TECNICAS DIGITALES I

DESCRIPCION DE MAQUINAS DE ESTADOS SINCRONAS EN VHDL

Diagrama General

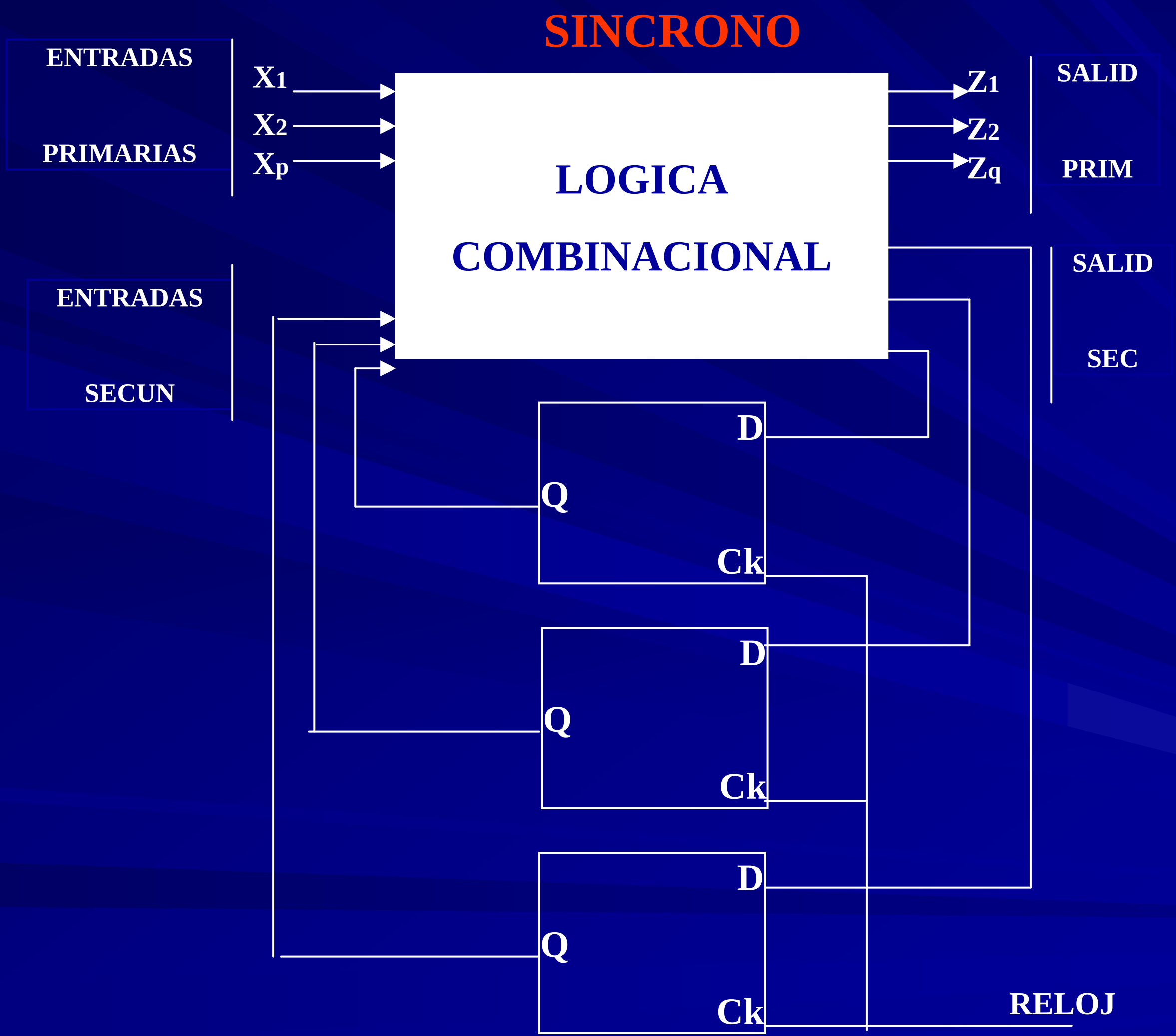
