CS221: Digital Design

RTL Design using ASM

A. Sahu

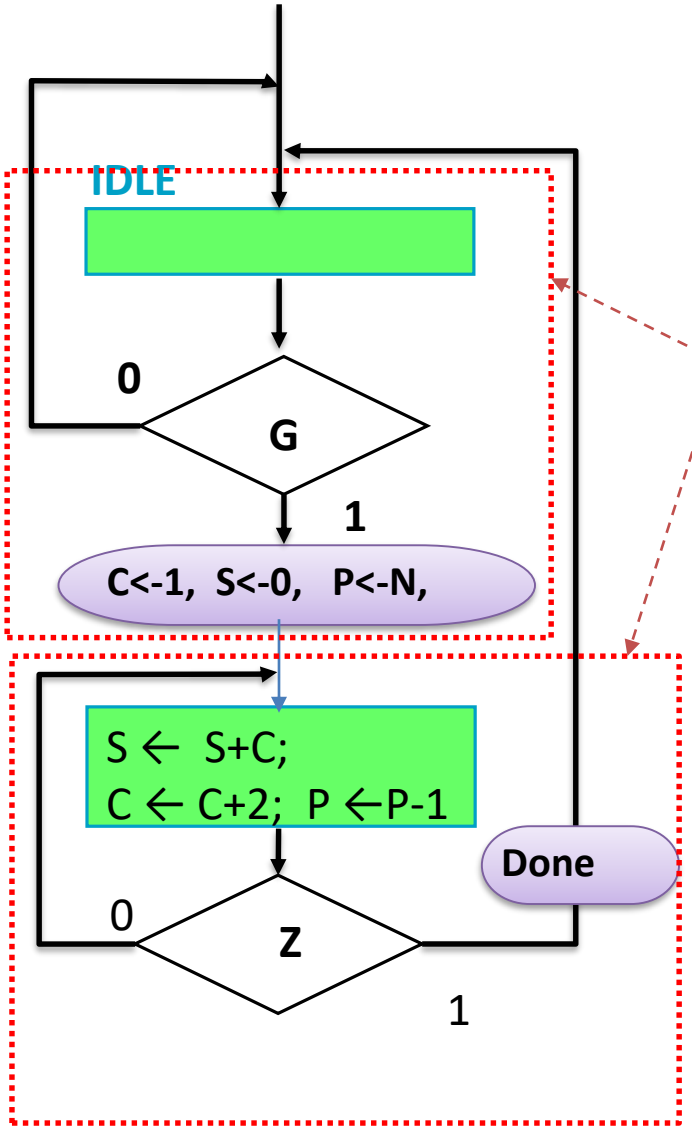
Dept of Comp. Sc. & Engg.

Indian Institute of Technology Guwahati

Example:
Design a Digital Circuit to Sum of First N odd numbers

It is N^2 but we want to use basic summation method 1+3+5+7..+(2n-1)

Sum of First N odd numbers



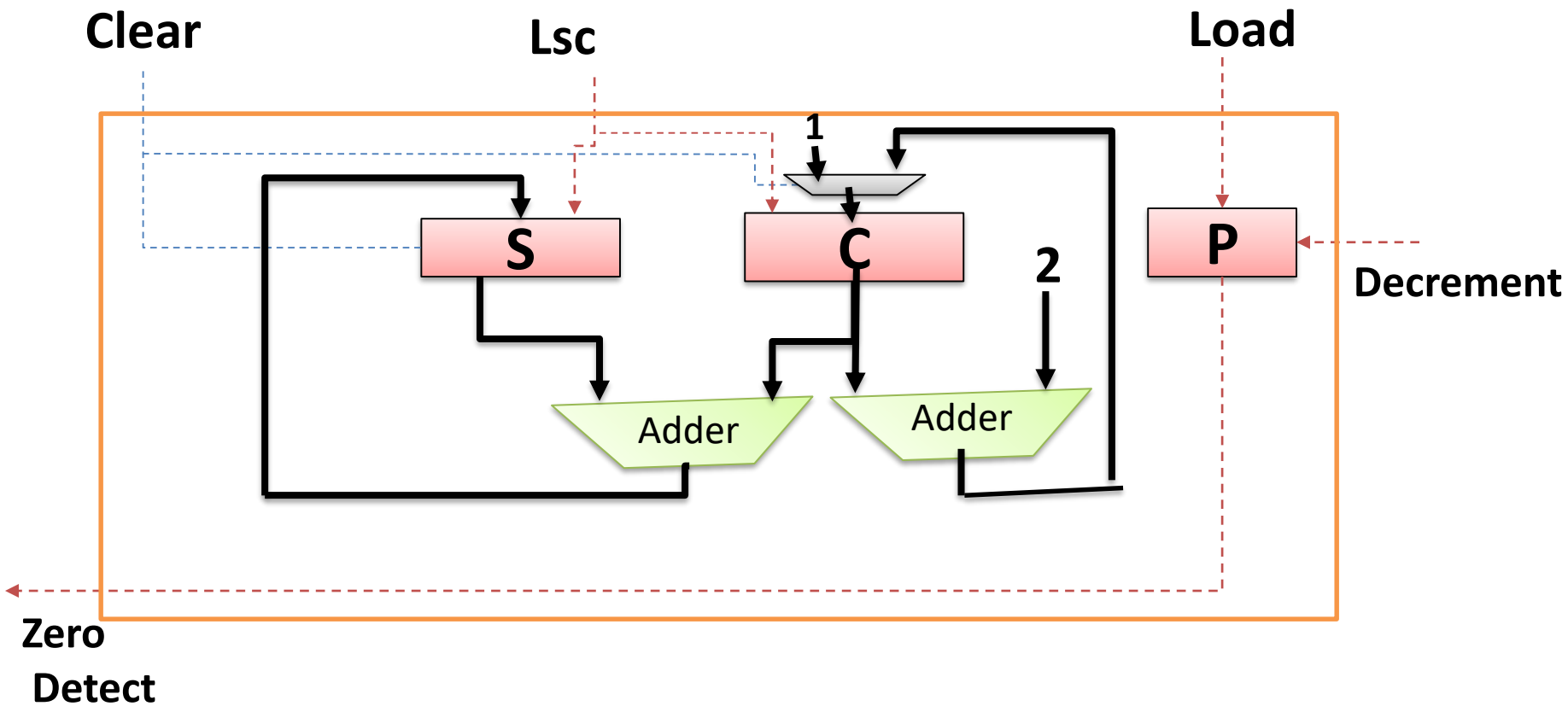
Require : two adders, decrement register, S reg, C reg

