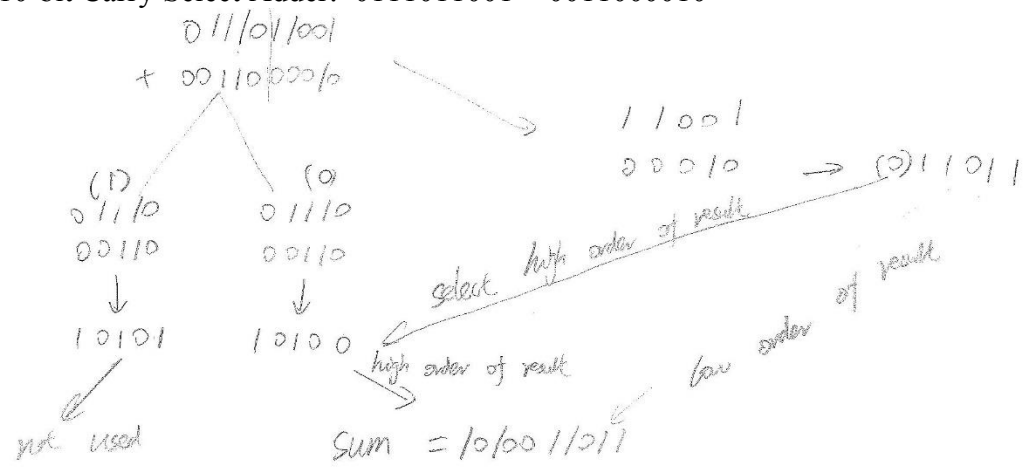
## Homework 3 Solutions

1. [Self Study]
   a) 10-bit Carry Select Adder: 0111011001 + 0011000010




   b) 10-bit subtraction circuit: 0111011001 – 0011000010

0111011001 − 0011000010

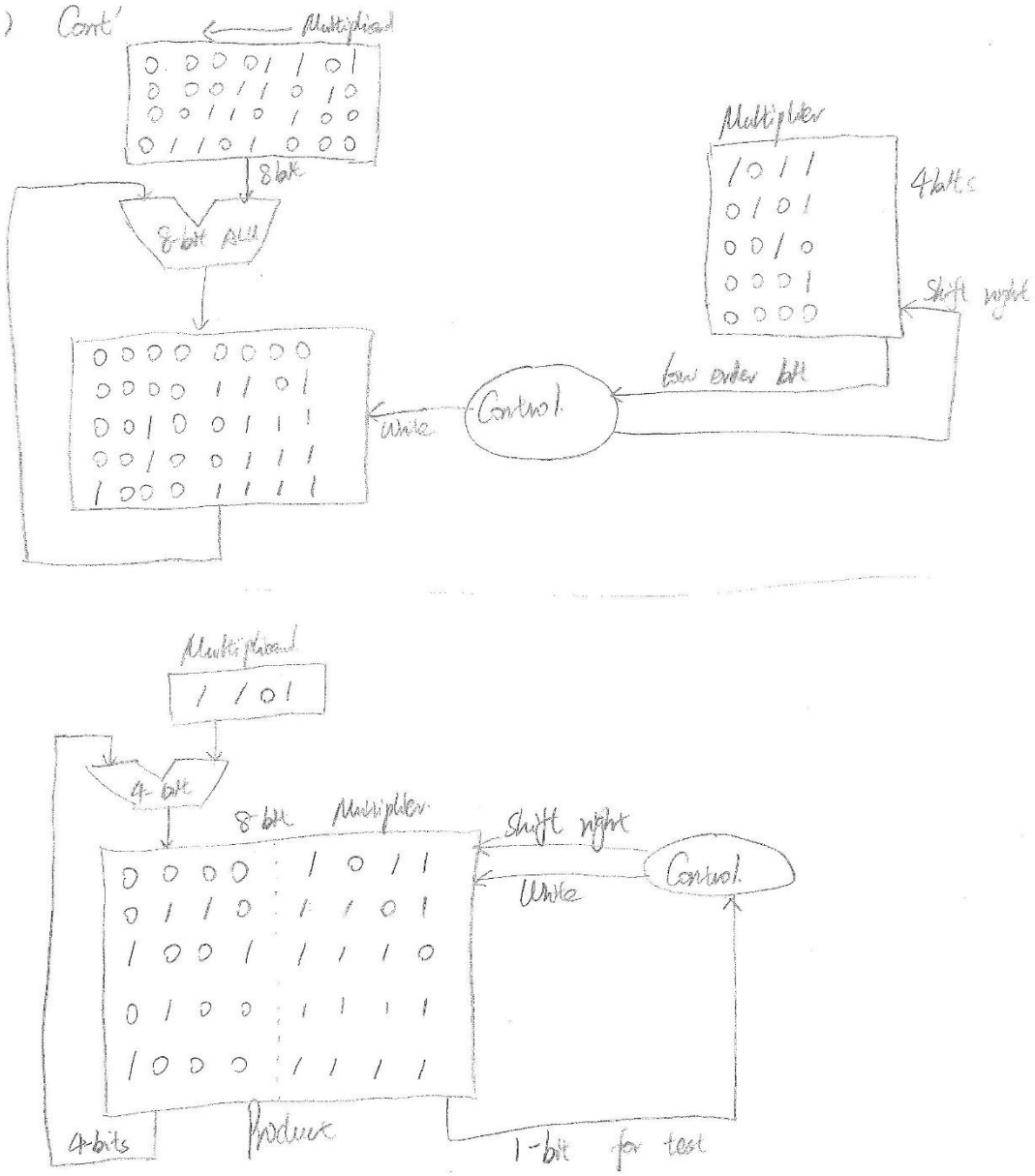
= 0111011001 + (1100111101) + 1

= 0111011001 + 1100111110

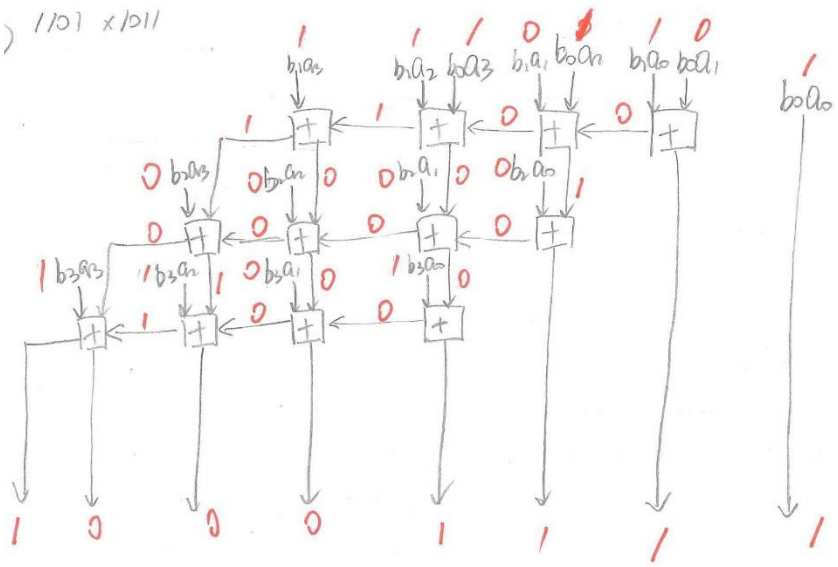
