Chapter 14
FSM with Datapath
SKEE2263 Digital Systems

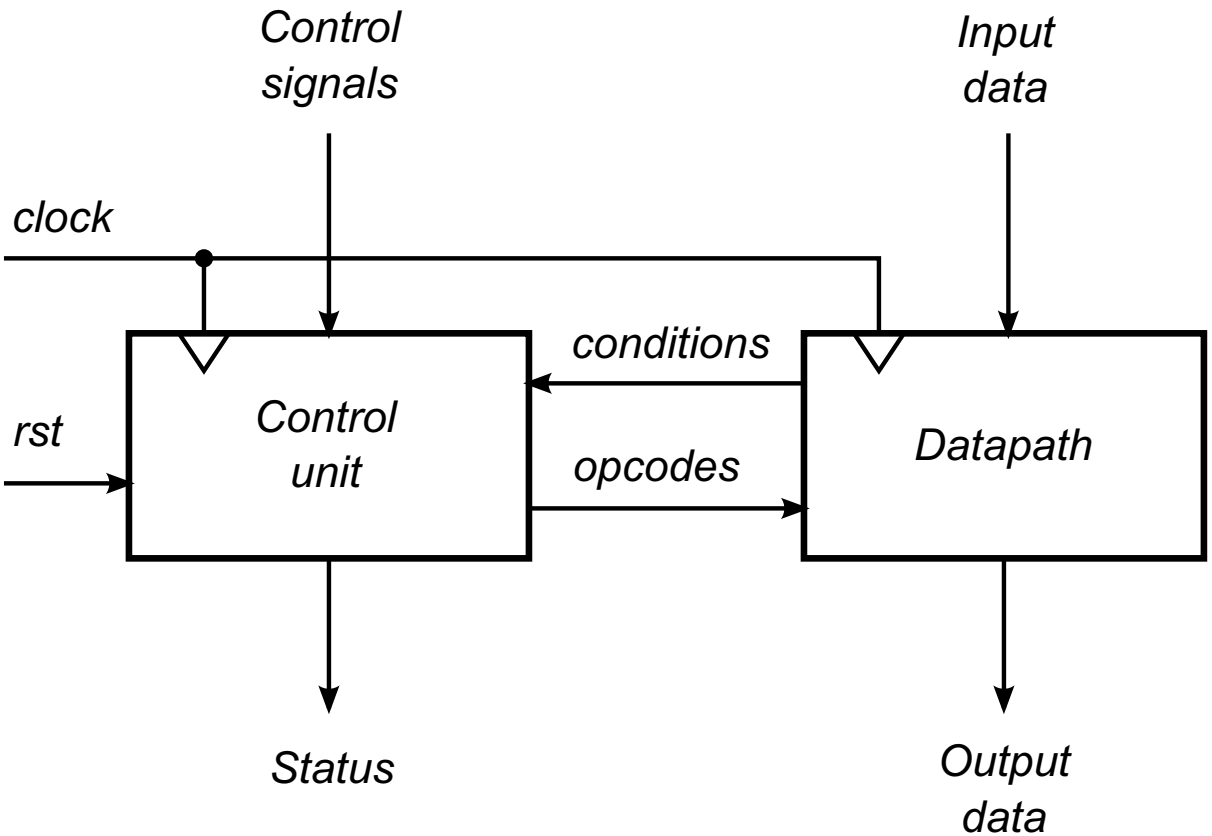
Mun'im Zabidi {munim@utm.my}
Ismahani Ismail {ismahani@fke.utm.my}
Izam Kamisian {e-izam@utm.my}

Faculty of Electrical Engineering, Universiti Teknologi Malaysia

April 30, 2018

Table of Contents

FSMD High Level Block Diagram



- FSMD = Finite State Machine with Datapath
- a.k.a. CU+DU

FSMD Components: Datapath Unit (DU)

The Datapath Unit (DU):

- Executes arithmetic and logic functions.
- Typically consists of
  - Registers to store variable of the algorithm
  - ALU and other combinational blocks that implement operation (operators) of the algorithm
  - Step counters that implement the loop counters.
  - Steering logic (multiplexers and buses)

FSMD Components: Control Unit (DU)

