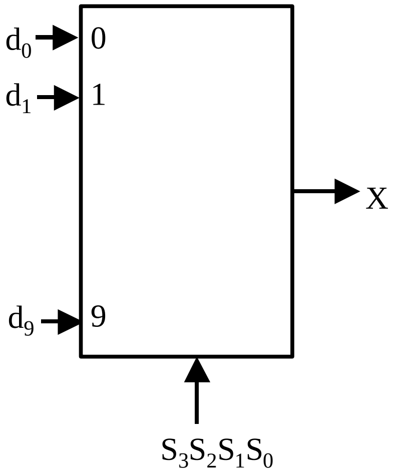
CS/EE 260 – Homework 5 Solutions

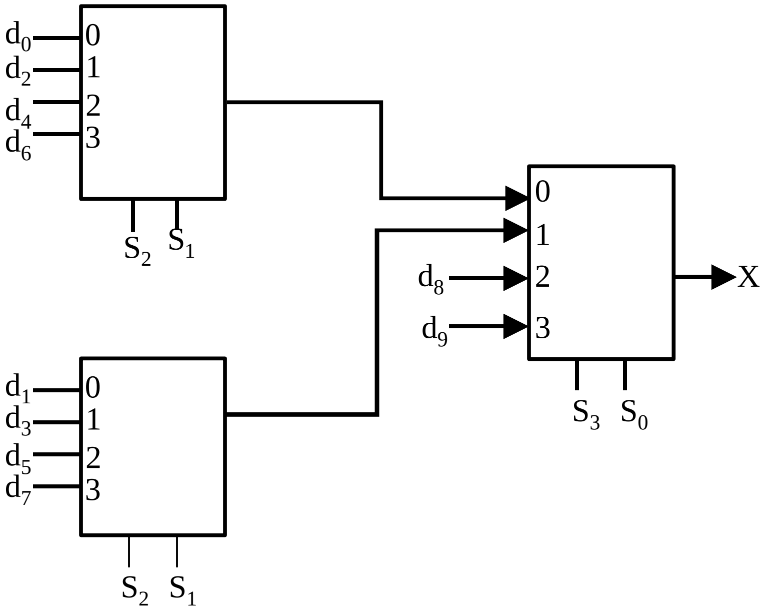
Spring 2000

1. (MK 3-23) Construct a 10-to-1 line multiplexer with three 4-to-1 line multiplexers. The multiplexers should be interconnected and inputs labeled so that the selection codes 0000 through 1001 can be directly applied to the multiplexer selections inputs without added logic.

10:1 mux

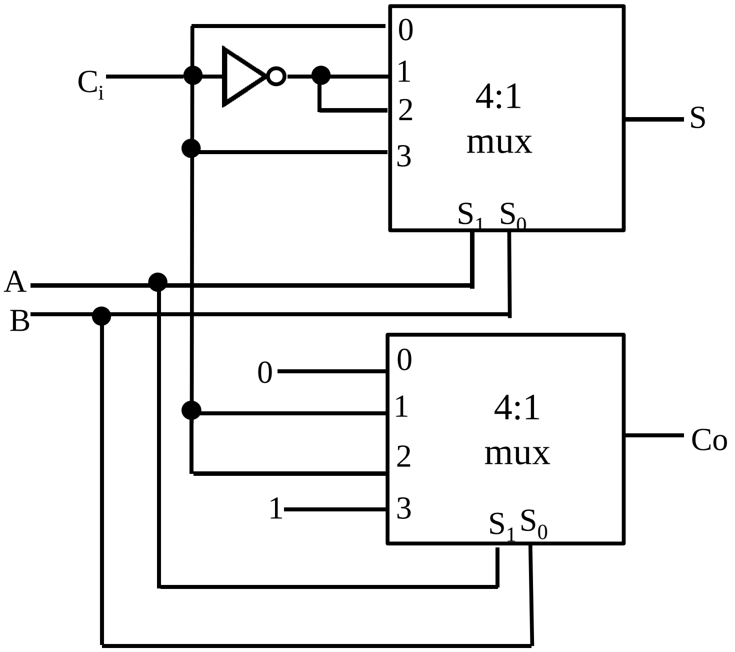


Implementation using 4:1 muxes.

