Steps in the Design of a Controller

Introduction

- Processor
  - Digital circuit that performs a computation tasks
  - It consists of a **controller** and a datapath
- Processor Types
  - Application-specific: variety of computation tasks
  - Single-purpose: one particular computation task
  - Custom single-purpose: non-standard task
- A Custom Single-Purpose Processor may be
  - Fast, small, low power
  - But, high NRE, longer time-to-market, less flexible

Objectives

- Identify methods used for a controller design
  - Finite State Machine (FSM)
    - Moore versus Mealy
    - FSM in VHDL
  - Controller Design

Combinational logic design

Combinational circuit is a logic circuit whose outputs at any time are determined directly and only from the present input combinations.

Combinational logic design process:

Step 1: Capture behavior

- Capture the function

Step 2: Convert to circuit

- Create equations
- Implement as gate-based circuit

Combinational logic design Example

A) Problem description

y is 1 if a is to 1, or b and c are 1. z is 1 if b or c is to 1, but not both, or if all are 1.

B) Truth table

Inputs			Outputs	
a	b	c	y	z
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	1	0
1	0	1	1	1
1	1	0	1	1
1	1	1	1	1

C) Output equations

y = a'bc + ab'c' + ab'c + abc' + abc

z = a'b'c + a'bc' + ab'c + abc' + abc

D) Minimized output equations

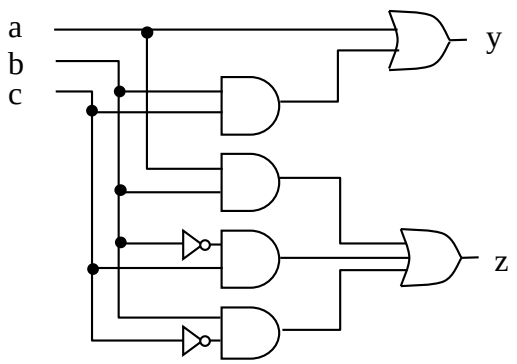
y: a \ bc	00	01	11	10
0	0	0	1	0
1	1	1	1	1

y = a + bc

z: a \ bc	00	01	11	10
0	0	1	0	1
1	0	1	1	1

z = ab + b'c + bc'

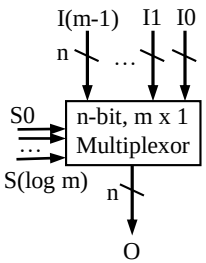
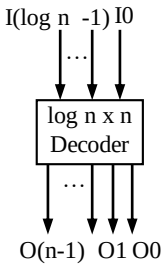
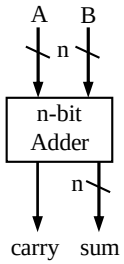
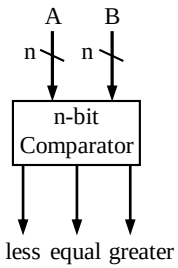
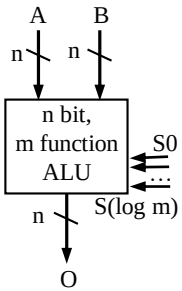
E) Logic Gates



RT-Level Combinational Components

- Register transfer (RT) methodology is used for the design of large combinational logic circuits.
- RT is a higher level of abstraction than logic gates.
- The building blocks of RT-Level design are shown, next slide.

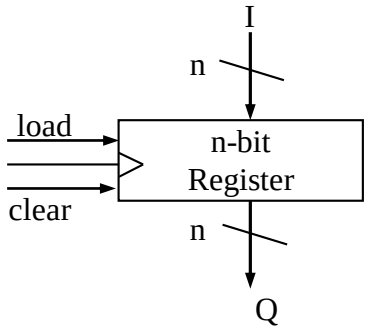
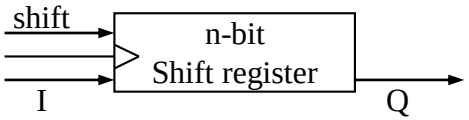
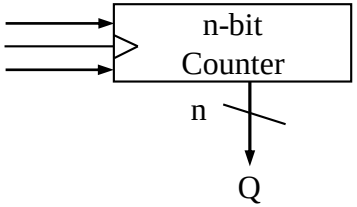
RT-Level Combinational Components

