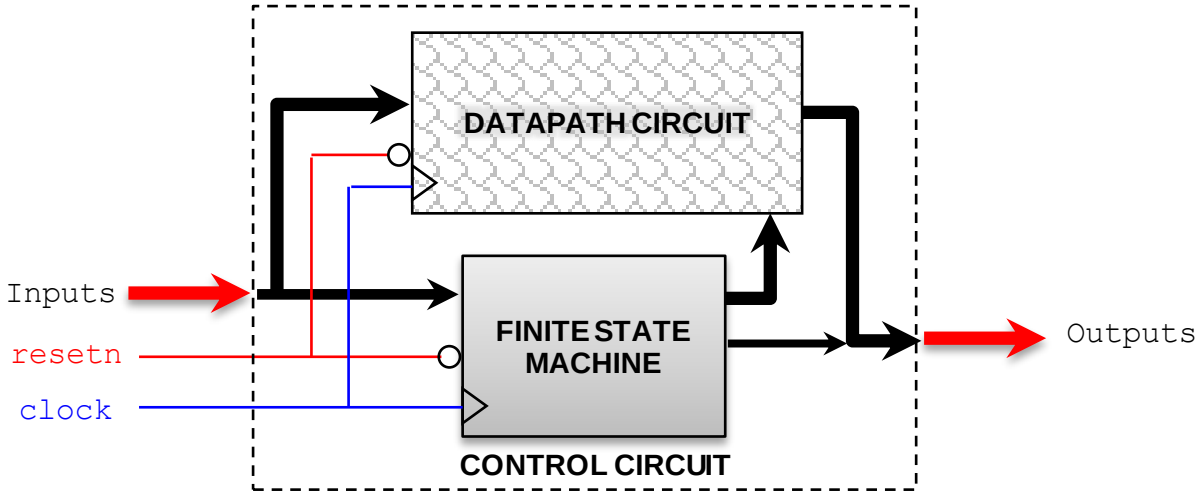
Notes - Unit 7

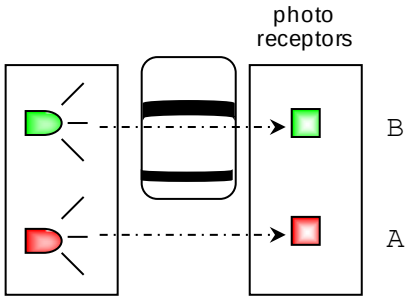
Introduction to Digital System Design

Digital System Model

- FSM + Datapath Circuit:



Example: Car Lot Counter



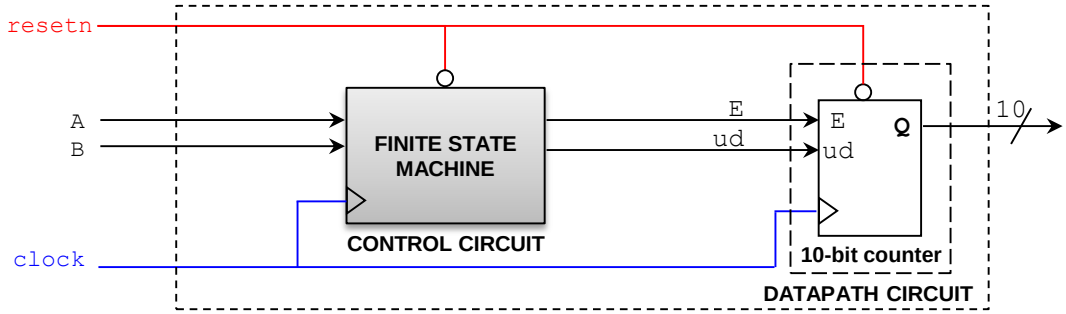
If A = 1 → No light received (car obstructing LED A)
If B = 1 → No light received (car obstructing LED B)

If car enters the lot, the following sequence (A|B) must be followed:
00 → 10 → 11 → 01 → 00
If car leaves the lot, the following sequence (A|B) must be followed:
00 → 01 → 11 → 10 → 00

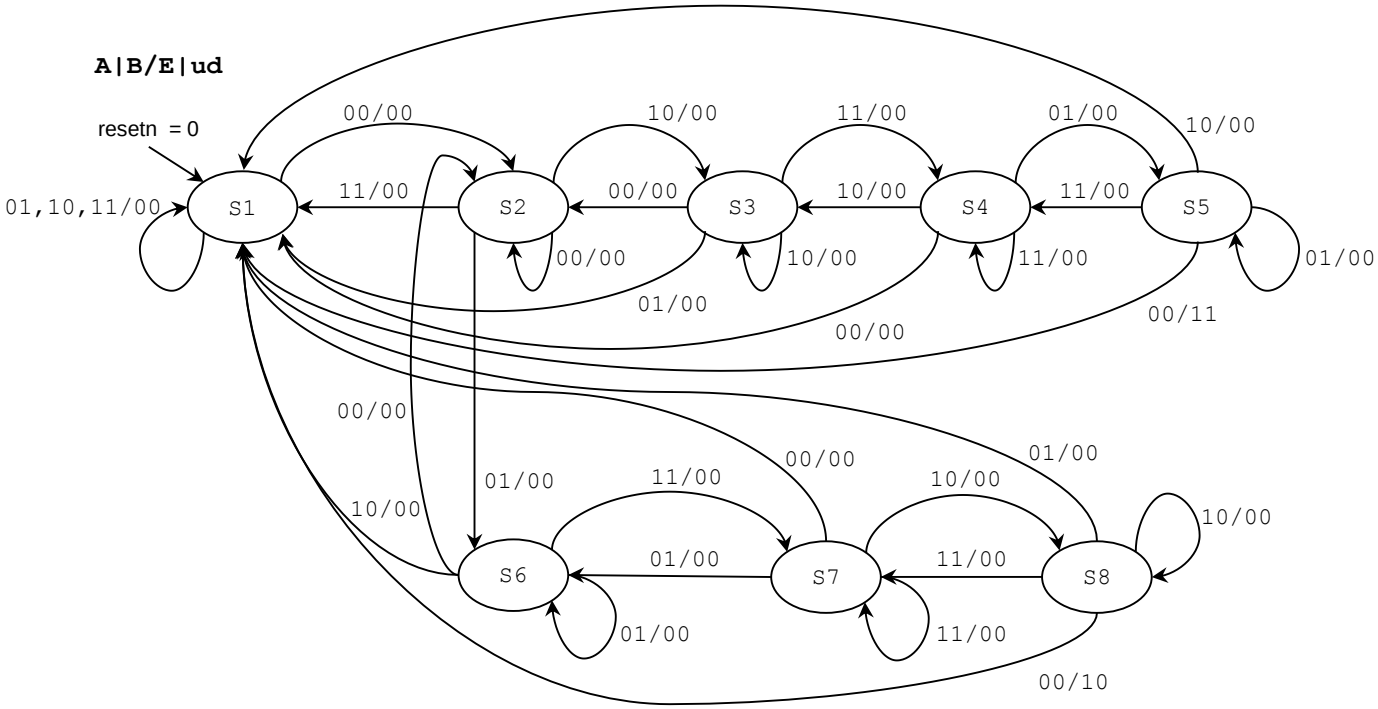
A car might stay in a state for many cycles since the car speed is very large compared to that of the clock frequency.