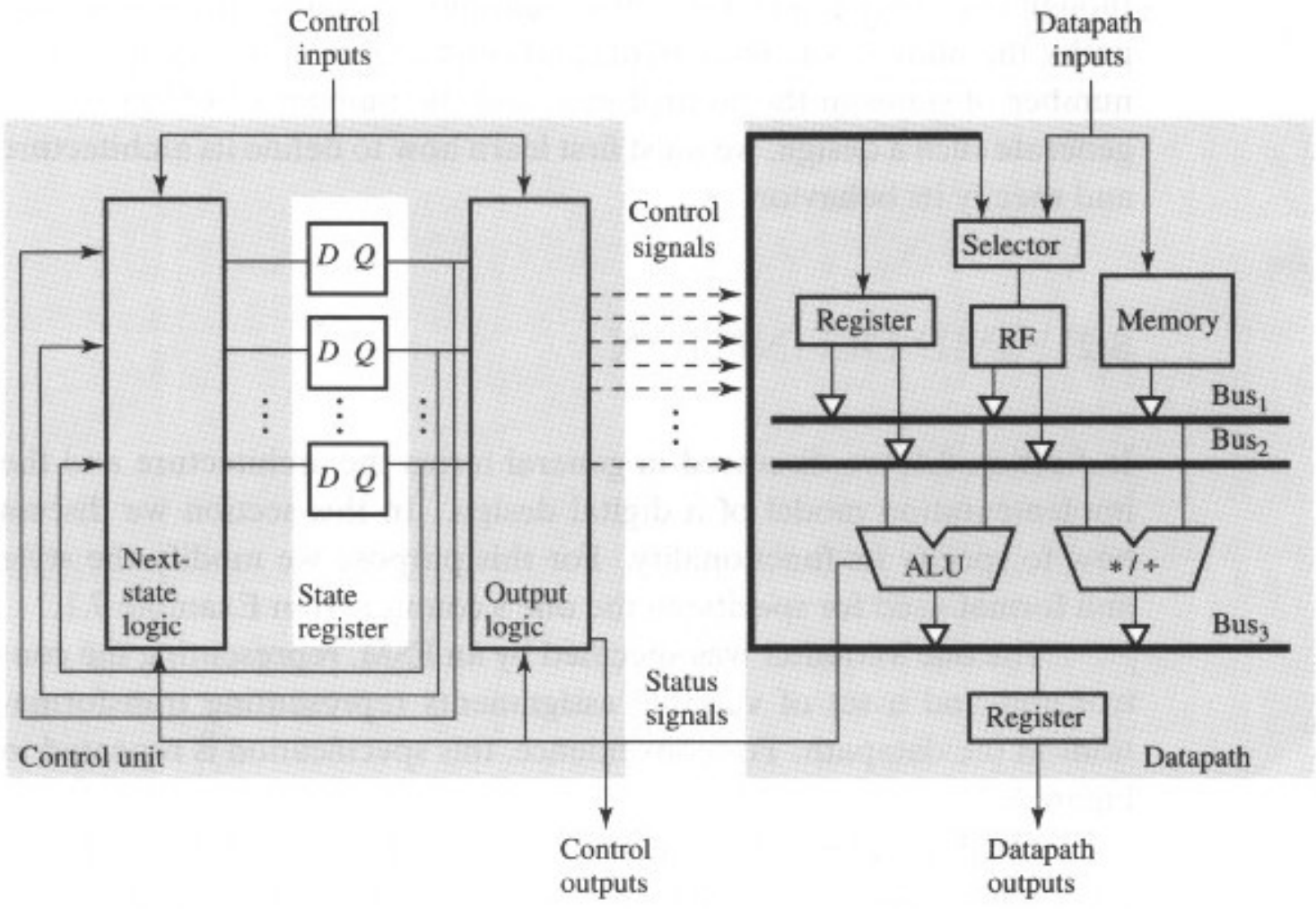
The Control Unit (CU):

- Generates control signals for data path operation.
  - Implemented as an FSM that may be designed in the conventional manner.
  - Receives status signals from data path (e.g. “operation complete”) and from outside world (e.g. “data ready”)
  - Generates two types of signals: opcodes to blocks of datapath, and status signals to the outside world

