EE/CSE 371
Design of Digital Circuits and Systems

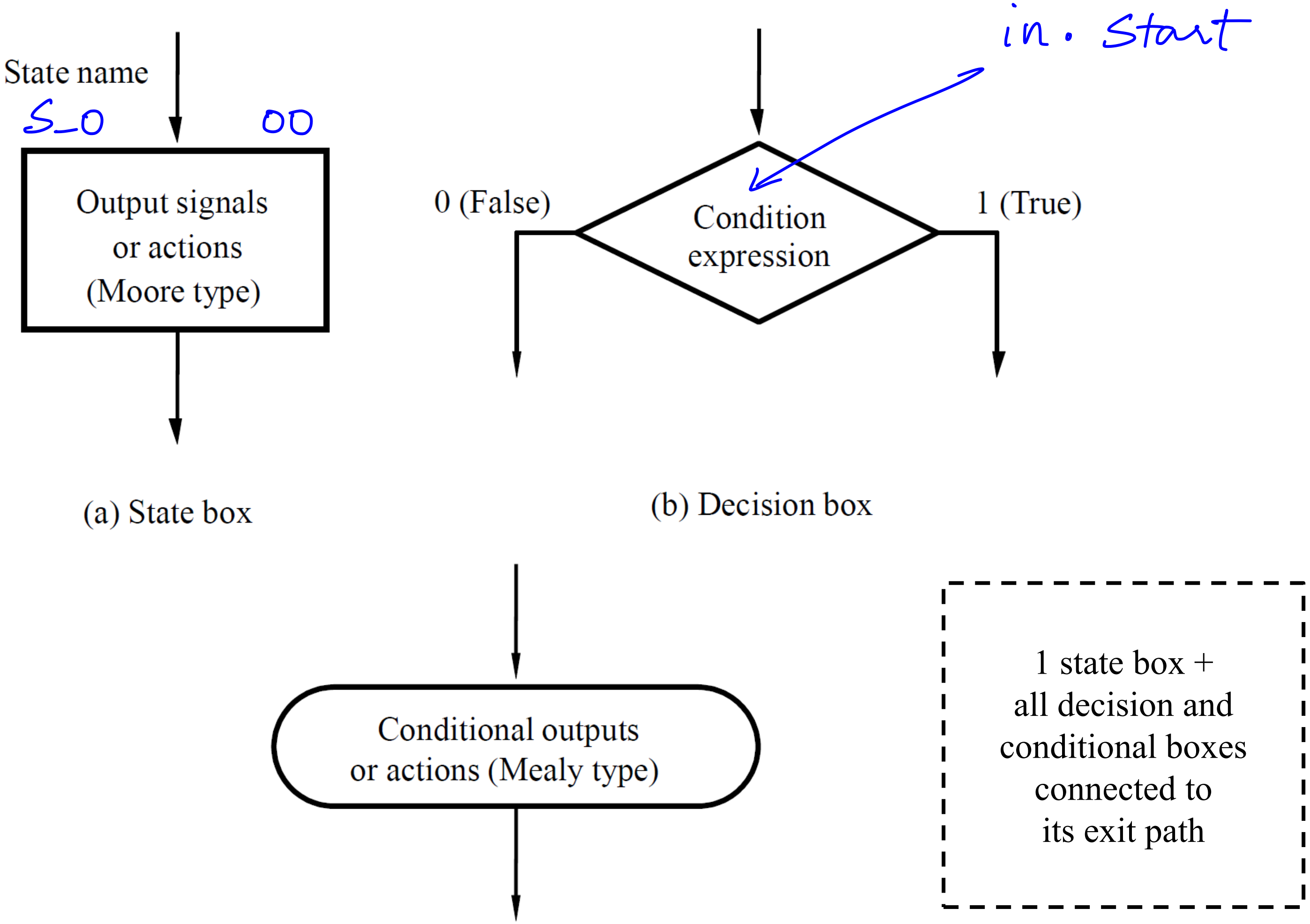
Lecture 6: ASM with Datapath I

Relevant Course Information

1. Homework 3 due Wednesday (10/18)

2. Lab 3 due Friday (10/27)

Review: ASM Chart



1 state box +
all decision and
conditional boxes
connected to
its exit path

(a) State box

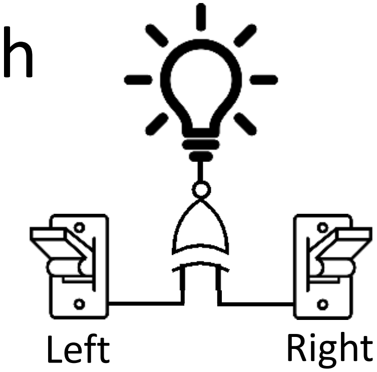
(b) Decision box

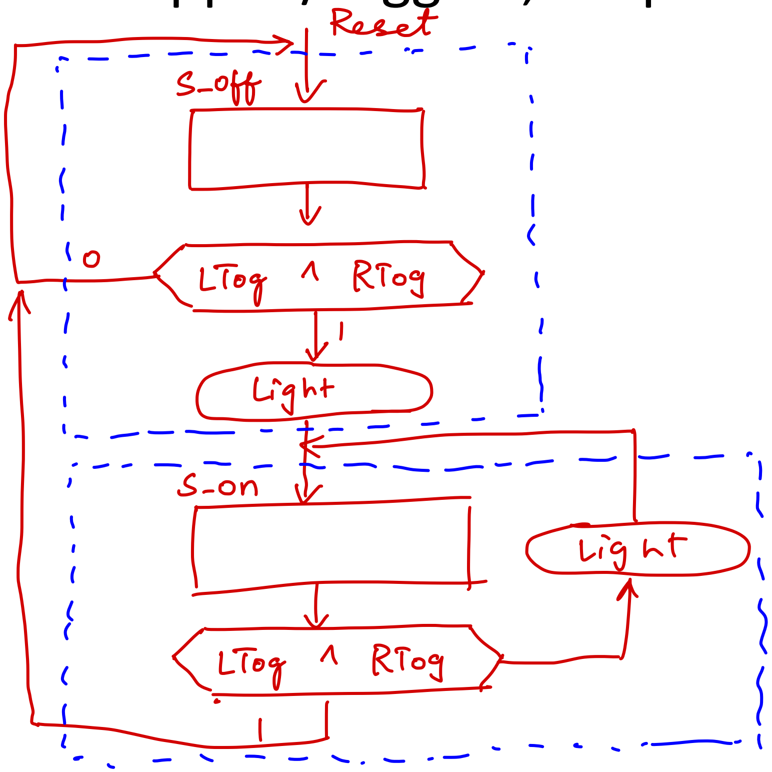
(c) Conditional output box

(d) ASM block

Review Question: 3-way Switch

- Create an ASM chart for a 3-way switch system using *Mealy*-type output
  - `LTog` and `RTog` pulse `1` when switch is flipped/toggled, output called `light`



