1. (5 points) Compute the difference: 011011 − 001101 using six bit binary arithmetic, by taking the 2s complement of the second operand and then adding.

\[ 011011 - 001101 = 011011 + (110010 + 1) = 011011 + 110011 = 001110 \]

2. (10 points) Show that the following Boolean equation is true using algebraic simplification (not Karnaugh maps).

\[ ABD' + BC' + CD = (B + (C \oplus D'))(A + C' + D)(B + C)(B + D) \]
\[ = (B + C'D' + CD)(A + C' + D)(B + CD) \]
\[ = (A + C' + D)(B + CD) \]
\[ = AB + ACD + BC' + BD + CD \]
\[ = AB + BC' + BD + CD \]
\[ = ABD + ABD' + BC' + BD + CD \]
\[ = ABD' + BC' + BD + CD \]
\[ = ABD' + BC' + BC'D + BCD + CD \]
\[ = ABD' + BC' + CD \]
3. (10 points) The simulation output below is from an execution of the simple processor introduced in section 1 of the lecture notes. The instruction set for the processor appears on the next page. Answer the following questions using the simulation output.

What does the instruction that is being executed at time 4.4 µs do? 1012 – stores value in accumulator to location 012.

What is the value of memory location 000e at time 4.7 µs? 4004

What is the value of the program counter at time 4.72 µs? 000f

What is the value of memory location 0010 at time 5 µs? a = 1010

Which component of the computer is putting a value on the address bus at time 5.0 µs? program counter
0xxx load ACC with value stored in memory word xxx
1xxx store value in ACC into memory word xxx
2xxx add value in memory word xxx to value in ACC
3000 negate the value in ACC
3001 halt
4xxx change the value of PC to xxx
5xxx if the value of ACC is zero, change PC value to xxx
6xxx load ACC with value whose address is stored in word xxx
7xxx store ACC value into word whose address is in word xxx
8xxx change ACC value to xxx
9xxx add xxx to the value in ACC

4. (6 points) What is the dual of the expression $BC' + A(B + D)' + (A' + B)(C + D')$?

The dual is $(B + C')(A + (BD)')(A'B + CD')$

What is its complement?

The complement is $(B' + C)(A' + (B'D')')(AB' + C'D)$
5. (10 points) Write code for a VHDL module that directly implements the circuit shown below. Please make your code complete and syntactically correct.

A module implementing the circuit appears below.

```vhdl
library IEEE;
use IEEE.std_logic_1164.all;

entity foo is
    port (  
        A, B, C, D: in std_logic;
        X, Y: out std_logic
    );
end foo;
architecture bar of foo is
begin
    X <= (A xor D) and B;
    Y <= (not ((A xor D) and B)) or (A xor D)  
        or (not (C and (not D)));
end bar;
```
6. (5 points) Consider the circuit and timing diagram shown below. Assuming that the inverters are all implemented using CMOS technology, which of the two curves labeled with a question mark corresponds to the signal B and which to signal C? Assume that all the inverters are identical. Explain why this is true.

The middle signal corresponds to output C, the bottom one to output B. The reason for this is that signal C will change faster than B, since the pair of inverters driving output C each have a separate pullup, which allows current to flow more quickly from the power supply to the output node.
7. (8 points) For each function given below, use a Karnaugh map to find the simplest sum-of-products expression for the function. Take full advantage of the don’t care conditions.

(a) \( F(X,Y,Z) = \sum m(0,2,6), \ d(X,Y,Z) = \sum m(3,4) \)

(b) \( F(A,B,C,D) = \sum m(0,2,5,7,13,14,15), \ d(A,B,C,D) = \sum m(1,3,6,8) \)
8. (12 points) The schematic below shows a ripple-carry adder and a section from a simulation of the adder. Fill in the waveforms for the outputs that are not shown. Make sure your transitions occur at the correct times, assuming each gate has a delay of 1 ns.
9. (10 points) Draw a logic diagram using simple gates and a 4:1 multiplexor that is equivalent to the VHDL code shown below. Be sure to label all the inputs of your multiplexor correctly.

```vhdl
library IEEE;
use IEEE.std_logic_1164.all;

entity foo is
  port (
    brownie, frosting, cake: in std_logic;
    choice: in std_logic_vector (1 downto 0);
    dessert, chocolate: out std_logic
  );
end foo;

architecture bar of foo is
begin
  chocolate <= (brownie or not choice(0)) xor cake;
  dessert <= (cake xor brownie) when choice = "11" else
              brownie when choice = "10" else
              cake and frosting;
end bar;
```

![Logic Diagram](image-url)