				
O = I0 if S=0..00 I1 if S=0..01 … I(m-1) if S=1..11	O0 =1 if I=0..00 O1 =1 if I=0..01 … O(n-1) =1 if I=1..11	sum = A+B (first n bits) carry = (n+1)'th bit of A+B	less = 1 if A<B equal =1 if A=B greater=1 if A>B	O = A *op* B *op* determined by S.
	With enable input e → all O's are 0 if e=0	With carry-in input Ci→ sum = A + B + Ci		May have status outputs carry, zero, etc.

Sequential components

Sequential circuit employs memory element in addition to combinational logic gate. Their outputs are determined from the present input as well as the past states, on the memory cells.

Sequential components

		
Q = 0 if clear=1, I if load=1 and clock=1, Q(previous) otherwise.	Q = lsb - Content shifted - I stored in msb	Q = 0 if clear=1, Q(prev)+1 if count=1 and clock=1.

Finite-State Machine (FSM)

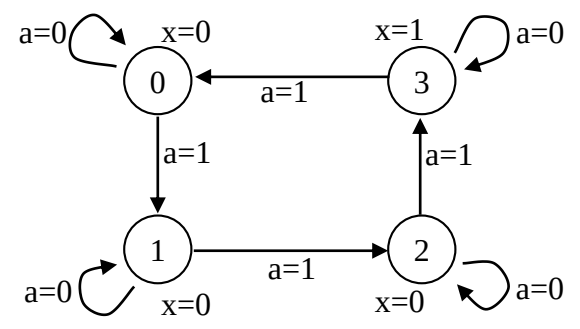
For sequential behavior, a Boolean equation is not enough, a more powerful mathematical formalism called FSM is used:

- Each state represents the current 'mode' of the circuit, serving as the circuit's memory of past input values
- In Moore FSM
  - desired out put values are listed next to each state
  - Input conditions that cause a transition from one state to another are shown next to each arc
  - Each arc condition is ANDed with rising (or falling) clock edge, all inputs are synchronous
- All inputs and outputs must be Boolean, all operations must be Boolean operations.

A) Problem Description

You want to construct a frequency divider. Slow down your pre-existing frequency so that you output a 1 cycle for every four cycles

B) State Diagram



Finite State Machines

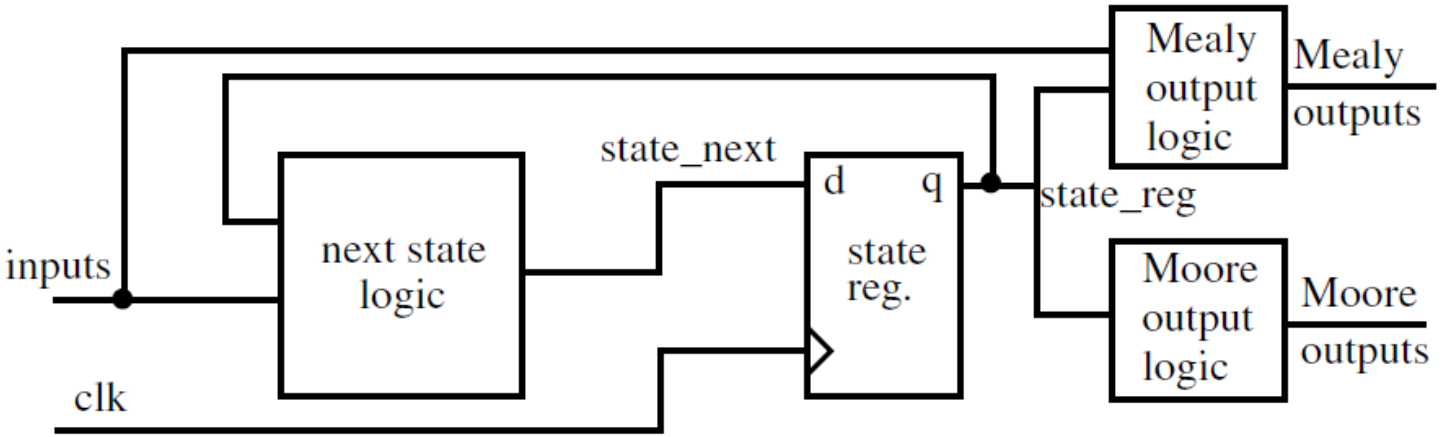
FSMs are sequential machines with "random" next-state logic

