprocessor system, addresses 0000 to 9FFF are occupied by RAM and addresses A000 to DFFF are occupied by ROM. The system also contains two peripheral devices: a serial port occupying addresses E100 to E107 and a parallel port occupying addresses E200 to E201. You have a supply of 8k×8 RAM integrated circuits and a supply of 16k×8 ROM integrated circuits.

a) State how many input address pins you would expect to find on each of the RAM integrated circuits, ecah of the ROM integrated circuits and on each of the peripheral device integrated circuits.

b) Derive Boolean expressions for the CE inputs of each memory and peripheral integrated circuit.

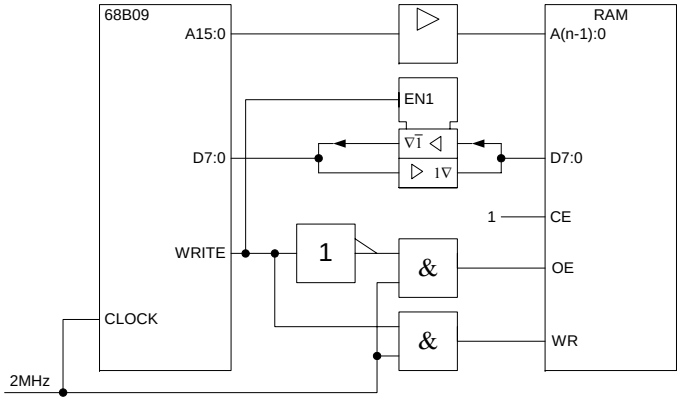
c) Say what is unusual about the byte ordering within the ROM.

8B. The diagram shows a Motorola 68B09 microprocessor connected to a memory circuit together with the timing diagram for a microprocessor read cycle..



Each logic gate has a propagation delay that may vary independently in the range 5 to 10 ns. Calculate the maximum permissible access times of the memory from (a) its address inputs, and (b) its OE input.

9C. The circuit of question 8 is altered by the introduction of buffers in the address lines and bi-directional buffers in the data lines.



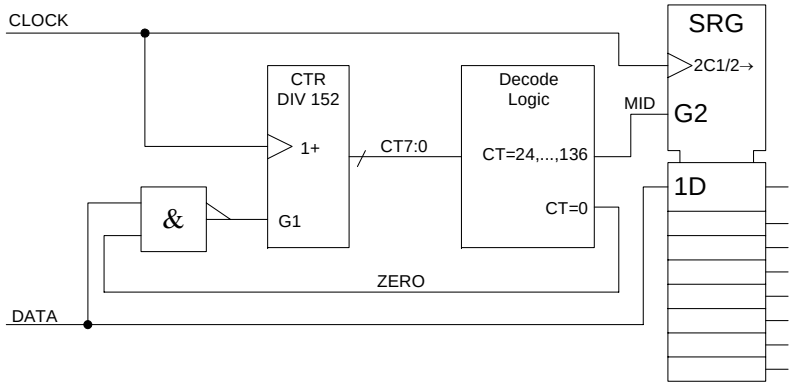
The address line buffers have a propagation delay of <18 ns while the data line buffers have a propagation delay of <12 ns, an enable time of <40 ns and a disable time of <25ns. The enable time applies when an output changes from a high impedance state to a driven state, the disable time applies when an output changes from a high impedance state.

Calculate the new values of the maximum permissible access times of the memory from (a) its address inputs, and (b) its OE input.

10A. Explain why a bi-directional buffer is normally designed to have a longer enable time than disable time.

11D. Buffers for address and data lines can be made faster if they have inverted outputs. Say how the operation of a microprocessor is affected if the address and data lines pass through inverting buffers between the microprocessor and (a) read-write RAM memory and (b) read-only ROM memory.

12B. Asynchronous bit serial data consisting of a start bit (logical 0), 8 data bits and a stop bit (logical 1) is received by the circuit below. The CLOCK frequency is 16× the transmission bit rate. Give a Boolean expression for the signal MID; simplify it where possible.



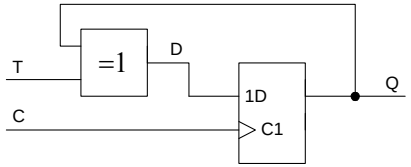
13C. In question 12, the CLOCK frequency is changed to 4× the transmission bit rate. Determine the appropriate counter division ratio and the counter values for which MID should now be high. Determine the clock accuracy required by the transmitter and receiver to ensure that the data is received correctly.

E2.11/ISE2.22 – Digital Electronics II

Solution Sheet 2

(Question ratings: A=Easy, …, E=Hard. All students should do questions rated A, B or C as a minimum)

1B. As seen in problem sheet 1, an XOR gate can be used to invert a signal or pass it through unchanged according to whether a control input is high or low.



2C. We define t=0 as the falling edge of CA.

Setup requirement: $\max(DB\uparrow\downarrow)+12 < \min(CB\uparrow)$
$50+22+12 < (13 + \frac{1}{2}T)$
$\frac{1}{2}T > 71 \Rightarrow f < 7$ MHz

Hold requirement: $\max(CB\uparrow) + 27 > \min(T + DB\uparrow\downarrow)$
$\frac{1}{2}T+22 + 27 > T + 5+13$
$\frac{1}{2}T > 31$ (less severe restriction than above)

Note the extra T term in the hold requirement: this is because we want the *second* transition of DB to occur >27 ns after CB↑. The Hold requirement is so easily satisfied that it wouldn't normally be necessary to calculate it exactly.

3C.

µPA → flipflop CA↓=0	Setup:	$\max(DA\uparrow\downarrow) +5 < \min(CA\uparrow)$ $50+5 < \frac{1}{2}T$ $\frac{1}{2}T > 55 \Rightarrow$ **f < 9 MHz**	Hold:	$\max(CA\uparrow)+3< \min(T+DA\uparrow\downarrow)$ $\frac{1}{2}T+3< T + 5$ $\frac{1}{2}T > -2$ ☑
flipflop → flipflop CA↑=0	Setup:	$\max(DB\uparrow\downarrow)+5 < \min(CB\downarrow)$ $10+22+5 < \frac{1}{2}T+13$ $\frac{1}{2}T > 24 \Rightarrow f < 21$ MHz	Hold:	$\max(CB\downarrow)+3< \min(T+DB\uparrow\downarrow)$ $\frac{1}{2}T+22+3< T +2+13$ $\frac{1}{2}T > 10 \Rightarrow f < 50$ MHz
flipflop → µPB CB↓=0	Setup:	$\max(FB\uparrow\downarrow)+12<\min(CB\uparrow)$ $10+12<\frac{1}{2}T$ $\frac{1}{2}T > 22 \Rightarrow f < 23$ MHz	Hold:	$\max(CB\uparrow)+27<\min(T+FB\uparrow\downarrow)$ $\frac{1}{2}T+27<T + 2$ $\frac{1}{2}T > 25 \Rightarrow f < 20$ MHz

It can be seen that the critical figure is the setup time for the first flipflop: this is because the microprocessor takes such a long time (up to 50 ns) to output its data. Question 4 (which doesn't work) and question 5 (which does) show how to relax this constraint. In the third row of the previous table, I have cancelled out the delay of the clock line driver/receiver from the two sides of the inequality. This is only valid if we can assume that the propagation delays for rising and falling edges are the same (not generally true).

4C. The first flipflop now responds to a falling clock edge: this means that µPA now has a full clock cycle to output its data rather than only a half cycle. We have therefore doubled maximum clock frequency of the circuit. (Note that the middle row of this table is unchanged from the previous question).

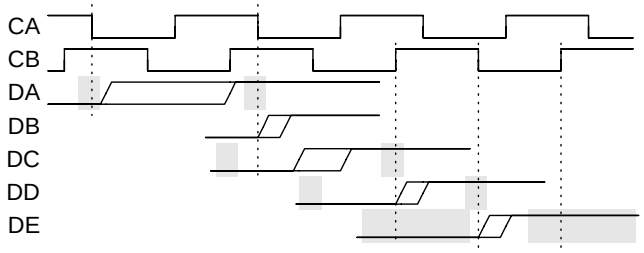
The problem is that the output from the second flipflop now changes on the rising clock edge and therefore fails to meet the hold time of µPB.

µPA → flipflop CA↓=0	Setup:	$\max(DA\uparrow\downarrow)+5<\min(T+CA\downarrow)$ $50+5<T$ $T > 55 \Rightarrow$ **f < 18 MHz**	Hold:	$\max(CA\downarrow)+3<\min(DA\uparrow\downarrow)$ $0+3<5$ ☑
flipflop → flipflop CA↓=0	Setup:	$\max(DC\uparrow\downarrow)+5<\min(CB\uparrow)$ $10+22+5 < \frac{1}{2}T+13$ $\frac{1}{2}T > 24 \Rightarrow f < 21$ MHz	Hold:	$\max(CB\uparrow)+3<\min(T+DC\uparrow\downarrow)$ $\frac{1}{2}T+22+3<T + 2+13$ $\frac{1}{2}T > 10 \Rightarrow f < 50$ MHz
flipflop → µPB CB↑=0	Setup:	$\max(DD\uparrow\downarrow)+12<\min(T+CB\uparrow)$ $10+12<T$ $T > 22 \Rightarrow f < 46$ MHz	Hold:	$\max(CB\uparrow)+27<\min(DD\uparrow\downarrow)$ **2 > 27** ☒

5D. We can fix the hold problem by adding a third flipflop. The last row of the previous table is now replaced by the two rows below and the maximum frequency is now 18 MHz.

flipflop → flipflop CB↑=0	Setup:	$\max(DD\uparrow\downarrow)+5<\min(CB\downarrow)$ $10+5<\frac{1}{2}T$ $\frac{1}{2}T > 15 \Rightarrow f < 33$ MHz	Hold:	$\max(CB\downarrow)+3<\min(T+DD\uparrow\downarrow)$ $\frac{1}{2}T+3<T + 2$ $\frac{1}{2}T > 1 \Rightarrow f < 500$ MHz
flipflop → µPB CB↓=0	Setup:	$\max(DE\uparrow\downarrow)+12<\min(CB\uparrow)$ $10+12<\frac{1}{2}T$ $\frac{1}{2}T > 22 \Rightarrow f < 23$ MHz	Hold:	$\max(CB\uparrow)+27<\min(T+DE\uparrow\downarrow)$ $\frac{1}{2}T+27<T + 2$ $\frac{1}{2}T > 25 \Rightarrow f < 20$ MHz

The timing of this circuit with a clock period of about 60 ns (16.7 MHz) is shown below with setup/hold windows shaded:



6A. Two reasons. Firstly many memories, though not all, use the same pins for data input as for data output: the outputs must therefore be turned off (or "tristated") to allow new data to be written into a memory location. Secondly, a large memory system contains several memory integrated circuits which are enabled one at a time according to the address range selected (as in question 7 below). The use of tri-state outputs allows all memory data lines to be connected together without the need for an external multiplexer to switch between them.