In state : We can put RTL like statement

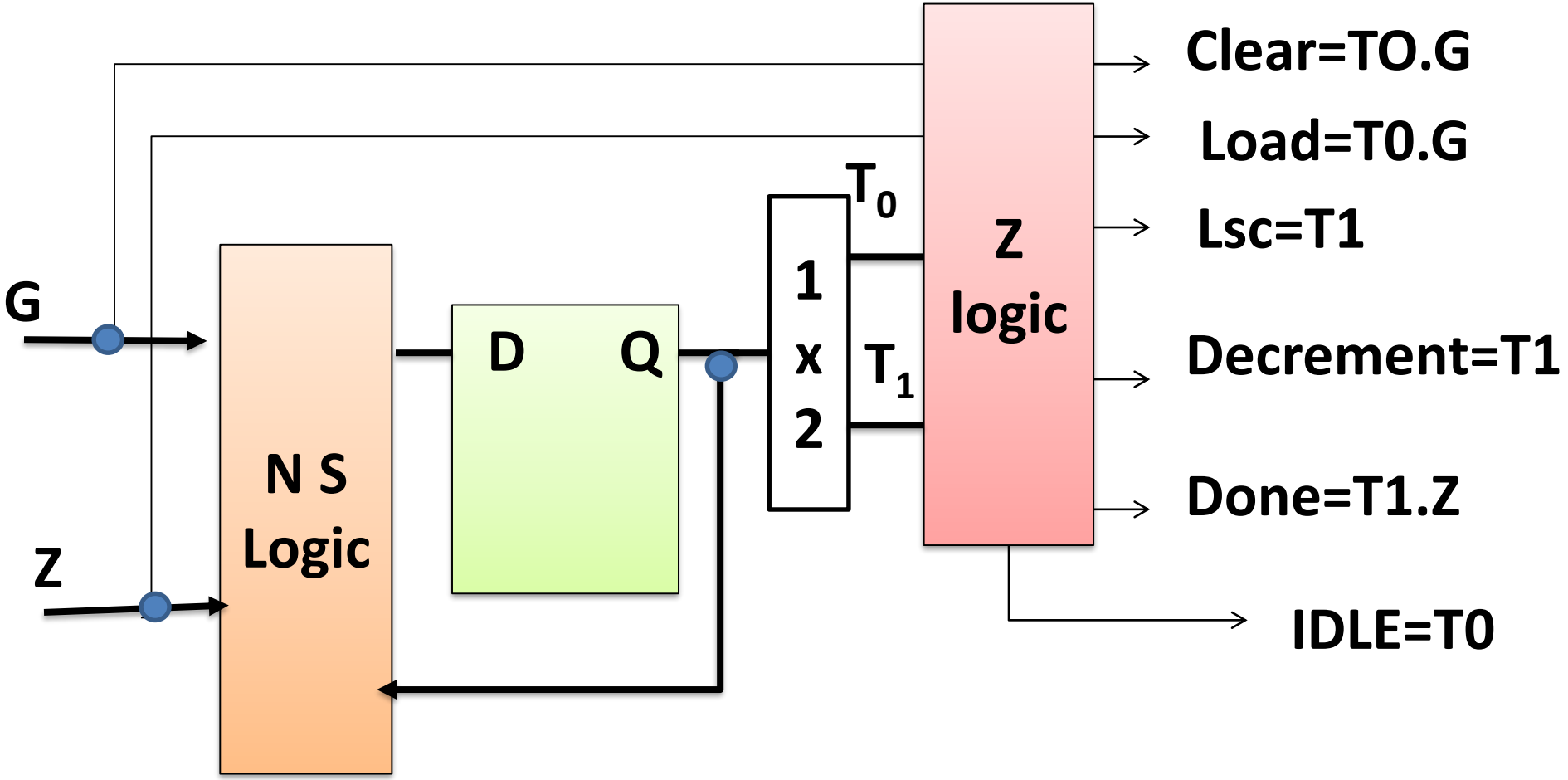
$S \leftarrow S+C,\ C \leftarrow C+2,\ P \leftarrow P-1$