Used to implement functions that are realized by carrying out a **sequence of steps** -- commonly used as a controller in a large system

The state transitions within an FSM are more complicated than for regular sequential logic such as a shift register

An FSM is specified using five entities: *symbolic states*, *input signals*, *output signals*, *next-state function* and *output function*

- Mealy vs Moore output



Mealy versus Moore FSM

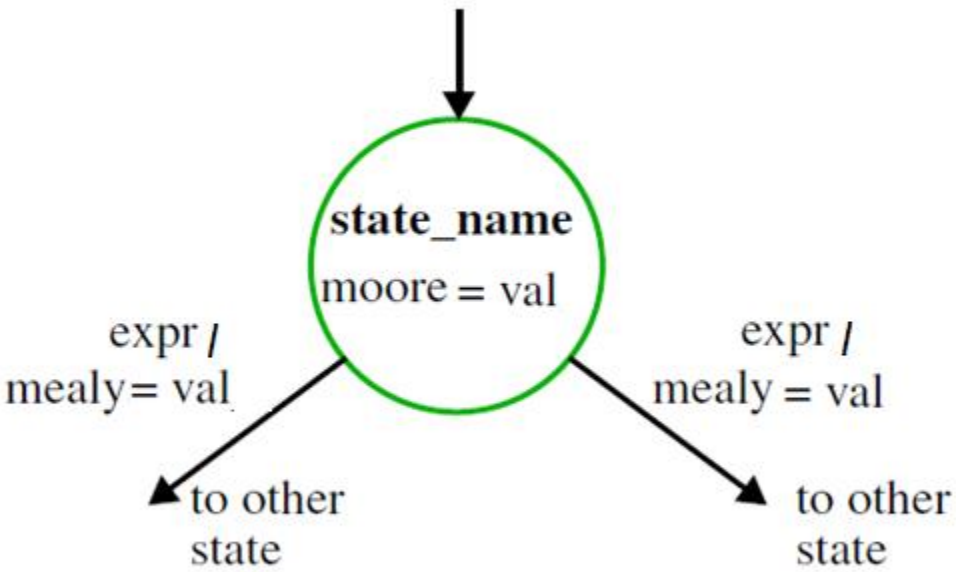
- Mealy FSM
  - Output depends both on current state and inputs.
- Moore FSM
  - Output depends only on current state.

In theory, for any Mealy machine, there is an equivalent Moore machine, and vice versa.

In practice, one or the other will be most appropriate.

Finite State Machine

State diagram



A *node* represents a unique state

An *arc* represents a transition from one state to another
Is labeled with the condition that causes the transition