DIGITAL SYSTEM (FSM + Datapath circuit)

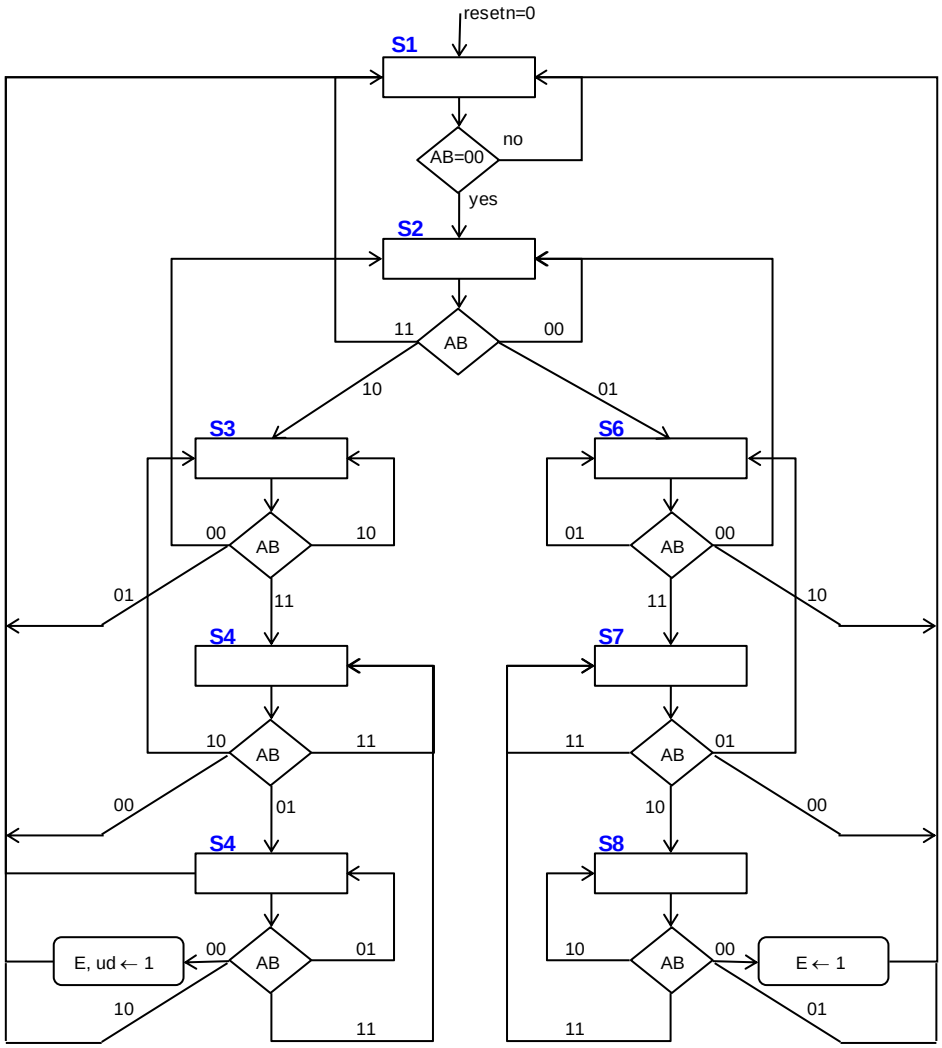
- Usually, when 'resetn' (asynchronous clear) and 'clock' are not drawn, they are implied.



- **Finite State Machine (FSM)**:



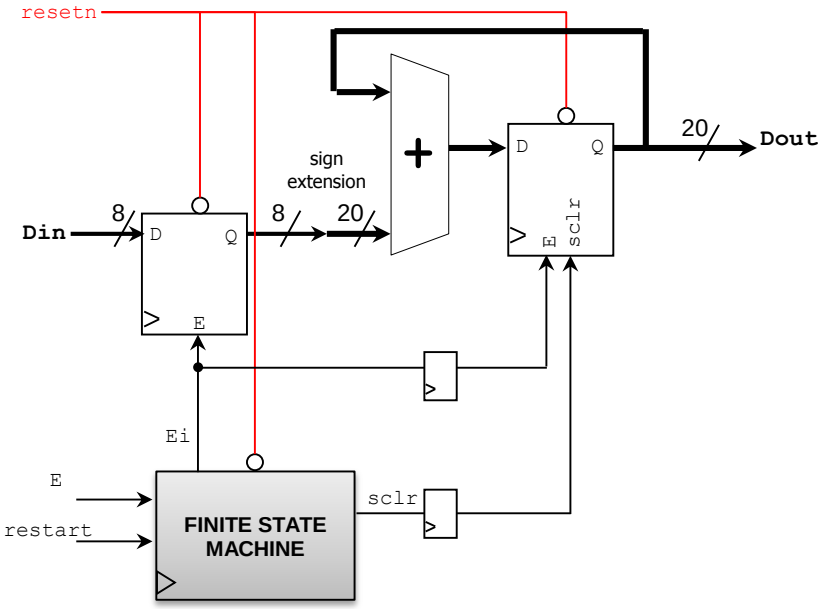
- **Algorithmic State Machine (ASM) chart:**



EXAMPLE: ACCUMULATOR

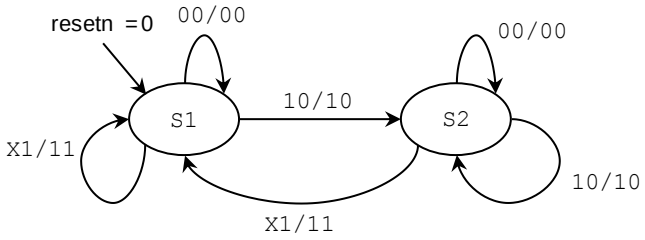
DIGITAL SYSTEM (FSM + Datapath circuit)

- `sclr`: Synchronous clear. If E = '1' and sclr = '1', then the output bits of the registers are set to zero.