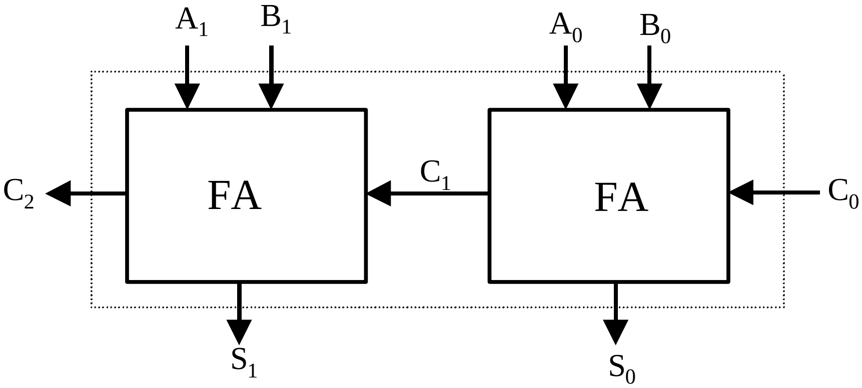


2. (MK 3-27) Implement a binary full adder with a dual 4-to-1 line multiplexer and a single inverter.

AB	C_i	S		C_o	
00	0	0	C_i	0	0
00	1	1		0	
01	0	1	C_i'	0	C_i
01	1	0		1	
10	0	1	C_i'	0	C_i
10	1	0		1	
11	0	0	C_i	1	1
11	1	1		1	



3. (MK 3-34) Design a combinational circuit that forms the 2-bit binary sum S_1S_0 of two 2-bit numbers A_1A_0 and B_1B_0 and has both input C_0 and a carry output C_2. Do not use half adders or full adders, but instead use a two-level circuit plus inverters for the input variables, as needed. Design the circuit by starting with the following equations for each of the two bits of the adder.



$S_0 = A_0\,B_0\,C_0 + A_0\,B_0\,'C_0' + A_0'B_0C_I' + A_0\,'B_0'\,C_0 = \text{odd}\,(A_{0\,,}B_0, C_i)$

$C_1 = A_0\,B_0 + A_0C_0 + B_0C_0 = \text{majority}\,(A_{0\,,}B_0, C_0)$

$S_1 = \text{odd}\,(A_1, B_1, C_1) = \text{odd}\,(A_1, B_1, \text{majority}\,(A_{0\,,}B_0, C_0))$

$= A_1\,B_1\,\text{majority}\,(A_{0\,,}B_0, C_0) + A_1\,B_1'\,\text{majority}\,(A_{0\,,}B_0, C_0)'$
$\quad + A_1'\,B_1\,\text{majority}\,(A_{0\,,}B_0, C_0)' + A_1'\,B_1'\,\text{majority}\,(A_{0\,,}B_0, C_0)$

$= (A_1\,B_1 + A_1'\,B_1')\,\text{majority}\,(A_{0\,,}B_0, C_0) + (A_1\,B_1' + A_1'\,B_1)\,\text{majority}(A_0'{}_{,}B_0', C_0')$

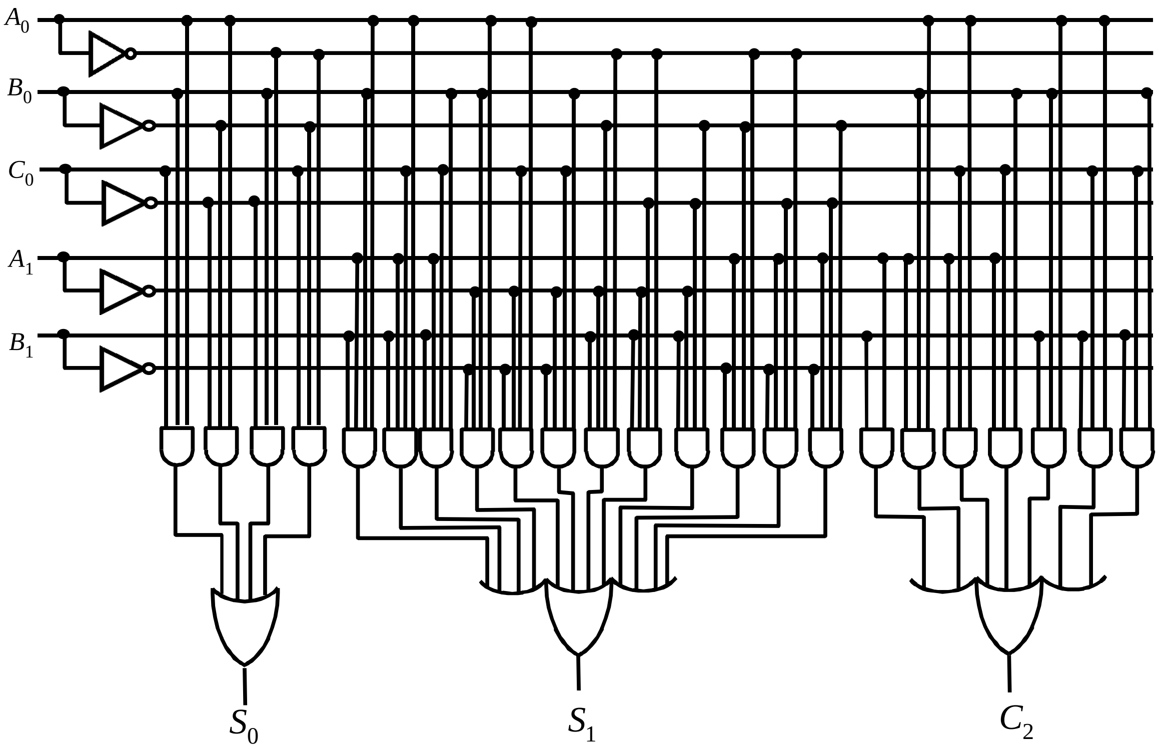
$= (A_1\,B_1 + A_1'\,B_1')\,(A_0\,B_0 + A_0C_0 + B_0\,C_0) + (A_1'\,B_1 + A_1B_1')\,(A_0'\,B_0' + A_0'\,C_0' + B_0'\,C_0')$

