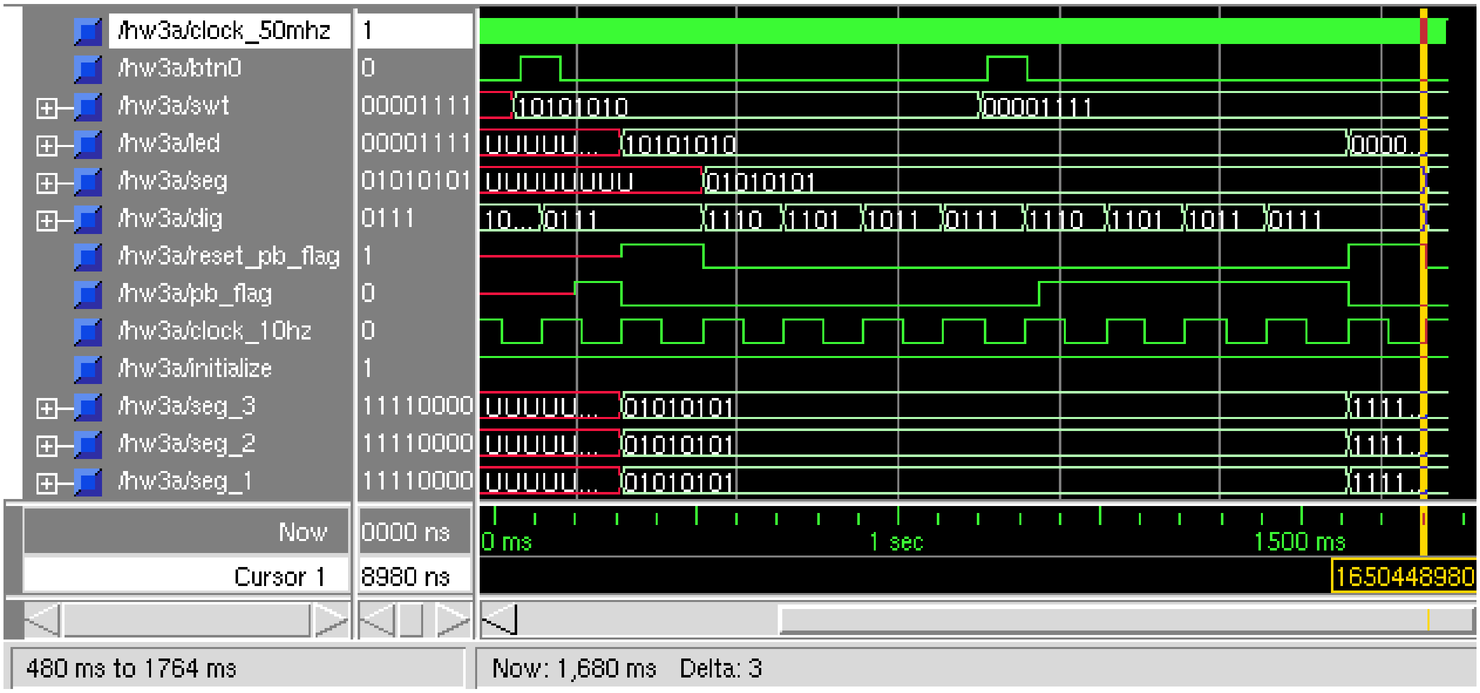
CHECKING THE SPARTAN3 INPUTS/OUTPUTS

Prof. Don Bouldin

1. **cp ~bouldin/webhome/protected/551-hw3a.tar.gz .**
2. **gunzip 551-hw3a.tar**
3. **tar –xvf 551-hw3a.tar**
4. **cd 551-hw3a**
5. **./presynth-sim**

This will bring up the following window:



```
/hw3a/clock_50mhz     1
/hw3a/btn0            0
/hw3a/swt             00001111
/hw3a/led             00001111
/hw3a/seg             01010101
/hw3a/dig             0111
/hw3a/reset_pb_flag   1
/hw3a/pb_flag         0
/hw3a/clock_10hz      0
/hw3a/initialize      1
/hw3a/seg_3           11110000
/hw3a/seg_2           11110000
/hw3a/seg_1           11110000
Now                   0000 ns
Cursor 1              8980 ns
```

480 ms to 1764 ms

Now: 1,680 ms   Delta: 3

After btn0 is pressed (properly), the pb_flag is set.
Then while dig "0111" is displayed , swt<7:0> are read and pb_flag is reset.