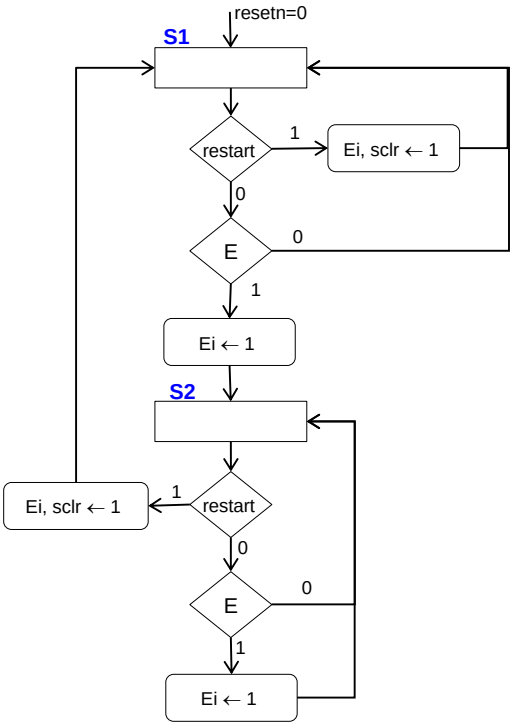


- **Finite State Machine (FSM)**:

E|restart/Ei|sclr



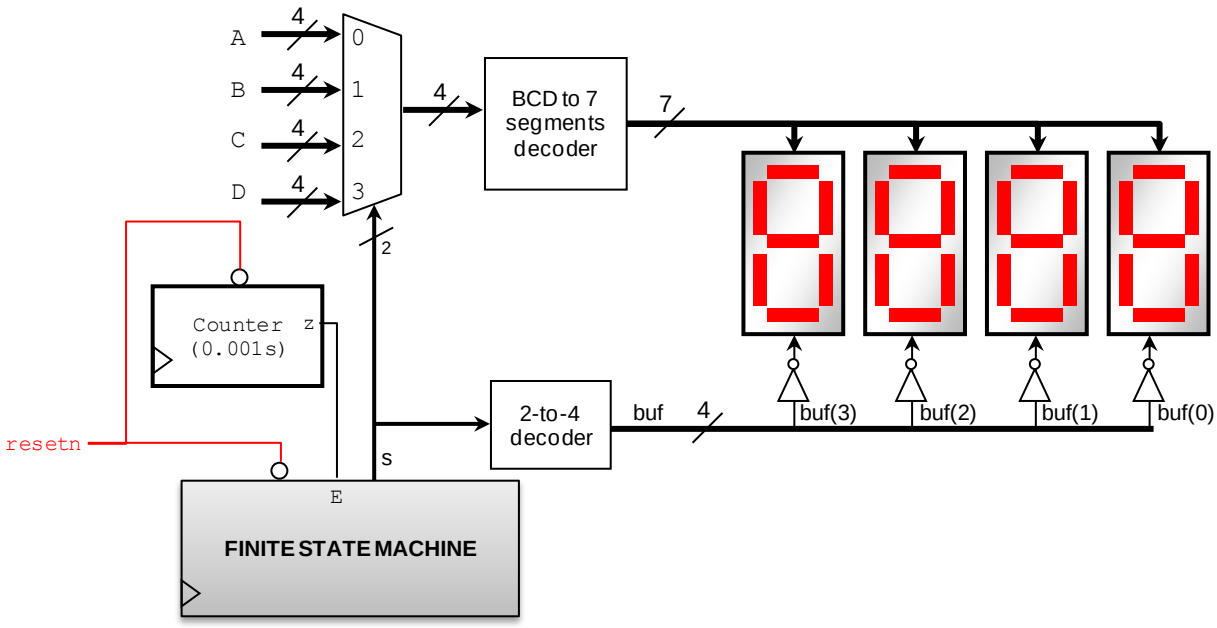
- **Algorithmic State Machine (ASM)**:



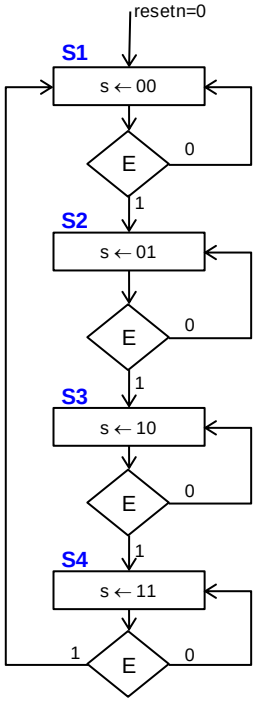
EXAMPLE: 7-SEGMENT SERIALIZER

DIGITAL SYSTEM (FSM + Datapath circuit)

- Most FPGA Development board have a number of 7-segment displays (e.g., 4, 8). However, only one can be used at a time.
- If we want to display four digits (inputs A, B, C, D), we can design a serializer that will only show one digit at a time on the 7-segment displays.
- Since only one 7-segment display can be used at a time, we need to serialize the four BCD outputs. In order for each digit to appear bright and continuously illuminated, each digit is illuminated for 1 ms every 4 ms (i.e. a digit is un-illuminated for 3 ms and illuminated for 1 ms). This is taken care of by feeding the output 'z' of the counter to 0.001 s to the enable input of the FSM. This way, state transitions only occur each 0.001 s.
- In the figure, the enable signals for the four 7-segment displays are active low (this is usually the case).



- **Algorithmic State Machine (ASM) chart:** This is a Moore-type FSM.