$= A_1B_1\,A_0\,B_0 + A_1\,B_1\,A_0C_0 + A_1\,B_1\,B_0C_0 + A_1'\,B_1'\,A_0\,B_0 + A_1'\,B_1'\,A_0C_0 + A_1'B_1'\,B_0C_0$
$\quad + A_1'B_1A_0'\,B_0' + A_1'B_1A_0'C_0' + A_1'B_1B_0'C_0' + A_1B_1'A_0'\,B_0' + A_1B_1'A_0'C_0' + A_1B_1'\,B_0'C_0'$

$C_2 = \text{majority}\,(A_1, B_1, C_1) = \text{majority}\,(A_1, B_1, \text{majority}\,(A_0, B_0, C_0))$

$= A_1B_1 + A_1\text{majority}\,(A_0, B_0, C_0) + B_1\text{majority}\,(A_0, B_0, C_0)$

$= A_1B_1 + A_1(A_0B_0 + A_0C_0 + B_0B_0) + B_1(A_0B_0 + A_0C_0 + B_0B_0)$



4. This problem is to design a simple 4 bit arithmetic unit. Your circuit will have two data inputs $A=a_3a_2a_1a_0$ and $B=b_3b_2b_1b_0$ and a data output $X=x_3x_2x_1x_0$. It will also have a control input $C=c_2c_1c_0$ which determines what operation your circuit will perform.

   if C=0 then the output X is equal to input A
   if C=1 then the output X is equal to input B
   if C=2 then the output X is equal to the 2's complement of input A
   if C=3 then the output X is equal to the 2's complement of input B
   if C=4 then the output X is equal to A+B where A and B are interpreted as unsigned integer values in the range 0..15
   if C=5 then the output X is equal to A+B where A and B are interpreted as signed integer values in the range –8..+7 (2s complement)
   if C=6 then the output X is equal to A-B where A and B are interpreted as signed integer values in the range –8..+7 (2s complement)
   if C=7 then the output X is equal to B-A where A and B are interpreted as signed integer values in the range –8..+7 (2s complement)

   In addition, your circuit will have a status output V which is 1, if the requested operation results in an invalid result. For example, negating an input whose value is –8 should cause the V output to go high. An unsigned addition of values 8 and 13 should also cause the V bit to go high.

As a first step in designing your circuit, draw a *block diagram* that includes a 4 bit adder, one or more 1's complement circuits, incrementers and multiplexors. Include a control block that the *C* input connects to and that generates the *V* output and the control signals for the other components. Show each circuit component as a block labeled by its function (e.g. adder) and with all inputs and outputs clearly labeled.

Write logic equations for each output of the control block that will cause the circuit as a whole to implement the specification given above.

Complete your design using the schematic editor. Use only AND, OR, EXOR gates and inverters. On the printout of the schematic, outline in pencil the gates that correspond to the blocks in your block diagram and label them as in your block diagram. Arrange your schematic so that it corresponds as closely as possible to your block diagram.