```
  0111011001
+ 1100111110
------------
(1)0101000111
```

   c) 4-bit multiplication -- do for "pencil-and-paper" on both multiply circuits (3.4 and 3.6): 1101 x 1011

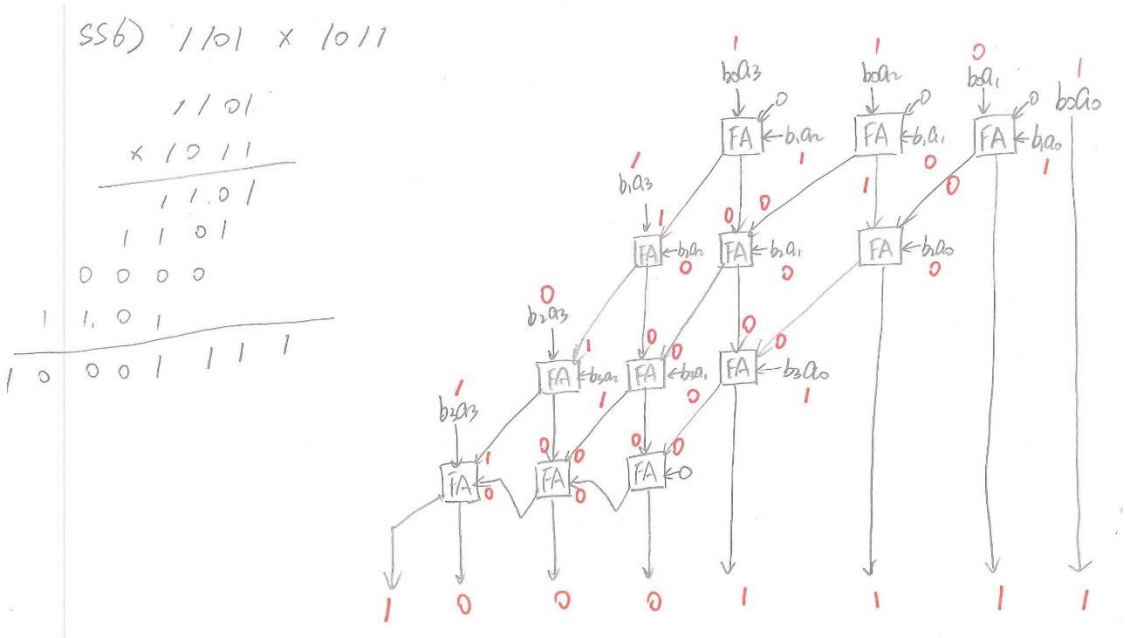
```
SSS)   1101   multiplicand
     x 1011   multiplier
     ------
       1101
      1101
     0000
    1101
    ---------
    10001111
```