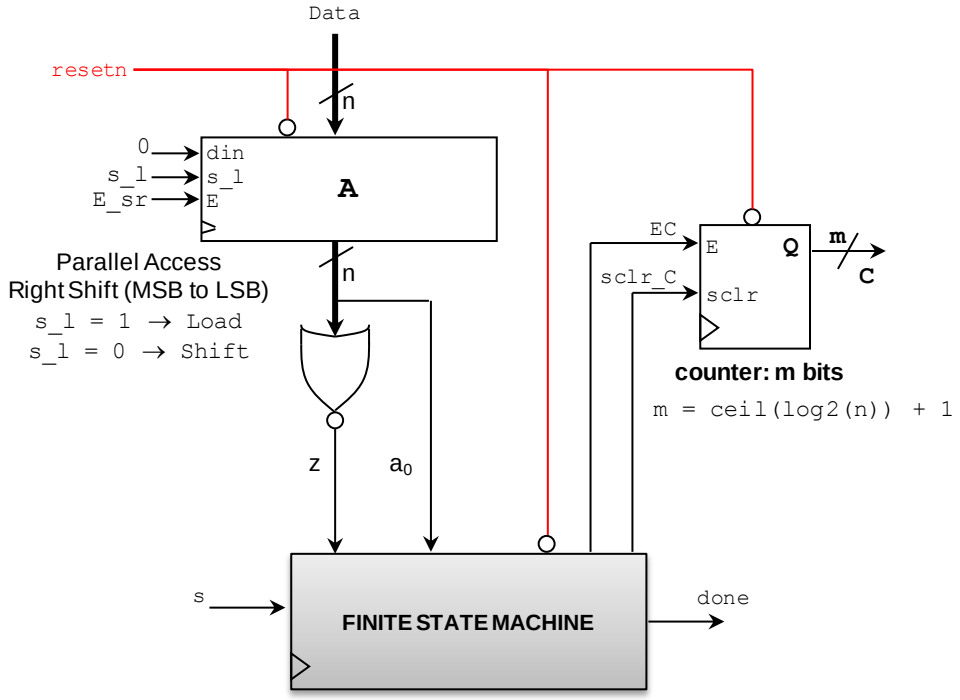
EXAMPLE: BIT-COUNTING CIRCUIT

SEQUENTIAL ALGORITHM

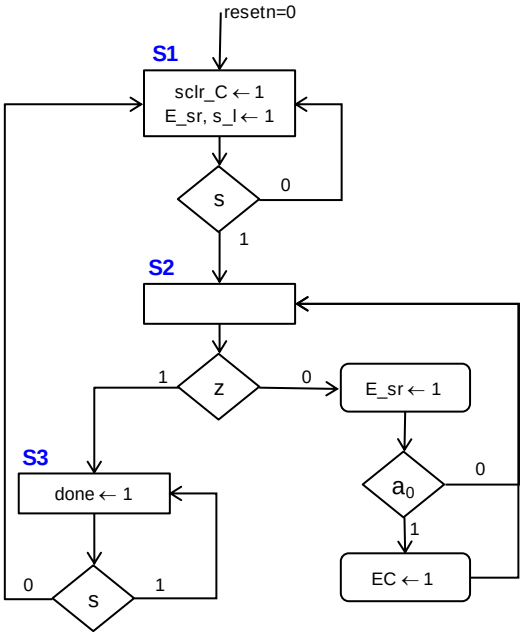
```
C ← 0
while A ≠ 0
   if a0 = 1 then
      C ← C + 1
   end if
   right shift A
end while
```

DIGITAL SYSTEM (FSM + Datapath circuit)

- `sclr`: Synchronous clear. In this case, if sclr = '1', the count is initialized to zero (here, we do not need EC to be 1).