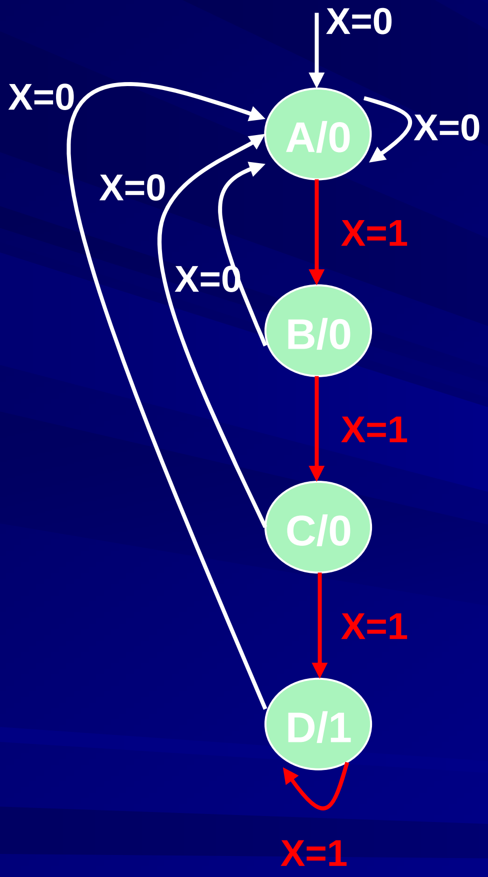
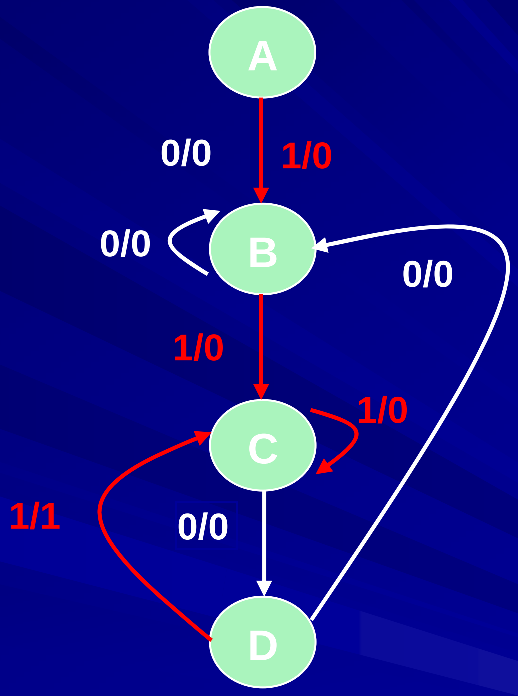
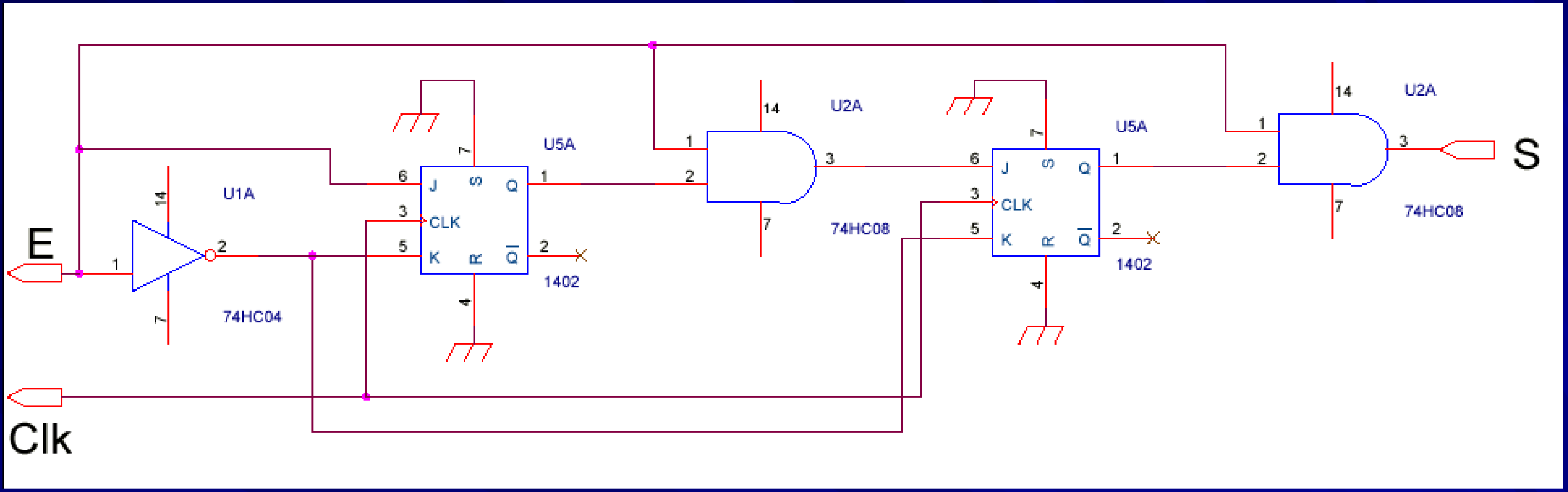