$C \leftarrow 1,\ S \leftarrow 0,\ P \leftarrow N$

DP: Sum of First N odd numbers

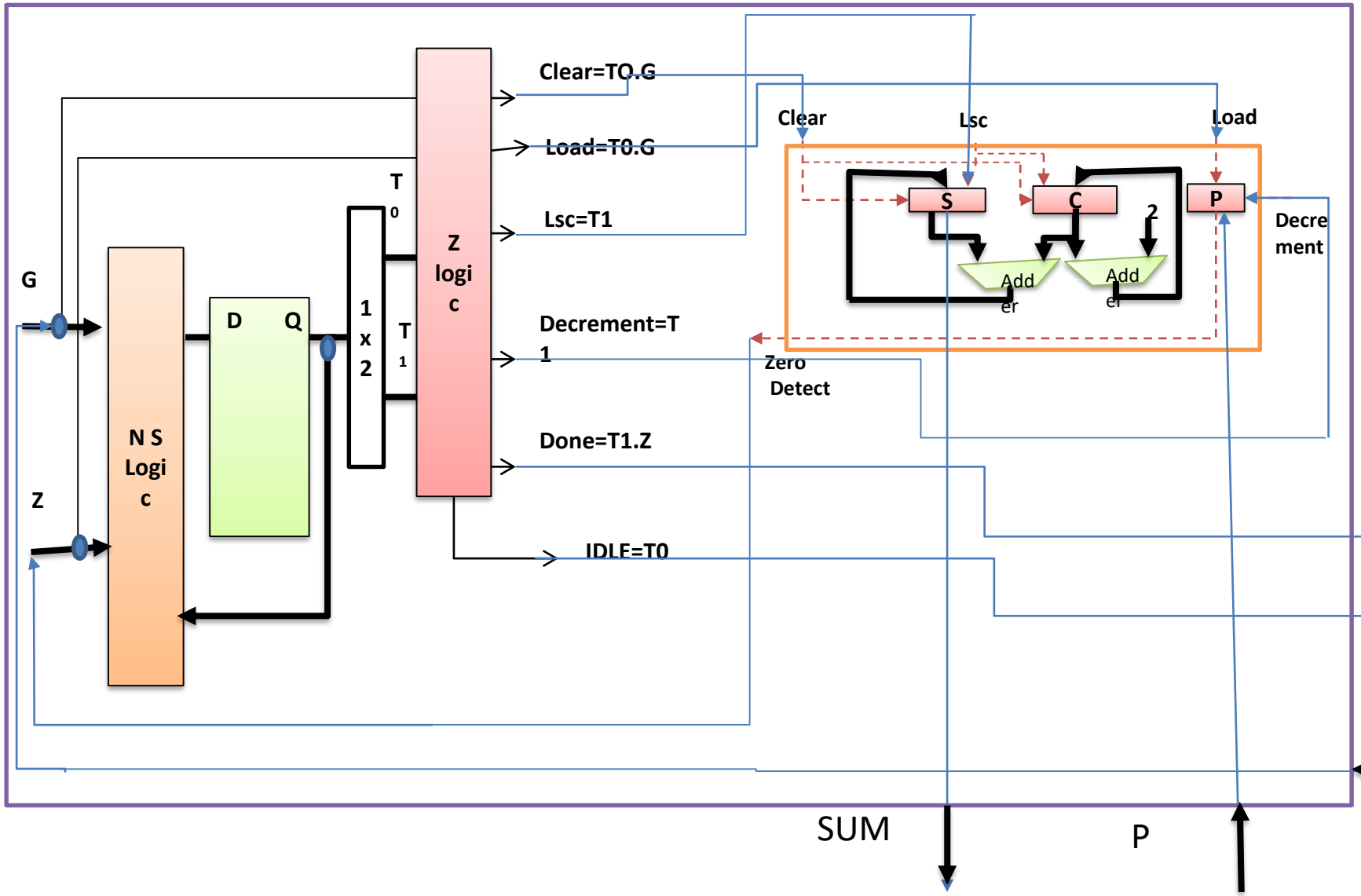


CP: Sum of First N odd numbers



D=T0.G+T1.Z’