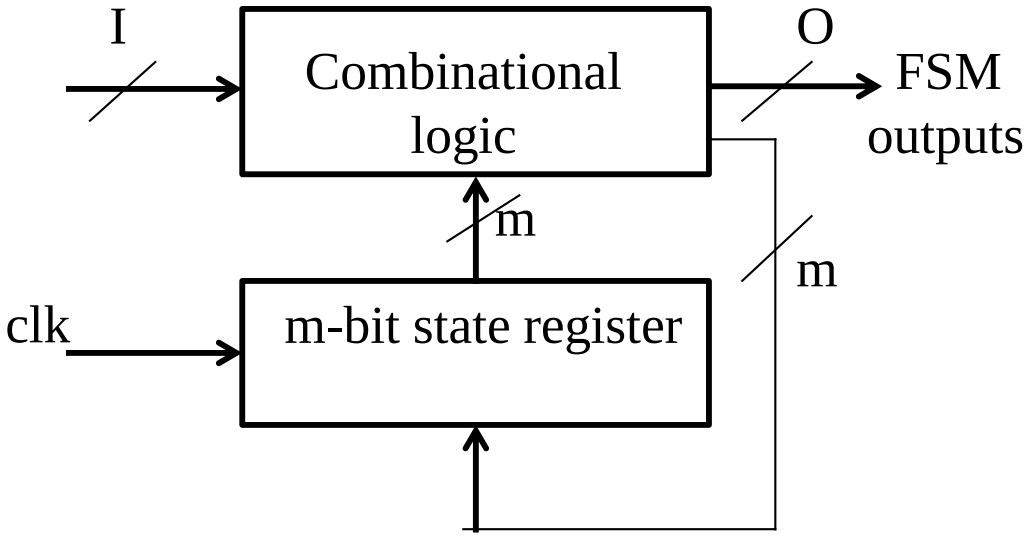
Moore outputs are shown inside the bubble

Mealy outputs are shown on the arcs

Only asserted outputs are listed

Controller Design

- The sequential circuit that implements an FSM is called a controller. A standard controller architecture consists of a register and combinational logic.
- Combinational logic provides for next state and the output logic.



Controller Design Process

Step1: Capture behavior

- Capture the Finite State Machine (FSM), *FSM describes the desired behavior of the controller. A state diagram is a graphical drawing of FSM.*

Step 2: Convert to circuit

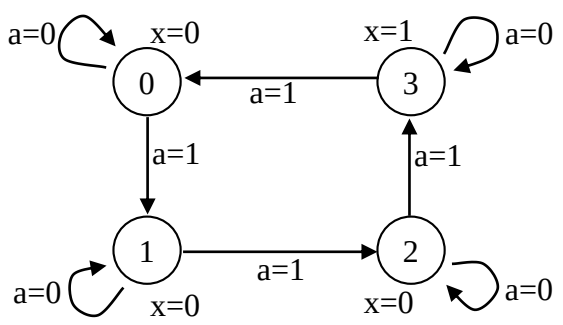
- Set up the standard architecture, by using a state register of appropriate width and combinational logic
  - State register width = $\log_2$(number of states)
- Encode states by assigning a unique binary number to each state
- Fill in the truth table, translate the FSM into a truth table for the combinational logic such that the logic will generate the outputs and next state signals for the given FSM
- Implement the combinational logic.

Sequential logic (Controller) design

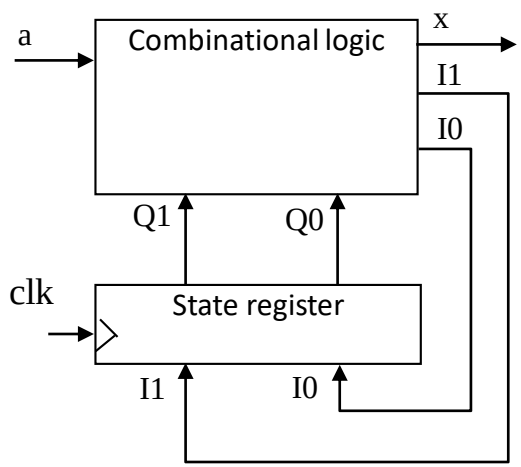
A) Problem Description

You want to construct a frequency divider. Slow down your pre-existing frequency so that you output a 1 cycle for every four cycles

B) State Diagram



C) Implementation Model



D) State Table (Moore-type)

Inputs			Outputs		
Q1	Q0	a	I1	I0	x
0	0	0	0	0	0
0	0	1	0	1	
0	1	0	0	1	0
0	1	1	1	0	
1	0	0	1	0	0
1	0	1	1	1	
1	1	0	1	1	1
1	1	1	0	0	

- Given this implementation model
  - Sequential logic design quickly reduces to combinational logic design

Sequential logic (Controller) design (cont.)