7B. a) RAM has 13 address inputs, ROM has 14, Serial Port has 3 and Parallel Port has 1.

b) We need 5 RAM chips and one ROM, serial and parallel chips. The CE inputs are given in the following table:

Chip	Address Range	CE
RAM	0000 – 1FFF	$\overline{A15}\cdot\overline{A14}\cdot\overline{A13}$
	2000 – 3FFF	$\overline{A15}\cdot\overline{A14}\cdot A13$
	4000 – 5FFF	$\overline{A15}\cdot A14\cdot\overline{A13}$
	6000 – 7FFF	$\overline{A15}\cdot A14\cdot A13$
	8000 – 9FFF	$A15\cdot\overline{A14}\cdot\overline{A13}$
ROM	A000 – DFFF	$A15\cdot\left(\overline{A14}\cdot A13+A14\cdot\overline{A13}\right)$
Serial	E100 – E107	$A15\cdot A14\cdot A13\cdot A8$
Parallel	E200 – E201	$A15\cdot A14\cdot A13\cdot A9$

Note that I have not included all the address lines needed to fully decode the Serial and Parallel port adress ranges. The microprocessor should never access the undefined memory locations in the range E000 to FFFF so it does not matter if the peripheral ports respond to several of them. As defined above, the following addresses will all refer to the serial port's lowest location: E100, E108, E110, E118, …, FFF8. Of the 16 address lines, 3 are direct inputs to the serial port, 4 are used in forming its CE and 9 are unused. The 9 unused lines can take on 512 possible values and so the serial port will appear 512 times in the memory map. If a PAL is being used to generate the CE signals, then the number of PAL inputs required may be reduced by not decoding peripheral address ranges fully.

c) The bytes are in the wrong order in the ROM. The ROM has 14 address inputs, namely A13:0. Addresses A000 to BFFF have A13=1 and will therefore be mapped to the second half of the ROM; addresses C000 to DFFF have A13=0 and will be mapped to the first half of the ROM. This situation could be corrected by inverting A13 before connecting it to the ROM but this would add delay: a neater solution is to use A14 as the most significant address bit rather than A13.

8B. Note that only the setup time matters for this question. Let A be the access time (or propagation delay) from the address inputs and E be the access time from the OE input. The clock period is 500 ns. P is the propagation delay of a gate (i.e. 5 to 10 ns)

$\max(105+A)+40<500 \Rightarrow$ **A < 355 ns**

$\max(P+E)+40<250 \Rightarrow$ **E < 200 ns** (taking P=10, its maximum value)

9C. We now have three possible paths to consider:

A15:0→Mem→D7:0	$\max(105+18+A+12)+40<500 \Rightarrow$ **A < 325 ns**
CLOCK↑→Mem:OE→D7:0	$\max(P+E+12)+40<250 \Rightarrow$ **E < 188 ns**
WRITE→Buff:EN1→D7:0	$\max(105+40)+40<500 \Rightarrow$ **185>500** ☑

Note that in the symbol for the bidirectional buffer: the ∇ denotes tristate outputs: the output marked with a 1 is enabled when EN1 is high while the output marked with a $\overline{1}$ is enabled when EN1 is low. The ∇ always goes immediately next to the output pin. The symbol ▷ denotes a buffer: a gate with a higher than normal output current capability. Any signals that flow right to left instead of the more normal left to right must be marked with a ←.

10A. If the same wire is driven high by the output of one device and low by that of another, a high current will flow which will waste power and which may even damage the integrated circuits involved. To reduce the possibility of two devices trying to drive the same wire simultaneously, tristate outputs are almost always designed to turn off more quickly than they turn on.

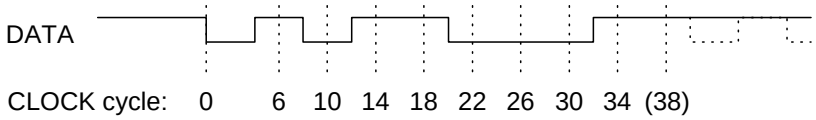
11D. The effect of inverting the address lines is to subtract them from FFFF. Thus what were memory locations 0000, 0001 and 0002 now become FFFF, FFFE and FFFD. Providing the chip enables are generated correctly, this reversal does not matter at all for a RAM: as long as each distinct address refers to a unique memory location the microprocessor does not care where it is inside the chip. For a ROM, the contents must be preprogrammed in the correct locations: thus the program would need to be stored backwards.

12B. The counter counts up from 0 to 151.

MID=CT3•!CT2•!CT1•!CT0 will get all odd multiples of 8, i.e. 8, 24, …, 136.

To eliminate the first of these we make MID=(CT7+CT6+CT5+CT4)•CT3•!CT2•!CT1•!CT0

13C. The counter should now divide by 38:



The eighth bit will now be clocked in at the end of the clock cycle in which CT5:0 equals 34. This is at time 34P+t from the beginning of the START bit where 0<t<P and P is the receiver clock period. This time must be in the range 8T to 9T where T is the duration of a bit cell. Thus

$$8T<34P+0 \Rightarrow T/P<4.25$$
$$9T>34P+P \Rightarrow T/P>3.89$$

If we assume that T and P have nominal values of T_0 and P_0 with a fractional tolerance of $\pm x$, we have

$$T_0(1-x) < T < T_0(1+x) \quad \text{and} \quad P_0(1-x) < P < P_0(1+x)$$

By considering the maximum and minimum possible values of the ratio T/P, we get:

$$\frac{T_0(1+x)}{P_0(1-x)} = 4.25 \quad \text{and} \quad \frac{T_0(1-x)}{P_0(1+x)} = 3.89$$

From which

$$\frac{(1+x)^2}{(1-x)^2} = \frac{4.25}{3.89} \quad \Rightarrow \quad (1+x) = 1.045(1-x) \quad \Rightarrow \quad x = \frac{0.045}{2.045} = 2.2\%$$

Substituting this value for x into the original equations gives $T_0/P_0 = 4.07$

Hence both the transmit and receive clocks must have a ratio of 4.07 and a tolerance of ±2.2%.This requirement is slightly more stringent than for the more usual 16× clock.

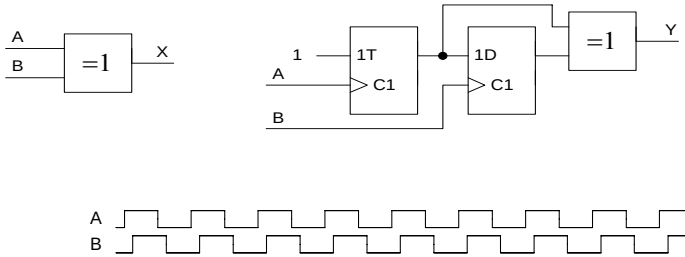
E2.11/ISE2.22 – Digital Electronics II

Problem Sheet 3

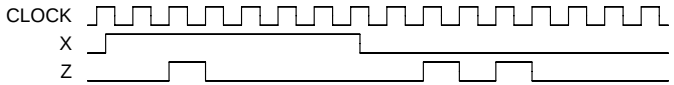
(Question ratings: A=Easy, …, E=Hard. All students should do questions rated A, B or C as a minimum)

1B. Q2:0 is the output of a 3-bit binary counter whose input is a constant frequency squarewave, CLOCK. Give a Boolean expression for Z in terms of Q2:0 such that Z is high whenever Q2:0 has the value 6. Draw a timing diagram showing the waveforms of CLOCK and Z and the value of Q2:0 during each clock cycle. Indicate on your diagram where glitches might occur in Z.

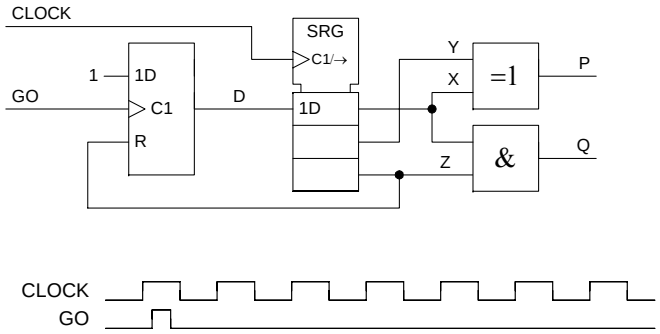
2C. The diagram shows two *phase-detector* circuits. Inputs A and B are symmetrical squarewaves with the same frequency but differing phases. Complete the timing diagram by showing the waveforms of X and Y for the case when B lags A by 45°. If logical 0 and 1 correspond to 0 V and 5 V respectively, sketch graphs showing how the DC components (i.e. average values) of X and Y vary with the phase difference.



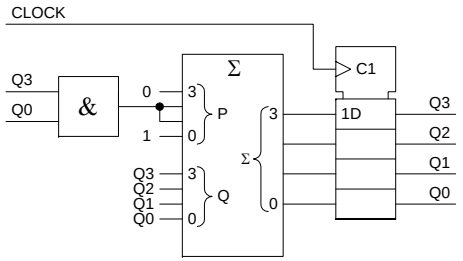
3B. The signal X forms the input to a shift register that is clocked by CLOCK↑. As shown in the timing diagram, the signal Z gives one pulse when X goes high and two pulses when it returns low. If the successive outputs from the shift register are A, B, C, … derive a Boolean expression for Z.