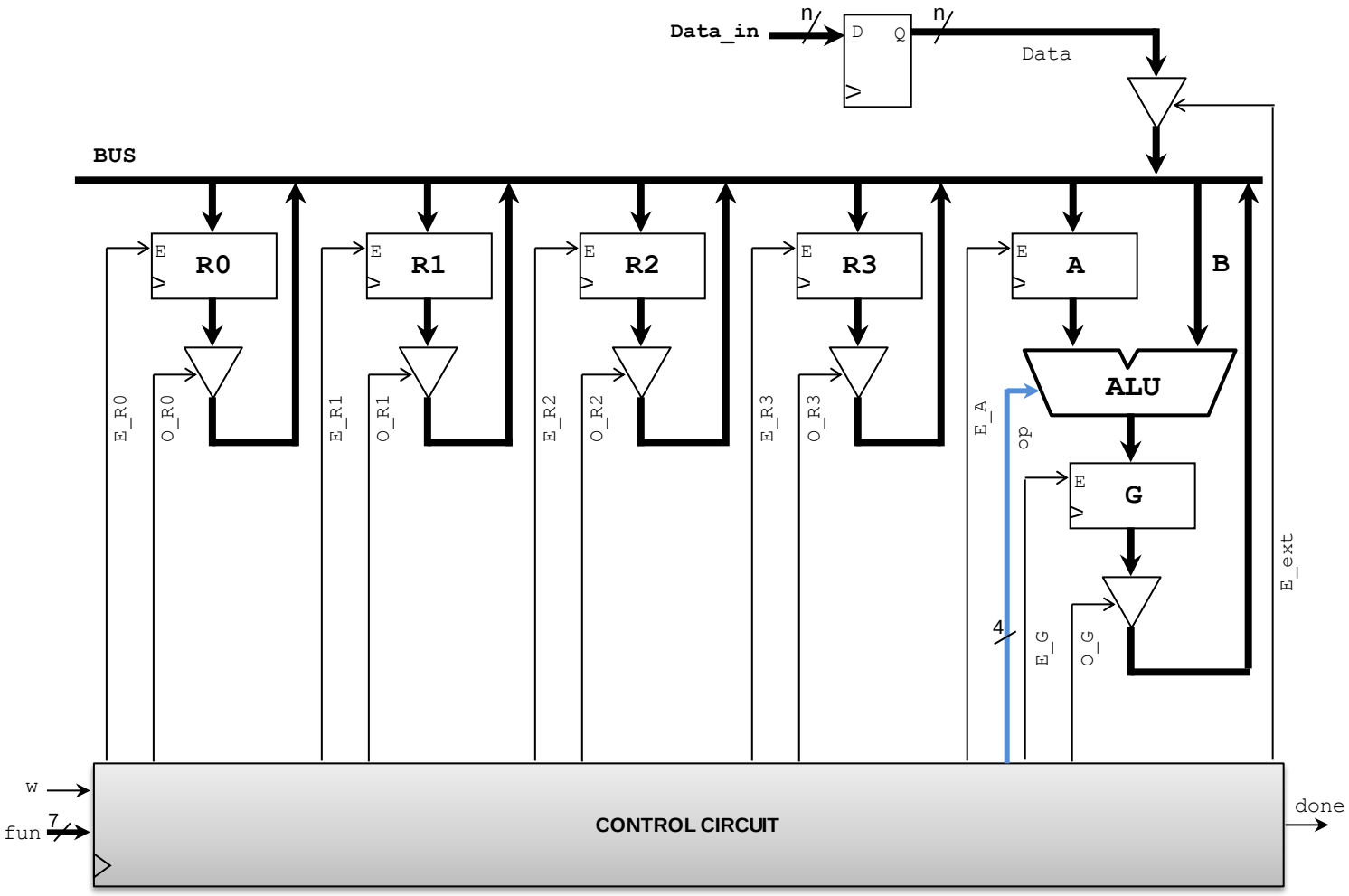


- **Algorithmic State Machine (ASM) chart:**



EXAMPLE: SIMPLE PROCESSOR

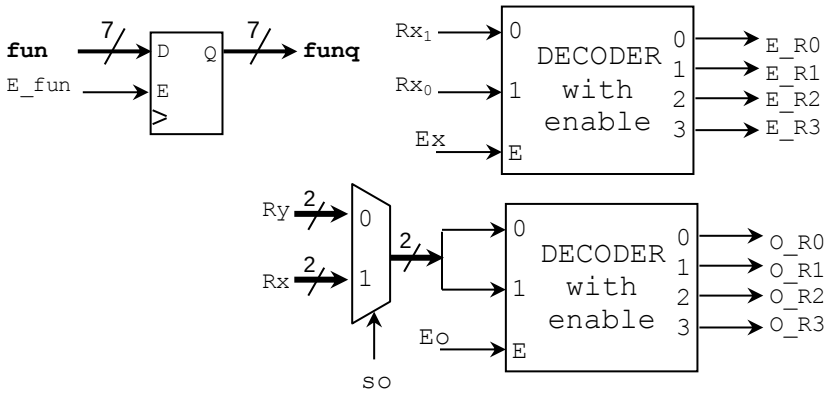
DIGITAL SYSTEM (FSM + Datapath circuit)



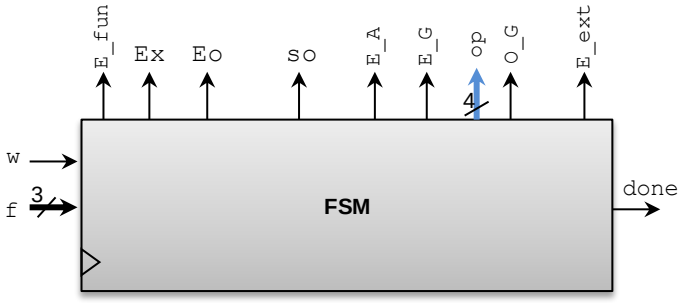
- Operation: Every time w = '1', we grab the instruction from fun and execute it:
- $funq = |f_2|f_1|f_0|Ry_1|Ry_0|Rx_1|Rx_0|$

f	Operation	Function
000	Load Rx, Data	Rx ← Data
001	Move Rx, Ry	Rx ← Ry
010	Add Rx, Ry	Rx ← Rx + Ry
011	Sub Rx, Ry	Rx ← Rx - Ry
100	Not Rx	Rx ← NOT (Rx)
101	And Rx, Ry	Rx ← Rx AND Ry
110	Or Rx, Ry	Rx ← Rx OR Ry
111	Xor Rx, Ry	Rx ← Rx XOR Ry

- **Control Circuit:**



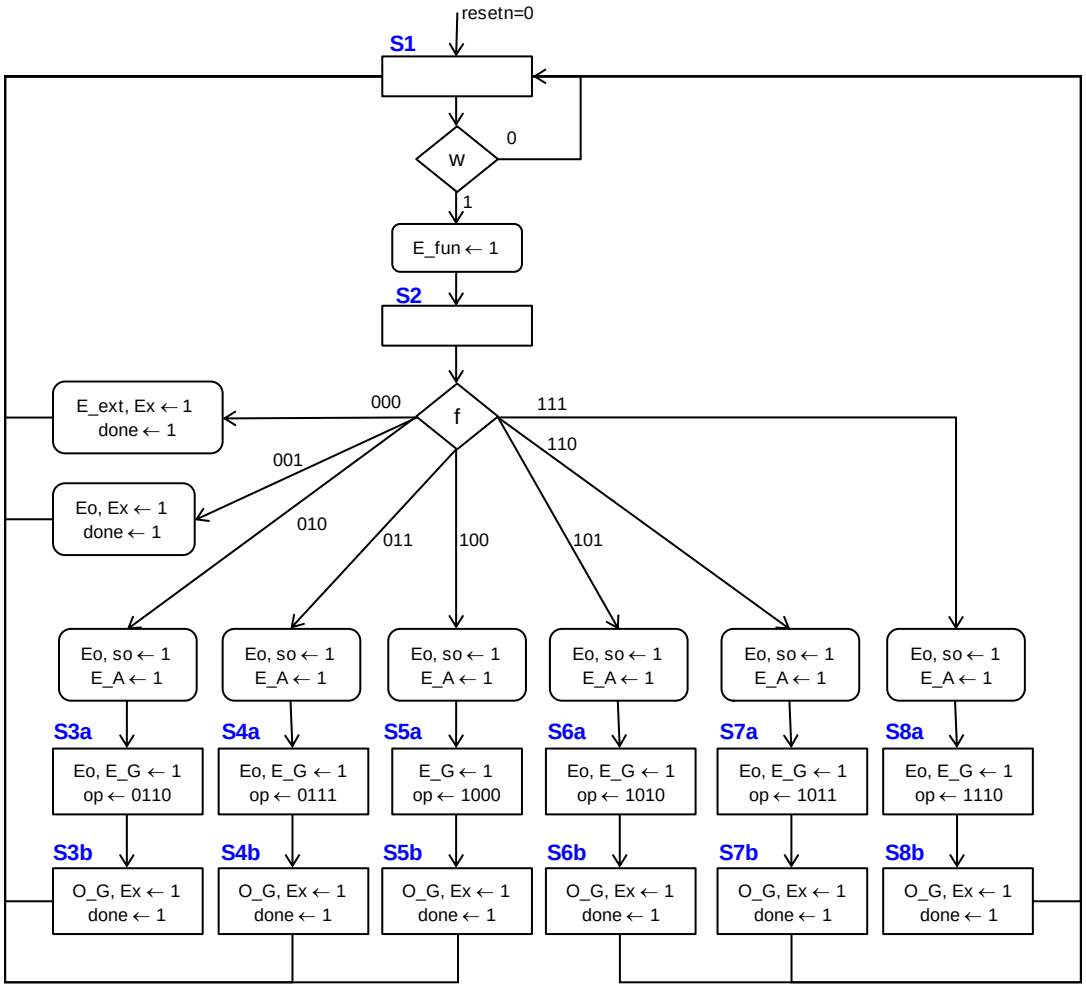
funq = |f_2|f_1|f_0|Ry_1|Ry_0|Rx_1|Rx_0|