CP+DP: Sum of First N odd numbers

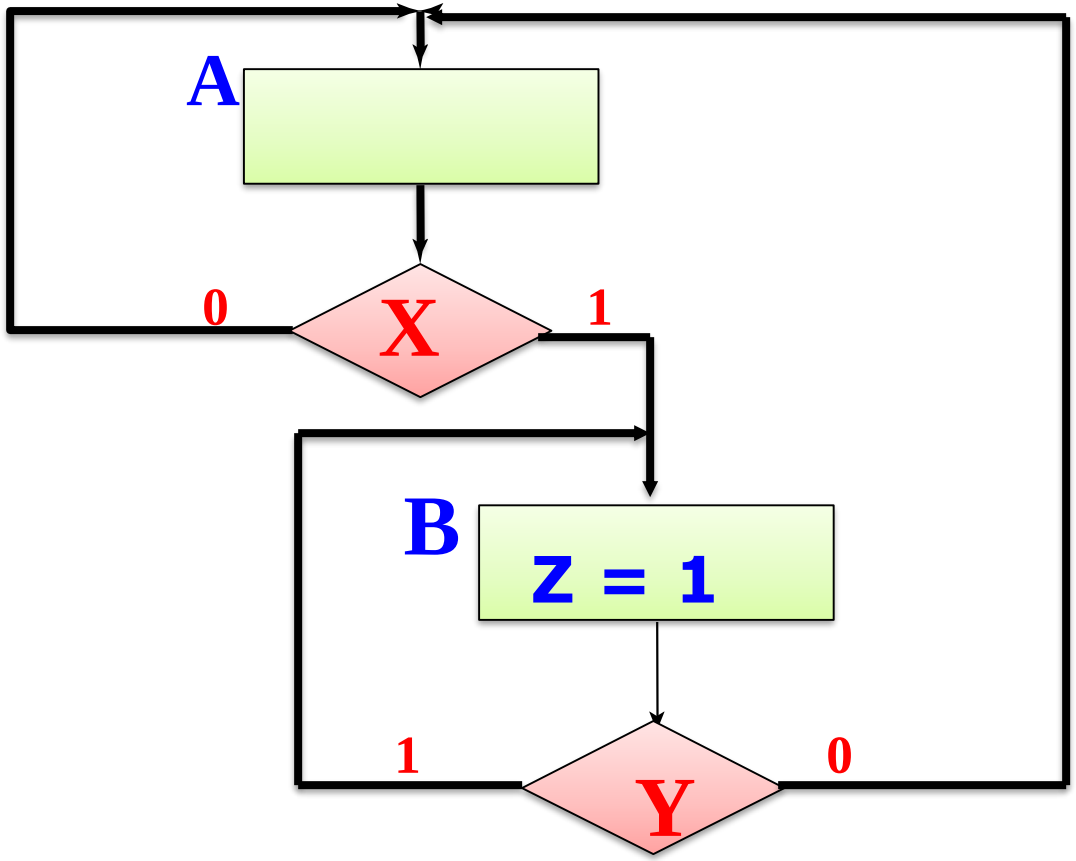


Another Example

ASM Design Example

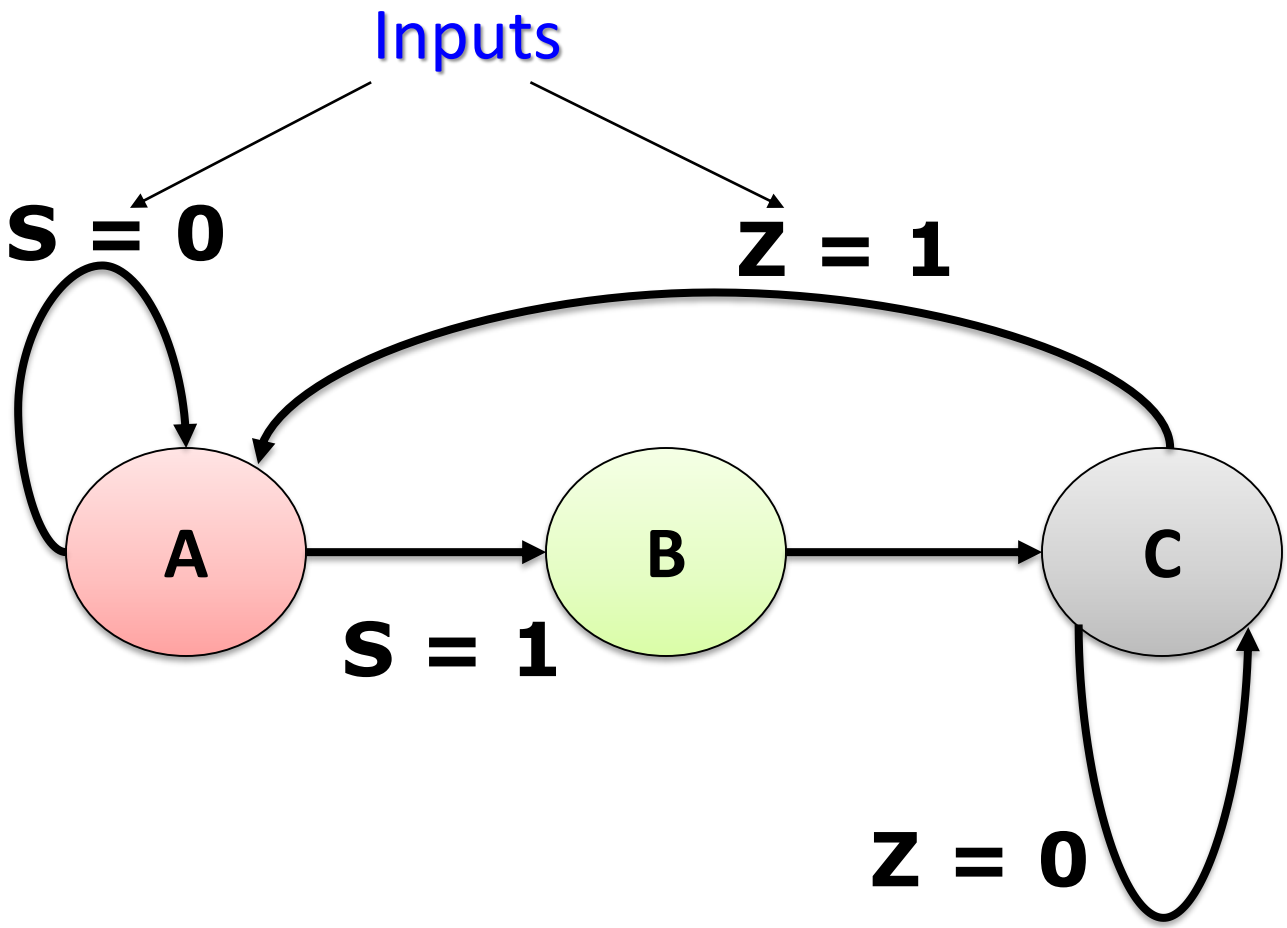
- Find the ASM chart corresponding to the following description
  - **There are two states A, B**
  - **If in state A and input X is `0' then the next state is A**
  - **If in state A and input X is `1' then the next state is B**
  - If in state B and input Y is `1' then the next state is B
  - If in state B and input Y is `0' then the next state is A
  - **Output Z is equal to `1' while the circuit is in state B**
- Solution:
  - Total States → 2
  - Two Inputs → X, Y
  - One Output → Z