4B. Complete the timing diagram by drawing the waveform of P and Q. Explain why only one of these signals is certain to be glitch-free. If the GO pulse occurs at a random time with respect to the CLOCK, determine the average time delay in CLOCK periods between the GO↑ edge and the Q↑ edge.

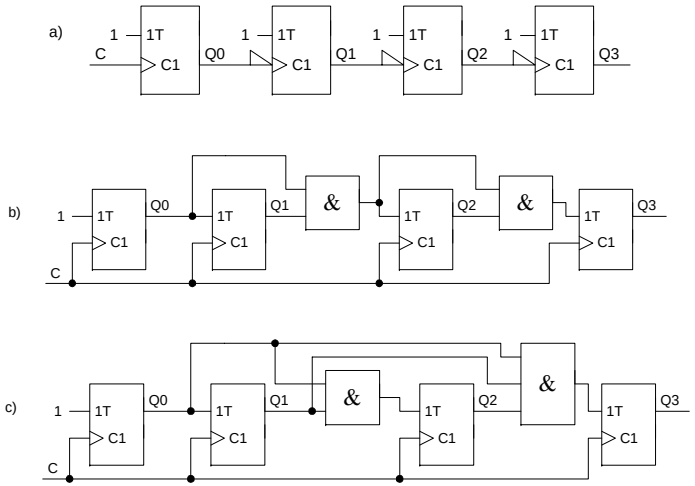


5C. The diagram shows an AND gate, a 4-bit register and an adder connected together to form a counter. List the values taken by the P input of the adder for all possible values of Q3:0. Draw a state diagram showing the sequence of values taken by Q3:0 on successive CLOCK pulses.



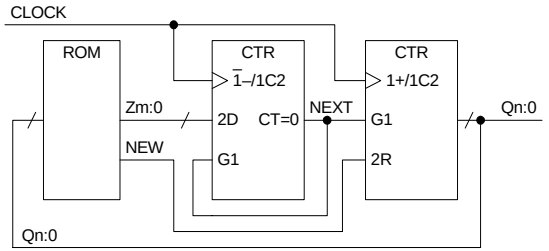
6C. Modify the above circuit so that it follows the count sequence 1, 2, 3, …, 9, 10, 1, 2, 3, …. Draw a state diagram for your revised circuit.

7C. In the following counter circuits, the propagation delays of gates and flipflops are 5 and 10 ns respectively and the setup time for the flipflops is 2 ns. In addition, a flipflop clock input must stay high for at least 5 ns and low for at least 2 ns. For each circuit, calculate the maximum clock frequency and the maximum propagation delay from the C input to any of the Qn outputs. Say what your answers would be for 32-bit counters designed in the same manner. Note that for design (c), each successive AND gate has one additional input.



8D. The diagram shows the circuit for a programmable pulse generator consisting of a read-only memory (ROM) and two counters of length m+1 and n+1 bits respectively. On the CLOCK↑ the leftmost counter is loaded with the value Z if NEXT=1 and counts down if NEXT=0 while the rightmost counter counts up if NEW=0 and is reset to zero if NEW=1. When the contents of the leftmost counter equal zero, its CT=0 output goes high. Determine the waveform of Q0 if the ROM contents are:

Q1:0	Z2:0	NEW
0	5	0
1	0	0
2	3	0
3	2	1

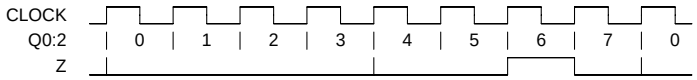


E2.11/ISE2.22 – Digital Electronics II

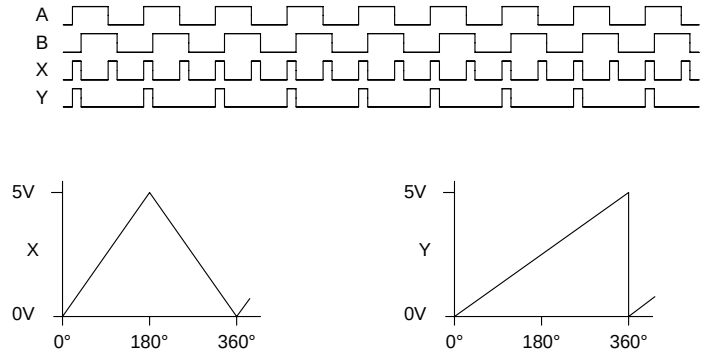
Solution Sheet 3

(Question ratings: A=Easy, …, E=Hard. All students should do questions rated A, B or C as a minimum)

1B. Z = Q2•Q1•!Q0. Note that (a) Q2 is always the MSB and (b) we must include the !Q0 term. Glitches in Z are possible for the transitions 3→4 and 7→0.



2C. The XOR gate goes high twice per cycle whereas the more complicated circuit only goes high once per cycle. The advantage of the complicated circuit is that it covers a full 360° monotonically.

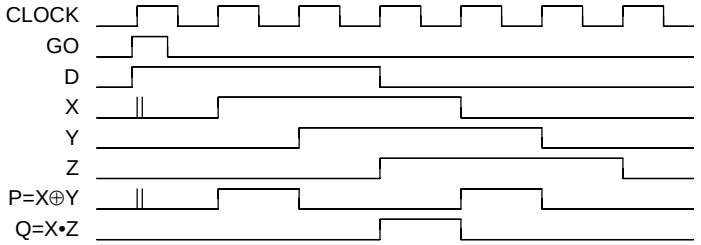


3B. Z = B ⊕ C + !D • E

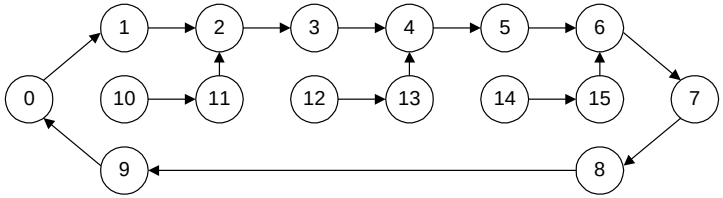
Note that since this expression does not involve A, it will be glitch-free.

4C. The output of the first shift-register stage can go metastable if D↑ occurs just before the CLOCK↑ edge. This will only affect the P output because Z will be low at the time which will force Q low regardless of X.

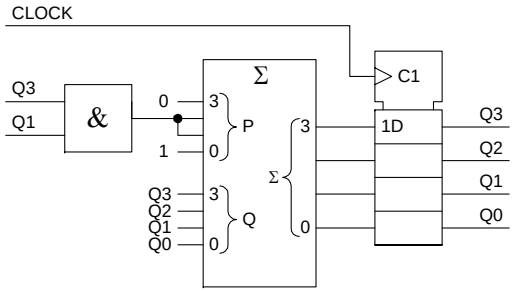
The average time delay between GO↑ and Q↑ will be 2½ clock periods.

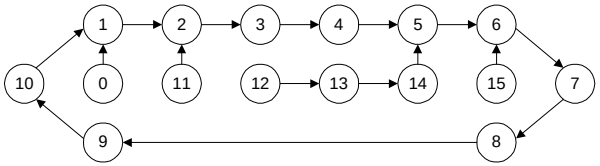


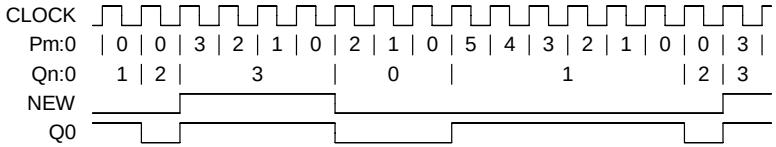
5C. The P input of the adder equals 7 when Q is 9, 11, 13 or 15. For all other values of Q it equals 1. Bearing in mind that the adder result is modulo 16 (i.e. 10+7=1), this results in the following state diagram:



6C. We want to make 10 the maximum count rather than 9, so we need to detect when Q3 and Q1 are high. We will now add 7 onto Q in states 10, 11, 14 and 15.







7C. For the 4-bit counters shown:

(a) Frequency limited only by the CLOCK constraints: 1/7ns =143 MHz (or 100 Hz if it must be symmetrical). The worst-case propagation delay is the change from 1111 to 0000 which takes 40 ns.

(b) $T - (10+5+5) > 2 \Rightarrow T > 22 \Rightarrow f < 45$ MHz. Delay = 10 ns.

(c) $T - (10+5) > 2 \Rightarrow T > 17 \Rightarrow f < 59$ MHz. Delay = 10 ns.

For a 32-bit counter:

(a) Worst-case delay increases to 320 ns

(b) We now have $T > 12 + 30 \times 5 = 162 \Rightarrow f < 6$ MHz.

(c) Unchanged.

The ripple counter (a) has the highest clock speed but the longest propagation delay. As the counter length increases, the propagation delay of design (a) increases while the maximum clock frequency of design (b) decreases. Design (c) maintains its high performance regardless of counter length but requires some very large gates.

8D. The Q0 output goes alternately high and low for a length of time determined by the leftmost counter. Thus the ROM output Zm:0 that is associated with each value of Qn:0 is one less than the length of the next count sequence. When NEW is high, the whole sequence starts again from Q=0. For the ROM contents given in the question, we get a total cycle length of 14 clock cycles.

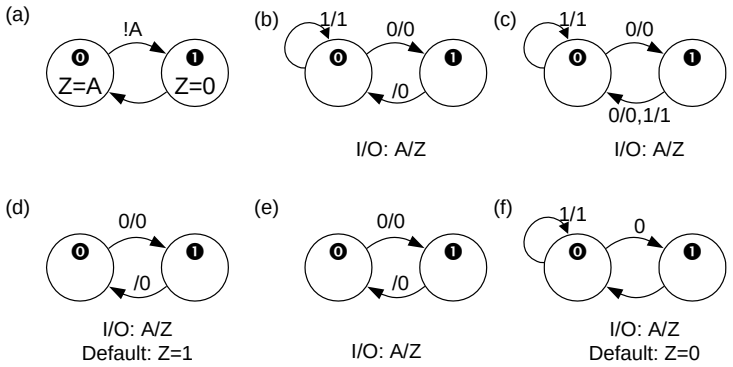
In the timing diagram, Pm:0 is the contents of the leftmost counter.