E) Minimized Output Equations

I1

a \ Q1Q0	00	01	11	10
0	0	0	1	1
1	0	1	0	1

I1 = Q1'Q0a + Q1a' + Q1Q0'

I0

a \ Q1Q0	00	01	11	10
0	0	1	1	0
1	1	0	0	1

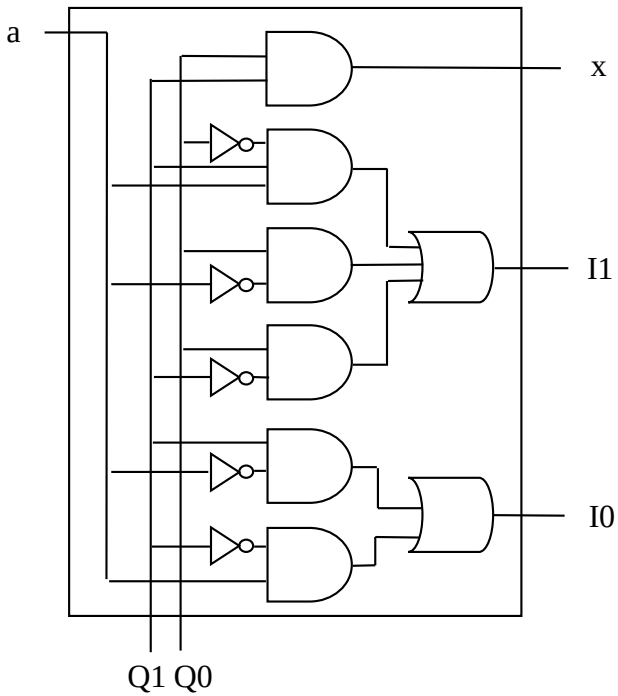
I0 = Q0a' + Q0'a

x

a \ Q1Q0	00	01	11	10
0	0	0	1	0
1	0	0	1	0

x = Q1Q0

F) Combinational Logic



Modeling Finite-State Machine in VHDL

- An `enumeration` type can be used for the states:

```
type frequency_divider is
        (zero, one, two, three);
```

- We can declare a signal to represent the current state of state machine:

```
signal current_state, next_state: frequency-divider;
```

Modeling Finite-State Machine in VHDL

- An `enumeration` type can be used for the states:

```
type frequency_divider is
        (zero, one, two, three);
```

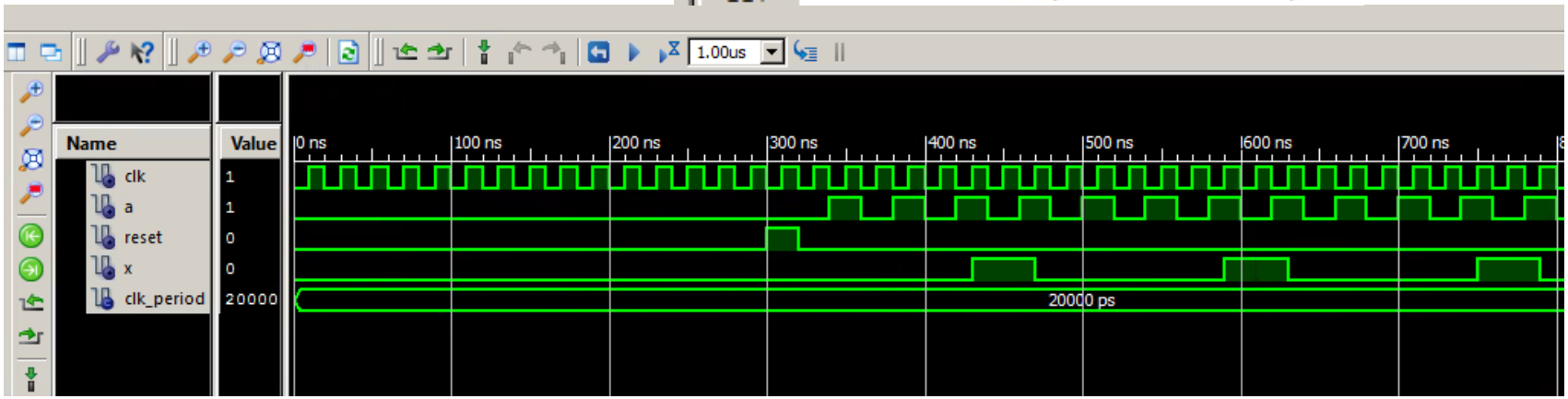
- We can declare a signal to represent the current state of state machine:

```
signal current_state, next_state: frequency-divider;
```