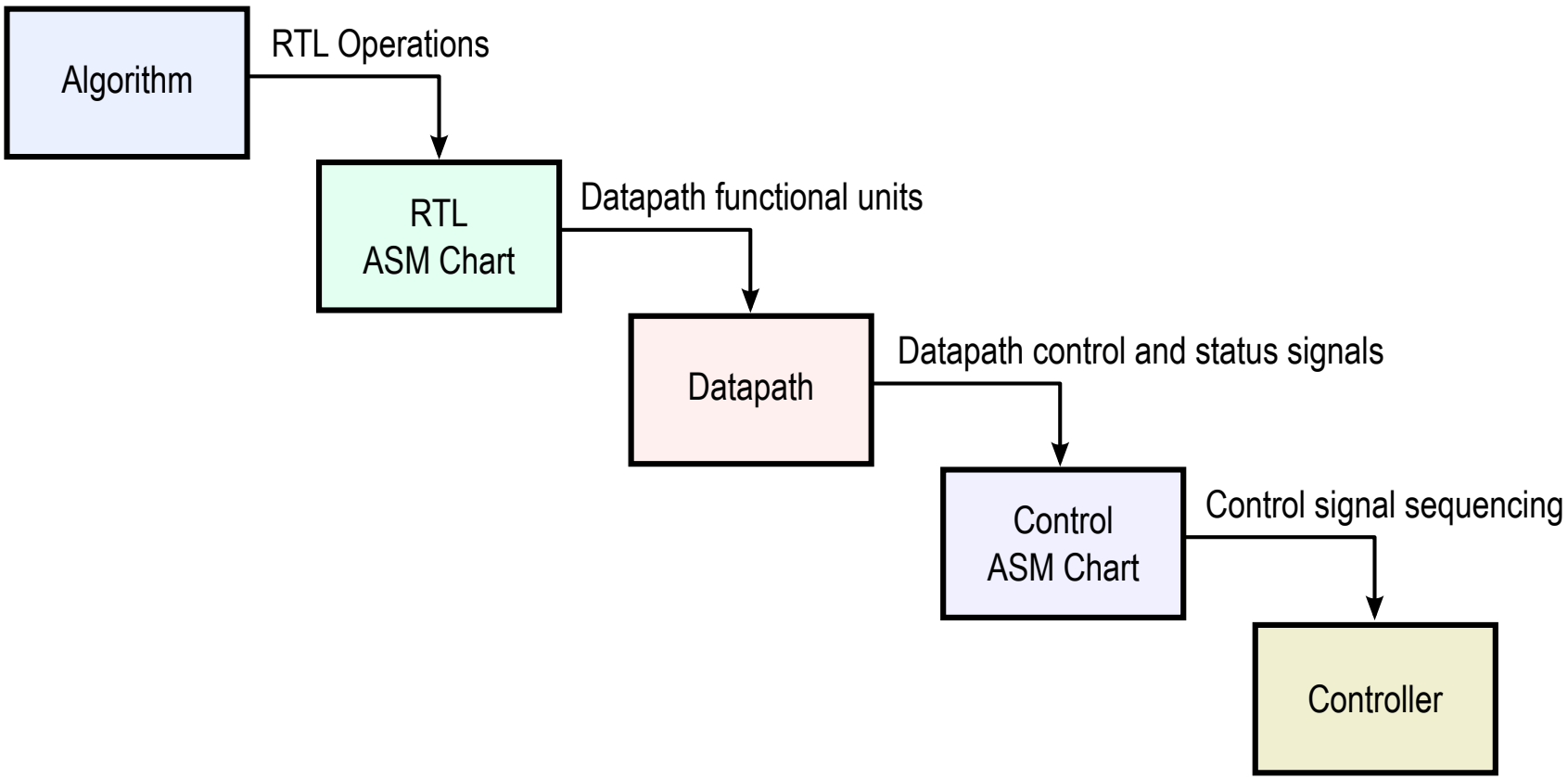
FSMD Modeling

- FSMD can be simple to very complex. Use the appropriate modeling tools.
  - State diagrams are usable for simple FSMDs.
  - Algorithmic State Machines (ASMs) almost like state diagram, but looks like flowcharts. Therefore, they can describe more complex decision making sequence.
  - Register Transfer Language (RTL) models how data is transferred from one place to another in a system.