E2.11/ISE2.22 – Digital Electronics II

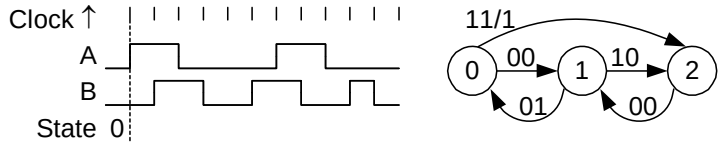
Problem Sheet 4

(Question ratings: A=Easy, …, E=Hard. All students should do questions rated A, B or C as a minimum)

1B. Say which of the following state diagrams denote the same state machine as version (a). Where an arrow is marked 0/1, for example, it means when A=0, the output Z will be 1 and the transition will be taken at the next CLOCK rising edge.



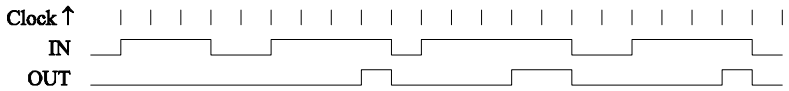
2C. The state diagram and input waveforms of a state machine are shown below. All input and state transitions occur shortly after the clock rising edge. Complete the timing diagram by indicating the value of the state during each clock cycle and by drawing the waveform of X. The initial state is 0 as shown.



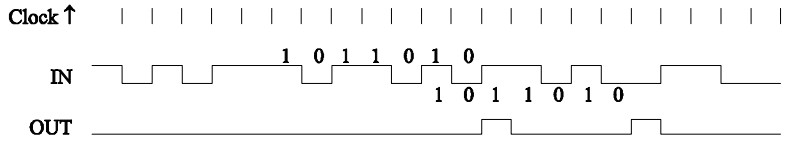
I/O Signals: A,B/X Default: X=0

3B. A synchronous state machine has its state represented by the 2-bit number S1:0 and has a single input signal DIR. The current state is stored in a D-type register whose input NS1:0 is defined by: $NS1 = S0 \oplus DIR$ and $NS0 = \overline{S1 \oplus DIR}$. Draw the state diagram for the state machine.

4C. Draw the state diagram for a state machine whose output goes high when the input is high for four or more clock cycles. As shown in the timing diagram, the output should go high during the fourth clock cycle and remain high so long as the input does. Input and state transitions occur shortly after the clock rising edge.



5D. Draw the state diagram for a state machine whose output goes high during the clock cycle following the reception of the input sequence 1011010. The trigger sequences can overlap as in the example below. Indicate the sequence of states followed by your design.for the input sequence given below.

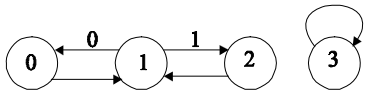


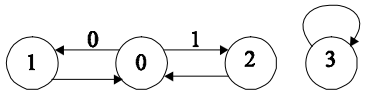
6C. A counter is required that follows the sequence 1, 2, 3, 1, 2, 3, …. Design a state machine to follow this sequence using D-type flipflops and as few gates as possible. You should ensure that the counter will reach the desired sequence regardless of its initial state.

7B. State the circumstances under which an input to a state machine should be passed through a register before going to the logic that generates the next-state bits.

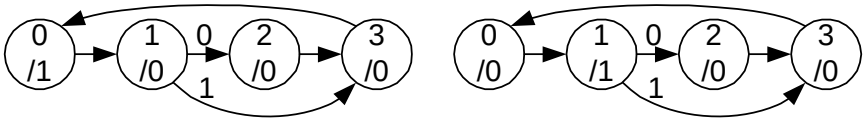
State the circumstances under which an output from a state machine should be passed through a register before being used elsewhere in a circuit.

8C. Two possible numberings for a state machine are shown below. Explain why it is essential for the input to be synchronised with the clock in one case but not in the other.

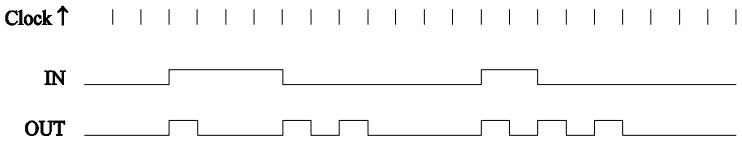




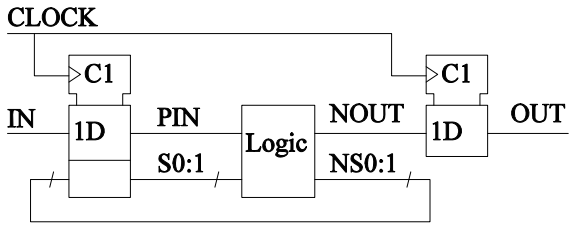
9C. Show that for one of the state machines shown below it is possible to renumber the states to avoid output glitches but that this is not possible for the other one. Assume the input is synchronized with the clock.

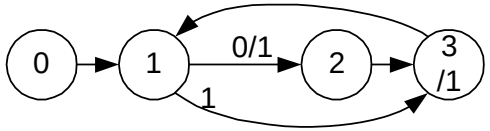


10C. Construct the state diagram for a state machine that emits a single pulse on each rising edge of its input and a double pulse on each falling edge as shown below. Each output pulse should last exactly one clock cycle. Assume that the input signal has been synchronized with the clock rising edge. How does your design react to an input signal that goes low for less than four clock cycles?

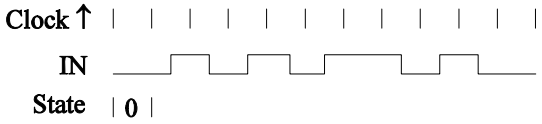


11C. In the state machine illustrated below, the contents of the logic block are defined by: $NS1 = S1 \oplus S0$, $NS0 = PIN + S1 + \overline{S0}$, $NOUT = \overline{PIN} \cdot S0 + S1 \cdot S0$ which gives the state diagram shown. Transitions of the input signal IN occur on the *falling* edge of the clock. Complete the timing diagram by indicating the sequence of states and the signals PIN, NOUT and OUT.

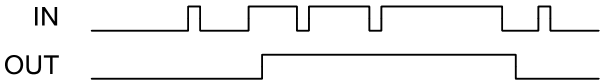




I/O Signals: PIN/POUT
Default: NOUT=0



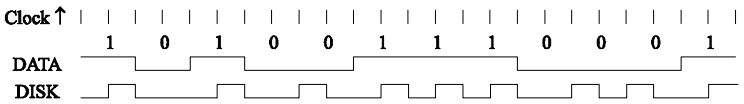
12D. In the notes (page 3.35) the "noise pulse eliminator" is designed as a Moore machine and introduces a two-cycle delay in the output. Show that if it is designed as a Mealy machine it will only require three states and will introduce only one cycle delay as shown (all transitions occur on the rising clock edge):



Design the state machine and give Boolean expressions for the outputs of the logic block..

13E. Data is recorded onto floppy disks in a Modified form of Frequency Modulation known as MFM in which each bit of data is recorded as a pair of bits on the disk.. A logical 1 is always recorded as 01 whereas a logical 0 is recorded as either 10 or 00 according to whether the previous bit was 0 or 1. This recording scheme ensures that successive 1's on the disk are separated by between one and three 0's. The timing diagram shows the input (DATA) and the recorded bit-pairs (DISK) for the sequence 101001110001.

Design the state diagram of a state machine which converts an input stream of data bits into the bit-pairs that must be recorded onto the disk as shown in the timing diagram below. Each DATA bit lasts for two clock cycles and all state and input transitions occur shortly after the clock's rising edge. If possible, your design should function correctly regardless of its initial state.



E2.11/ISE2.22 – Digital Electronics II

Solution Sheet 4

(Question ratings: A=Easy, …, E=Hard. All students should do questions rated A, B or C as a minimum)

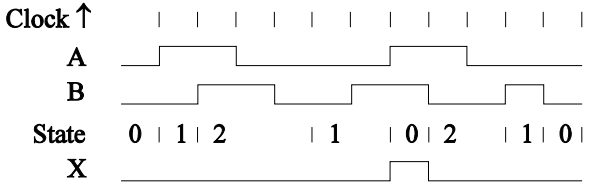
1B. In comparing state diagrams, you should first check the transitions and then check that the outputs are the same in each state. It can be seen that the transitions are the same for all the versions. However the outputs are incorrect in version (c) and version (e).

When output are marked on arrows they refer to the state from which the arrows originate. In version (c) therefore, state 1 has an output Z=A instead of Z=0 as it should be.

In version (e), the output is not specified for the case A=1 in state 0. It must either be specified by default as in version (d) or else explicitly as in version (b).

2C. You should first determine the state sequence. The transitions depend on the value of A and B immediately *before* the Clock↑ edge. A common mistake is to use the values *after* the edge.

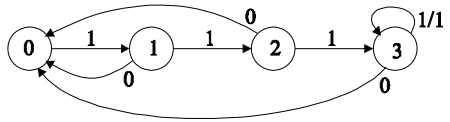
Note that X is only ever high in state 0 and then only if A and B are high. A common mistake is to make X high in state 2 rather than state 0: remember that outputs on transition arrows refer to the preceding state.

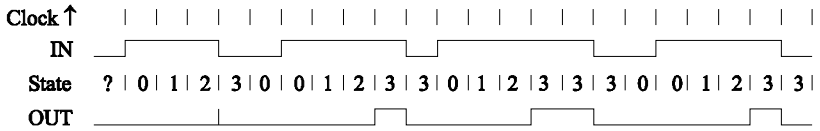


3B. This represents a 2-bit bidirectional counter whose counting sequence has only one bit changing at a time. This unit-distance property means that you can decode the outputs without risk of glitches.