- Arithmetic-Logic Unit (ALU):

op	Operation	Function	Unit
0000	y <= A	Transfer 'A'	Arithmetic
0001	y <= A + 1	Increment 'A'	
0010	y <= A - 1	Decrement 'A'	
0011	y <= B	Transfer 'B'	
0100	y <= B + 1	Increment 'B'	
0101	y <= B - 1	Decrement 'B'	
0110	y <= A + B	Add 'A' and 'B'	
0111	y <= A - B	Subtract 'B' from 'A'	
1000	y <= not A	Complement 'A'	Logic
1001	y <= not B	Complement 'B'	
1010	y <= A AND B	AND	
1011	y <= A OR B	OR	
1100	y <= A NAND B	NAND	
1101	y <= A NOR B	NOR	
1110	y <= A XOR B	XOR	
1111	y <= A XNOR B	XNOR	

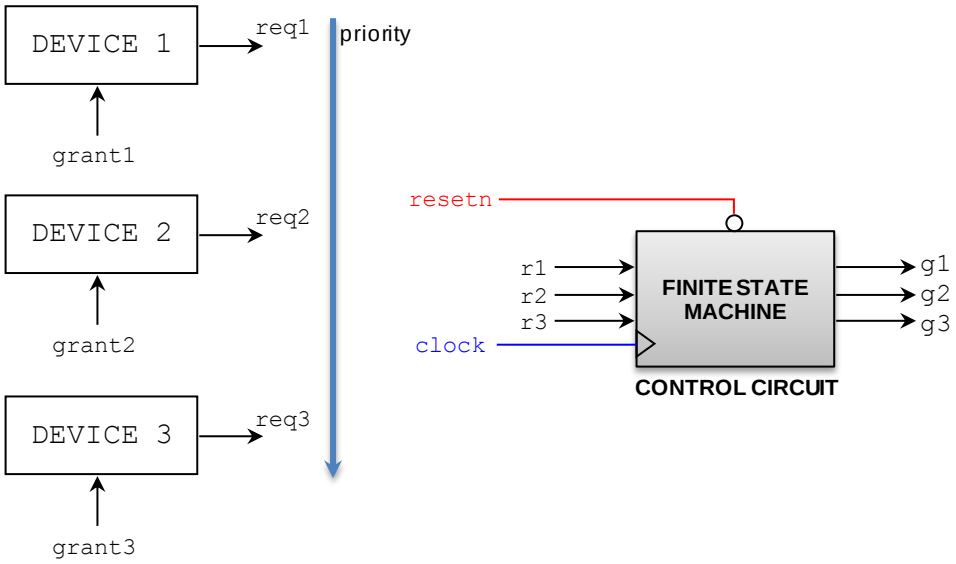
- **Algorithmic State Machine (ASM)**:



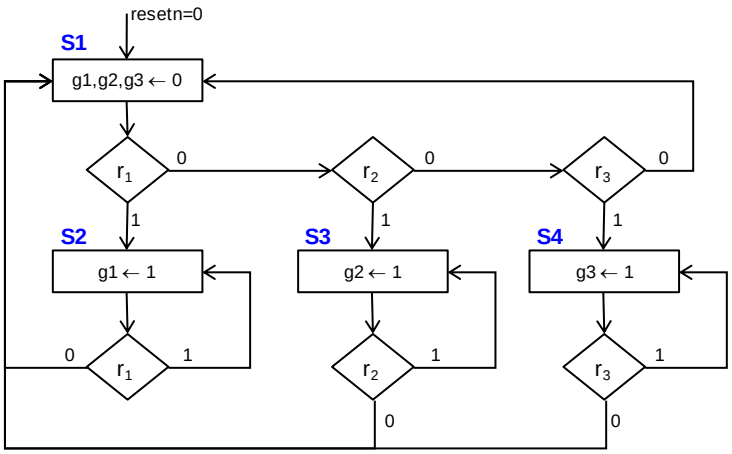
EXAMPLE: ARBITER CIRCUIT

DIGITAL SYSTEM (FSM + Datapath circuit)

- Three devices can request access to a certain resource at any time (example: access to a bus made of tri-state buffers, only one tri-state buffer can be enabled at a time). The FSM can only grant access to one device at a time.There should be a priority level among devices.
- If the FSM grants access to one device, one must wait until the request signal to that device is deasserted (i.e. set to zero) before granting access to a different device.

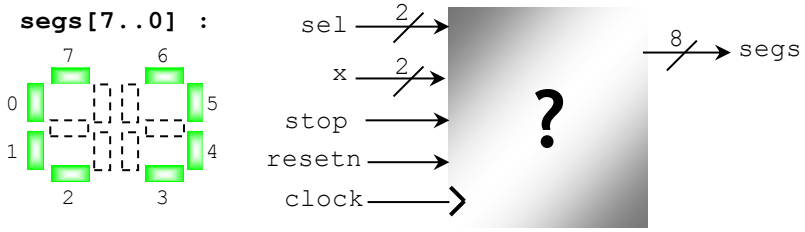


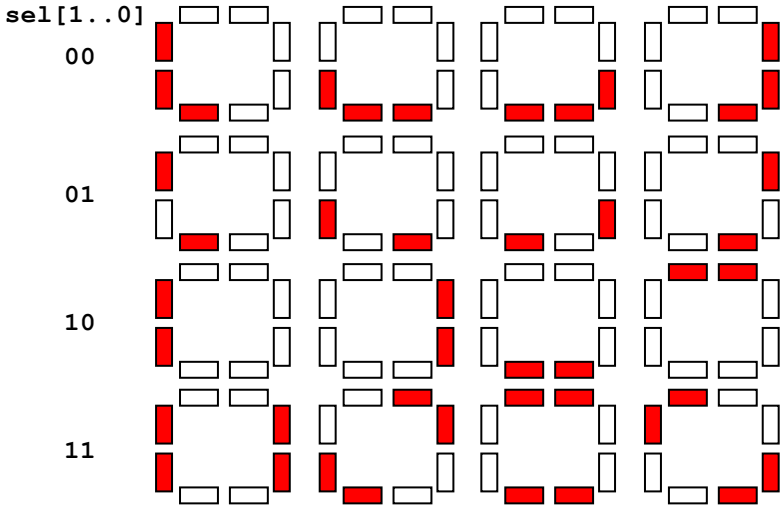
- **Algorithmic State Machine (ASM) chart:**