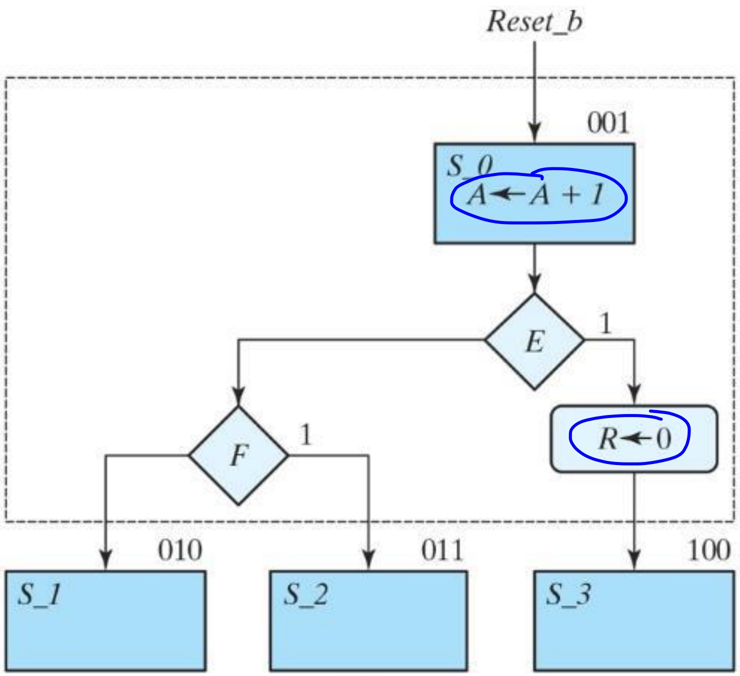


ASMD Charts

- An **A**lgorithmic **S**tate **M**achine with a **D**atapath chart is created by adding RTL operations to an ASM chart
  - Timing of operations can be confusing – *NOT* a flowchart

- School of Thought #1:
  - RTL operations are triggered by control signals, so they can appear anywhere an output signal can:

missing control signals that are asserted.

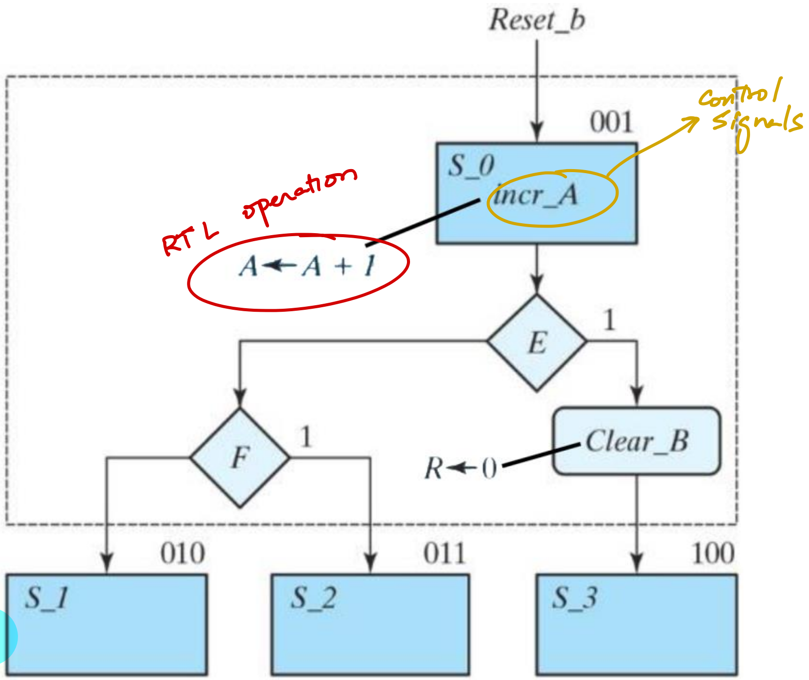


ASMD Charts

- An **A**lgorithmic **S**tate **M**achine with a **D**atapath chart is created by adding RTL operations to an ASM chart
  - Timing of operations can be confusing – *NOT* a flowchart

- School of Thought #2:
  - It's clearer to separate control signals (Control) from RTL operations (Datapath)