ASM Design Example



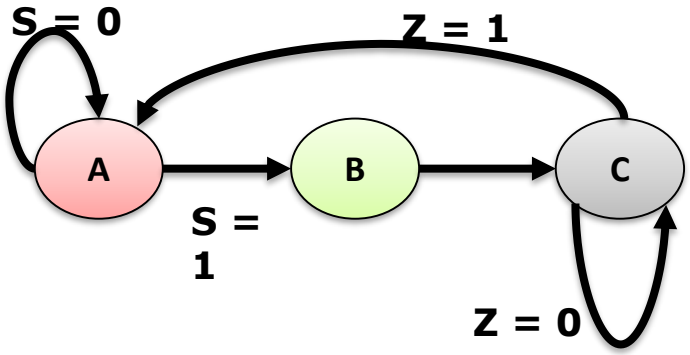
Another Example

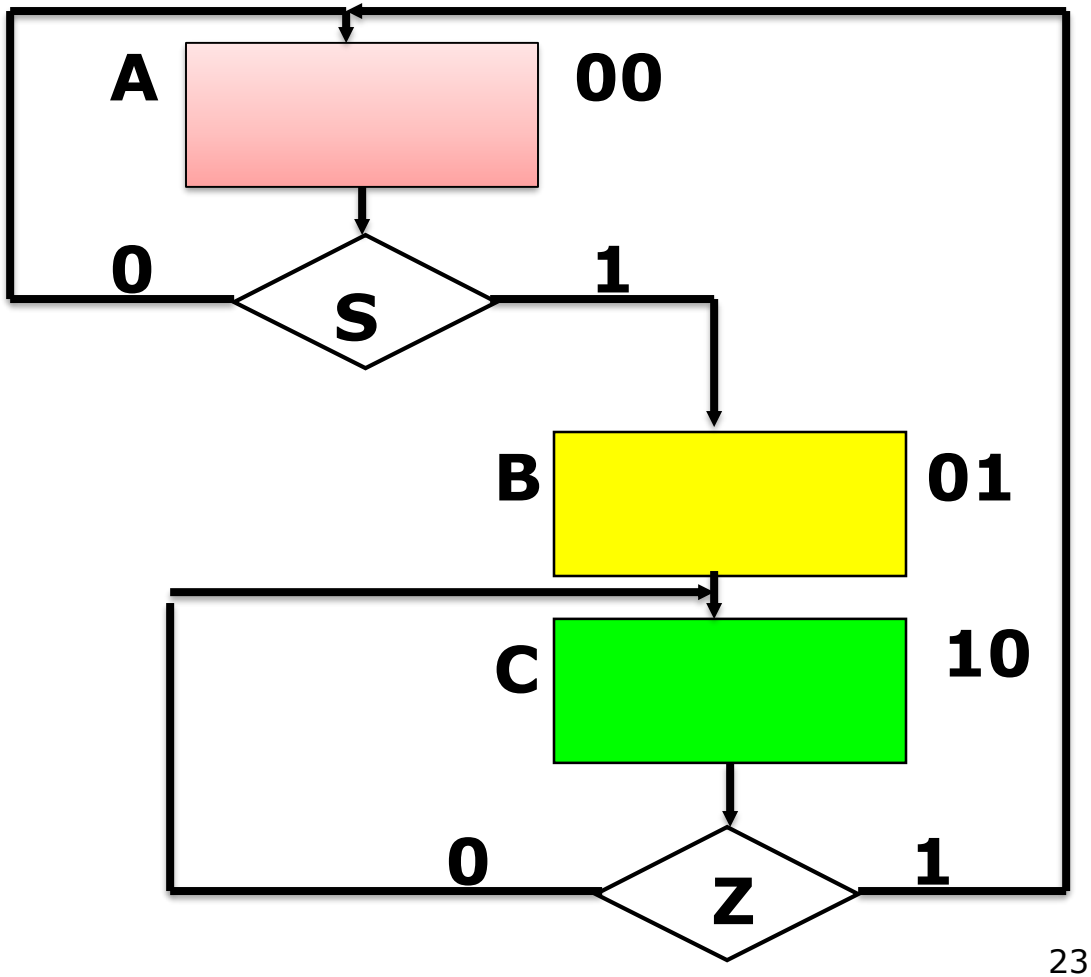
Another Example: From FSM to ASM



Similar to the ASM of : Sum of First N odd numbers

Another Example: From FSM to ASM

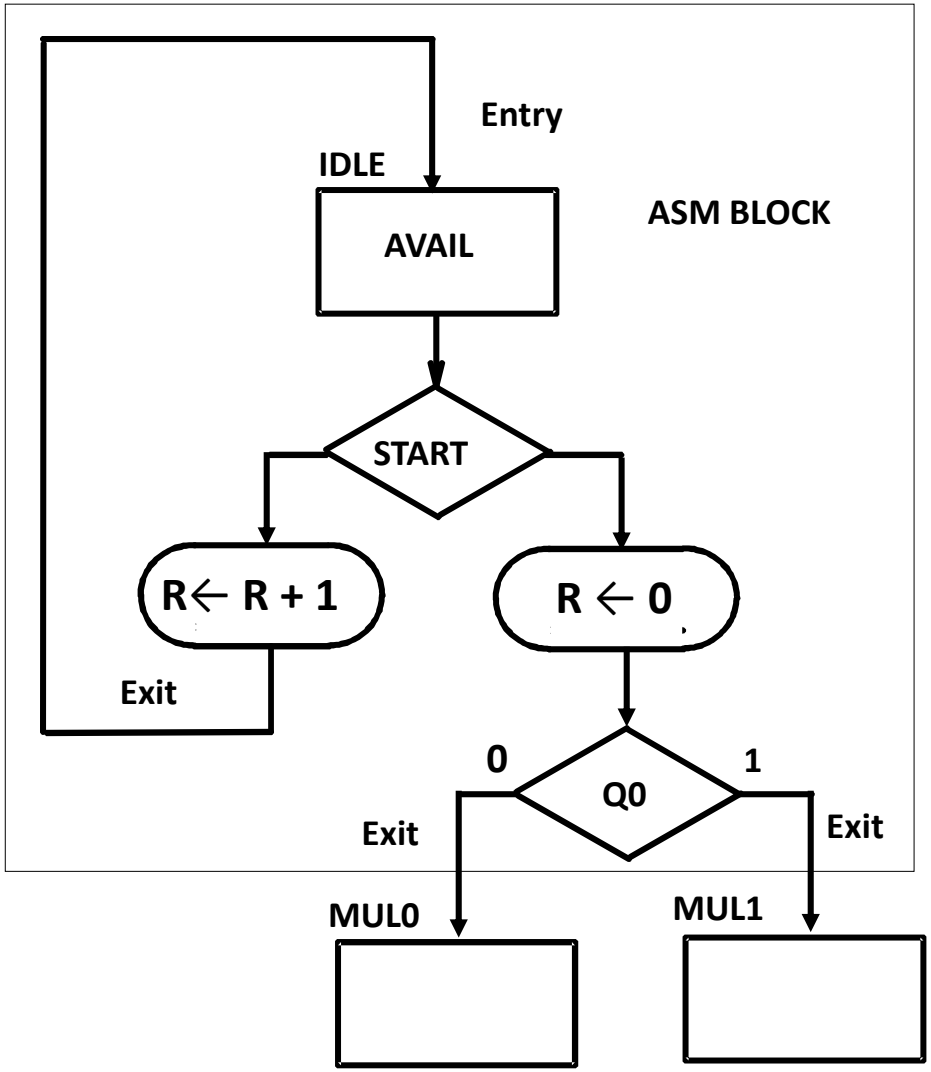




ASM Block

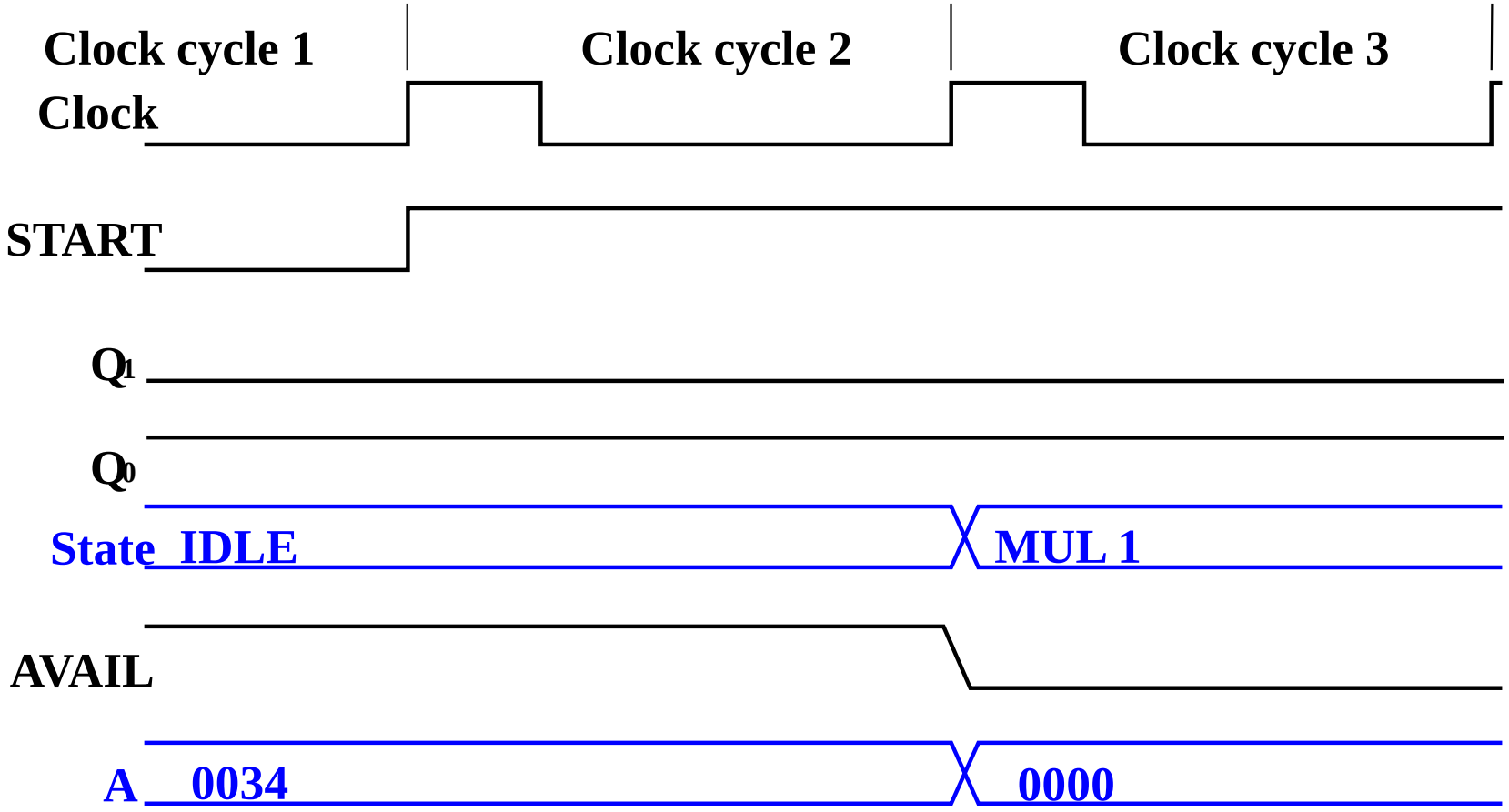
ASM Blocks

One ASM block execute in one cycle



ASM Timing

- Outputs appear while in the state
- Register transfers occur at the clock while exiting the state - New value occur in the next state!



Another Example

ASM : DP+CP Example

- Find the Data Path and ASM for the following problem:
  - We **first** need to load two registers (R1, R2) with some value.
  - We will **then** need to add the two Registers (R1, R2) and save the result in Register R3.
  - All these operations **should occur if** a "**start**" Signal is activated.