Now, synthesize the VHDL source file using *Synplify_Pro*
into the Spartan3 part by typing:

```
6.      synplify_premier_dp   –batch -tcl synplify-spartan3.tcl
```

The synthesized net-list is now under a subdirectory, rev_1, so copy the appropriate files:

```
7.      cp spartan3-fit                 rev_1

8.      cp spartan3-bitgen              rev_1

9.      cp stim-hw3a.do                 rev_1

10.     cp hw3a.ucf                    rev_1

11.     cp vsim-post-spartan3          rev_1

12.     cd                             rev_1
```

Generate the Spartan3 layout using the *Xilinx* fitter:

```
13.     ./spartan3-fit hw3a
```

Note the resources used by observing:

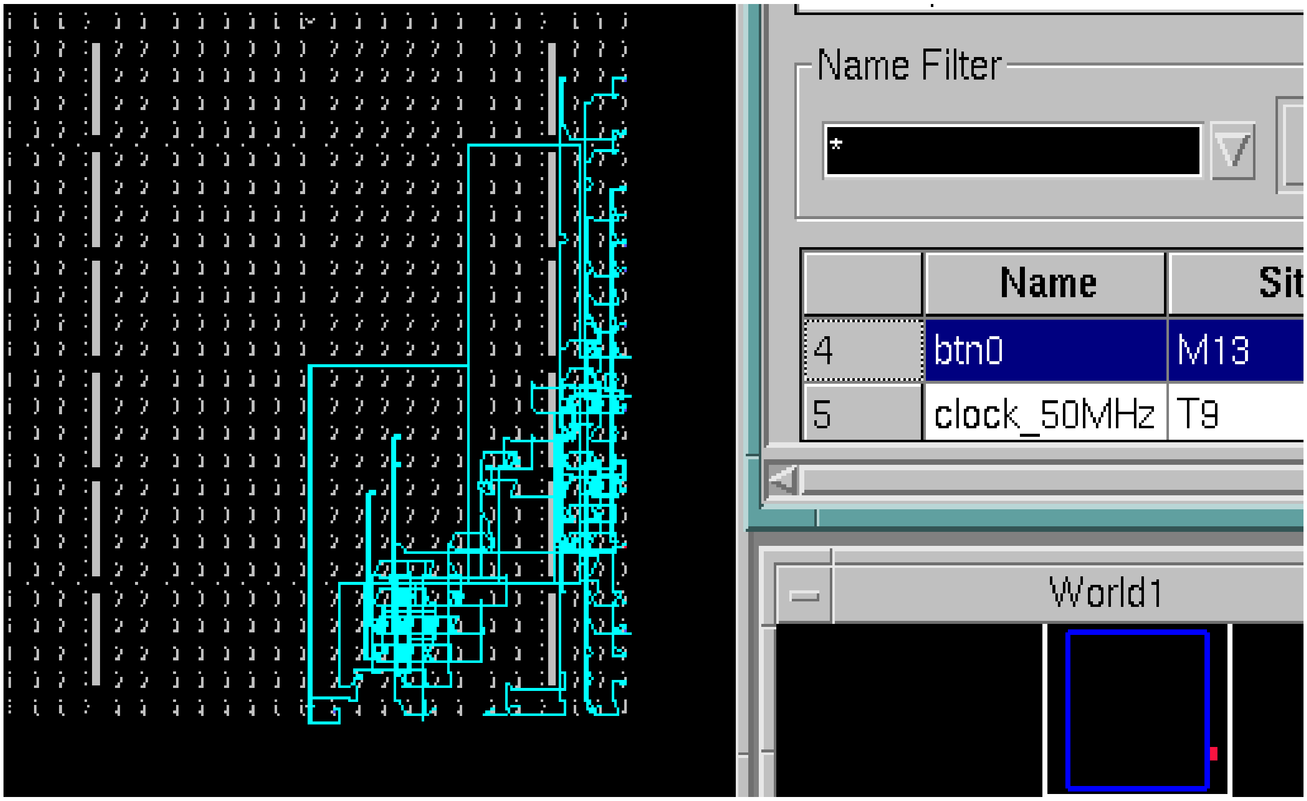
```
14.    grep Slices hw3a.mrp
```

Perform post-layout simulation by typing:

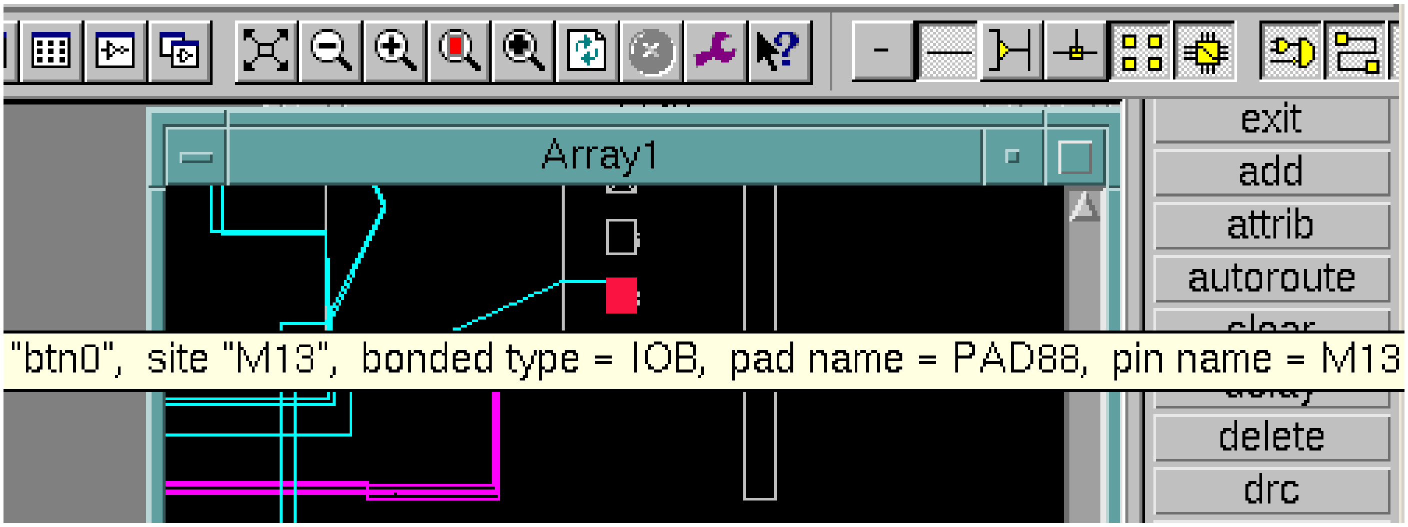
```
15.     ./vsim-post-spartan3   hw3a
```

Generate the configuration file (hw3a_r.bit) by typing:

```
16.    ./spartan3-bitgen
```



	Name	Sit
4	btn0	M13
5	clock_50MHz	T9



"btn0", site "M13", bonded type = IOB, pad name = PAD88, pin name = M13

Downloading

1. Use a PC which has a parallel port and the Xilinx ISE software installed.

2. Connect the power to the Spartan3 board. The digits should display “PASS”.

3. Connect the parallel cable between the PC and the JTAG connector on the board.

4. Transfer the file (hw3a_r.bit) to the PC using ssh and ftp.

5. Invoke IMPACT by selecting:

   START→All Programs→Xilinx ISE → Accessories → IMPACT

6. Under “mode”, select “Configuration”.

7. Right-click the mouse and select “cable auto connect”.

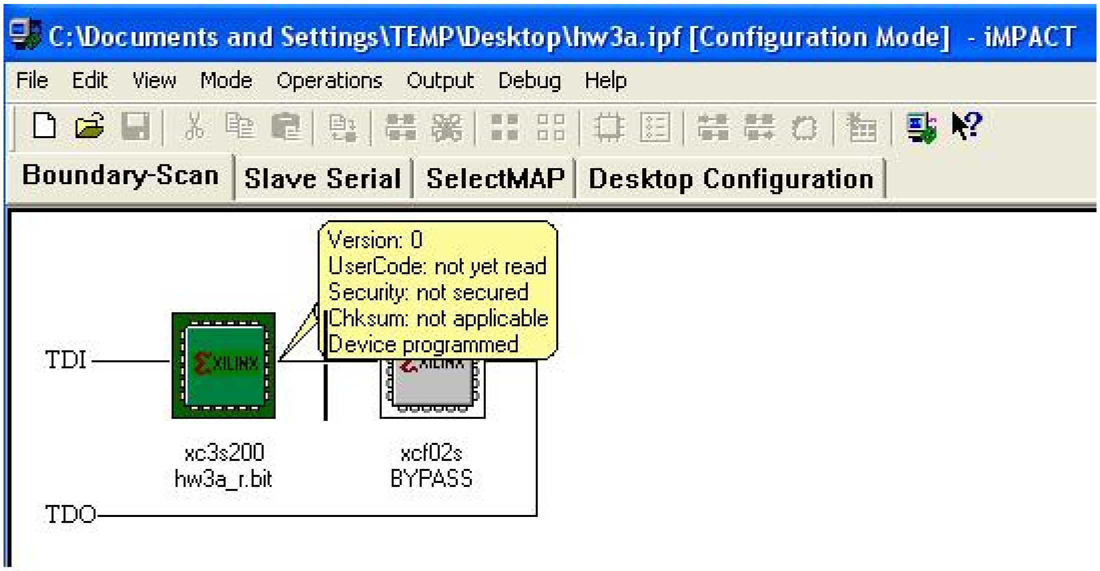
8. Right-click the mouse and select “initialize chain”.