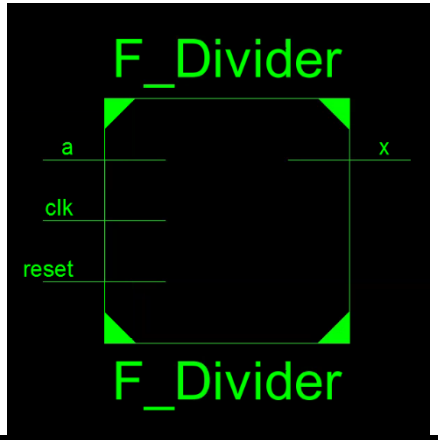
```
entity F_Divider is
    Port ( clk : in  STD_LOGIC;
           a : in  STD_LOGIC;
           reset : in  STD_LOGIC;
           x : out  STD_LOGIC);
end F_Divider;

architecture Behavioral of F_Divider is
   type statetype is(zero, one, two, three);
   signal state, nextstate: statetype;

begin
--state register
   process(clk) begin
   if (clk ='1' and clk' event) then
   if(reset='1') then state<=zero;
   elsif (a='0') then state<=state;
   else state<=nextstate;
   end if;
   end if;
   end process;

--next state logic
   nextstate<= one when state=zero else
               two when state=one else
               three when state=two else
               zero when state=three;

--output logic
   x<='1' when state=three else '0';
end Behavioral;
```

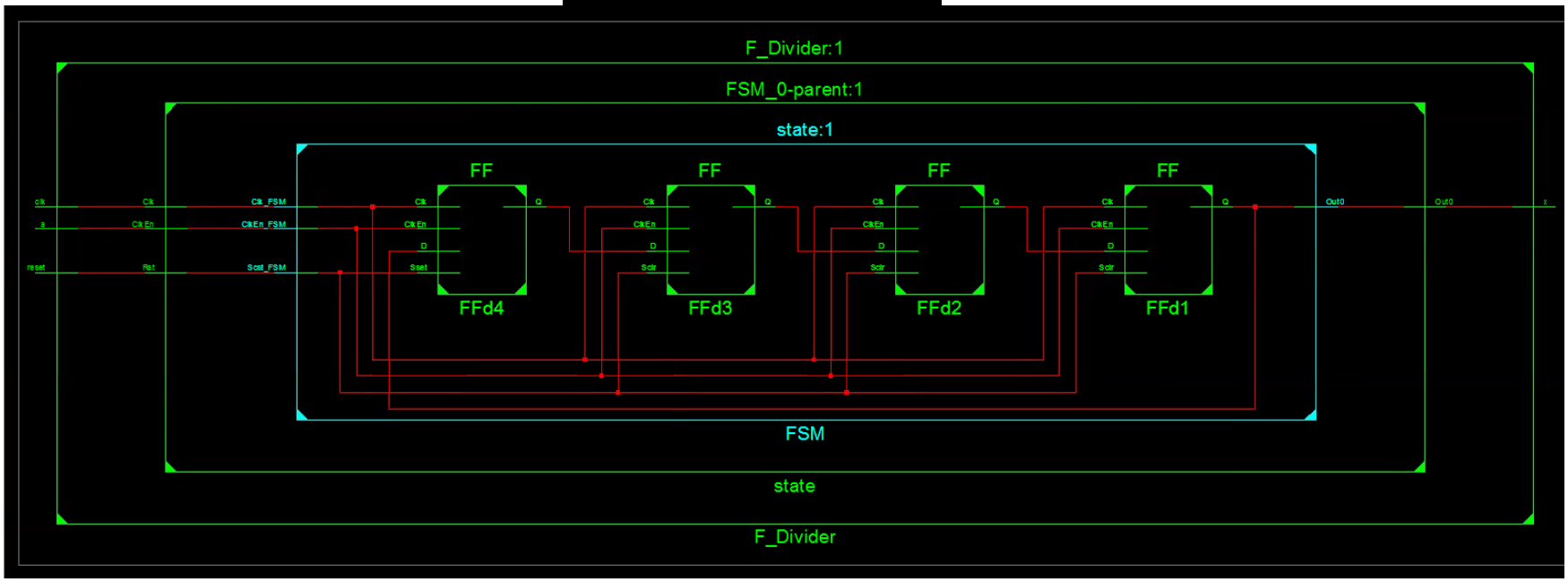
Test Bench

```
91      -- insert stimulus here
92      reset <='1'; wait for 20 ns;
93      reset <='0'; wait for 20 ns;
94         a <= '1'; wait for 20 ns;
95         a <= '0'; wait for 20 ns;
96         a <= '1'; wait for 20 ns;
97         a <= '0'; wait for 20 ns;
98         a <= '1'; wait for 20 ns;
99         a <= '0'; wait for 20 ns;
100        a <= '1'; wait for 20 ns;
101        a <= '0'; wait for 20 ns;
102        a <= '1'; wait for 20 ns;
103        a <= '0'; wait for 20 ns;
104        a <= '1'; wait for 20 ns;
105        a <= '0'; wait for 20 ns;
106        a <= '1'; wait for 20 ns;
107        a <= '0'; wait for 20 ns;
108        a <= '1'; wait for 20 ns;
109        a <= '0'; wait for 20 ns;
110        a <= '1'; wait for 20 ns;
111        a <= '0'; wait for 20 ns;
112        a <= '1'; wait for 20 ns;
113        a <= '0'; wait for 20 ns;
114        a <= '1'; wait for 20 ns;
115        a <= '0'; wait for 20 ns;
116        a <= '1'; wait for 20 ns;
117        a <= '0'; wait for 20 ns;
```