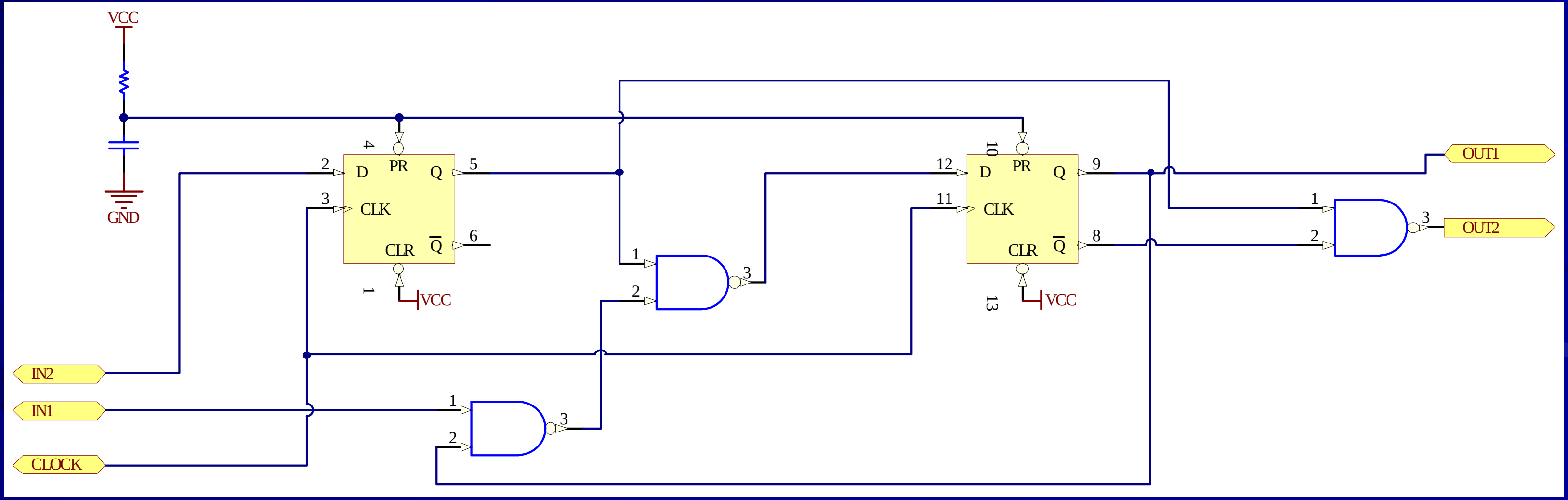
Diagrama de estados



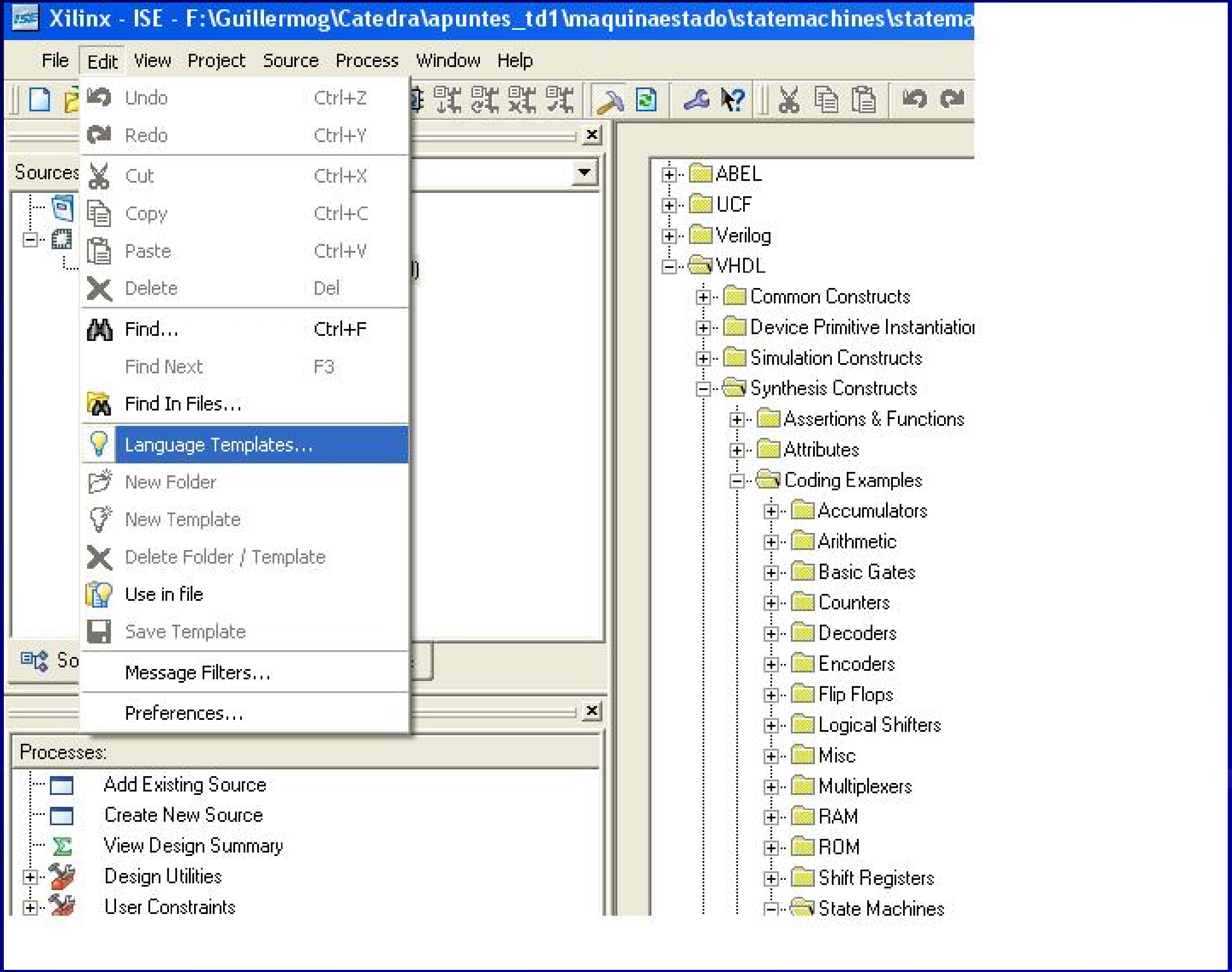


Moore y Mealy