Simulate your design for the following inputs and verify that the results are correct.

C=0, *A*=5; *C*=1, *B*=10; *C*=2, *A*=6; *C*=3, *B*=-8; *C*=4, *A*=5, *B*=14; *C*=4, *A*=5, *B*=7; *C*=5, *A*=6, *B*=5; *C*=5, *A*=3, *B*=4; *C*=6, *A*=5, *B*=2; *C*=6, *A*=4, *B*=6;
C=6, *A*=4, *B*=-6; *C*=6, *A*=2, *B*=3; *C*=7, *A*=5, *B*=-6; *C*=7, *A*=-2, *B*=6

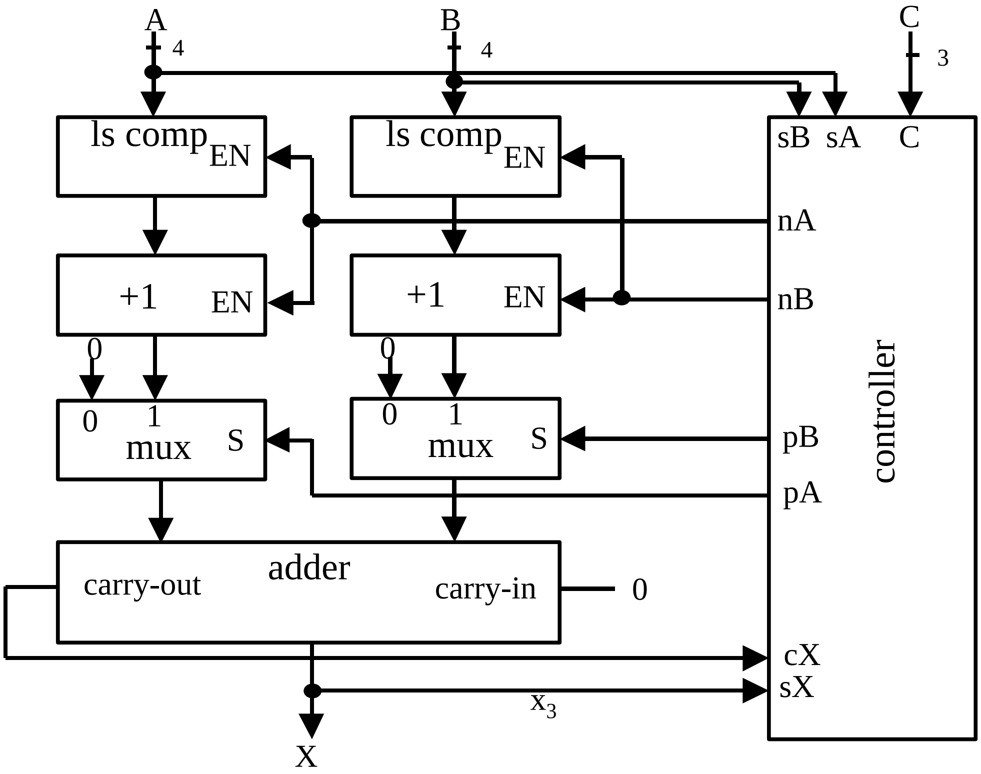


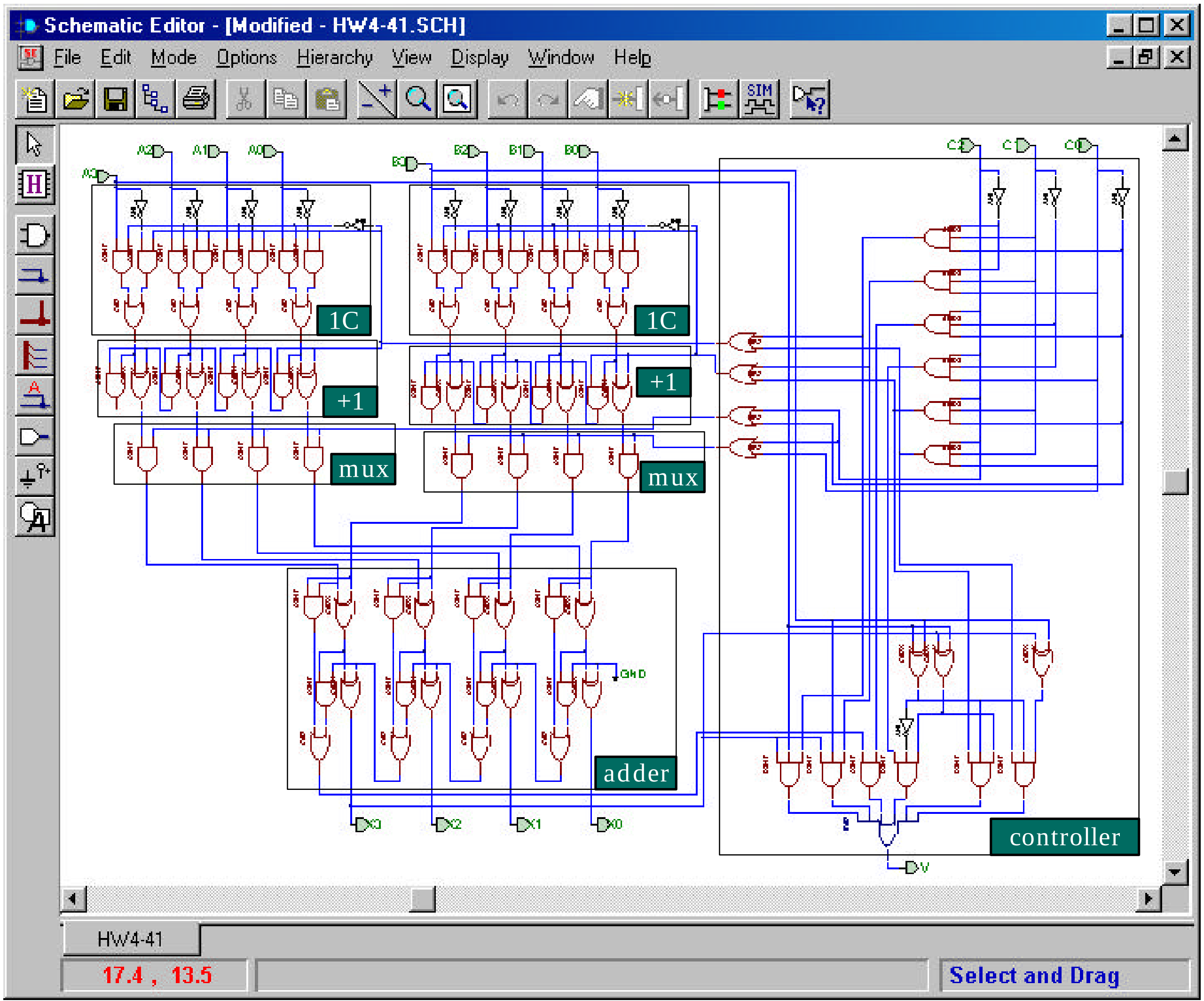
Table for control signals:

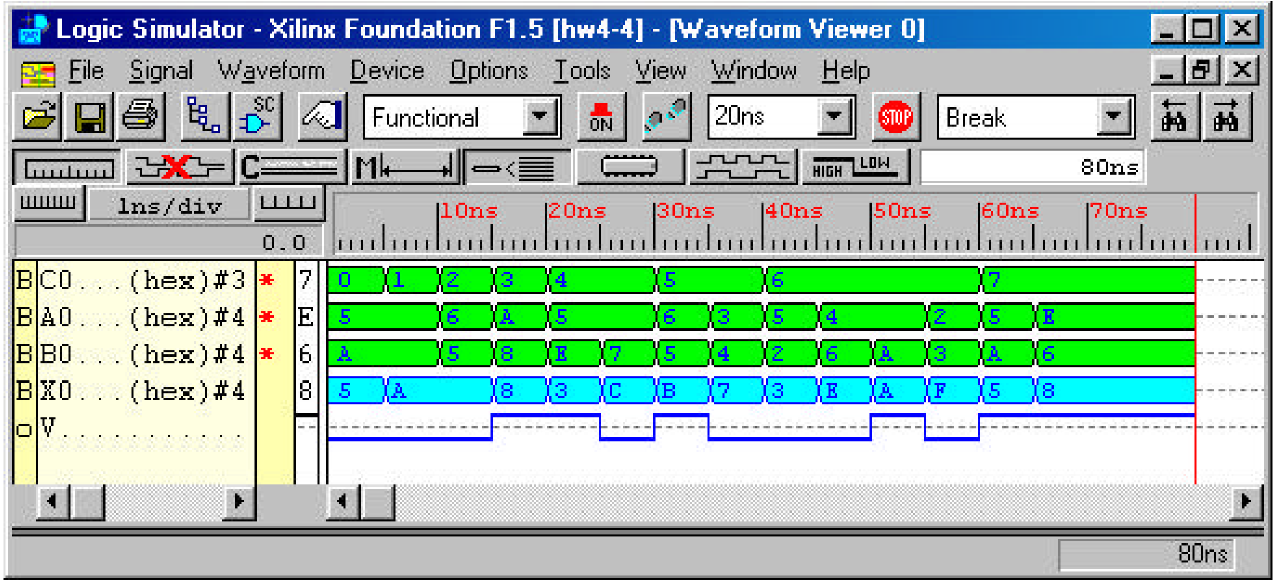
C	nA	pA	nB	pB	V
0	0	1	X	0	0
1	X	0	0	1	0
2	1	1	X	0	$sA \cdot sX$
3	X	0	1	1	$sB \cdot sX$
4	0	1	0	1	cX
5	0	1	0	1	$(sA=sB) \cdot (sA \neq sX)$
6	0	1	1	1	$(sA \neq sB) \cdot (sA \neq sX)$
7	1	1	0	1	$(sA \neq sB) \cdot (sB \neq sX)$