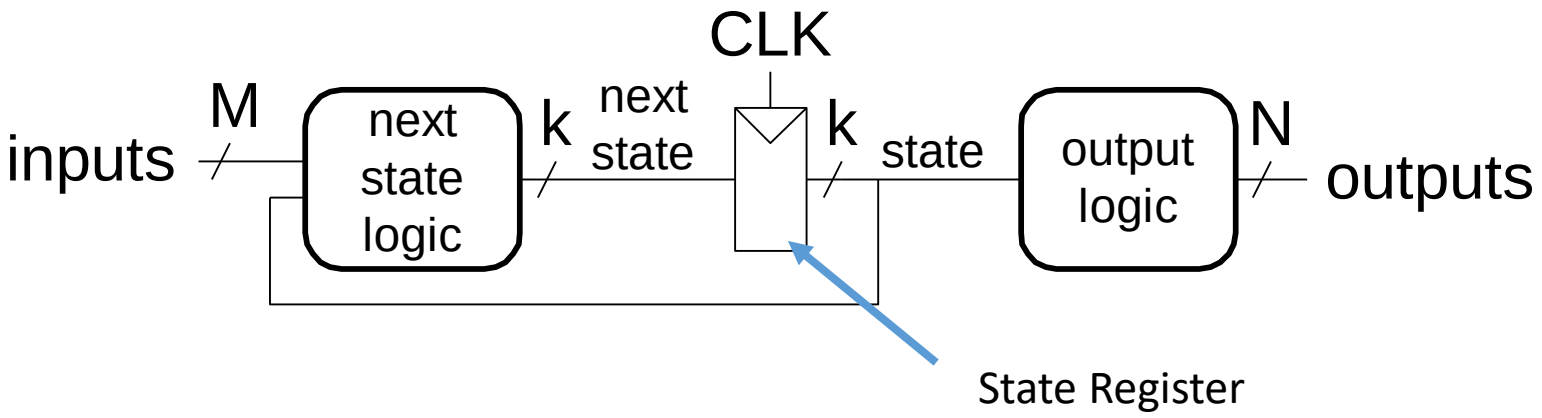
RTL Schematic



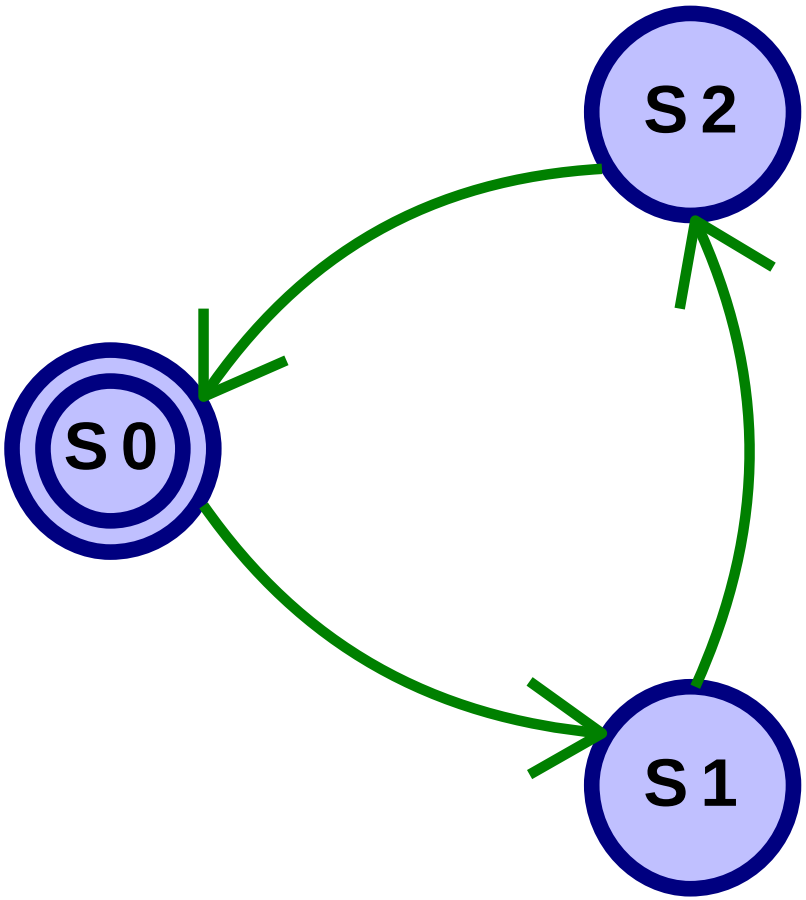


Moore Finite State Machine (FSM)

- **Three blocks:**
  - next state logic
  - state register
  - output logic



FSM Example: Divide



The double circle indicates the reset state

FSM in VHDL

```
library IEEE; use IEEE.STD_LOGIC_1164.all;
entity divideby3FSM is
  port (clk, reset: in  STD_LOGIC;
        y:          out STD_LOGIC);
end;

architecture synth of divideby3FSM is
  type statetype is (S0, S1, S2);
  signal state, nextstate: statetype;
begin
  -- state register
  process (clk, reset) begin
    if reset = '1' then state <= S0;
    elsif clk'event and clk = '1' then
      state <= nextstate;
    end if;
  end process;

  -- next state logic
  nextstate <= S1 when state = S0 else
               S2 when state = S1 else
               S0;

  -- output logic
  y <= '1' when state = S0 else '0';
end;
```

This example defines a new enumeration data type, `statetype`, with three possibilities: `s0, s1, s2,` `state` and `nextstate` are `statetype` signals. By using an enumeration instead of choosing the state encoding, VHDL frees the synthesizer to explore various state encodings to choose the best one.

Summary

- Designing a custom single-purpose processor for a given application requires an understanding of various aspects of digital design.
- Design of a circuit to implement Boolean functions requires combinational logic design.
- Design of a circuit to implement a state diagram requires sequential design, this circuit is commonly called a controller.
- A controller consists of a state register and a combinational logic block.
  - The combinational logic block provides for next state and output logic.
  - The size of the state register depends on the number of states.