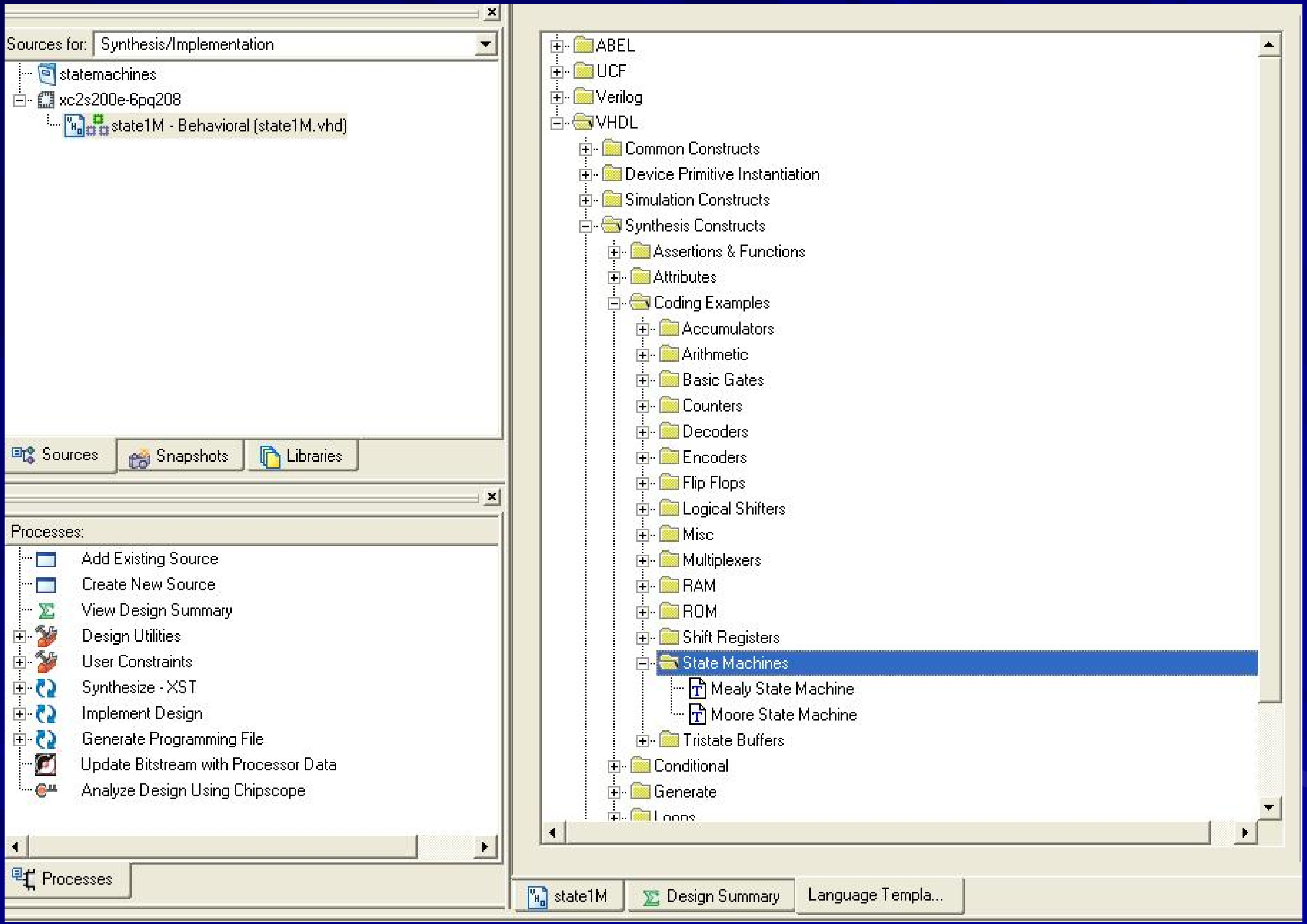


VHDL



Xilinx - ISE - F:\Guillermog\Catedra\apuntes_td1\maquinaestado\statemachines\statema

VHDL