d) "unrolled" 4-bit multiplier: 1101 x 1011



e) Carry-Save-Adder (CSA) based 4-bit multiplier -- do for both"pencil-and-paper" and the circuit (the scan from Kavanagh).



2. [Carry Select Adders]

a) Delay = 2*k + 2*(j-2), where k = bits of RCA, j = number of blocks
Delay = 2*4 + 2*(2-2) = 8+0 = 8

b) Delay = 2*k + 2*(j-2), where k = 4, j=3
Delay = 2*4 + 2*(3-2) = 8+2 = 810

c) The question states that we have only a single size RCA. As a result, we know that k=N/j, where k = bits of RCA, j = number of blocks.
We can then plug this in to the CSA delay equation to solve for k:
2k + 2(N/k) – 4
We have the smallest delay when the RCA and MUX delays are approximately equal, so we can rewrite this as:
2k = 2(N/k) ➔ k = $\sqrt{N}$
For this question, N=64, so k=8.
Going back to k=N/j, solving for j gives j=8 as well.
With 8 stages, we have a total of 15 RCAs: one for the first state + 2 for the rest of the stages.
1+2*7=15 RCAs
Delay = 2*8 + 2(8-2) = 16+12 = 28

3. [Sequential Timing]
   a) 32 cycles, each with a 64-bit RCA, ignoring shift = 32 * 128 = 4096 cycles
   b) 32 cycles, each with a 32-bit RCA, ignoring shift = 32 * 64 = 2048 cycles

4. [Multiplication]
   this is similar to the previous question, except that P&H make different assumptions about how the operations are implemented. Please note that for the hardware version, #2 above is more realistic.

   P&H solution: For hardware, it takes 1 cycle to do the add, 1 cycle to do the shift, and 1 cycle to decide if we are done. So the loop takes (3 * A) cycles, with each cycle being B time units long.
   For a software implementation, it takes 1 cycle to decide what to add, 1 cycle to do the add, 1 cycle to do each shift, and 1 cycle to decide if we are done. So the loop takes (5 * A) cycles, with each cycle being B time units long.
   (3*8)*4tu = 96 time units for hardware
   (5*8)*4tu = 160 time units for software

5. [Multiplication]
   a) Unrolled using RCAs: Delay = 2N + 4(N-2) ≈ 6N
   b) Unrolled using CSAs: Delay = 2N + 2(N-2) ≈ 4N
   c) Unrolling is faster because it doesn't require row A to be fully computed for row A+1 to begin computation while a shift-and-add multiplier does.

6. [Multiplication]
   Regardless of how you change the CSA-based multiplier, the timing is not affected.

7. [Floating Point]
   **3.23** $63.25 \times 10^0 = 111111.01 \times 2^0$

   normalize, move binary point 5 to the left

   $1.1111101 \times 2^5$

   sign = positive, exp = 127+5=132

   Final bit pattern: 0 1000 0100 1111 1010 0000 0000 0000 000

   = 0100 0010 0111 1101 0000 0000 0000 0000 = 0x427D0000

8. [Floating Point]
   a) 0|0001|0000
   b) 0|1110|1111
   c) 0|0111|0001
   d) 0|0110|1111
   e) 0|1101|1010
   f) 0|1111|0000
   g) 0|1000|0101
   h) 0|0110|1110
   i) 0|1000|0001