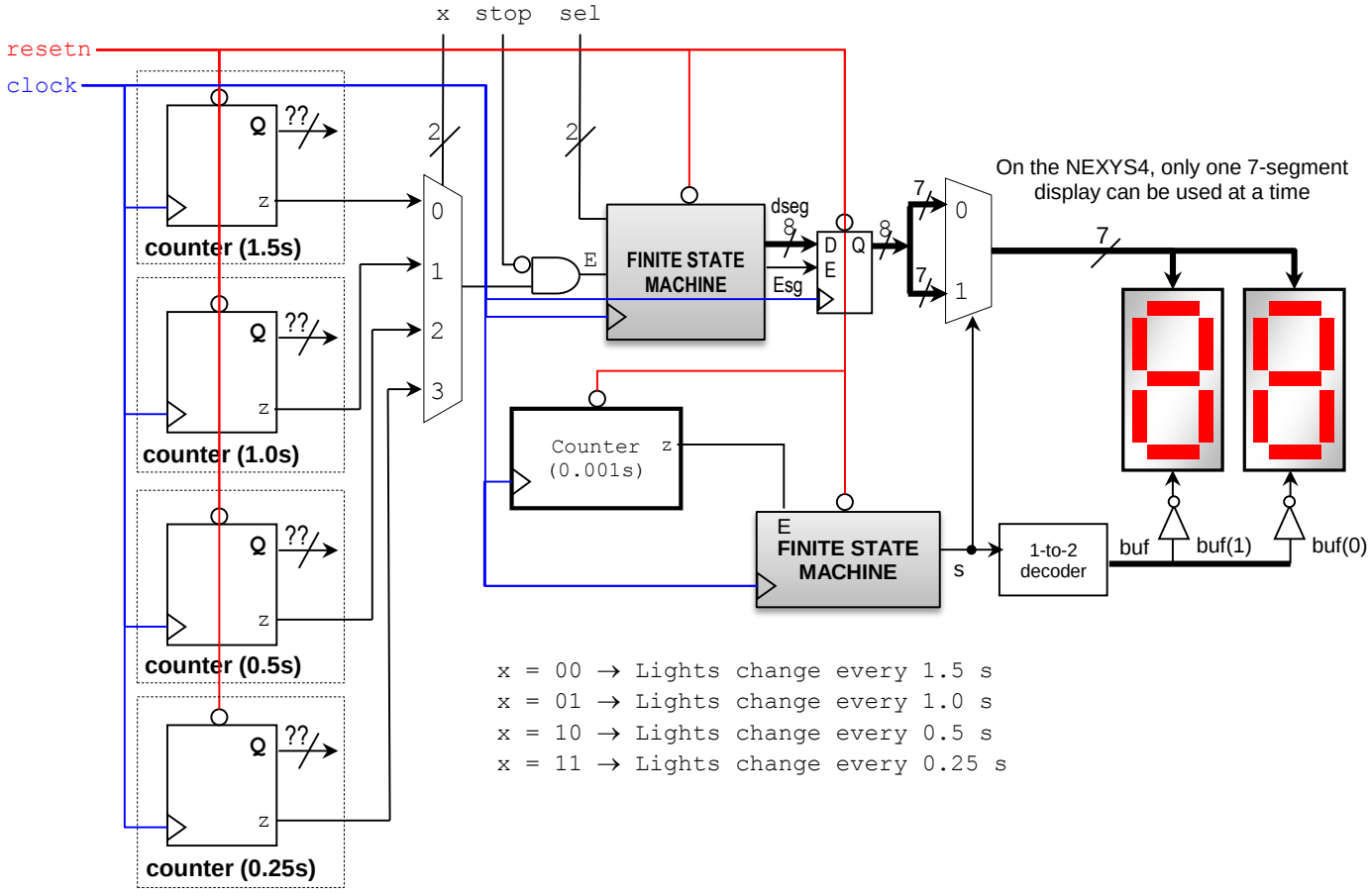
EXAMPLE: DISPLAYING PATTERNS ON 7-SEGMENT DISPLAYS

- Different patterns are shown based on the selector 'sel' signal. Two 7-segment displays are used.
- 'stop' input: If it is asserted (stop = 1), the lights' pattern freezes.
- The input 'x' selects the rate of change (every 1.5, 1.0, 0.5, or 0.25 seconds).