Sources for: Synthesis/Implementation

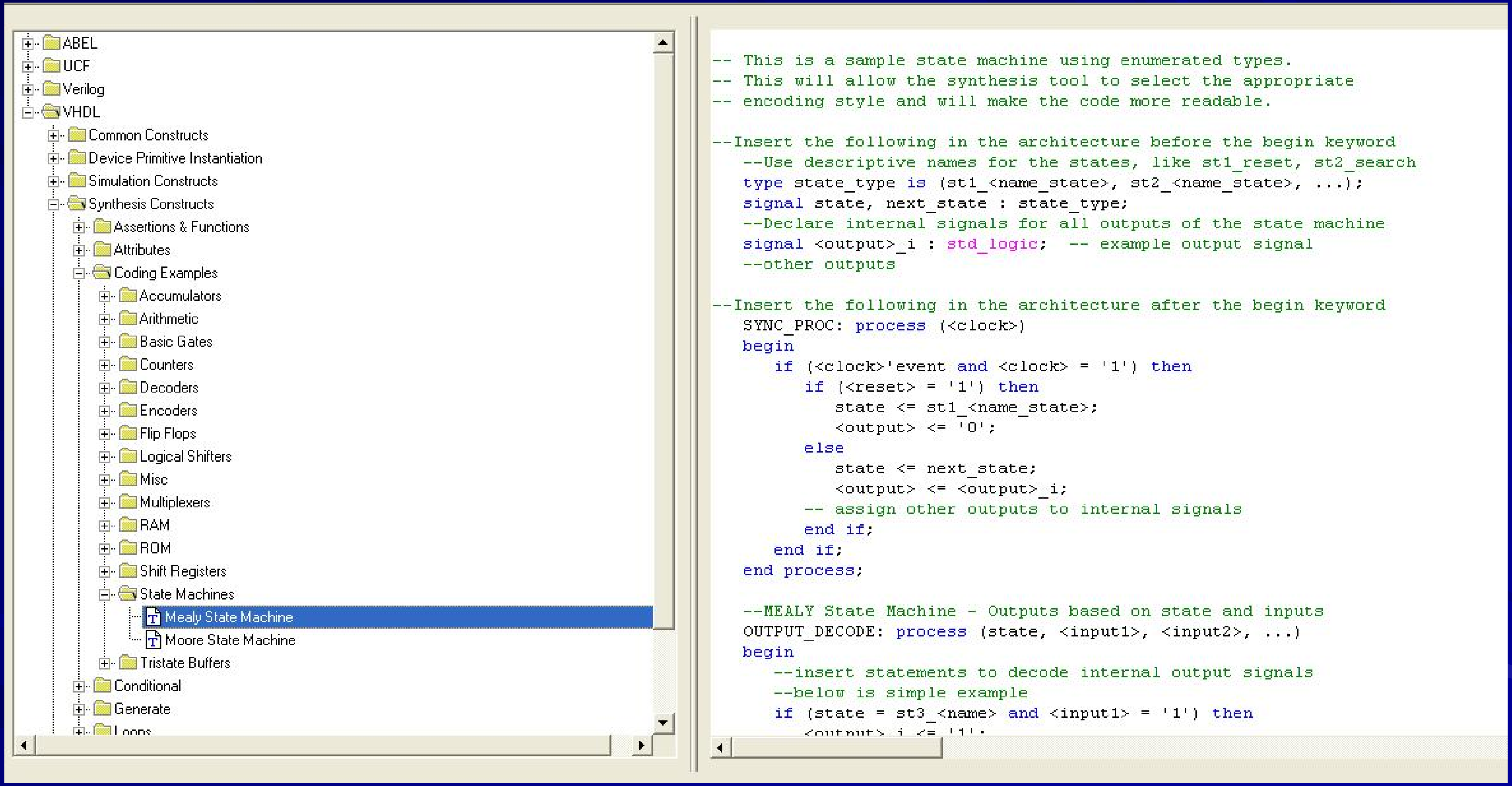
- statemachines
- xc2s200e-6pq208
  - state1M - Behavioral (state1M.vhd)