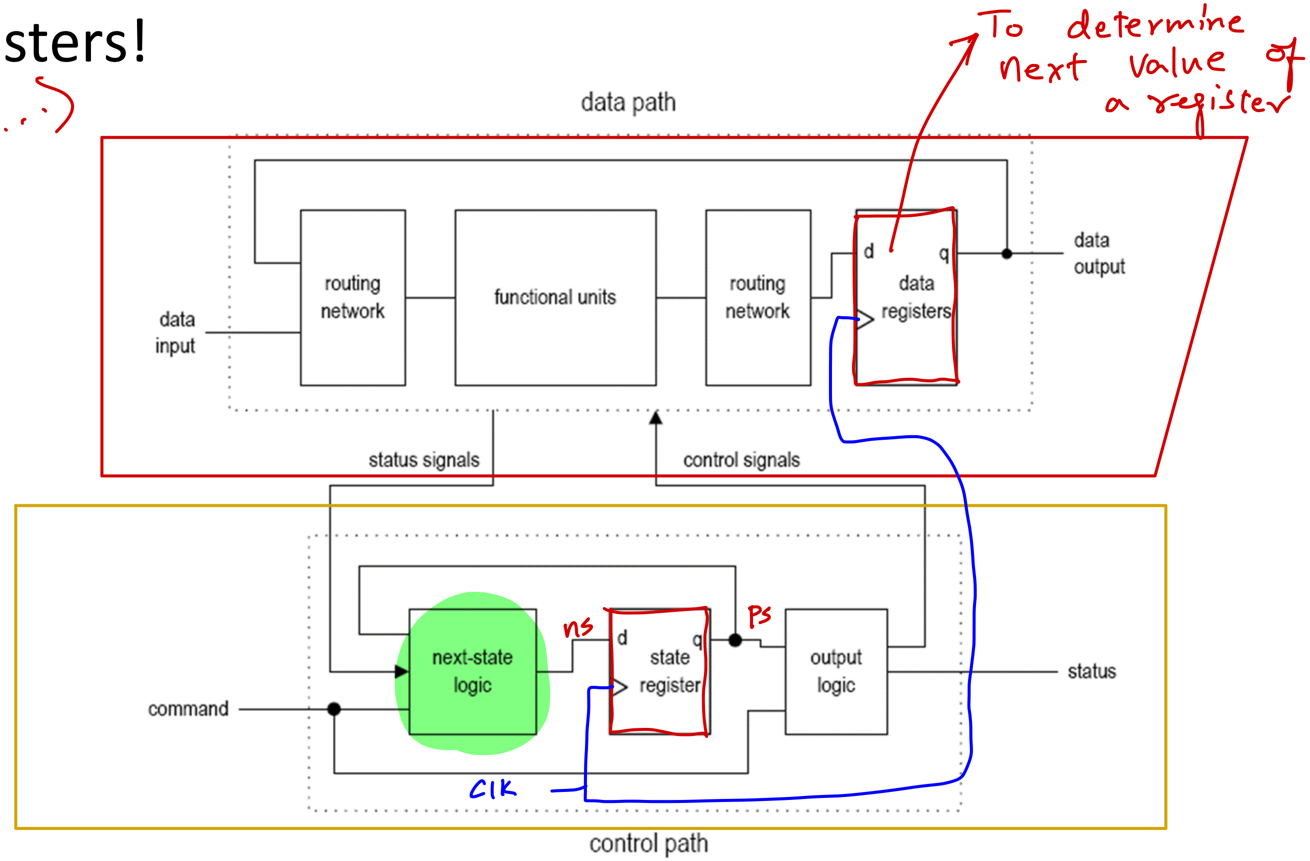
- There isn't a set standard
  - You may see both or variants
  - I will use School of Thought #2



ASMD Hardware

- State transitions and RTL operations are both controlled by the clock
  - It's often helpful to remember the underlying hardware – registers!

$r_{dst} \leftarrow f(r_{src1}, \ldots)$



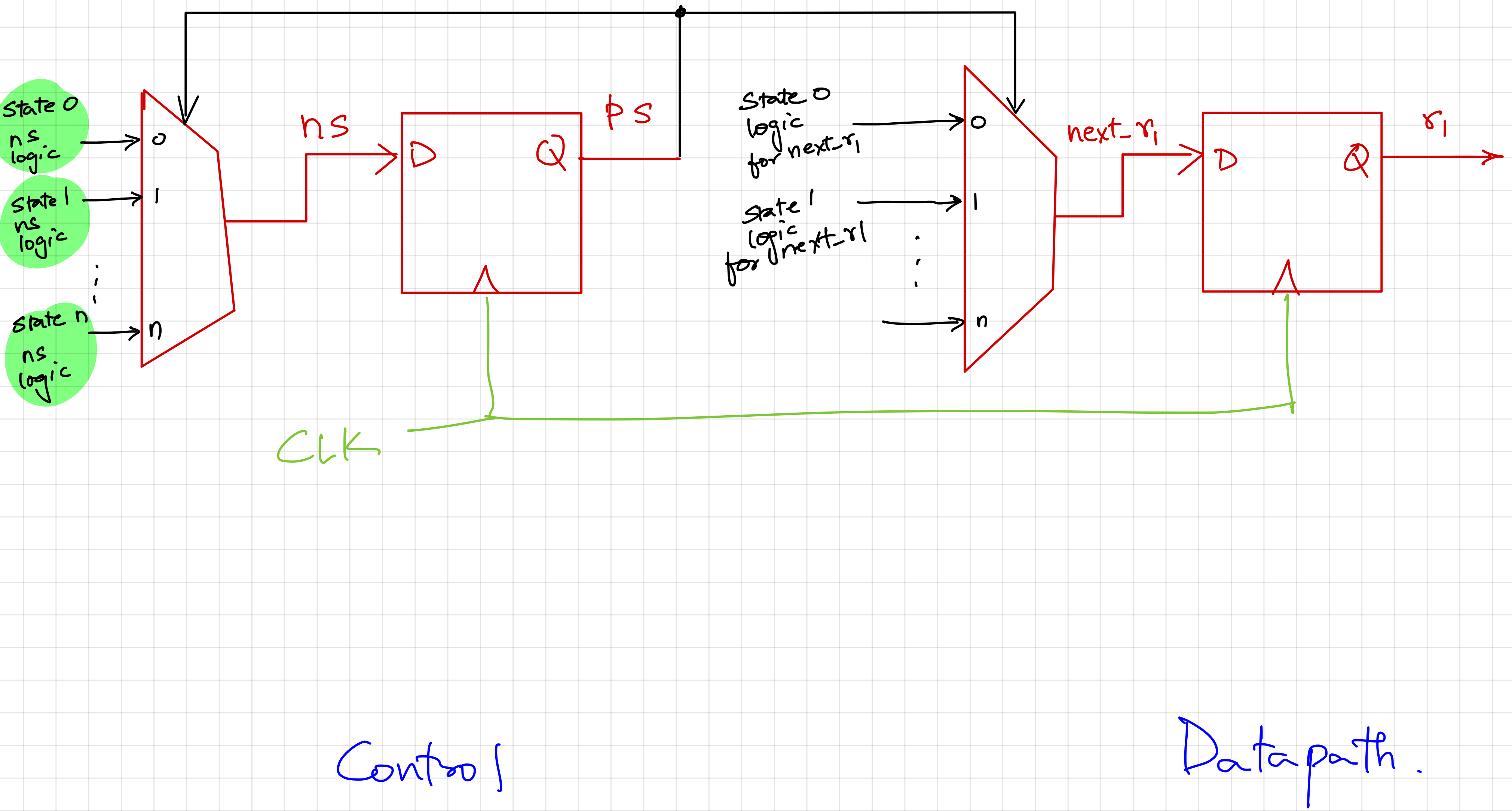
ASMD Hardware

Control Datapath.