9. Locate “hw3a_r.bit” and click OK.

10. At the next prompt, select BYPASS.

11. Point the mouse cursor over the xc3s200 part and right-click to select “PROGRAM”. Click OK.

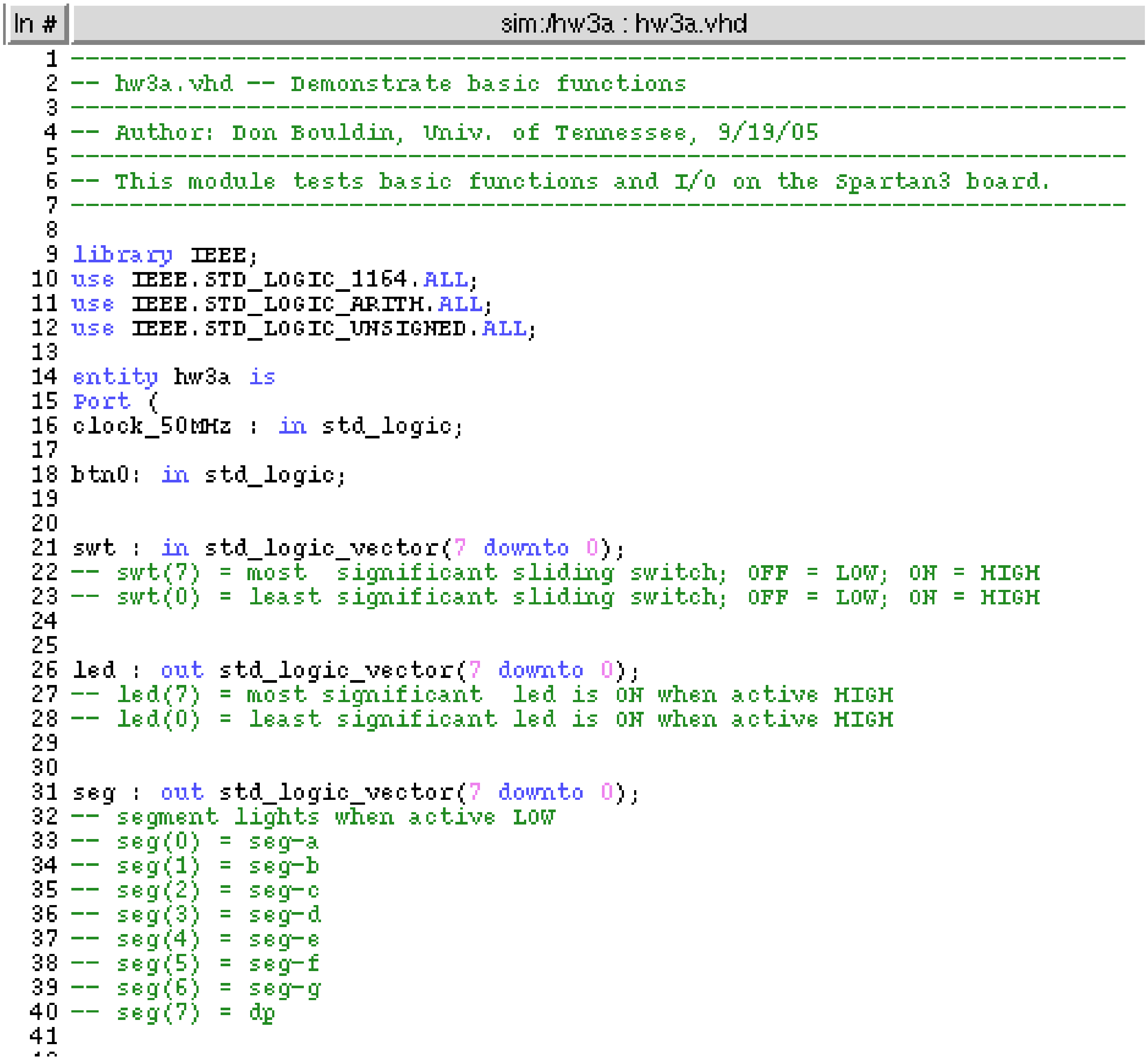
12. When finished, select File→ Exit.


C:\Documents and Settings\TEMP\Desktop\hw3a.ipf [Configuration Mode] - iMPACT
File Edit View Mode Operations Output Debug Help
Boundary-Scan
Slave Serial
SelectMAP
Desktop Configuration
TDI
XILINX
XILINX
xc3s200
hw3a_r.bit
xcf02s
BYPASS
TDO