Processes:

- Add Existing Source
- Create New Source
- View Design Summary
- Design Utilities
- User Constraints
- Synthesize - XST
- Implement Design
- Generate Programming File
- Update Bitstream with Processor Data
- Analyze Design Using Chipscope

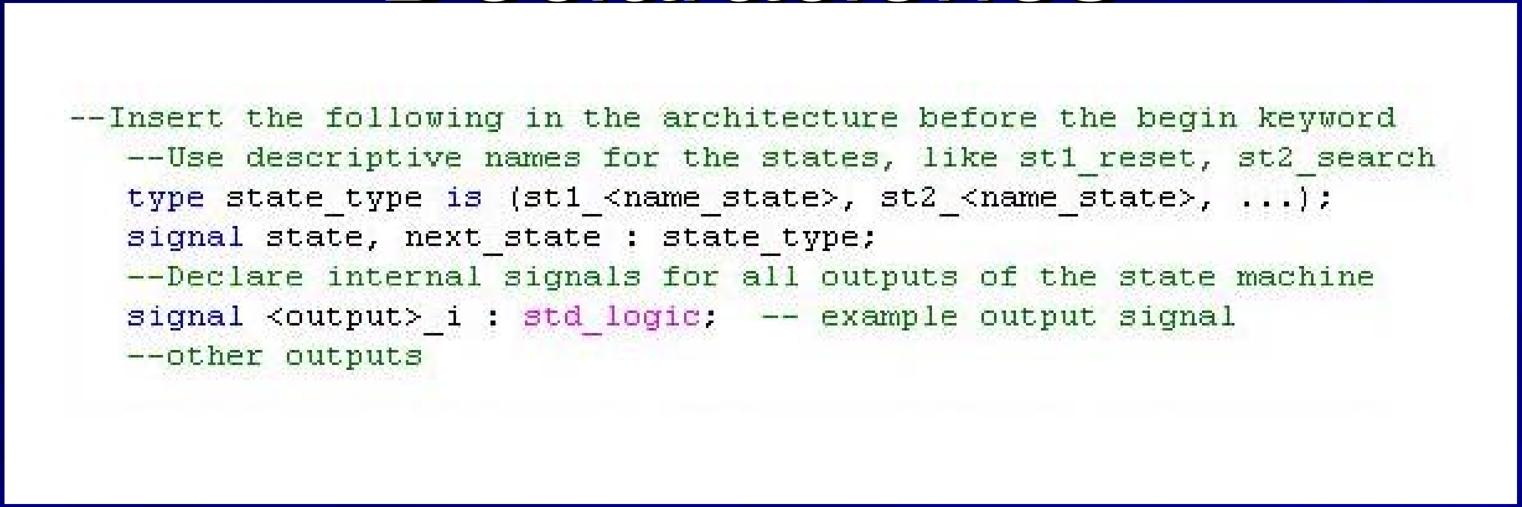
- ABEL
- UCF
- Verilog
- VHDL
  - Common Constructs
  - Device Primitive Instantiation
  - Simulation Constructs
  - Synthesis Constructs
    - Assertions & Functions
    - Attributes
    - Coding Examples
      - Accumulators
      - Arithmetic
      - Basic Gates
      - Counters
      - Decoders
      - Encoders
      - Flip Flops
      - Logical Shifters
      - Misc
      - Multiplexers
      - RAM
      - ROM
      - Shift Registers
      - State Machines
        - Mealy State Machine
        - Moore State Machine
      - Tristate Buffers
    - Conditional
    - Generate
    - Loops