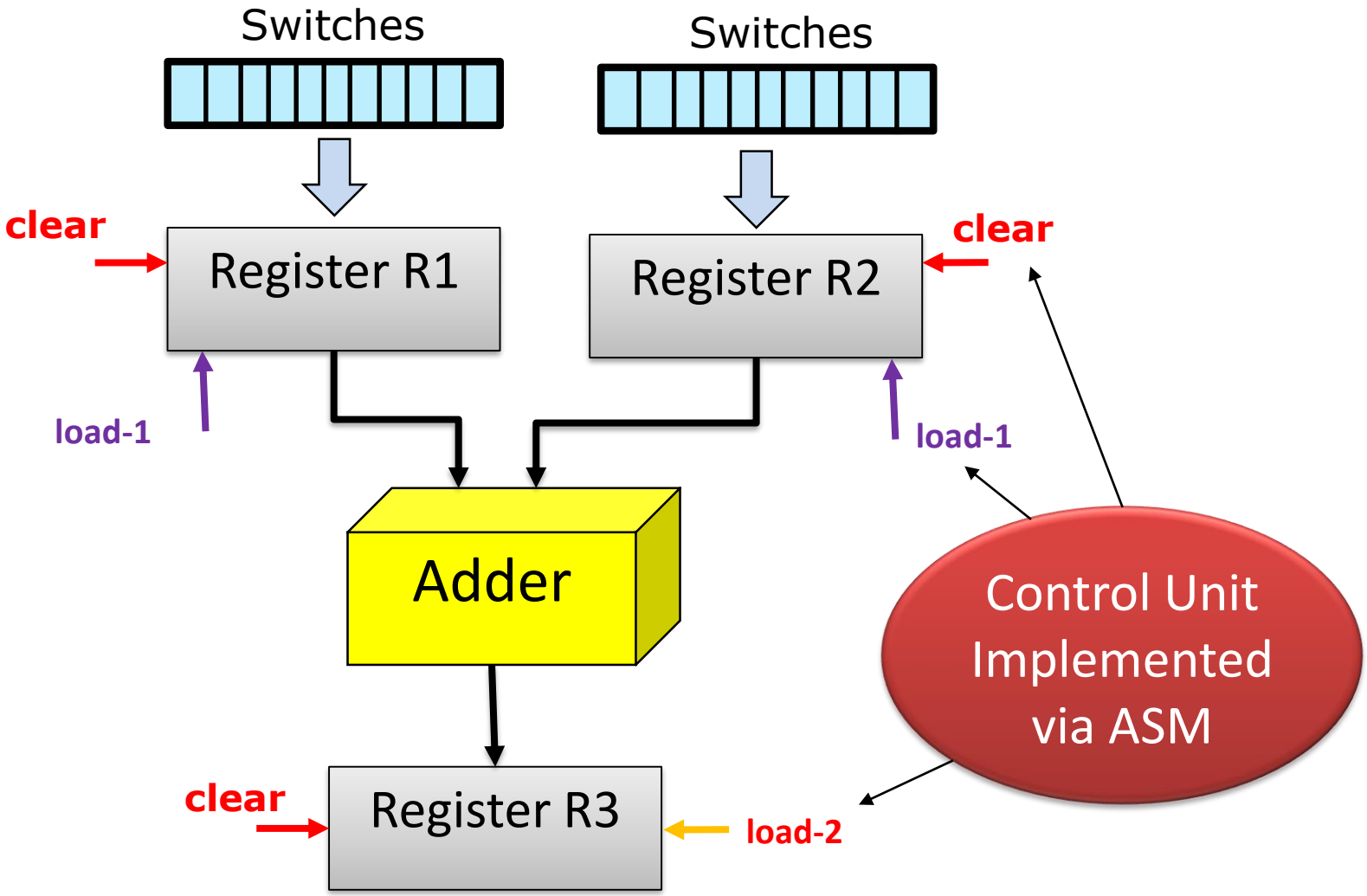
ASM : DP+CP Example

- Translation to Hardware:
  - We need to clear the registers first.
  - If the "**start**" signal is set to 0, I do nothing
  - Else If the "**start**" signal is set to 1, I will **load** R1, R2 with values
  - Next, enable R3 to be loaded (load-2)by the results of R1+R2