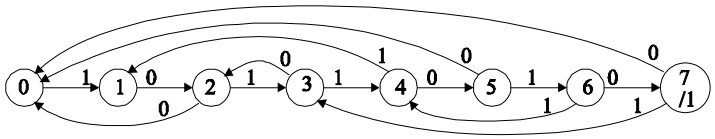
DIR	S1	S0	NS1	NS0
0	0	0	0	1
0	0	1	1	1
0	1	0	0	0
0	1	1	1	0
1	0	0	1	0
1	0	1	0	0
1	1	0	1	1
1	1	1	0	1

4C. Since the output must go high during the fourth clock cycle in response to the value in that cycle, we must have a Mealy machine: a Moore machine would insert too much delay. If IN=1 during the current cycle then we want OUT=1 if the previous three (or more) cycles had IN=1. We therefore need to remember how many of the previous cycles had IN=1: 0, 1, 2 or ≥3. We therefore need four states. Note that an unavoidable glitch possibility exists if IN goes high for three clock cycles; this can only be eliminated reliably by delaying the output for an extra cycle.

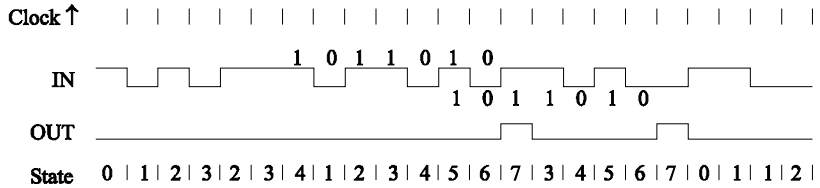




5D. The previous question could be regarded as recognising the sequence 1111. This question is pretty similar but with a different pattern to recognise. There are two significant differences. Firstly, when an input bit does not conform to the required sequence, we cannot always just branch back to state 0; the last few bits of the rejected input sequence may be the first few bits of the correct one. Secondly, the output must go high during the cycle *following* the trigger sequence; this requires an extra state at the end and allows us to use a Moore machine.



I/O Signals: IN/OUT Default: OUT=0

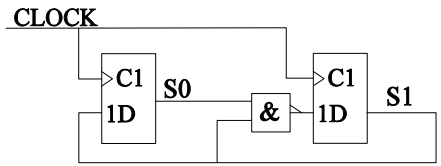


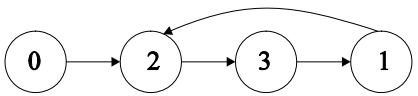
6C. The following table therefore lists both the value of the next state (NS1:0):

S1	S0	NS1	NS0
0	0	X	X
0	1	1	0
1	1	0	1
1	0	1	1

Choosing the "don't care" entries to simplify the expressions we get:

D flipflop inputs: $NS1 = \overline{S1} + \overline{S0}$ $NS0 = S1$





If our flipflops possess set inputs, we don't require the gate. We can avoid state 0 either by forcing S0 high whenever S1 is low or by forcing S1 high whenever S0 is low. The former approach does not work because the set input to S0 will not be released in time when the state machine goes from state 1 to state 2. We can redraw our table with the assumption that S1 is forced high whenever S0 is low; this means that the S1 flipflop's data input can be "don't care" whenever NS0 is equal to 0:

S1	S0	NS1	NS0
0	0	X	X
0	1	X	0
1	1	0	1
1	0	1	1

Choosing the "don't care" entries to simplify the expressions we get:

D flipflop inputs: $NS1 = \overline{S0}$ $NS0 = S1$

7B. Any inputs that are not already synchronized with the clock must be passed through a register before going to the next-state logic. The only exception to this is if the level of a particular input only ever selects between two states whose state numbers differ in a single bit position.

Any output that is prone to glitches must be passed through a register before being connected to a clock, set or reset input of a subsequent circuit either directly or via combinational logic. Another way of putting this is that a glitch-prone output should not be connected to a glitch-sensitive input.

If ROM or RAM is used for the combinational logic then all outputs are glitch-prone. If hazard-free combinational logic is used then glitches are possible if an input depends on two or more inputs/state-bits that can change simultaneously and if any of their 2^n possible combinations would cause the output to change. Another way of looking at this is that since absolute simultaneity is impossible, any inputs to the combinational logic that change together might in fact change in any conceivable sequence; an output is glitch-prone if any of these sequences would cause it to change.

8C. In the rightmost diagram, the input signal selects between states 1 and 2 (01 and 10 in binary); it thus causes both state bits to change. If the input changes just before the clock edge, both inputs to the state register will be changing within the setup-hold window and may end up taking any value. In particular the circuit may end up in state 3 whence it returneth not.

In the leftmost state diagram, the input signal selects between states 0 and 2 (00 and 10 in binary). The LSB is 0 in both cases and so the state machine cannot possibly go to any state other than these two.

9C. We can assume without any loss of generality that the state in which the output is high is numbered 3 (we can always invert one or both of the state bits to make this true). The output will therefore be generated by means of an AND gate. The AND gate output will be prone to glitches if its two inputs (S0 and S1) ever change in opposite directions simultaneously; this is because there will always be a chance that they may briefly be high together. S0 and S1 change simultaneously if and only if we ever have a transition from 1 to 2 (01 to 10) or from 2 to 1 (10 to 01).

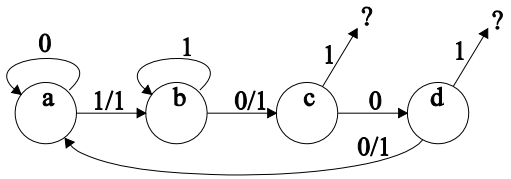
In the leftmost state diagram, the three states in which the output is 0 are all mutually connected and so whichever two of them are numbered 1 and 2, there must be a transition joining them and hence a possibility of a glitch.

In the rightmost state diagram we can number the states from left to right 2,3,1,0. There is now no direct link between 1 and 2 and no glitch is possible on the output. In general, any numbering scheme will work in which the second and fourth states from the left have numbers adding up to 3.

Note that even with the leftmost diagram, we can suppress the potential glitch by numbering the states 3,1,2,0 from the left and then inserting a delay in the S1 output of the state register. This delay will ensure that S1 effectively changes later than S0 and that the dangerous transition follows the sequence $1 \rightarrow 0 \rightarrow 2$ without any possibility of passing through state 3.

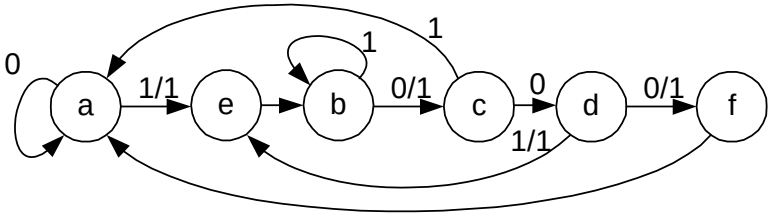
Note too that we can also make the left diagram work by numbering the states 4, 1, 2, 3 from the left and ensuring that the output is high only in state 4 (out of the possible 8 states). This now requires 3 state bits which is inefficient but none of the transitions between states 1, 2 and 3 can generate a high output since the most significant state bit will remain low.

10C. The basic diagram is shown below; note that we *must* use a Mealy machine in order to get zero delay between IN and OUT. The only two points of difficulty is what to do if the input goes high in the middle of the double pulse sequence and whether we wish to ensure that consecutive pulses are separated by at least one clock cycle.



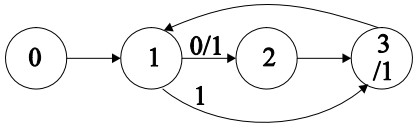
I/O Signals: IN/OUT
Defaults: OUT=0

The following diagram ensures that pulses are distinct (by the addition of states e and f) and abandons pulse sequences when another input transition occurs:

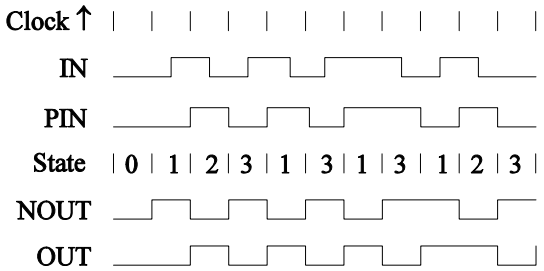


I/O Signals: IN/OUT Default: OUT=0

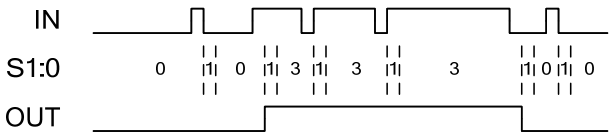
11C.



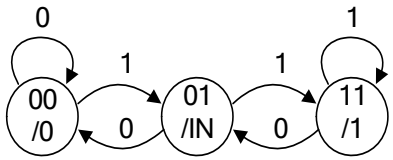
I/O Signals: PIN/NOUT
Default: NOUT=0



12D. We design the state diagram in the same way as in the lecture notes. Now however, the output in state 1 has to depend on IN (and therefore the circuit must be a Mealy machine); this is illustrated in the first two occurrences of state 1 in the timing diagrm.



We can now draw the state diagram:



State Numbers: S1,S0
Inputs/Outputs: IN/OUT

We now make a Karnaugh map for the three outputs: NS1,NS0/OUT. The last row of the K-map is all "don't care".