VHDL



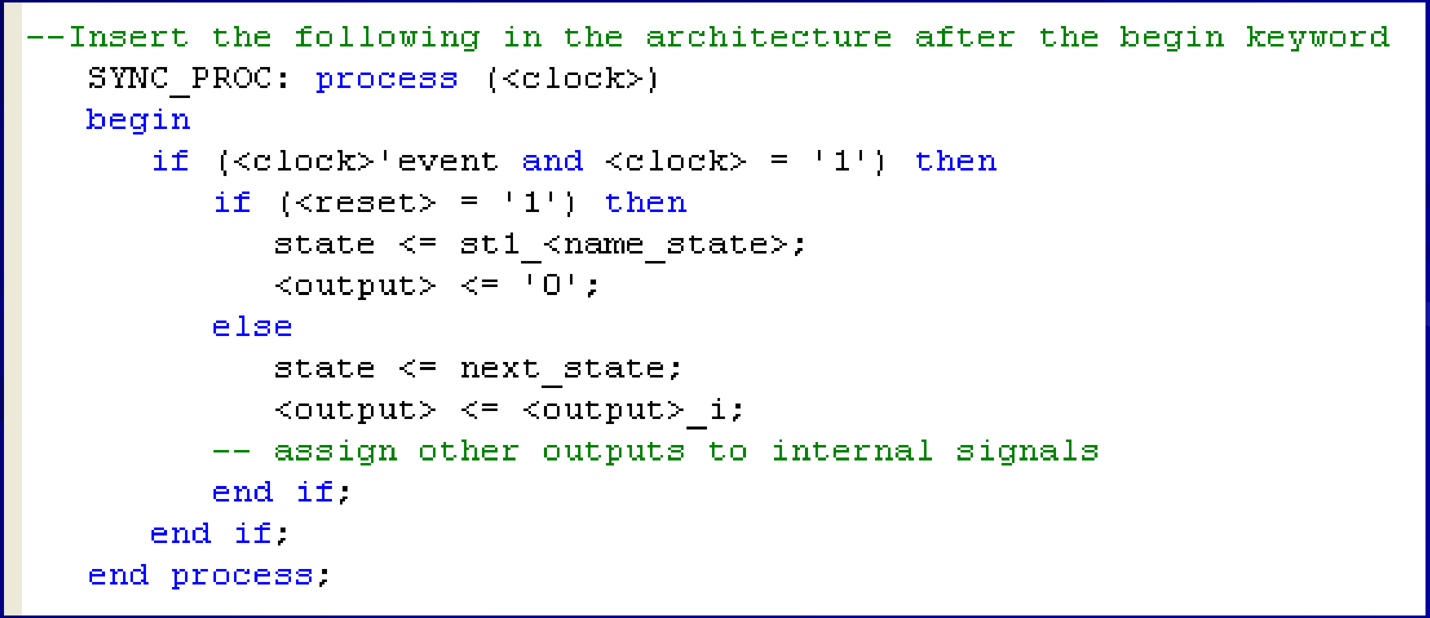
```
-- This is a sample state machine using enumerated types.
-- This will allow the synthesis tool to select the appropriate
-- encoding style and will make the code more readable.

--Insert the following in the architecture before the begin keyword
   --Use descriptive names for the states, like st1_reset, st2_search
   type state_type is (st1_<name_state>, st2_<name_state>, ...);
   signal state, next_state : state_type;
   --Declare internal signals for all outputs of the state machine
   signal <output>_i : std_logic;  -- example output signal
   --other outputs

--Insert the following in the architecture after the begin keyword
   SYNC_PROC: process (<clock>)
   begin
      if (<clock>'event and <clock> = '1') then
         if (<reset> = '1') then
            state <= st1_<name_state>;
            <output> <= '0';
         else
            state <= next_state;
            <output> <= <output>_i;
         -- assign other outputs to internal signals
         end if;
      end if;
   end process;

   --MEALY State Machine - Outputs based on state and inputs
   OUTPUT_DECODE: process (state, <input1>, <input2>, ...)
   begin
      --insert statements to decode internal output signals
      --below is simple example
      if (state = st3_<name> and <input1> = '1') then
```

Declaraciones



```
--Insert the following in the architecture before the begin keyword
   --Use descriptive names for the states, like st1_reset, st2_search
   type state_type is (st1_<name_state>, st2_<name_state>, ...);
   signal state, next_state : state_type;
   --Declare internal signals for all outputs of the state machine
   signal <output>_i : std_logic;  -- example output signal
   --other outputs
```