- State transitions and RTL operations are both controlled by the clock
  - It's often helpful to remember the underlying hardware – registers!

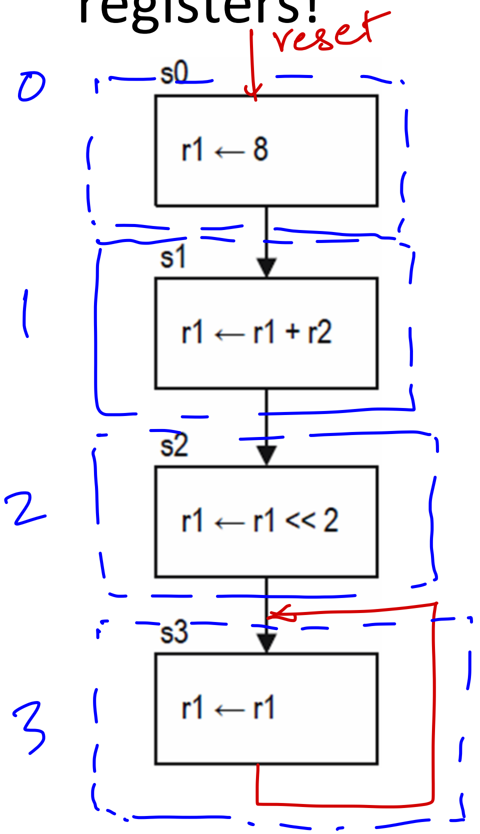
- The behavior of both state and data registers depend on the current control state
  - Can conceptually think of as a MUX to the registers' inputs that uses the current state as its selector bits

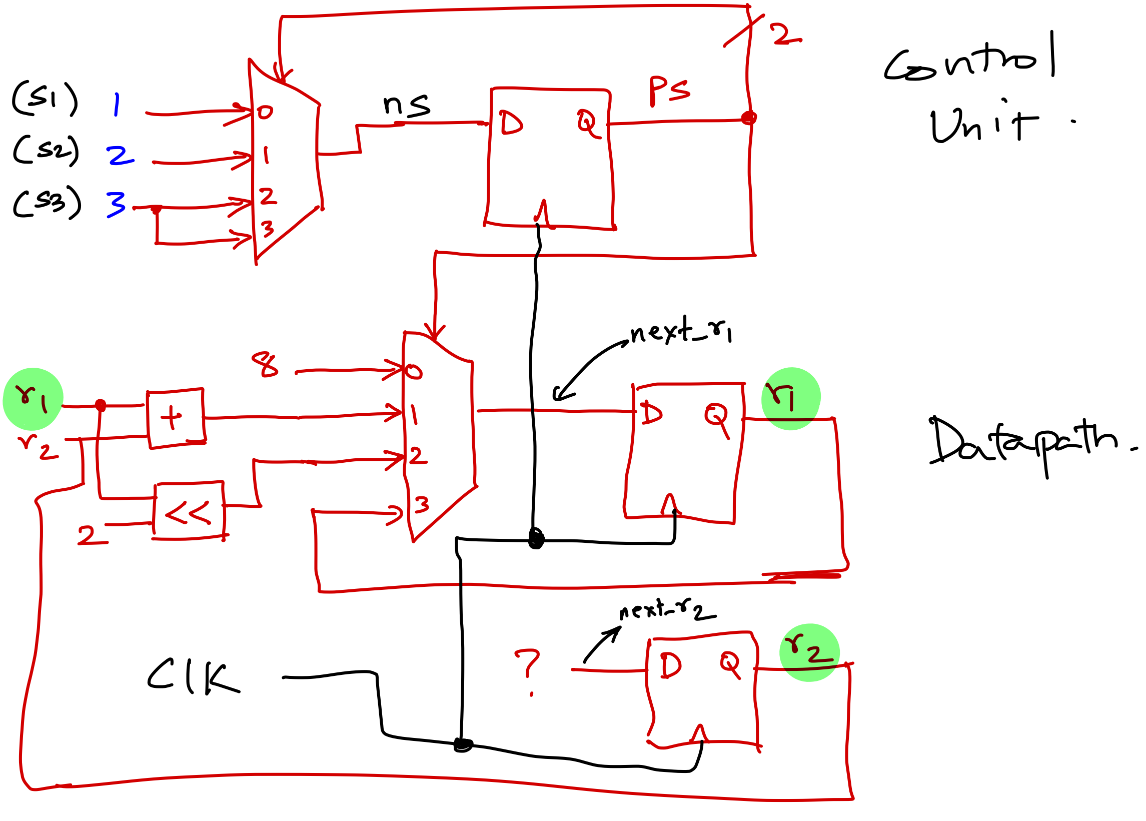
next slide


State 0 ns logic
state 1 ns logic
state n ns logic
0
1
n
ns
D
Q
ps
CLK
State 0 logic for next_r1
state 1 logic for next_r1
0
1
n
next_r1
D
Q
r1
Control
Datapath.

Hardware Example #1

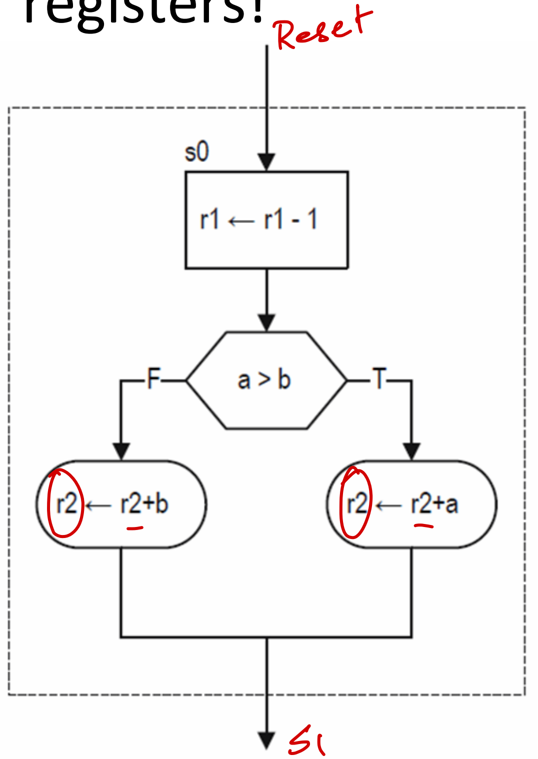
- State transitions and RTL operations are both controlled by the clock
  - It's often helpful to remember the underlying hardware – registers!