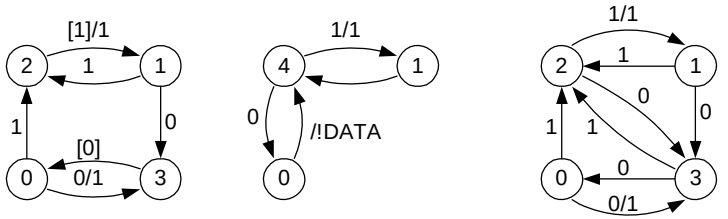
S1	S0	IN=0	IN=1
0	0	00/0	01/0
0	1	00/0	11/1
1	1	01/1	11/1
1	0	X**X**/**X**	X**X**/X

Choosing the "don't care" entries to simplify the expressions we get:

NS1=S0.IN, NS0=S1+IN, OUT=S1 + S0.IN = S1 + NS1

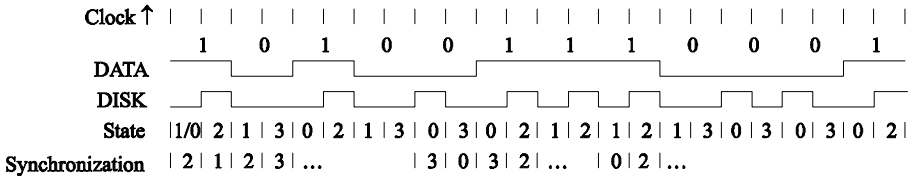
The X's in the final row of the state table are bold where these expressions are true. Note that the final row always branches to state 1 and so we can never get trapped in state 2.

13E. The starting point is the leftmost state diagram. Here states 0 and 1 are occupied during the first half of each input bit and 2 and 3 are occupied during the second half. Since the input data should not change between the two halves, we do not really need to check its value in states 2 and 3; I have therefore put the branch conditions in brackets. The top two states correspond to DATA=1 while the lower two correspond to DATA=0.



I/O Signals: DATA/DISK; DEFAULT: DISK=0

If we assume that DATA is correct in states 2 and 3, we can merge them to form state 4 as in the second diagram. A better solution is to branch 2→3 or 3→2 if DATA is incorrect as in the third diagram. This will resynchronise the circuit and make if function correctly regardless of its initial state. The figure shows several examples of this.



E2.11/ISE2.22 – Digital Electronics II

Problem Sheet 5

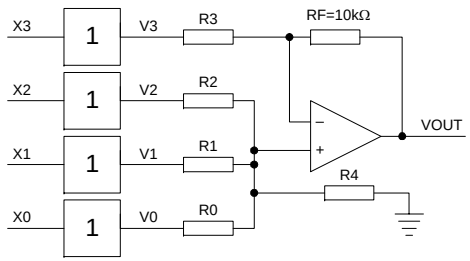
(Question ratings: A=Easy, …, E=Hard. All students should do questions rated A, B or C as a minimum)

1B. A 3½ digit Digital Voltmeter has a display range of ±1999 and an accuracy of ±2 on the display. How many bits would a binary A/D converter need to have for its ±0.5 LSB accuracy to be as good as that of the DVM?

2B. A 12-bit converter has a resolution of 1 mV (i.e. 1 LSB = 1 mV) and input voltages in the range ±0.5 mV are converted to the value 0. What range of input voltages will be converted to –2047 ?

3B. A 10-bit converter converts an input voltage x to the value floor(x / 10mV). If 1 V < x < 8 V, what range of output values will be obtained ?

4C. X3:0 is a 4-bit signed number whose value, X, lies in the range –8 to +7. If the logic levels of V3:0 are 0 V and +5 V, choose values for R0 to R4 so that VOUT is equal to X/8 volts.



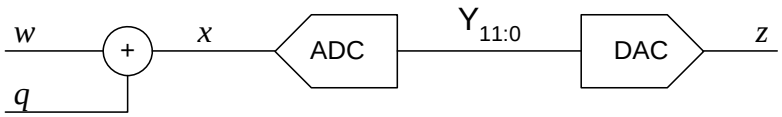
5C. The composite video signal to drive a monochrome TV monitor takes one of three different voltages according to the values of two digital signals DATA and SYNC:

DATA	SYNC	V_{OUT}
0	0	0.0
1	0	0.7
0	1	–0.3
1	1	Don't Care

Design a circuit to generate VOUT having a 50 Ω output impedance. You may assume that output logic levels are 0 and 5 V and that +5 V and –5 V power supplies are available should you need them. You do not need any op-amps although you will need at least one logic gate.

6B. Signals on a compact disc are stored as sequences of 16-bit numbers. Determine the maximum undistorted signal-to-noise ratio obtainable for a music signal whose peak amplitude is 10 times as great as its RMS value.

7B. Traingular pdf dither, q, of amplitude ±1 mV is added to an input signal, w, before conversion to a 12 bit number Y11:0. This is then sent to a DAC to generate an output voltage z. If all voltages are measured in mV then $z = \text{round}(w+q)$ and the pdf of q is equal to $p(q) = 1 - |q|$ for $|q| < 1$.



(a) Assuming that $|w| < 0.5$, show that the probability that $z = -1$ is given by $pr(z=-1) = 0.125 \times (2w-1)^2$.

(b) Derive similar expressions for $pr(z=0)$ and $pr(z=+1)$ for $|w| < 0.5$.

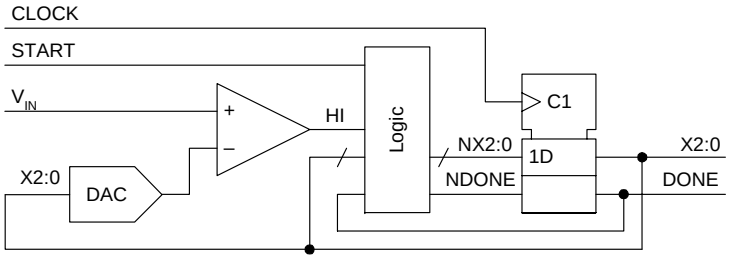
(c) Determine the mean and variance of z in terms of w.

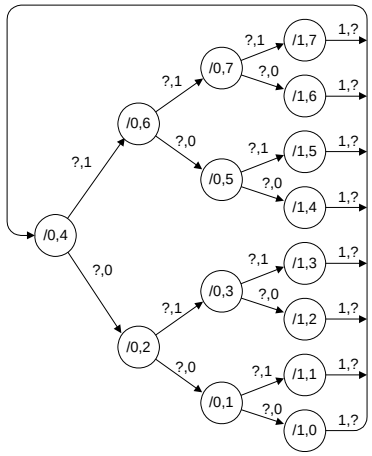
8C. A sample-hold circuit is used to store the input voltage of a 12-bit A/D converter during each conversion. The sample-hold circuit has an aperture uncertainty of 5 ns and a leakage current of ±1 nA. The A/D converter has an input voltage range of ±10 V.

If the input voltage is a sine wave of amplitude 10 V, calculate the input frequency at which the aperture uncertainty will result in an error of ±0.5 LSB [surprisingly low].

If the sample-hold uses a storage capacitor of 200 pF calculate how long the input voltage can be held before it changes by 0.5 LSB due to the leakage current.

9D. The circuit and state diagrams for a successive approximation converter are shown below. The output signals X2:0 and DONE are also used as the state bits. Derive Boolean equations for NX2:0 and NDONE. You should ensure that your circuit can never get stuck.





I/O Signals: START,HI/DONE,X0:2

10B. In the questions below, u represents a 4-bit unsigned binary number in the range 0 to 15 and x represents a signed 4-bit binary number in the range –8 to +7. Determine the range of possible values that each expression can take and give a Boolean expression for each bit of the corresponding binary number (signed or unsigned as appropriate). The function floor(x) denotes the largest integer less than x.

(a) $15 - u$	(b) floor(u/8)	(c) $u - 8$
(d) floor(u/2)	(e) $-(x+1)$	(f) floor(x/2)
(g) –floor(x/8)	(h) x – 8×floor(x/8)	(i) $x - 16\times$floor(x/8)
(j) floor(u/2) – 4×floor(u/8)	(k) $2u - 15\times$floor(u/8)	(l) $u - 16\times$floor(u/8)

E2.11/ISE2.22 – Digital Electronics II

Solution Sheet 5

(Question ratings: A=Easy, …, E=Hard. All students should do questions rated A, B or C as a minimum)

1B. Full-scale range = 3998 so the accuracy is 2/3998 of full-scale range. For an N-bit binary A/D converter, the full-scale range is (2^N–1) LSB giving an accuracy of 0.5/(2^N–1).

Hence $\frac{0.5}{2^N - 1} \le \frac{2}{3998} \quad \Rightarrow \quad 2^N \ge 1000.5 \quad \Rightarrow \quad N \ge 9.96 \quad \Rightarrow \quad N = 10$

2B. –2047 mV ±0.5 mV, i.e. –2047.5 mV to –2046.5 mV.

3B. 1 V and 8 V correspond to output values of 100 and 800 respectively, so if 1 V < x < 8 V, the output will be in the range 100 to 799.

4C. A change of 5 V in V3 must give a change of –1 V in VOUT, a gain of –0.2. Hence RF/R3 = 0.2 ⇒ R3 = 50 kΩ.

When V3=0, the op-amp may be viewed an a non-inverting amplifier with a gain of (1 + RF/R3) = 1.2. The voltage at VOUT due to V2:0 is therefore given by:

$$V_{OUT} = 1.2 \times \frac{G_2 V_2 + G_1 V_1 + G_0 V_0}{G_4 + G_2 + G_1 + G_0}$$

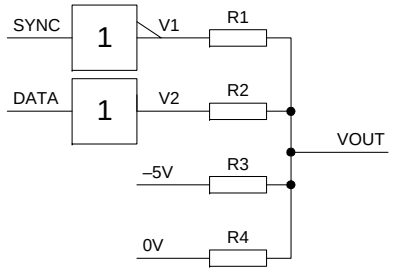
where G4:0 are the reciprocals of R4:0.

To minimize the effect of op-amp bias currents, we should make the Thévenin impedances at the input terminals equal. This means that G4+G2+G1+G0 = G3+GF = 120 µS.

The gains from V2, V1 and V0 to VOUT must be 0.1, 0.05 and 0.025 respectively. Thus we have G2 = 120 µS × 0.1/1.2 = 10 µS ⇒ R2 = 100 kΩ. Similarly, R1 = 200 kΩ and R0 = 400 kΩ.

Finally G4 = 120 µS – G2 – G1 – G0 = 102.5 µS ⇒ R4 = 9.8 kΩ.

5C. The SYNC signal needs inverting because SYNC going high must cause the output to decrease. We will need a negative bias voltage in order to obtain –0.3 V. Our circuit is therefore:



Taking Gn = 1/Rn we must have G1+G2+G3+G4 = 1/50Ω = 20 mS.