Transición de estado



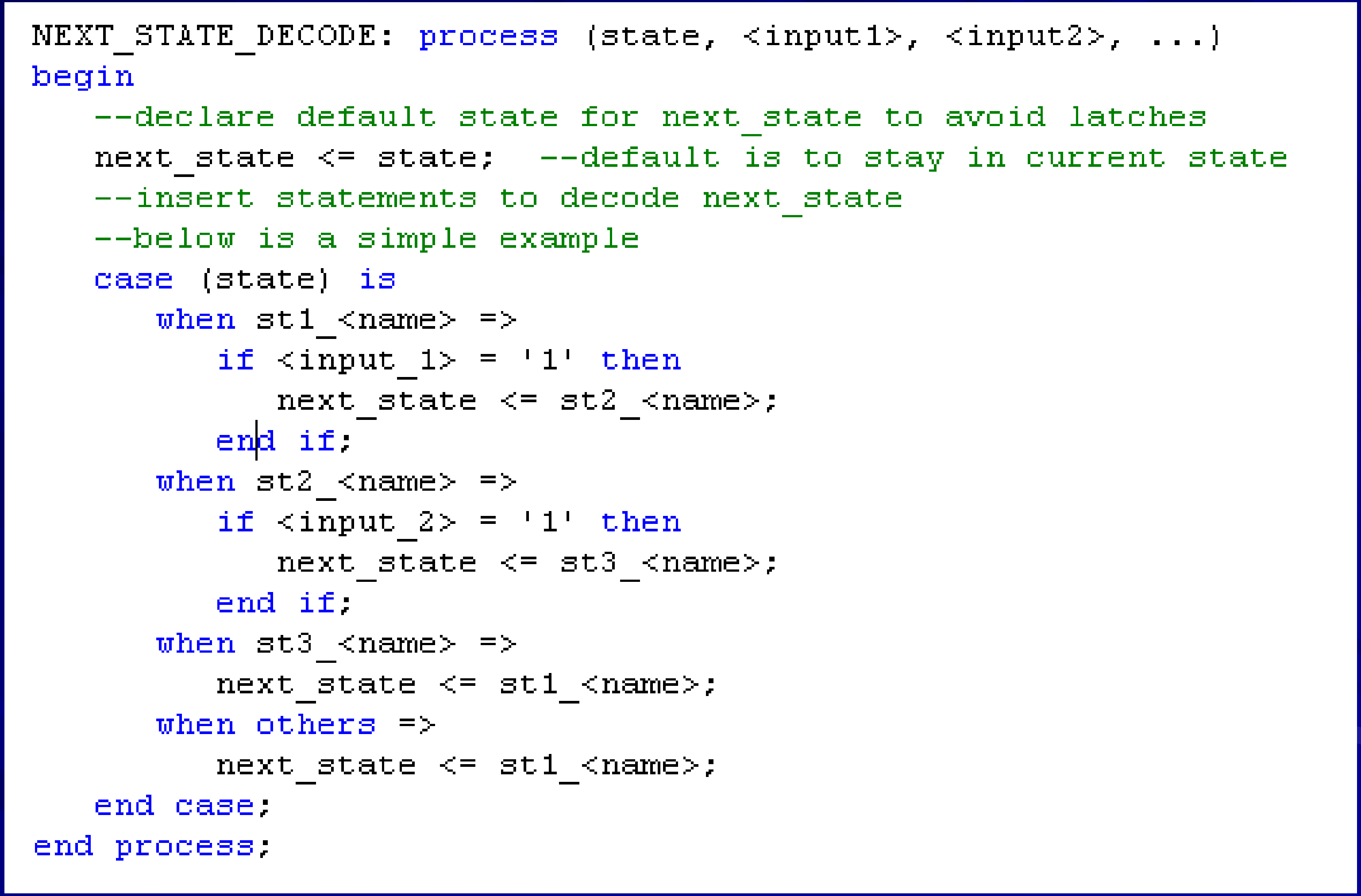
```
--Insert the following in the architecture after the begin keyword
   SYNC_PROC: process (<clock>)
   begin
      if (<clock>'event and <clock> = '1') then
         if (<reset> = '1') then
            state <= st1_<name_state>;
            <output> <= '0';
         else
            state <= next_state;
            <output> <= <output>_i;
         -- assign other outputs to internal signals
         end if;
      end if;
   end process;
```

Lógica de salida

```
--MEALY State Machine - Outputs based on state and inputs
OUTPUT_DECODE: process (state, <input1>, <input2>, ...)
begin
   --insert statements to decode internal output signals
   --below is simple example
   if (state = st3_<name> and <input1> = '1') then
      <output>_i <= '1';
   else
      <output>_i <= '0';
   end if;
end process;
```

```
--MOORE State Machine - Outputs based on state only
OUTPUT_DECODE: process (state)
begin
   --insert statements to decode internal output signals
   --below is simple example
   if state = st3_<name> then
      <output>_i <= '1';
   else
      <output>_i <= '0';
   end if;
end process;
```

Lógica de transición



```
NEXT_STATE_DECODE: process (state, <input1>, <input2>, ...)
begin
   --declare default state for next_state to avoid latches
   next_state <= state;  --default is to stay in current state
   --insert statements to decode next_state
   --below is a simple example
   case (state) is
      when st1_<name> =>
         if <input_1> = '1' then
            next_state <= st2_<name>;
         end if;
      when st2_<name> =>
         if <input_2> = '1' then
            next_state <= st3_<name>;
         end if;
      when st3_<name> =>
         next_state <= st1_<name>;
      when others =>
         next_state <= st1_<name>;
   end case;
end process;
```