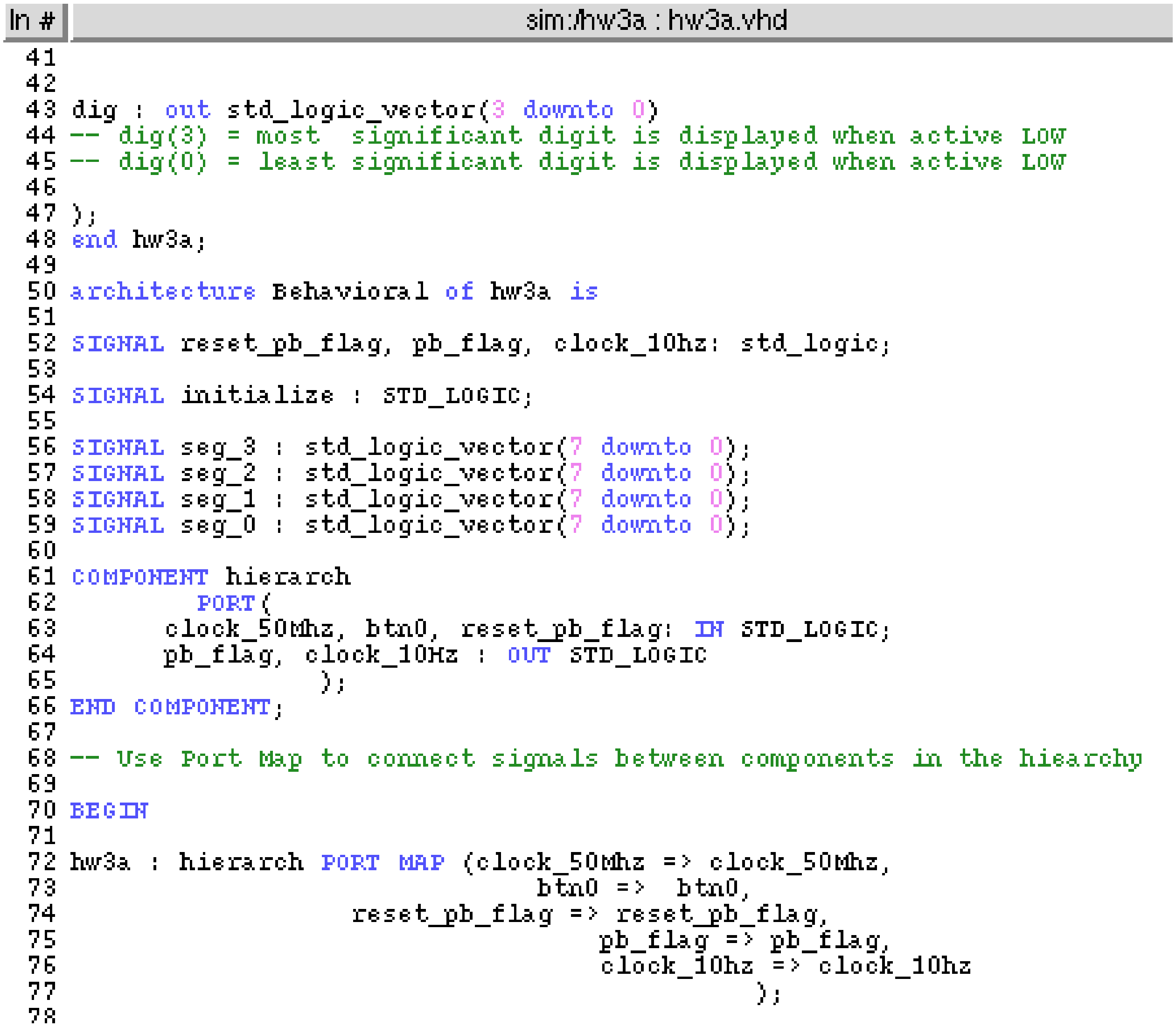
Version: 0
UserCode: not yet read
Security: not secured
Chksum: not applicable
Device programmed