Then VOUT = ($V_1G_1 + V_2G_2 – 5G_3$) / 20 mS.

From the truth table, we see that changes of 5 V in V1 and V2 must give changes in VOUT of 0.3 and 0.7 volts respectively; this means we need gains of 0.06 and 0.14. Hence:

G1 = 0.06 × 20 mS = 1.2 mS ⇒ R1 = 833Ω.
G2 = 0.14 × 20 mS = 2.8 mS ⇒ R2 = 357Ω.
To generate the –0.3 V offset: 5G3 = 0.3 × 20 mS = 6 mS ⇒ R3 = 833Ω.
G4 = 20 mS – G1 – G2 – G3 = 14.8 mS ⇒ R4 = 67.6Ω.

Note it is possible to take R4 to +5 V instead in which case R3 and R4 are 116Ω and 135Ω.

This circuit is very fast since it has no op-amps.

6B. The range of a 16-bit signed number is ±32767 and so to avoid distortion, the RMS value must be no higher than 3276.7. From the notes, the RMS value of quantisation noise is 0.289 LSB which gives a signal-to-noise ratio of 11338 which equals 81 dB

7B. (a) z will equal -1 when $x < -0.5$, so

$$\begin{aligned} pr(z=-1) &= pr(x<-0.5) = pr(q<-0.5-w) \\ &= \int_{q=-1}^{-0.5-w} p(q)dq = \int_{q=-1}^{-0.5-w} 1-|q|\,dq = \int_{q=-1}^{-0.5-w} 1+q\,dq \\ &= \left[q+0.5q^2\right]_{q=-1}^{-0.5-w} = 0.125\times(2w-1)^2 \end{aligned}$$

Note that because $|w|<0.5$ is given in the question, both integration limits are always negative and so we can replace $|q| \to -q$ in the integrand. You can also get this answer graphically (and more easily) by drawing the pdf and finding the area of the triangle representing $pr(q<-0.5-w)$.

(b) $pr(z=+1) = 0.125\times(2w+1)^2$

$$pr(z=0) = 1 - pr(z=-1) - pr(z=+1) = 0.75 - w^2$$

(c) We have

$$\begin{aligned} E(z) &= 1\times pr(z=+1) - 1\times pr(z=-1) \\ &= 0.125\times\left((2w+1)^2 - (2w-1)^2\right) = w \end{aligned}$$

$$\begin{aligned} Var(z) &= E(z^2) - E(z)^2 = E(z^2) - w^2 \\ &= 1\times pr(z=+1) + 1\times pr(z=-1) - w^2 \\ &= 0.125\times\left((2w+1)^2 + (2w-1)^2\right) - w^2 \\ &= w^2 + 0.25 - w^2 = 0.25 \end{aligned}$$

8C. Full-scale range of 20 V equals 4096 LSB so 0.5 LSB = 0.5 × 20/4096 = 2.44 mV.

The peak rate of change of a 10 V sinewave is $20\pi f$ volts per second. The voltage change in 5 ns is therefore $\pi f \times 10^{-7}$. These are equal when f = $2.44 \times 10^{-3} \times 10^{7} / \pi$ = 7.77 kHz.

For the second part I = C dV/dt from which $\Delta t = C\times\Delta V/I = 2\times10^{-10}\times2.44\times10^{-3}/10^{-9}$ = 488 μs.

9D. We send the all zero state to the initial state of a conversion.

$$\begin{aligned} NDONE &= DONE\cdot\overline{START} + \overline{DONE}\cdot X0 \\ NX2 &= DONE\cdot(X2+START) + \overline{DONE}\cdot X2\cdot(X0+X1+HI) + \overline{DONE}\cdot\overline{X2}\cdot\overline{X1}\cdot\overline{X0} \\ NX1 &= DONE\cdot X1\cdot\overline{START} + \overline{DONE}\cdot X1\cdot(X0+HI) + \overline{DONE}\cdot X2\cdot\overline{X1}\cdot\overline{X0} \\ NX0 &= DONE\cdot X0\cdot\overline{START} + \overline{DONE}\cdot X0.HI + \overline{DONE}\cdot X1\cdot\overline{X0} \end{aligned}$$

10B. We call the answer w or z according to whether it is unsigned or signed:

(a) 0<=w<=15, W_i=!U_i

(b) 0<=w<=1, W_0=U_3

(c) –8<=z<=7, Z_3=!U_3, Z_i=U_i for i=0,1,2

(d) 0<=w<=7, W_i=U_{i+1}

(e) –8<=z<=7, Z_i=!X_i

(f) –4<=z<=3, Z_i=X_{i+1}

(g) 0<=w<=1, W_0=X_3

(h) 0<=w<=7, W_i=X_i for i=0,1,2

(i) 0<=w<=15, W_i=X_i

(j) 0<=w<=3, W_i=U_{i+1} for i=0,1

(k) 0<=w<=15, W_0=U_3, W_i=U_{i-1} for i=1,2,3

(l) –8<=z<=7, Z_i=U_i

E2.11/ISE2.22 – Digital Electronics II

Problem Sheet 6

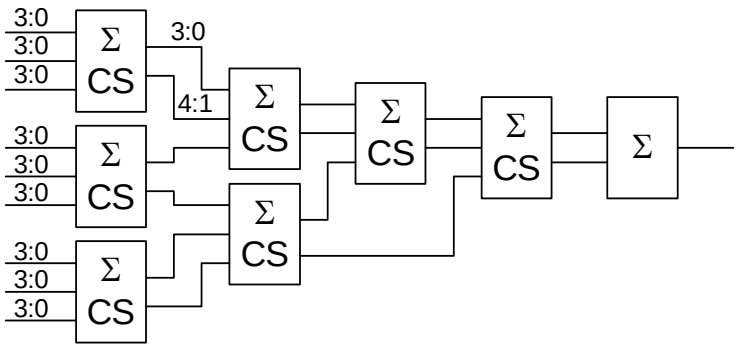
(Question ratings: A=Easy, …, E=Hard. All students should do questions rated A, B or C as a minimum)

1B+ A full-adder is a *symmetric* function of its inputs and is *self-dual*. Define the meaning of the italicised terms. Explain why self-duality implies that any NAND gate implementation of a full-adder can be converted into a NOR gate implementation with exactly the same gate interconnections.

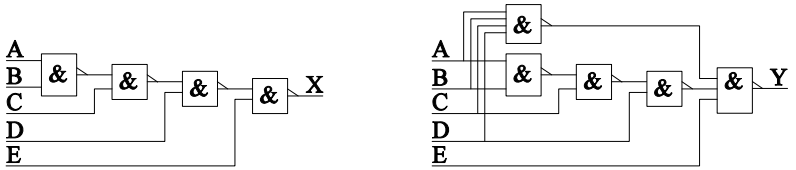
2B A number x lies in the range 0 to 9. Using only a full-adder, create a circuit that generates $y = 3x$ by adding together x and $2x$. Calculate the number of bits needed to represent y and hence use the smallest full-adder possible. How can the number of adder bits be reduced by using a single OR gate as well as the adder ?

3A. If CG = P•Q, CGP = P+Q and CP = P⊕Q, show using Boolean algebra that CG+CGP•CI = CG+CP•CI.

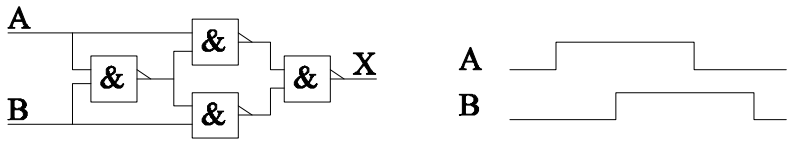
4C. The diagrams below show a possible interconnection for a carry-save addition tree in which each of the nine inputs is a 4-bit unsigned number. Label each of the intermediate and output numbers in the circuit with its range of bit numbers; thus 7:3 would indicate a 5-bit number whose least significant bit has a weight of 2^3. The outputs from the first carry-save block have already been labelled as an illustration.



5C. Show that the two circuits shown below generate the same Boolean function of their inputs. For each circuit, the inputs take the following sequence of values: ABCDE = 00000, 00001, 00011, 00111, 01111, 11111, 01111. For each circuit calculate the sequence of output values and the propagation delay each time the output changes. Each gate has a delay of 1. How is this circuit similar to the carry skip scheme?

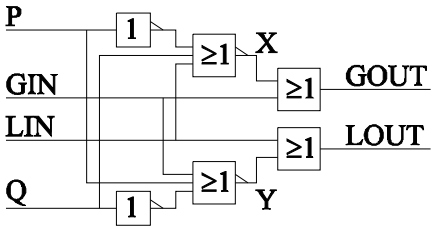


6C. The NAND gates in the following circuit have propagation delays of 5.5 ns when the output goes from low to high and 4 ns when it goes from high to low. If A and B have the waveforms shown, draw the waveform of X and give the propagation delay of the circuit for each output transition.

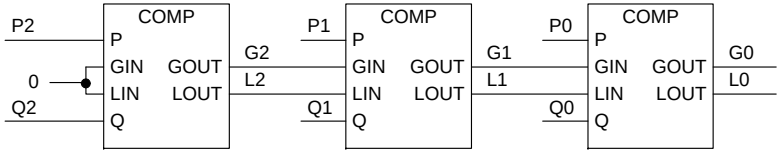


7B. Show how the circuit of a full-adder can be simplified if it is known that one of the input signals is always zero. Your circuit should use only NAND gates.

8C. The diagram shows the circuit of a single stage from a magnitude comparator. Calculate the worst-case propagation delays from each input to the GOUT output. In each case state what values the other inputs must have for the worst-case delay to occur.



