Department of Computing
Course 112 - Hardware
Tutorial 7

This tutorial contains exercises that are assessed. Make certain that your name, initials and group are clearly filled in. Do all your work on this sheet and hand it in at the end of the tutorial session.

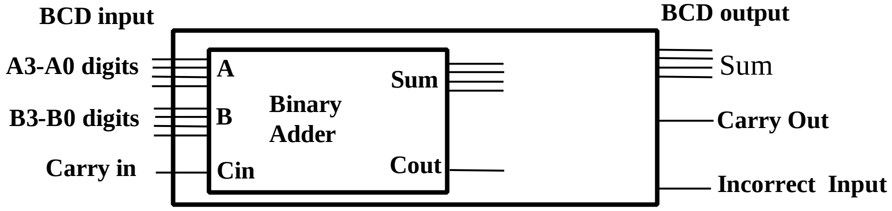
LAST NAME______________________________INITIALS_________GROUP_______

Small calculators work with **BCD** (binary-coded-decimal) digits. This means that four bits represent decimal digits **0** to **9** and thus six combinations of the four bits are not used. A common bit assignment for **BCD** digits are:

Bits	**0000**	**0001**	**0010**	**0011**	**0100**	**0101**	**0110**	**0111**	**1000**	**1001**
Digit	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**

Bits	**1010**	**1011**	**1100**	**1101**	**1110**	**1111**
Digit	**-**	**-**	**-**	**-**	**-**	**-**

In preparation for the design of a calculator, we want to build an **adder of two BCD digits**, which will be made up of a binary adder and some extra circuits.



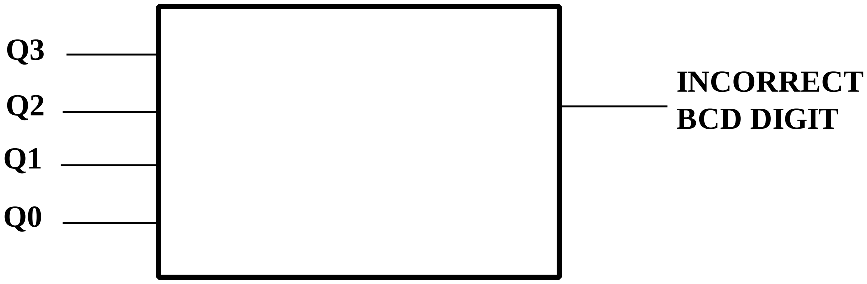
First, we need to design a circuit that recognises incorrect **BCD** input digits.

Problem 1

Design and draw the minimised circuit for the **"incorrect BCD digit"** signal generator. First draw the Karnaugh map and determine the minimised boolean equation, then draw the actual circuit.

		Q1 Q0			
		00	**01**	**11**	**10**
Q3Q2	**00**				
	01				
	11				
	10				

Incorrect BCD digit =



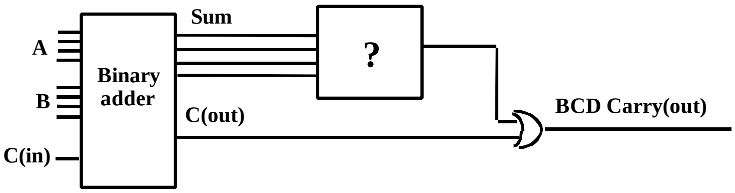
Two of the circuits of problem 1, and a two-input **OR** gate will be needed to provide an **Incorrect Input** signal which indicates that either or both input digits are made up of invalid bit combinations. We now plan to use a four-bit binary adder to provide the sum of two BCD digits.

Problem 2. Design the **BCD Carry Out**

For a sum of less than **9** the **Carry Out** signal should have a **0** value. For larger sums it should be **1**, equivalent to carry 10. For example two **BCD** digits, say **5** and **8** should produce **3** for the BCD output and a **Carry Out** of **1**, read as **13**. However the binary adder carry does not do this. The possibilities generating a carry are sumarised by the following table.

Sum	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**
Binary Sum	**1010**	**1011**	**1100**	**1101**	**1110**	**1111**	**0000**	**0001**	**0010**	**0011**
Binary Cout	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**1**
BCD Sum	**0000**	**0001**	**0010**	**0011**	**0100**	**0101**	**0110**	**0111**	**1000**	**1001**
BCD Cout	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

An ingenious classmate of yours comes up with the following scheme for the **BCD Carry Out** generator:



It works! Find the combinational circuit (shown with the question mark above) that will make it work:

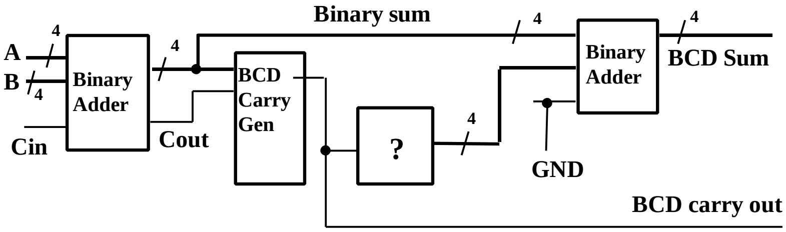
Q1 Q0

Q3Q2	**00**	**01**	**11**	**10**
00				
01				
11				
10				

BCD Carry =

Problem 3.

Only the binary sum of the two BCD digits has been generated so far. The same ingenious classmate of yours has a solution for generating the correct **BCD** output sum as well. All you have to do (she says) is to use another binary adder to fix up the output when a BCD carry is generated. What should the circuit (**?** on the diagram below) be? (**Hint:** The results will be simpler than you think!)


A
4
B
4
Cin
Binary Adder
4
Cout
BCD Carry Gen
Binary sum
?
4
4
Binary Adder
GND
4
BCD Sum
BCD carry out