$nA = (C=2) + (C=7)$ $pA = C_2 + C_0'$

$nB = (C=3) + (C=6)$ $pB = C_2 + C_0$

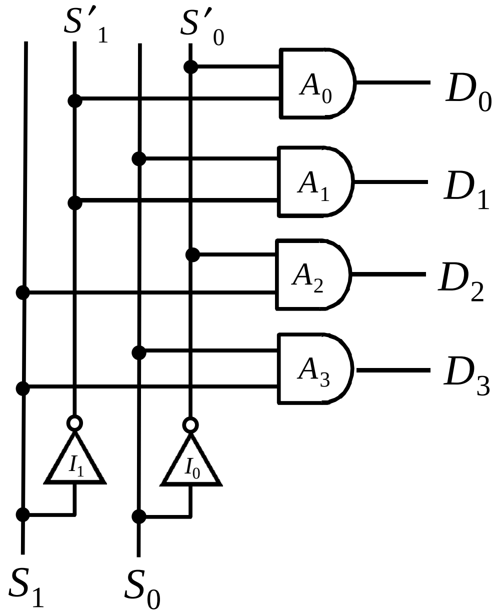
$$V = (C=2) \cdot sA \cdot sX + (C=3) \cdot sB \cdot sX + (C=4) \cdot cX + (C=5) \cdot (sA \oplus sB)' \, (sA \oplus sX)$$
$$+ (C=6) \cdot (sA \oplus sB) \cdot (sA \oplus sX) + (C=7) \cdot (sA \oplus sB) \cdot (sB \oplus sX)$$