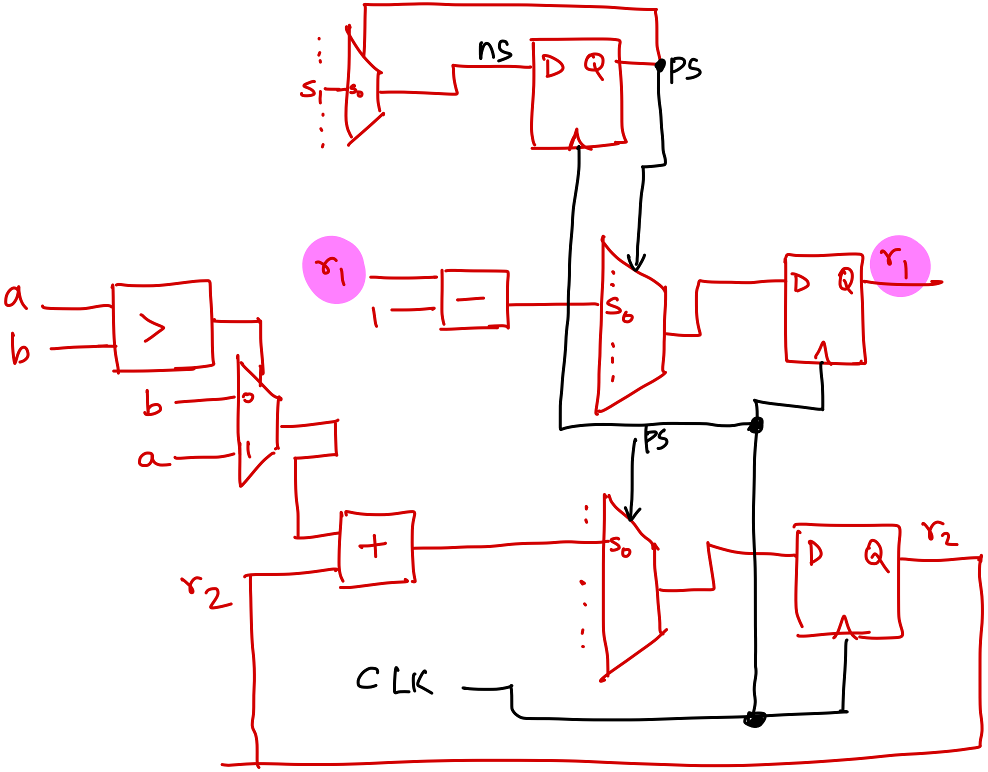




Hardware Example #2

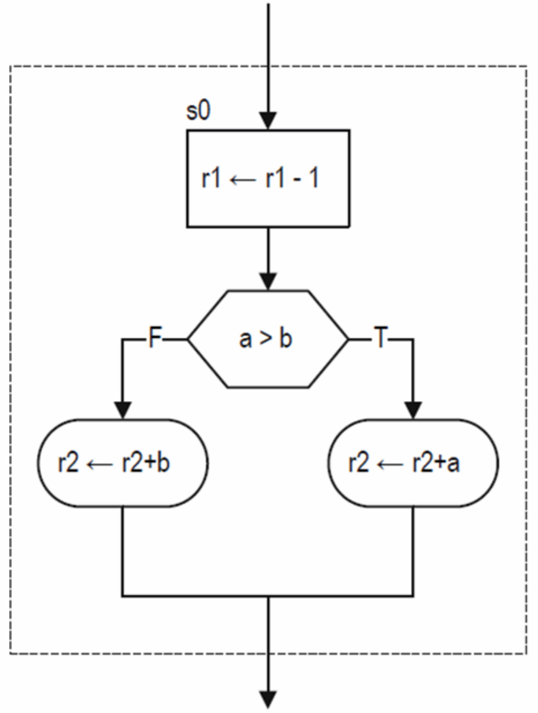
- State transitions and RTL operations are both controlled by the clock
  - It’s often helpful to remember the underlying hardware – registers!