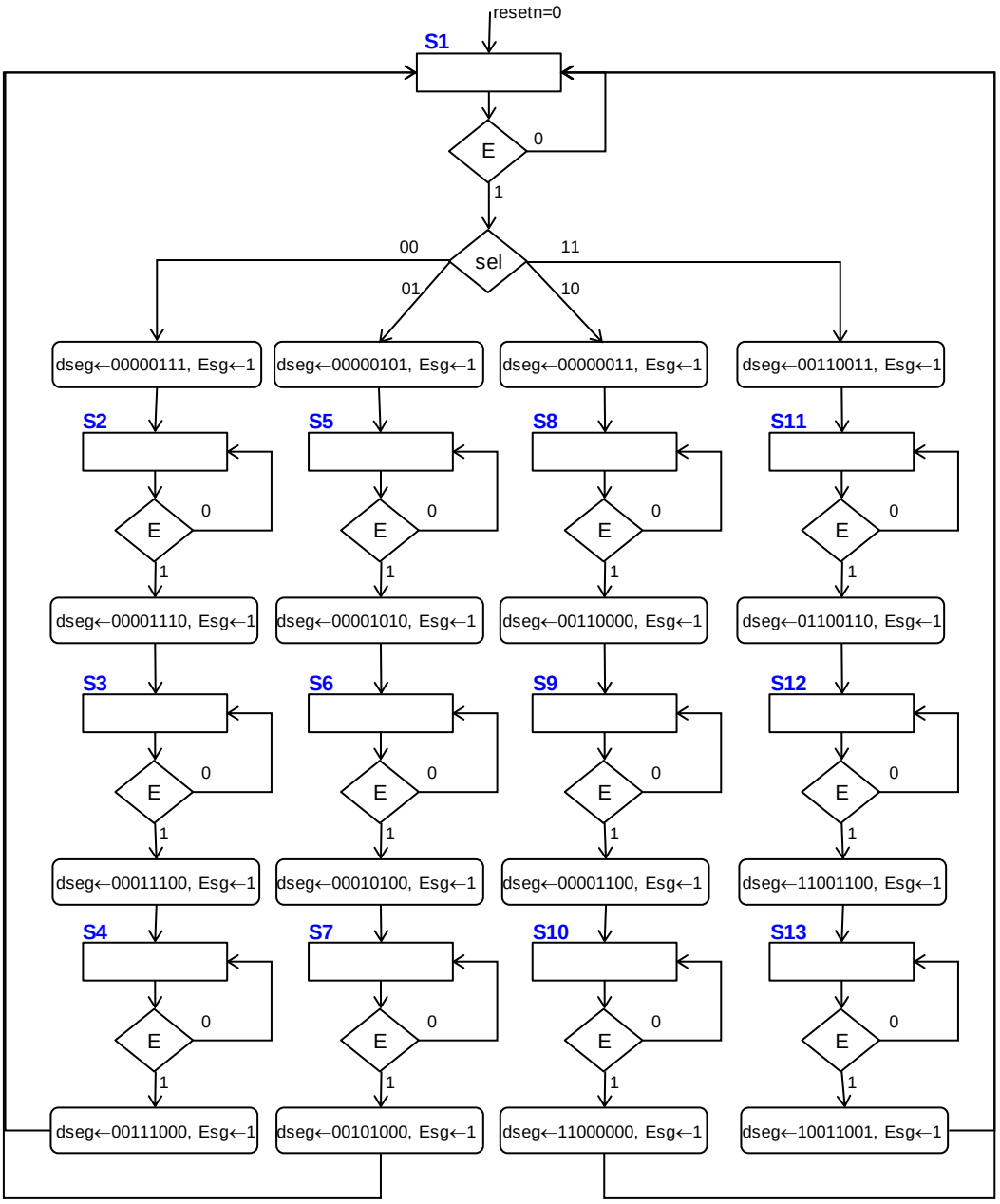




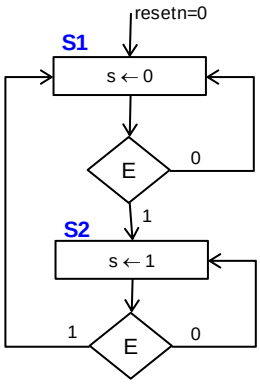
DIGITAL SYSTEM (FSM + Datapath circuit)



- **Algorithmic State Machine (ASM) chart:**



- **Algorithmic State Machine (ASM) chart:** This is the FSM that controls the output MUX



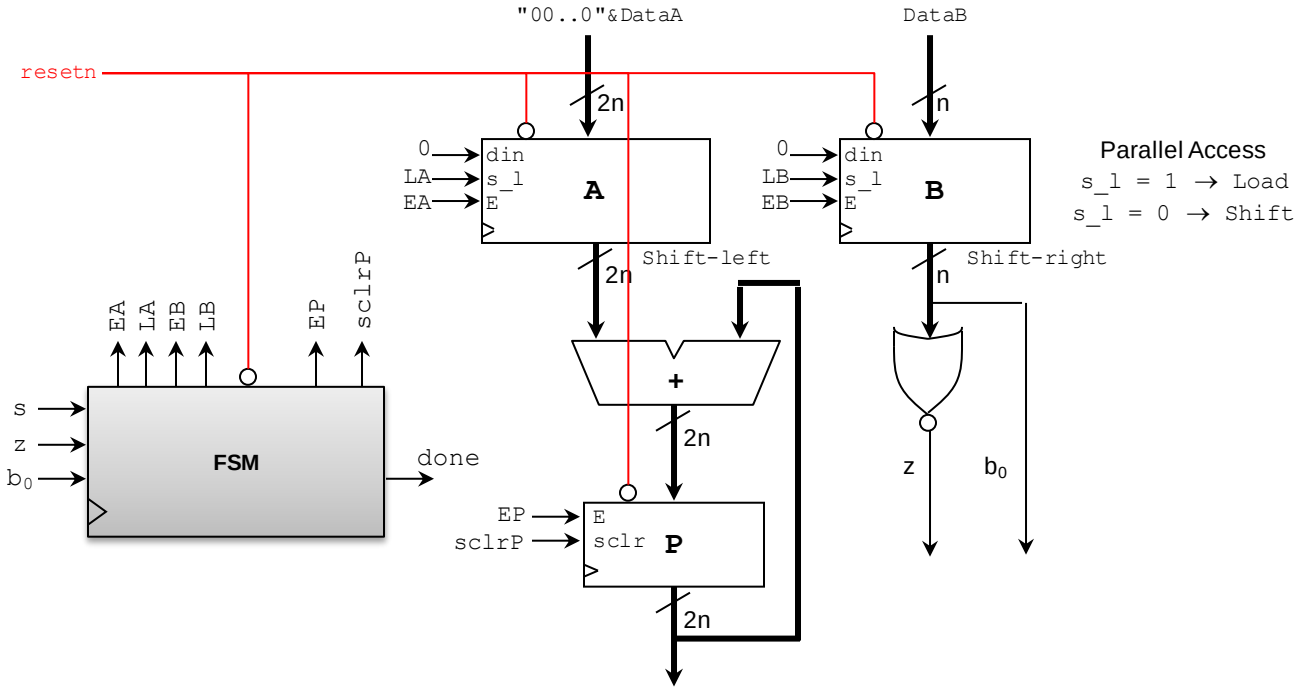
EXAMPLE: SERIAL MULTIPLIER

SEQUENTIAL ALGORITHM

```
P ← 0, Load A,B
while B ≠ 0
   if b0 = 1 then
      P ← P + A
   end if
   left shift A
   right shift B
end while
```

DIGITAL SYSTEM (FSM + Datapath circuit)

- Note that this algorithm can also be run on a simple processor. Here, we use dedicated circuitry.
  sclr: Synchronous clear. In this case, if sclr = '1' and E ='1', the register contents are initialized to 0.



- **Algorithmic State Machine (ASM) chart:**