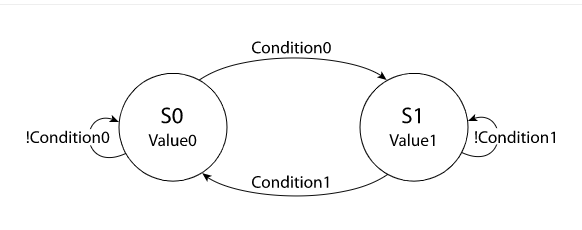
Active low digital stopwatch(DSW)

In entity we declared inputs(clk,start\_stop,min,sec) and outputs (finish,sec\_out,min\_out)

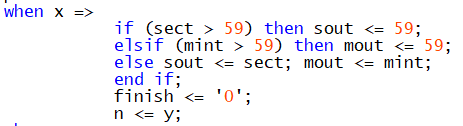
In architecture we can see that our project is one process state machine so we use type state

**ONE-PROCESS STATE MACHINE**

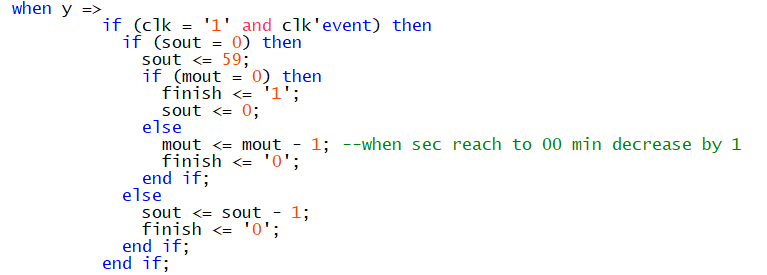
The one-process FSM design is the one you are most likely to encounter. It is usually implemented as a fully synchronous process. This results in the state, as well as the output being implemented in registers.



In this code we have two states (x,y) the output value of state signal (p,n). The vertices represent state changes, and the text on the vertices are boolean conditions that will cause the FSM to take that path. They are simply named Condition0



and Condition1



.Assume the output values are constants.

State-machines in VHDL are clocked processes whose outputs are controlled by the value of a state signal. The state signal serves as an internal memory of what happened in the previous iteration.

**Code**

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

use IEEE.STD\_LOGIC\_ARITH.ALL;

use IEEE.STD\_LOGIC\_UNSIGNED.ALL;

entity DSW is

port(clk,start\_stop : in std\_logic;

min,sec : in unsigned(5 downto 0);

finish : out std\_logic;

sec\_out,min\_out : out unsigned(5 downto 0));

end DSW;

architecture BEH of DSW is

type state is (x,y);

signal p,n : state:=x;

signal sect,mint,sout,mout : integer range 0 to 63; --(2to the power6)-1

begin

sect <= conv\_integer(sec); --convert binary to integer

mint <= conv\_integer(min);

sec\_out <= conv\_unsigned(sout,6); --6 bit convert like "111011"=59 and asigned sec\_out in sout signal

min\_out <= conv\_unsigned(mout,6);

process(clk,start\_stop) --process senstive to clock and start\_stop

begin

if (start\_stop = '1') then --initial value recorded before start

p <= n;

case p is

when x =>

if (sect > 59) then sout <= 59;

elsif (mint > 59) then mout <= 59;

else sout <= sect; mout <= mint; -- active low stopwatch

end if;

finish <= '0';

n <= y;

when y =>

if (clk = '1' and clk'event) then

if (sout = 0) then

sout <= 59;

if (mout = 0) then

finish <= '1'; --end process

sout <= 0;

else

mout <= mout - 1; --minutes countdown

finish <= '0';

end if;

else

sout <= sout - 1; --seconds countdown

finish <= '0';

end if;

end if;

end case;

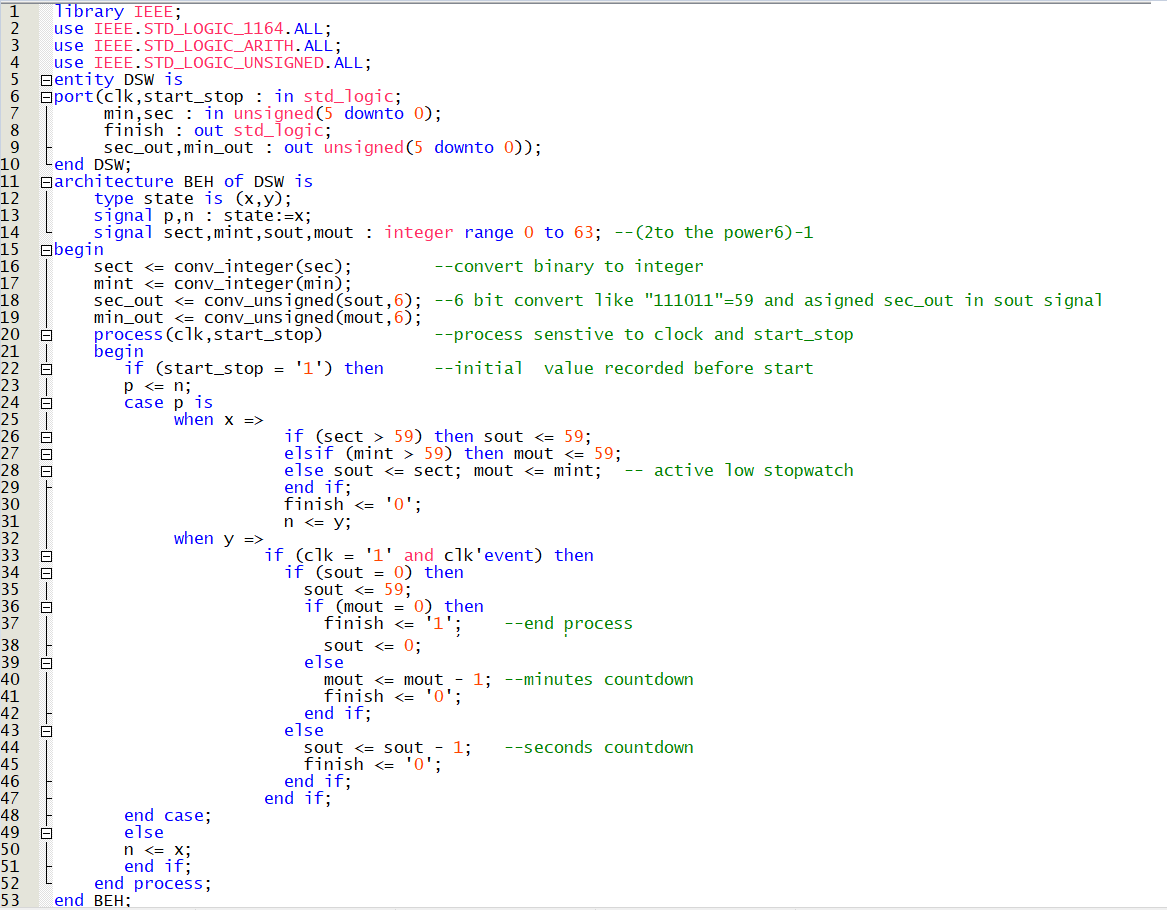
else

n <= x;

end if;

end process;

end BEH;



Schematic