APPENDIX

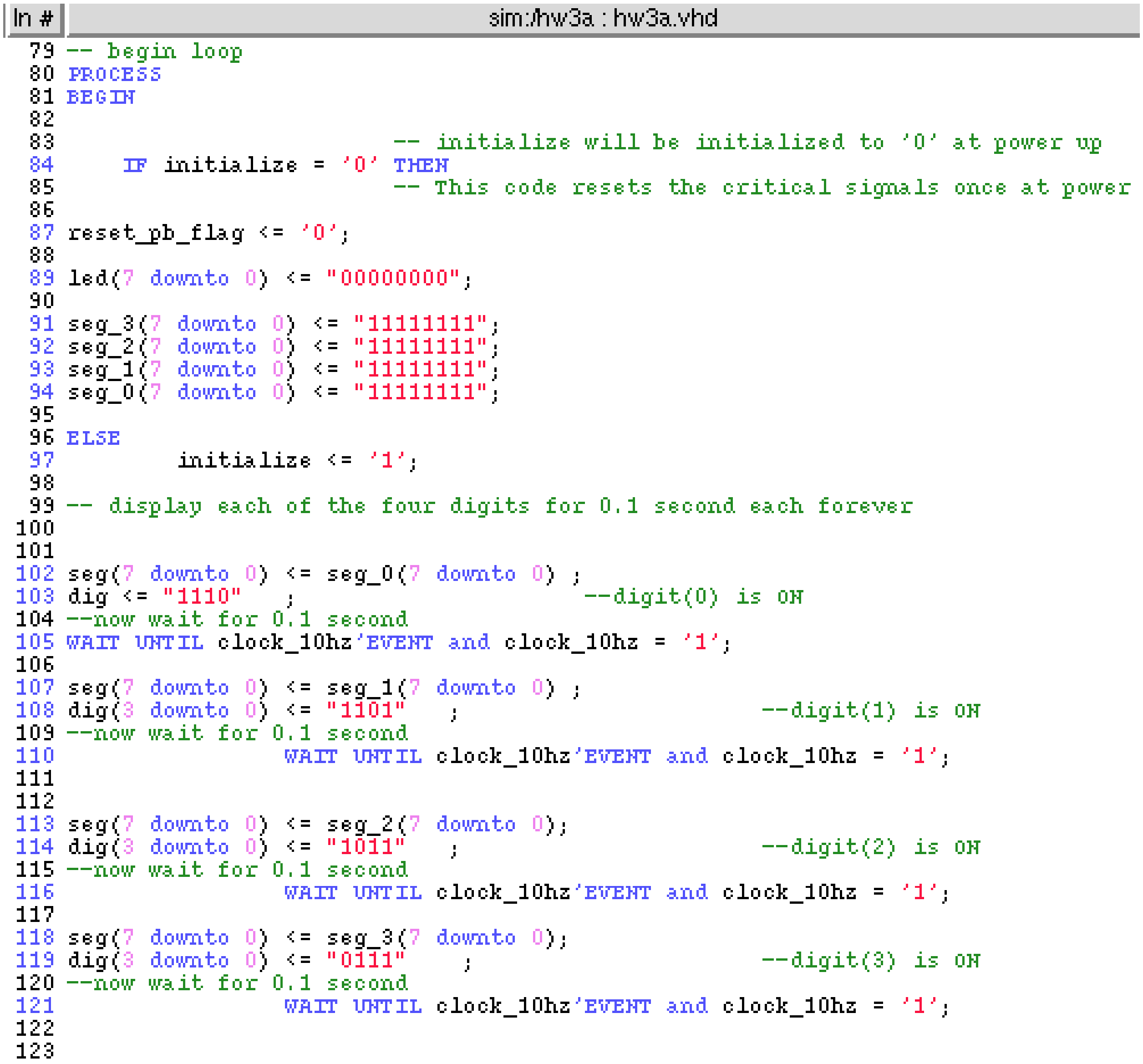
hw3a.vhd



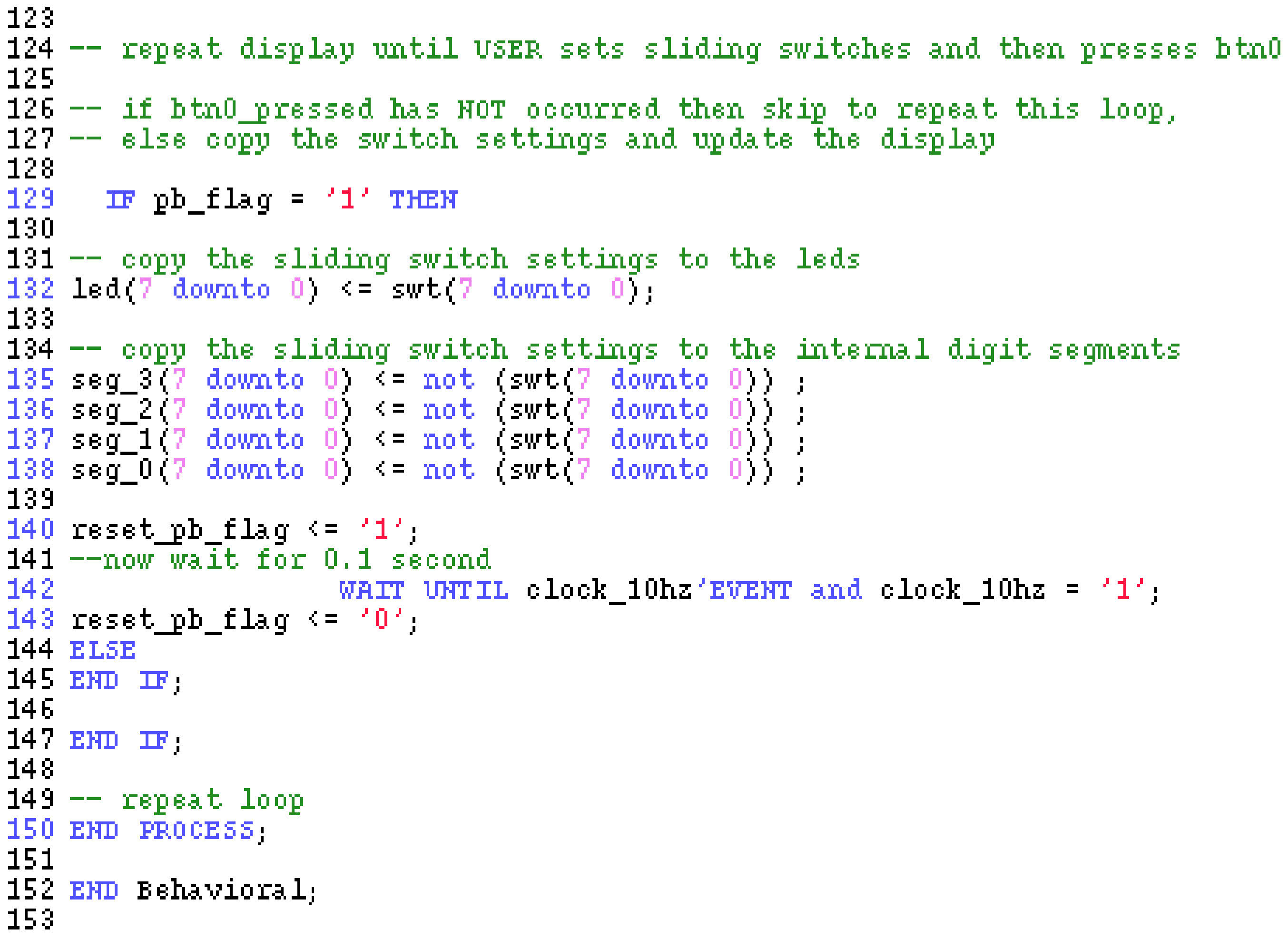
```
---------------------------------------------------------------------------
-- hw3a.vhd -- Demonstrate basic functions
---------------------------------------------------------------------------
-- Author: Don Bouldin, Univ. of Tennessee, 9/19/05
---------------------------------------------------------------------------
-- This module tests basic functions and I/O on the Spartan3 board.
---------------------------------------------------------------------------

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity hw3a is
Port (
clock_50MHz : in std_logic;

btn0: in std_logic;


swt : in std_logic_vector(7 downto 0);
-- swt(7) = most  significant sliding switch; OFF = LOW; ON = HIGH
-- swt(0) = least significant sliding switch; OFF = LOW; ON = HIGH


led : out std_logic_vector(7 downto 0);
-- led(7) = most significant  led is ON when active HIGH
-- led(0) = least significant led is ON when active HIGH


seg : out std_logic_vector(7 downto 0);
-- segment lights when active LOW
-- seg(0) = seg-a
-- seg(1) = seg-b
-- seg(2) = seg-c
-- seg(3) = seg-d
-- seg(4) = seg-e
-- seg(5) = seg-f
-- seg(6) = seg-g
-- seg(7) = dp