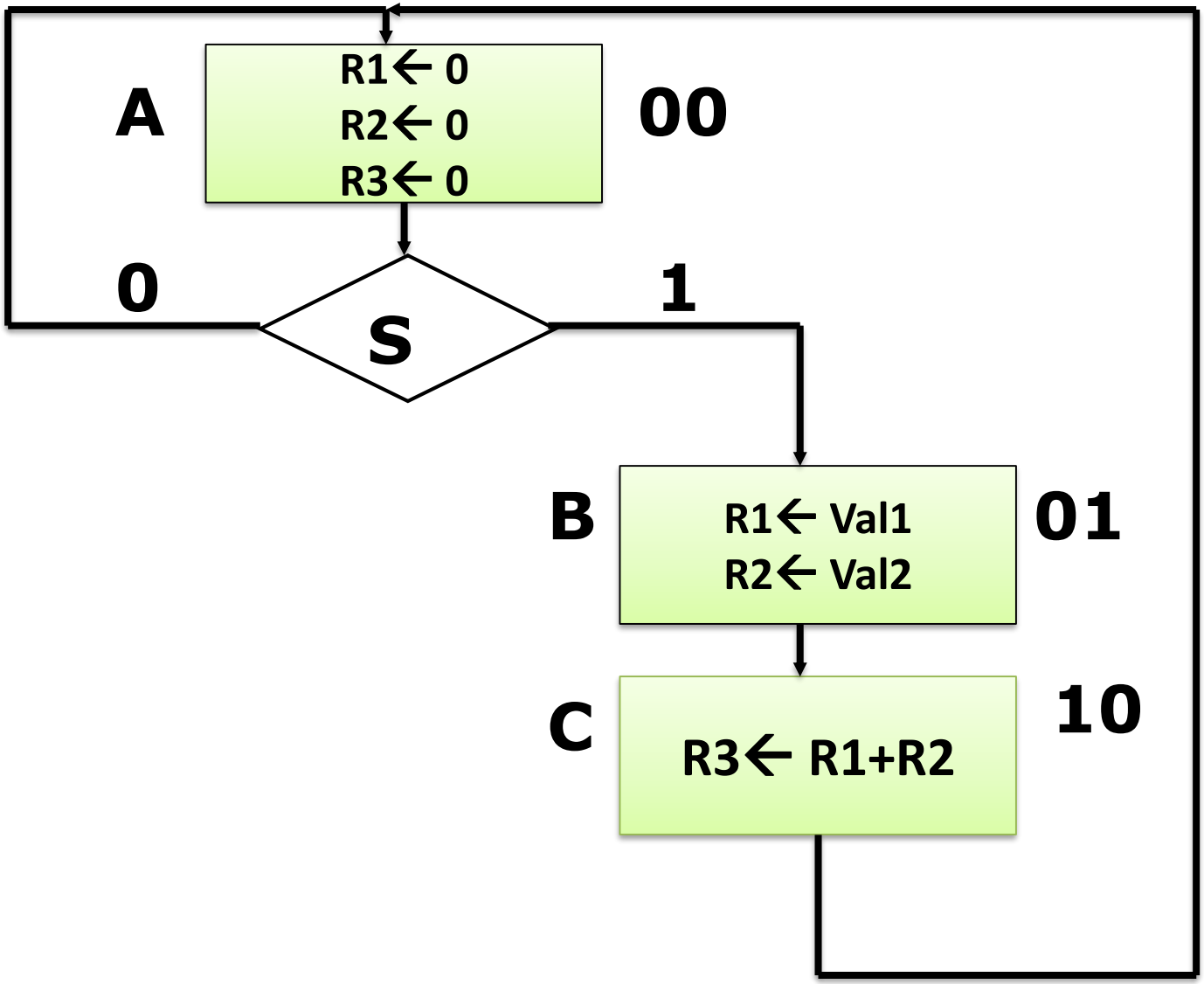
ASM : DP+CP Example

- Inputs/Outputs:
  - **start** is an input signal (Use a switch)
  - **clear** is an output signal generated and connected to R1, R2, R3
  - **load-1** is an output signal generated and connected to R1, R2
  - **load-2** is an output signal generated and connected to R3

Data Path


Switches
Switches
clear
Register R1
Register R2
clear
load-1
load-1
Adder
Control Unit
Implemented
via ASM
clear
Register R3
load-2

ASM : DP+CP Example



ASM : DP+CP Example Controller

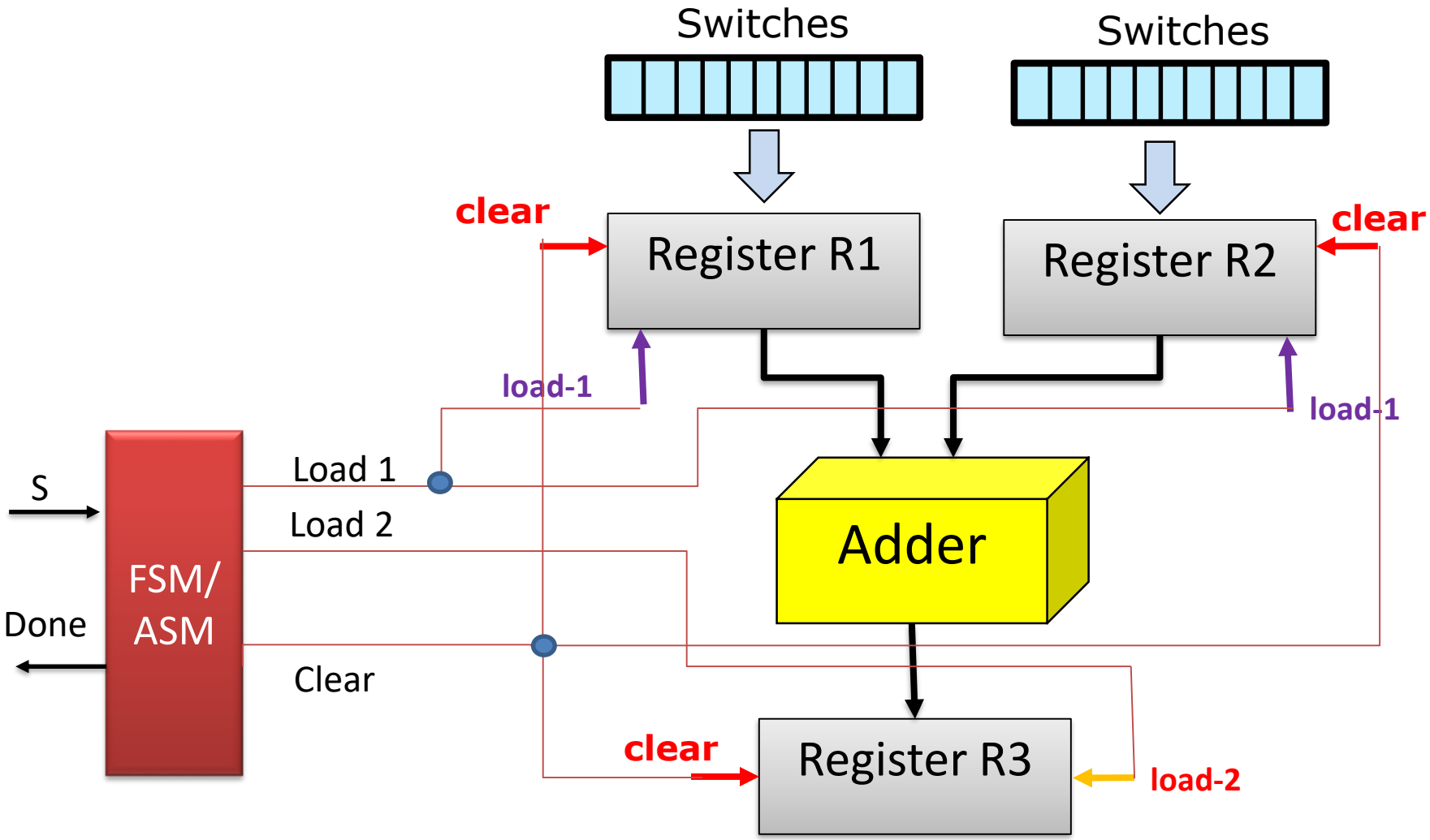
PS		S	NS	
0	0	0	0	0
0	0	1	0	1
0	1	X	1	0
1	0	X	0	0
1	1	X	0	0

PS		CLR	L1	L2
0	0	1	0	0
0	1	0	1	0
1	0	0	0	1
1	1	X	X	X

D1=Q1'.Q0 D2=S.Q1'.Q0'

CLR=Q1'.Q0' L1=Q0'.Q1 L2=Q1.Q0'

Data Path


Switches
Switches
clear
Register R1
Register R2
clear
load-1
load-1
S
Load 1
Load 2
FSM/
ASM
Done
Clear
Adder
clear
Register R3
load-2