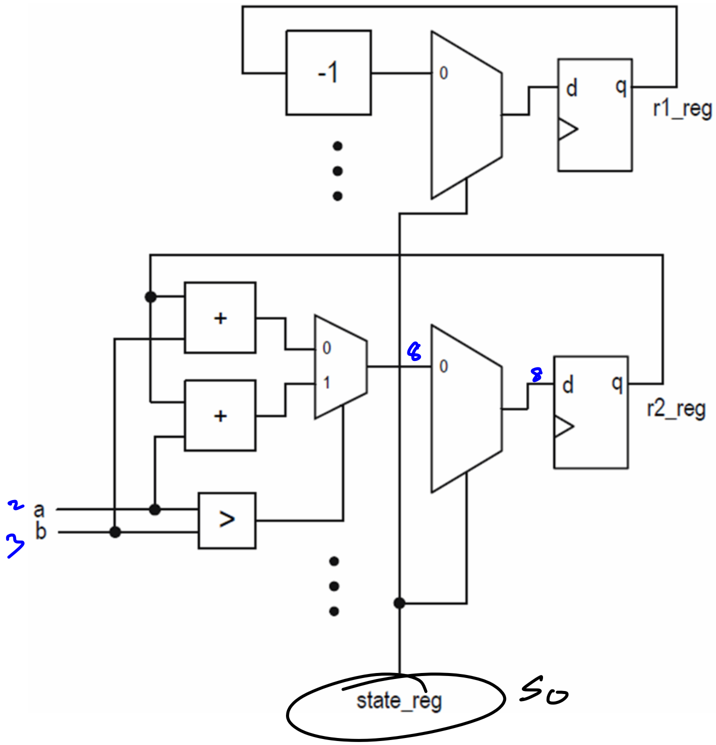




ASMD Timing

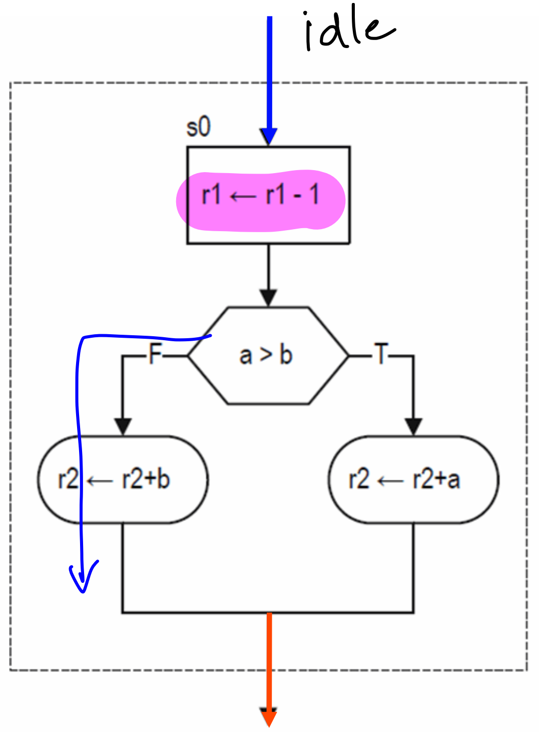
- Everything (registers!) within an ASM block occurs simultaneously at the *next* clock trigger
  - Differs from a flowchart – changes occur at state *exit* rather than *entrance*

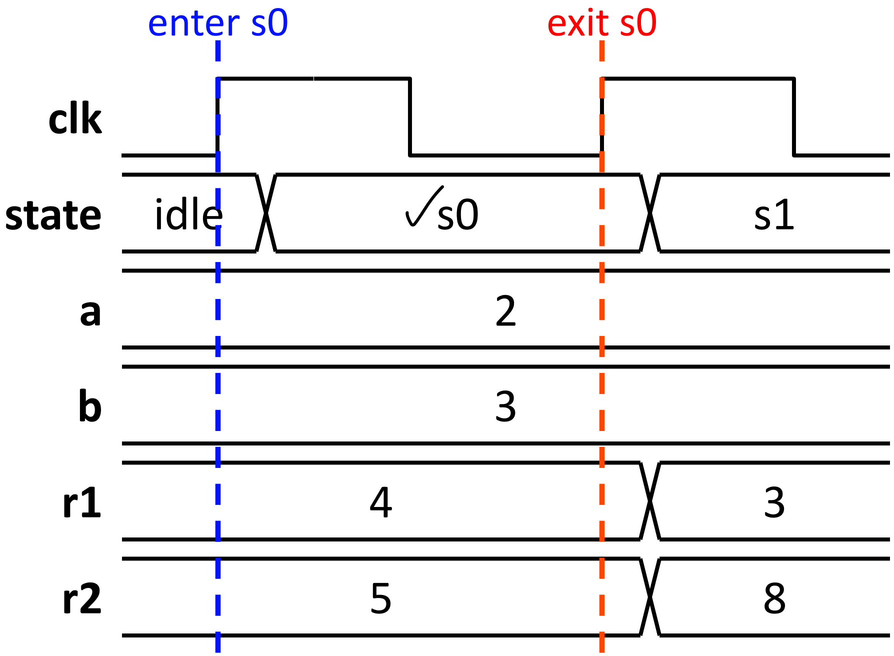




ASMD Timing

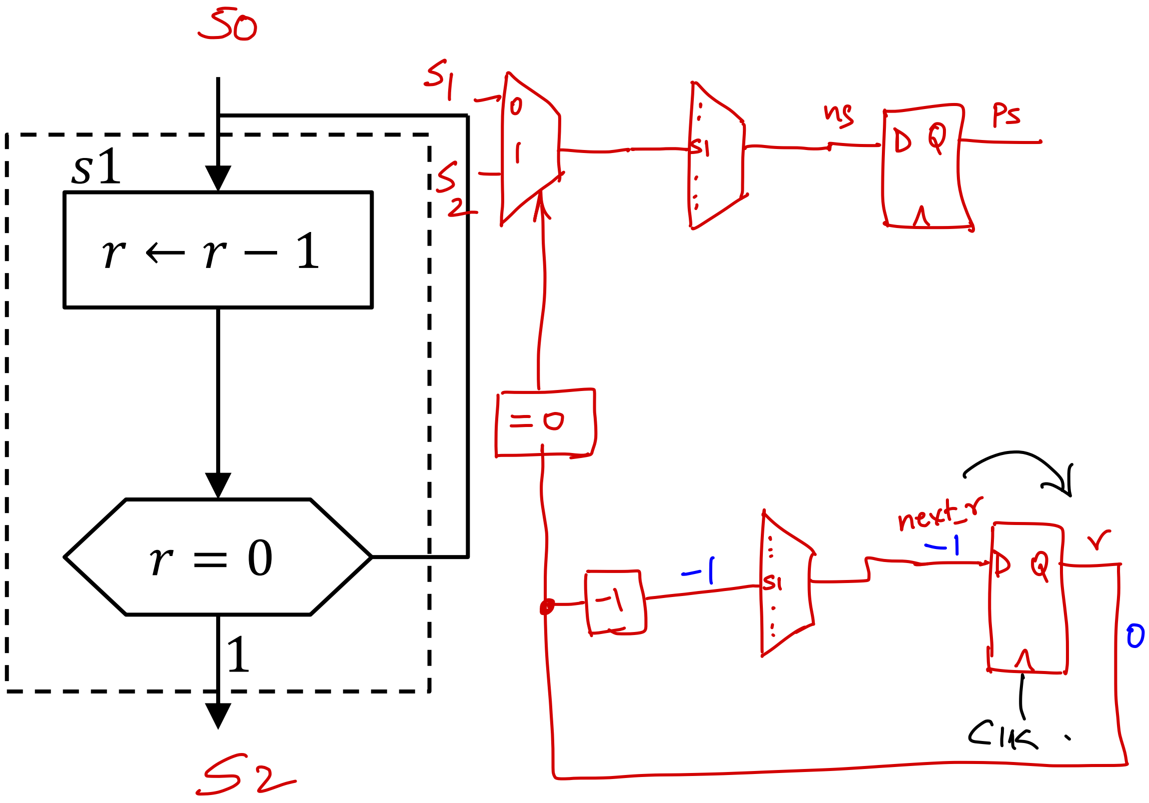
- Everything (registers!) within an ASM block occurs simultaneously at the *next* clock trigger
  - Differs from a flowchart – changes occur at state *exit* rather than *entrance*