```



```


dig : out std_logic_vector(3 downto 0)
-- dig(3) = most  significant digit is displayed when active LOW
-- dig(0) = least significant digit is displayed when active LOW

);
end hw3a;

architecture Behavioral of hw3a is

SIGNAL reset_pb_flag, pb_flag, clock_10hz: std_logic;

SIGNAL initialize : STD_LOGIC;

SIGNAL seg_3 : std_logic_vector(7 downto 0);
SIGNAL seg_2 : std_logic_vector(7 downto 0);
SIGNAL seg_1 : std_logic_vector(7 downto 0);
SIGNAL seg_0 : std_logic_vector(7 downto 0);

COMPONENT hierarch
        PORT(
      clock_50Mhz, btn0, reset_pb_flag: IN STD_LOGIC;
      pb_flag, clock_10Hz : OUT STD_LOGIC
                );
END COMPONENT;

-- Use Port Map to connect signals between components in the hiearchy

BEGIN

hw3a : hierarch PORT MAP (clock_50Mhz => clock_50Mhz,
                              btn0 =>  btn0,
                  reset_pb_flag => reset_pb_flag,
                                  pb_flag => pb_flag,
                                  clock_10hz => clock_10hz
                                             );
```



```
-- begin loop
PROCESS
BEGIN

                        -- initialize will be initialized to '0' at power up
    IF initialize = '0' THEN
                        -- This code resets the critical signals once at power

reset_pb_flag <= '0';

led(7 downto 0) <= "00000000";

seg_3(7 downto 0) <= "11111111";
seg_2(7 downto 0) <= "11111111";
seg_1(7 downto 0) <= "11111111";
seg_0(7 downto 0) <= "11111111";

ELSE
        initialize <= '1';

-- display each of the four digits for 0.1 second each forever


seg(7 downto 0) <= seg_0(7 downto 0) ;
dig <= "1110"   ;                     --digit(0) is ON
--now wait for 0.1 second
WAIT UNTIL clock_10hz'EVENT and clock_10hz = '1';

seg(7 downto 0) <= seg_1(7 downto 0) ;
dig(3 downto 0) <= "1101"   ;                         --digit(1) is ON
--now wait for 0.1 second
                WAIT UNTIL clock_10hz'EVENT and clock_10hz = '1';


seg(7 downto 0) <= seg_2(7 downto 0);
dig(3 downto 0) <= "1011"   ;                         --digit(2) is ON
--now wait for 0.1 second
                WAIT UNTIL clock_10hz'EVENT and clock_10hz = '1';

seg(7 downto 0) <= seg_3(7 downto 0);
dig(3 downto 0) <= "0111"    ;                        --digit(3) is ON
--now wait for 0.1 second
                WAIT UNTIL clock_10hz'EVENT and clock_10hz = '1';