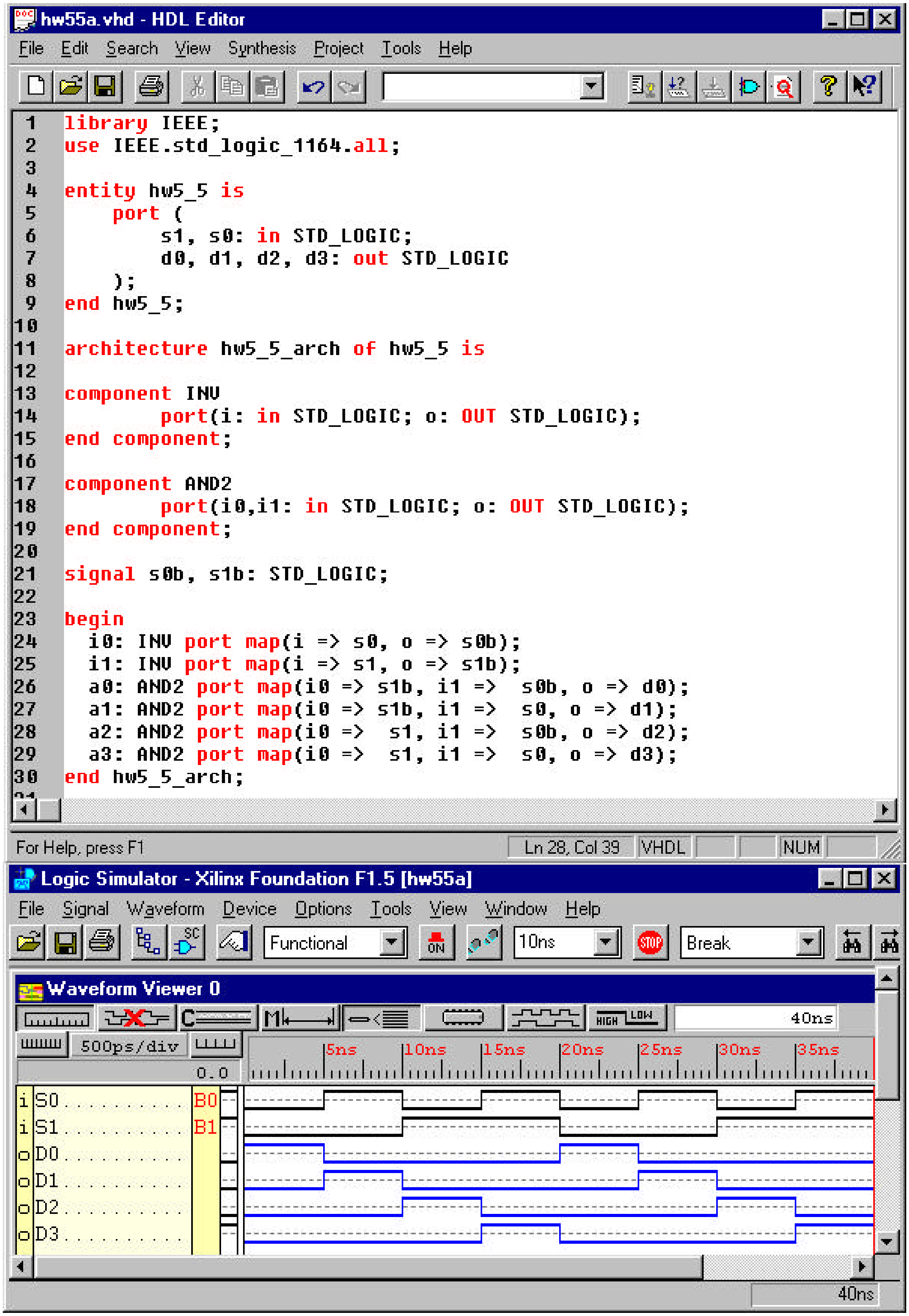

Schematic Editor - [Modified - HW4-41.SCH]
File Edit Mode Options Hierarchy View Display Window Help
A2 A1 A0 B2 B1 B0 C2 C1 C0
1C
1C
+1
+1
mux
mux
GND
adder
controller
X3 X2 X1 X0
V
HW4-41
17.4 , 13.5
Select and Drag


Logic Simulator - Xilinx Foundation F1.5 [hw4-4] - [Waveform Viewer 0]
File Signal Waveform Device Options Tools View Window Help
Functional
ON
20ns
Break
HIGH LOW
80ns
1ns/div
0.0
10ns 20ns 30ns 40ns 50ns 60ns 70ns
80ns