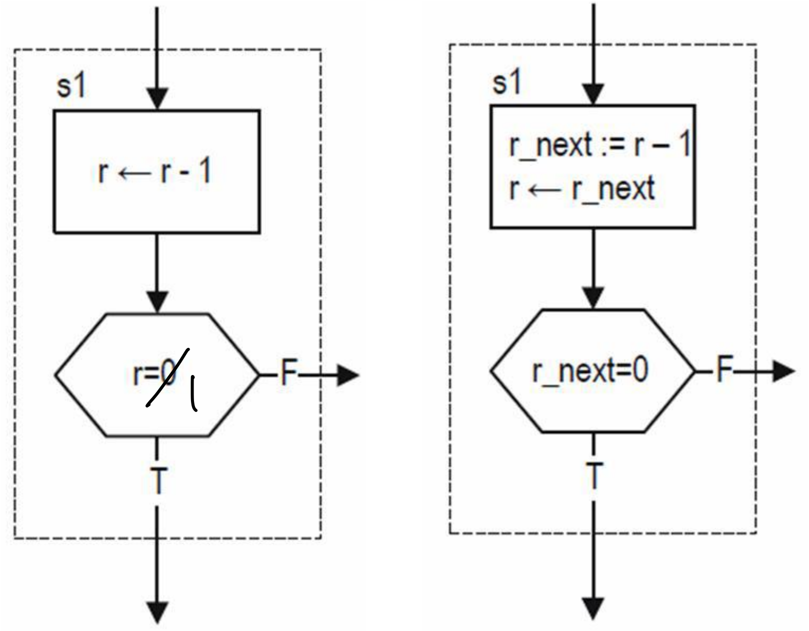
ASMD Timing Question

- What value will be stored in r when we transition from state $s1$ to the next state? -1, 0, 1



ASMD Timing

- When a registered output ($e.g.$, r) is used in a decision box, its effect may appear to be delayed by one clock
  - Can define a next-state value ($e.g.$, r_next) to use instead



(a) Use old value of r (b) Use new value of r

Short Tech Break

ASMD Design Procedure

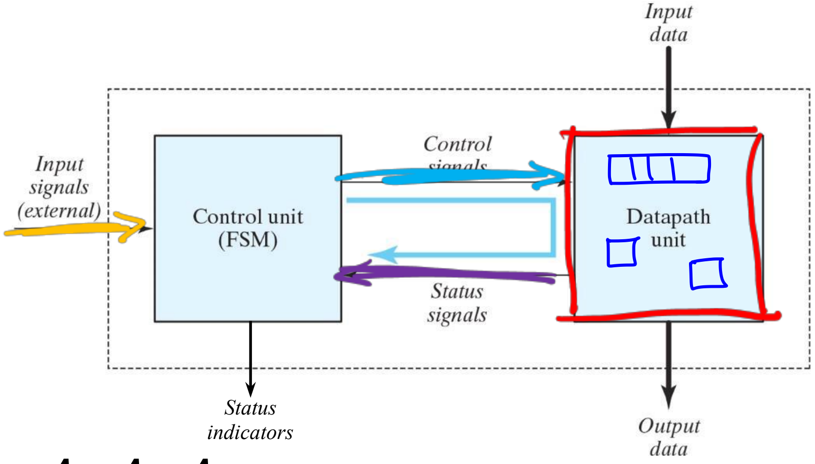
- From problem description or algorithm pseudocode:

1) **Identify necessary datapath components and operations**
2) **Identify states and signals that cause state transitions** (external inputs and status signals), based on the necessary sequencing of operations

Control

3) **Name the control signals** that are generated by the controller that cause the indicated operations in the datapath unit
4) **Form an ASM chart for your controller**, using states, decision boxes, and signals determined above
5) **Add the datapath RTL operations** associated with each control signal