```



```

-- repeat display until USER sets sliding switches and then presses btn0

-- if btn0_pressed has NOT occurred then skip to repeat this loop,
-- else copy the switch settings and update the display

  IF pb_flag = '1' THEN

-- copy the sliding switch settings to the leds
led(7 downto 0) <= swt(7 downto 0);

-- copy the sliding switch settings to the internal digit segments
seg_3(7 downto 0) <= not (swt(7 downto 0)) ;
seg_2(7 downto 0) <= not (swt(7 downto 0)) ;
seg_1(7 downto 0) <= not (swt(7 downto 0)) ;
seg_0(7 downto 0) <= not (swt(7 downto 0)) ;

reset_pb_flag <= '1';
--now wait for 0.1 second
                WAIT UNTIL clock_10hz'EVENT and clock_10hz = '1';
reset_pb_flag <= '0';
ELSE
END IF;

END IF;

-- repeat loop
END PROCESS;

END Behavioral;

```

presynth-sim

```
#./presynth-sim $1
vlib work
vcom -work work $1.vhd
vsim $1 -do stim-$1.do
```

stim-hw3a.do

```
add wave btn0 pb_flag swt led seg dig
force clock_50mhz 1 0, 0 10 –r 20
force btn0 0
run 520 ms
force swt “10101010”
run 10 ms
force btn0 1
run 50 ms
force btn0 0
run 520 ms
force swt “00001111”
run 10 ms
force btn0 1
run 50 ms
force btn0 0
run 520 ms
```

synplify_spartan3.tcl

```
#synplify_premier_dp -batch -tcl synplify-spartan3.tcl
add_file hw3a.vhd
impl -add rev_1
impl -active "rev_1"
set_option -technology spartan3
set_option -part xc3s200
set_option -package ft256
set_option -grade -5
set_option -synthesis_onoff_pragma 0
#map options
set_option -frequency 50.00
set_option -fanout_limit 500
set_option -pipe 0
set_option -retiming 0
set_option -fixgatedclocks 0
project -run synthesis
```

spartan3-fit

```
#./spartan3-fit filename
source /usr/local/xilnx/10.1i/ISE/settings.sh
ngdbuild   $1.edf
map -cm speed  -timing $1.ngd
par $1.ncd -w $1_r.ncd
trce -u 100  $1_r.ncd -o $1_r.twr
netgen -sta -w $1_r.ncd $1_sta.v -ofmt verilog
netgen -sim -tb -w $1_r.ncd $1_sim.vhd -ofmt vhdl
```

spartan3-bitgen

```
#./spartan3-bitgen
source /usr/local/xilnx/9.1i/ISE/settings.sh
bitgen hw3a_r.ncd
```

vsim-post-spartan3

```
#./vsim-post-spartan3 filename
vlib work
vmap simprim /usr/local/xilinx/10.1/ISE/vhdl/mti_se/simprim
vcom -work work $1_sim.vhd
vsim $1 -do stim-$1.do -sdftyp $1_sim.sdf
```

hw3a.ucf

```
#hw3-05 I/O Pin Assignments
#
NET "btn0" LOC = "M13" ;
#NET "btn1" LOC = "M14" ;
#NET "btn2" LOC = "L13" ;
#NET "btn3" LOC = "L14" ;
#
NET "dig(0)" LOC = "E13" ;
NET "dig(1)" LOC = "F14" ;
NET "dig(2)" LOC = "G14" ;
NET "dig(3)" LOC = "d14" ;
NET "led(0)" LOC = "K12" ;
NET "led(1)" LOC = "P14" ;
NET "led(2)" LOC = "L12" ;
NET "led(3)" LOC = "N14" ;
NET "led(4)" LOC = "P13" ;
NET "led(5)" LOC = "N12" ;
NET "led(6)" LOC = "P12" ;
NET "led(7)" LOC = "P11" ;
NET "clock_50MHz" LOC = "T9" ;
NET "seg(0)" LOC = "E14" ;
NET "seg(1)" LOC = "G13" ;
NET "seg(2)" LOC = "N15" ;
NET "seg(3)" LOC = "P15" ;
NET "seg(4)" LOC = "R16" ;
NET "seg(5)" LOC = "F13" ;
NET "seg(6)" LOC = "N16" ;
NET "seg(7)" LOC = "P16" ;
NET "swt(0)" LOC = "F12" ;
NET "swt(1)" LOC = "G12" ;
NET "swt(2)" LOC = "H14" ;
NET "swt(3)" LOC = "H13" ;
NET "swt(4)" LOC = "J14" ;
NET "swt(5)" LOC = "J13" ;
NET "swt(6)" LOC = "K14" ;
NET "swt(7)" LOC = "K13" ;
```