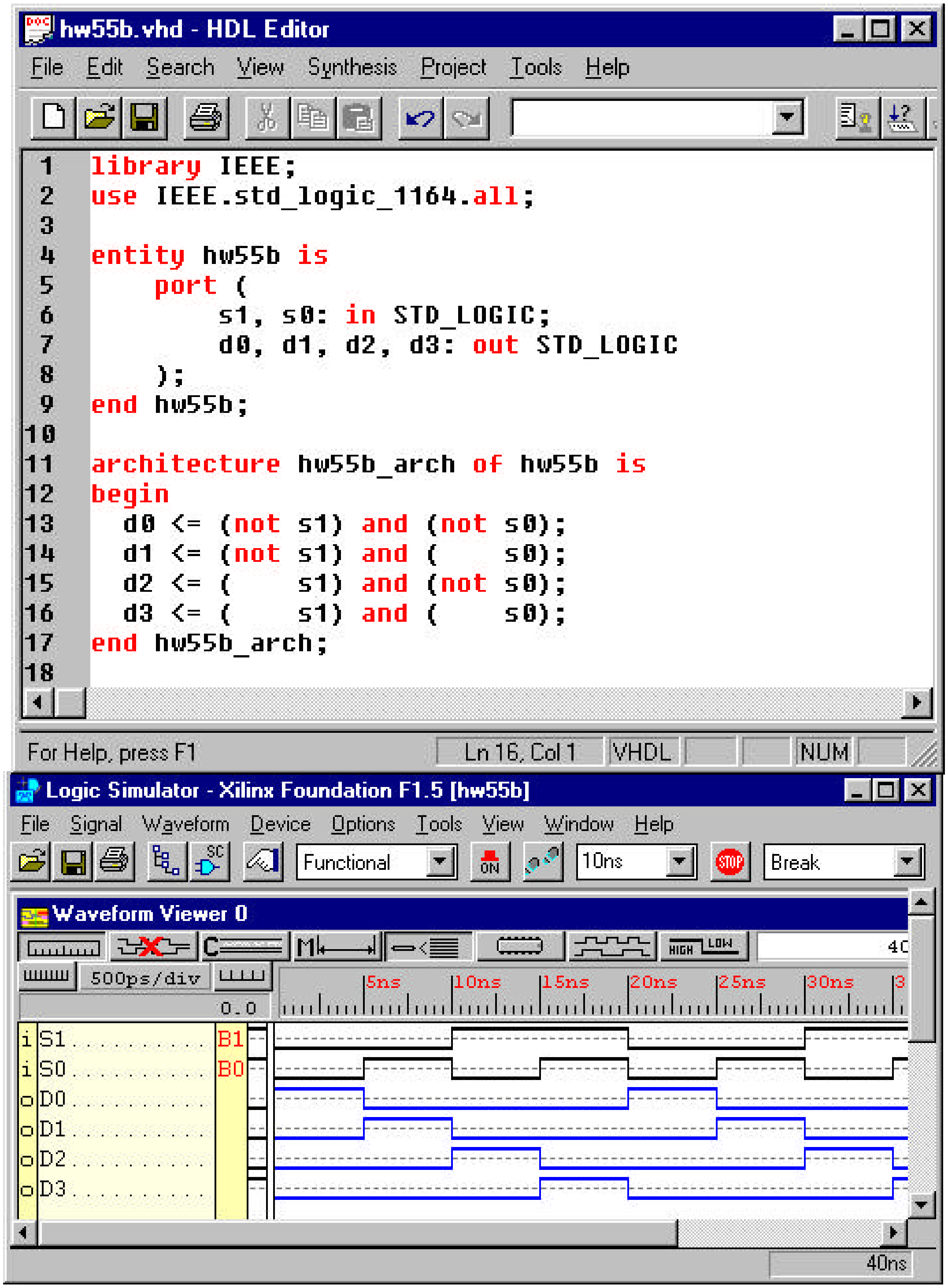
```
B C0...(hex)#3 * 7   0 1 2 3 4 5 6 7
B A0...(hex)#4 * E   5 6 A 5 6 3 5 4 2 5 E
B B0...(hex)#4 * 6   A 5 8 E 7 5 4 2 6 A 3 A 6
B X0...(hex)#4   8   5 A 8 3 C B 7 3 E A F 5 8
o V...........
```

5. Draw (by hand) a schematic diagram for a 2 input, 4 output decoder using simple 2 input gates. Label each input, output and gate. Write a structural VHDL specification that directly implements the circuit in your schematic. Simulate your decoder on all possible inputs. Write a dataflow VHDL specification of the decoder and simulate this version as well. Hand in the schematic, the VHDL listings and the waveforms from your simulation runs.





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

entity hw5_5 is
  port (
    s1, s0: in STD_LOGIC;
    d0, d1, d2, d3: out STD_LOGIC
  );
end hw5_5;

architecture hw5_5_arch of hw5_5 is

component INV
    port(i: in STD_LOGIC; o: OUT STD_LOGIC);
end component;

component AND2
    port(i0,i1: in STD_LOGIC; o: OUT STD_LOGIC);
end component;

signal s0b, s1b: STD_LOGIC;

begin
 i0: INV port map(i => s0, o => s0b);
 i1: INV port map(i => s1, o => s1b);
 a0: AND2 port map(i0 => s1b, i1 =>  s0b, o => d0);
 a1: AND2 port map(i0 => s1b, i1 =>  s0, o => d1);
 a2: AND2 port map(i0 =>  s1, i1 =>  s0b, o => d2);
 a3: AND2 port map(i0 =>  s1, i1 =>  s0, o => d3);
end hw5_5_arch;
```



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

entity hw55b is
    port (
        s1, s0: in STD_LOGIC;
        d0, d1, d2, d3: out STD_LOGIC
    );
end hw55b;

architecture hw55b_arch of hw55b is
begin
  d0 <= (not s1) and (not s0);
  d1 <= (not s1) and (    s0);
  d2 <= (    s1) and (not s0);
  d3 <= (    s1) and (    s0);
end hw55b_arch;
```