Design Example #1

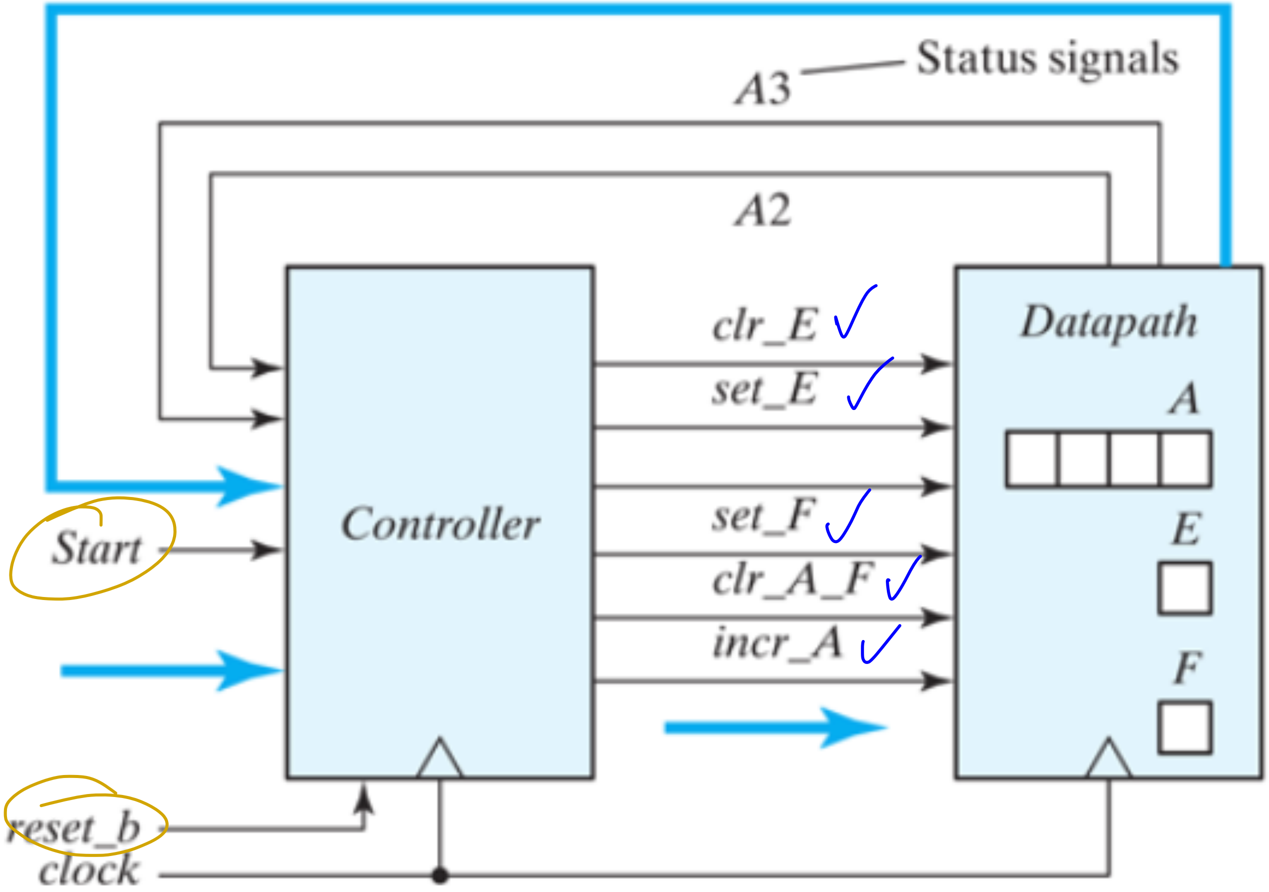


- System specification:
  - datapath
  - Flip-flops E and F
  - 4-bit binary up-counter $A = 0\text{b}A_3A_2A_1A_0$
  - Active-low reset signal $reset\_b$ puts us in state S_idle, where we remain while signal $Start = 0$
  - clr_A_F
  - Status Signals
  - $Start = 1$ initiates the system's operation by clearing A and F. At each subsequent clock pulse, the counter is incremented by 1 until the operations stop. incr_A
  - Control Signals in blue.
  - Bits A_2 and A_3 determine the sequence of operations:
    - clr_E
    - If $A_2 = 0$, set E to 0 and the count continues
    - set_E
    - If $A_2 = 1$, set E to 1; additionally, if $A_3 = 0$, the count continues, otherwise, wait one clock pulse to set F to 1 and stop counting (*i.e.*, back to S_idle)
    - Set_F

When does counting stops
A= ?
1101

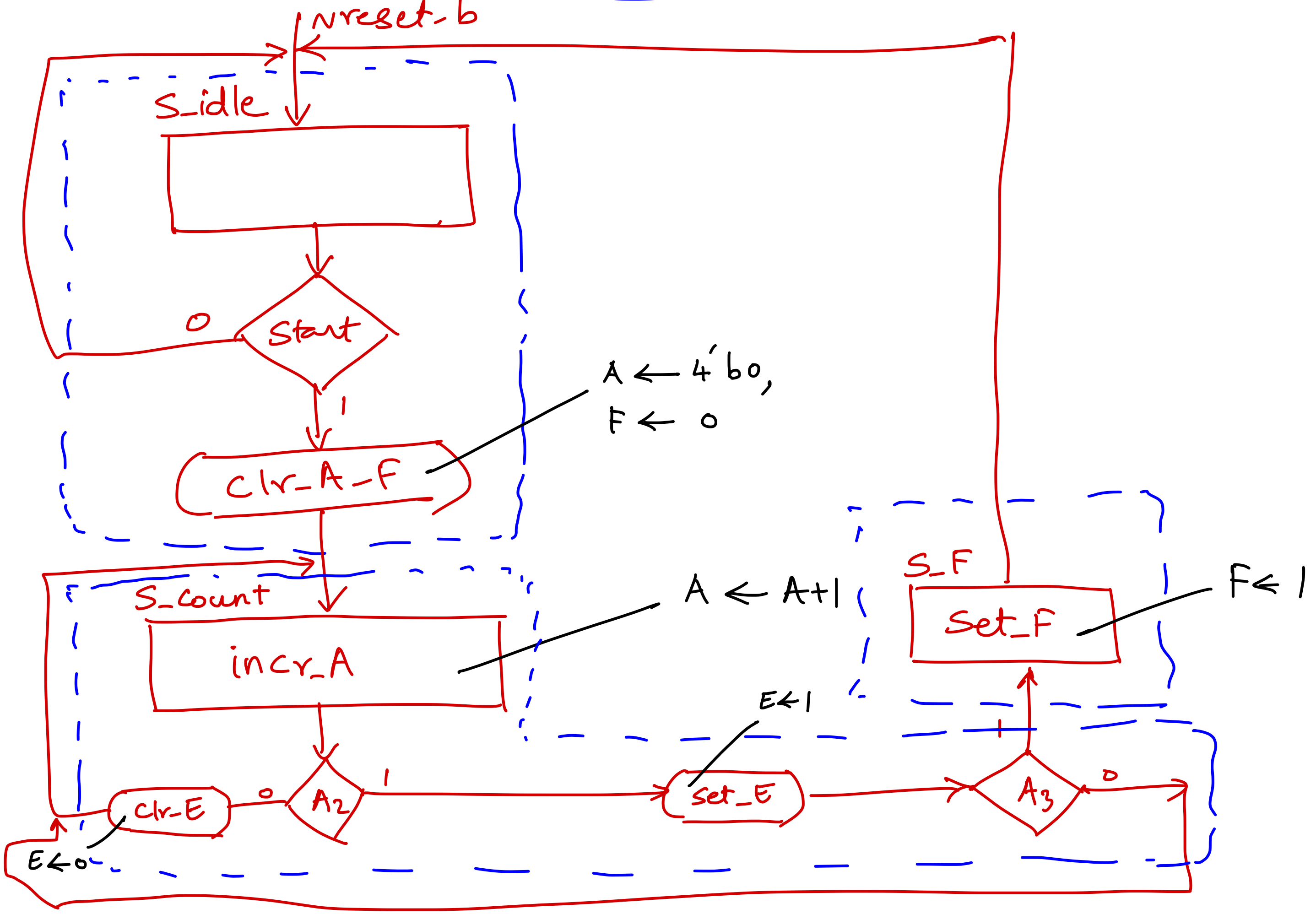
Design Example #1

- The system can be represented by the following block diagram:



Design Example #1 (ASM → ASMD Chart)

- Synchronous or asynchronous reset?



nreset_b
S_idle
0
Start
1
clr_A_F
A ← 4'b0,
F ← 0
S_count
incr_A
A ← A+1
S_F
Set_F
F ← 1
E ← 1
clr_E
0
A2
1
set_E
A3
0
1
E ← 0