9D. A 3-bit comparator for unsigned numbers is made by combining 3 copies of the circuit in the previous question:

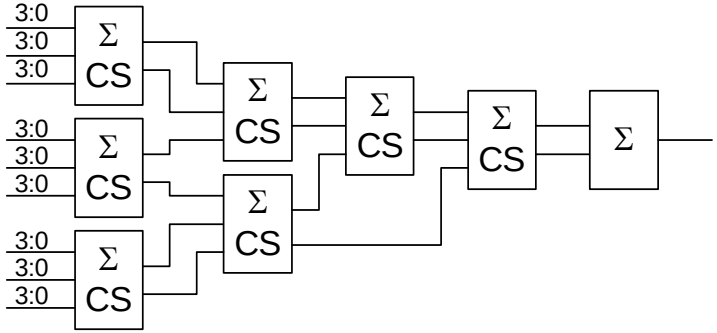


(a) Show that G0=1 if and only if P2:0 > Q2:0.

(b) Determine the worst-case delay from P2 to G0 and give an example of when it occurs (i.e. give the initial values of P2:0 and Q2:0 before P2 changes).

10D. The circuit shows a carry-save tree for adding together nine 4-bit unsigned numbers. Calculate the propagation delay of the circuit and give an example where this delay occurs. Each carry-save bit is implemented using the nine NAND gate circuit from the notes. The final stage is a carry-lookahead adder with a delay of 6.

Show how the delay may be reduced by inserting inverters between the columns of carry-save adder modules and merging the resultant AND gates into the following stages.



E2.11/ISE2.22 – Digital Electronics II

Solution Sheet 6

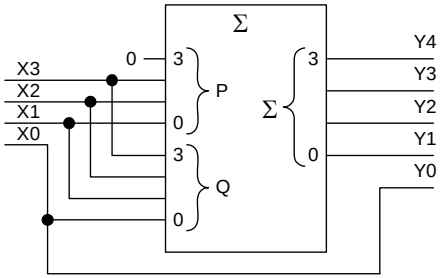
(Question ratings: A=Easy, …, E=Hard. All students should do questions rated A, B or C as a minimum)

1B+ *Symmetric* means that the inputs can be permuted arbitrarily without affecting any of the outputs.

Self-dual means that inverting all the inputs will cause all the outputs to be inverted.

It is possible to re-implement the circuit using *negative* logic in which a logic 1 is a low voltage and logic 0 is a high voltage. In negative logic a NAND function is performed by a positive logic NOR gate: thus wherever the original circuit had a NAND, you now need a NOR (and vice-versa). The input and output signals to the circuit still use positive logic, so they undergo logical inversion; the nett effect is therefore a full adder with inverters at the inputs and outputs. Since the adder is self-dual this is the same as a full adder.

2B *y* lies in the range 0 to 27 and therefore needs a 5 bit unsigned number. We generate 2*x* just by shifting the bits one space to the left; this doesn't need a shift register or any other circuitry – it just involves relabelling the bits. We don't need any adder stage for the LSB since X0 is always equal to Y0.

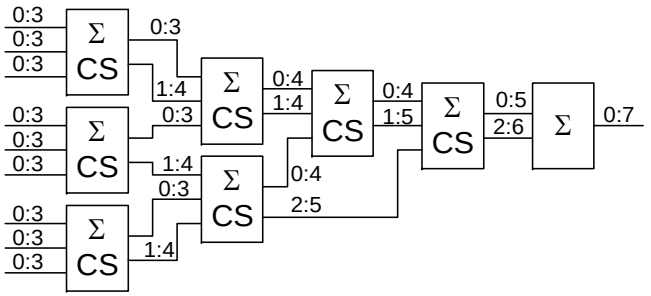


The Y4 output is the addition of Q3 and P3 together with the carry from the previous stage. Since P3=0 you can generate this by XORing Q3 with the carry. Since we know the result can never exceed 5 bits, we know that there can never be a carry out o bit 4 and hence that Q3 and the carry from stage 3 can never both be high at the same time. It follows that we can use an OR gate instead of the XOR that we would normally need. Hence we use a 3-bit full adder and OR X3 with the carry out of the adder to generate Y4.

3A. CGP = P+Q = P•(Q+!Q) + Q•(P+!P) = P•Q + P•!Q + !P•Q = CG + CP

CG + CGP•CI = CG + (CG+CP)•CI = CG•(1+CI) + CP•CI = CG + CP•CI

4C. The maximum output from the circuit is 9×15=135 which needs 8 bits.



The SUM output from a CS adder must go from the lowest bit position of any input to the highest bit position of any input. The CARRY output must go from one more than the lowest bit position with at least two inputs to one more than the highest bit position with at least two inputs. This is because we need at least two inputs to generate a carry.

Note that several of the carry-save modules have bit positions in which either one or two of the inputs are missing: in these cases the circuitry becomes either simpler or completely unnecessary. In the latter case, no carry output is possible at all.

We can actually save a small amount of circuitry by rearranging the connections between the first two stages so that all the 4:1 signals go to one carry-save module and all the 3:0 signals go to the other. This increases the number of columns that need no circuitry at all.

5C.

$$X = \overline{E} + D \cdot \left(\overline{C} + A \cdot B\right) = \overline{E} + \overline{C} \cdot D + A \cdot B \cdot D$$

$$Y = \overline{E} + \overline{C} \cdot D + A \cdot B \cdot D + A \cdot B \cdot C \cdot D = X$$

ABCDE	X=Y	X delay	Y delay	
00000	1			
00001	0	1	1	
00011	1	2	2	
00111	0	3	3	
01111	0			
11111	1	4	2	← Speedup
01111	0	4	4	← No Speedup

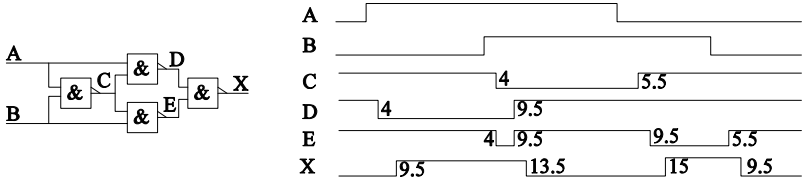
The circuit is similar to carry skip in that the situation causing the worst-case delay is recognised by the 4-input NAND gate and the chain of three NAND gates is then bypassed causing a speedup.

The circuit differs from carry skip in two respects:

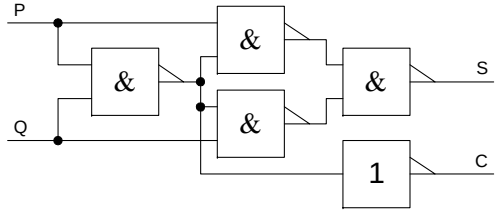
(a) Carry skip uses a multiplexer to switch between the fast and slow paths which means that it works for both rising and falling edges of its output. The use of a NAND gate in this circuit to combine the fast and slow paths means that the speedup is only effective for rising output transitions.

(b) Carry skip improves the worst-case delay at the expense of making the delay greater under most other conditions. This circuit doesn't make any delays worse.

6C. In the timing diagram, I have marked each edge with the time delay from the A or B transition that caused it. The circuit is an XOR function.



7B. Because a full-adder is symmetrical, it doesn't matter which of the inputs we take to be zero. Here I have assumed that it is the CI input.



8C. P→GOUT = 3 when !Q•!LIN•!GIN =1
GIN→GOUT = 1 when !P+Q+LIN=1
LIN→GOUT = 2 when P•!Q•!GIN=1
Q→GOUT = 2 when P•!LIN•!GIN =1

9D. The circuit compares the two numbers beginning at the most significant bit. If Px = Qx for all the bits down to and including bit n, then Gn=Ln=0. Otherwise, either Gn or Ln is high according to whether P is greater or less than Q. Pn and Qn are never high together.

Looking at the circuit of the previous question, we can see that GOUT is high either if GIN=1 (i.e. if higher order bits have determined that P>Q) or else if P=1 and Q=0 and LIN=0 (i.e. previous bits are identical but P>Q because of this bit).

The longest delay is P2→G0 = 7 when Q=2 and P changes from 5 to 1.

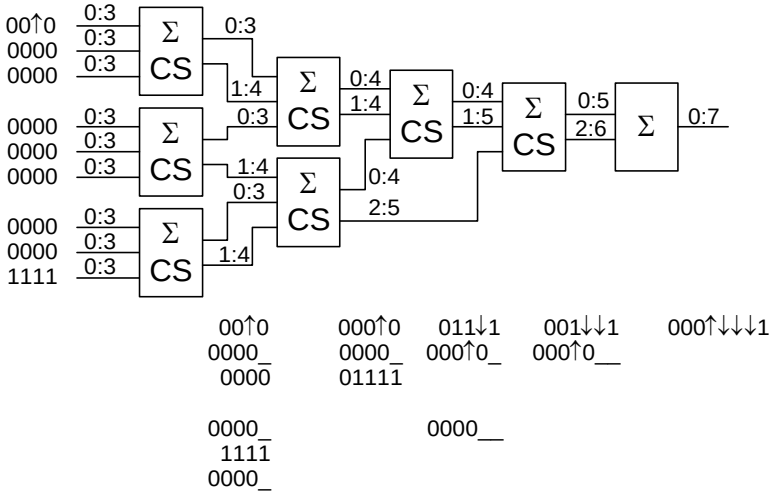
Time 0: P2↓
Time 1: !P2↑
Time 2: X2↓
Time 3: G2↓
Time 4: Y1↑
Time 5: L1↑
Time 6: X0↓
Time 7: G0↓

10D. Initial delay = 4×3 + 6 = 18.

The delay to the S output of the CS adder is 3 in any bit position where neither the initial or final states have all 3 inputs high. The delay of the final adder is 6 if any of its carry signals change.

Carry-save bits with only one active input (such as the least significant bit of the last carry-save module) require no circuitry and hence have no delay. To avoid these bits, we alter bit 1 rather than bit 0 in the example below.

The numbers written below the diagram show the value of each bit of each number. As indicated by the up-arrow, we are changing bit 1 of the uppermost input number from 0 to 1 (i.e. increasing the number's value by 2). The resultant changes in other values are indicated by the up or down arrows in the diagram.



By inserting inverters between stages, we can change the delay to 4×2+6 = 14. We should not insert an inverter in the signal path that skips a stage.