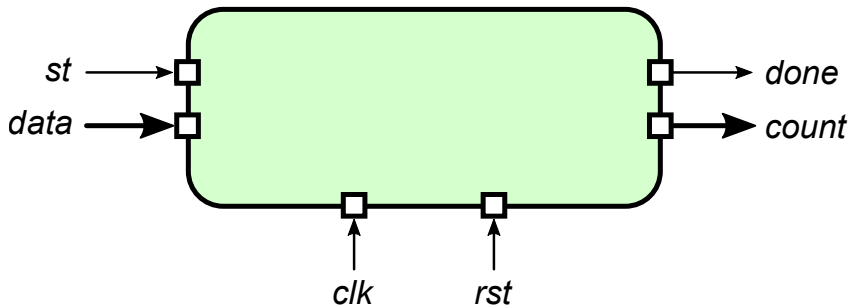
FSMD RTL View: Example of a Complex System



Designing a Complex Digital System

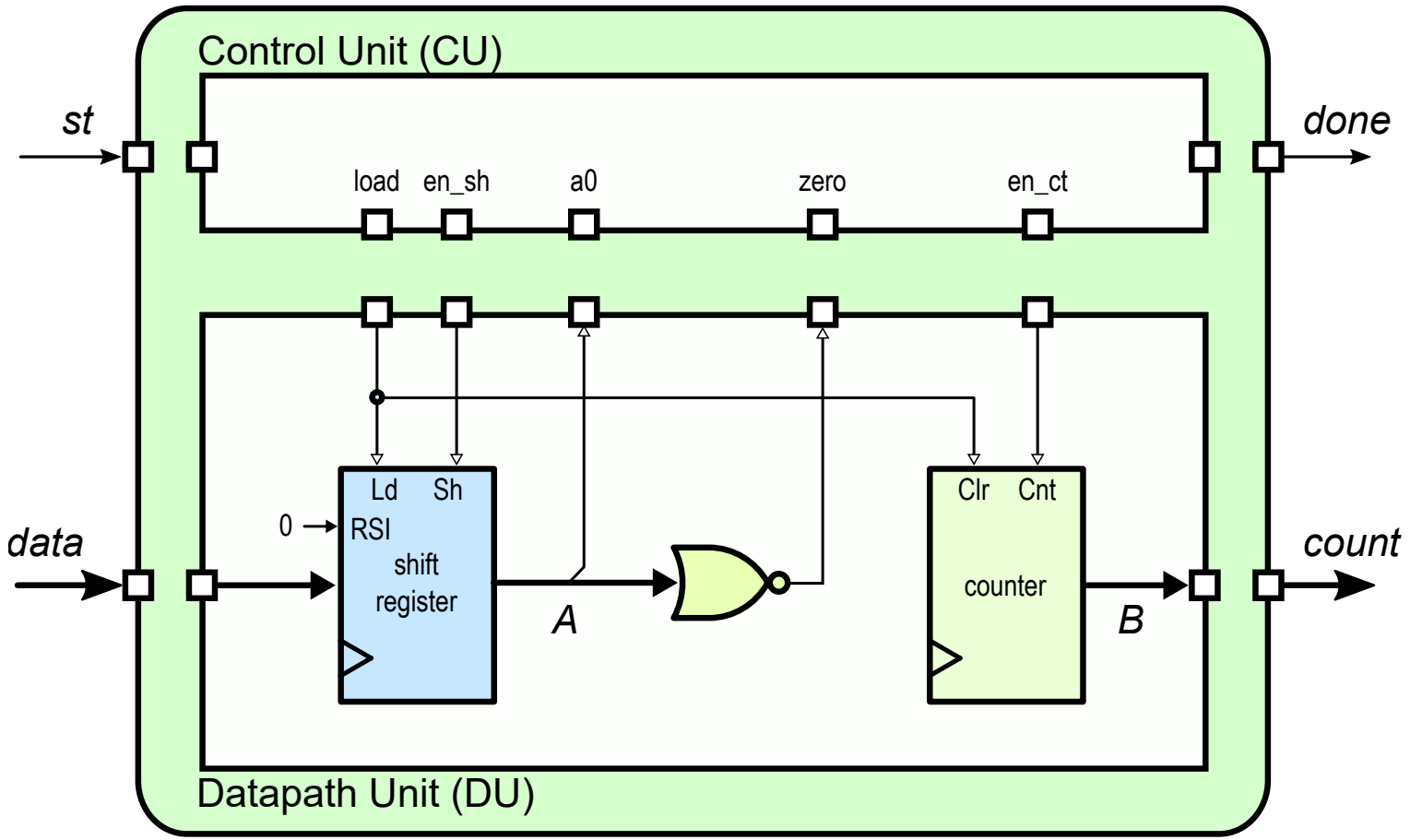


Ones Counter Entity and Algorithm

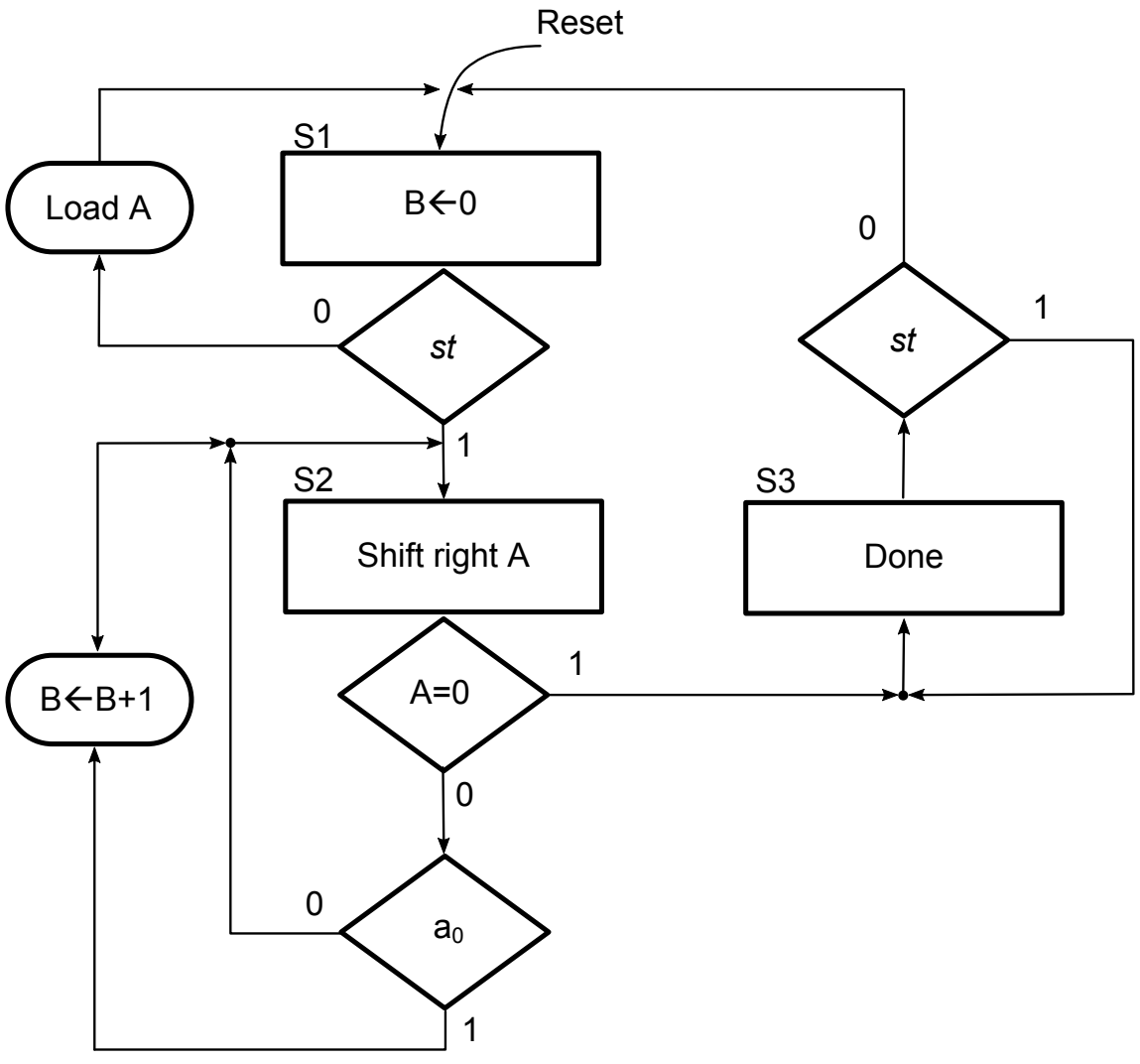



```
int onescount(unsigned data)
{
  unsigned count = 0;    // reset counter
  while( data != 0 ) {   // repeat as long as data has 1s
    if( data & 1 == 1 ) // if lsb = 1 then bump counter
      count++;
    data = data >> 1;    // shift data for next bit
  }
  return count;
}
```

Ones Counter Functional Block Diagram



High-Level ASM Chart

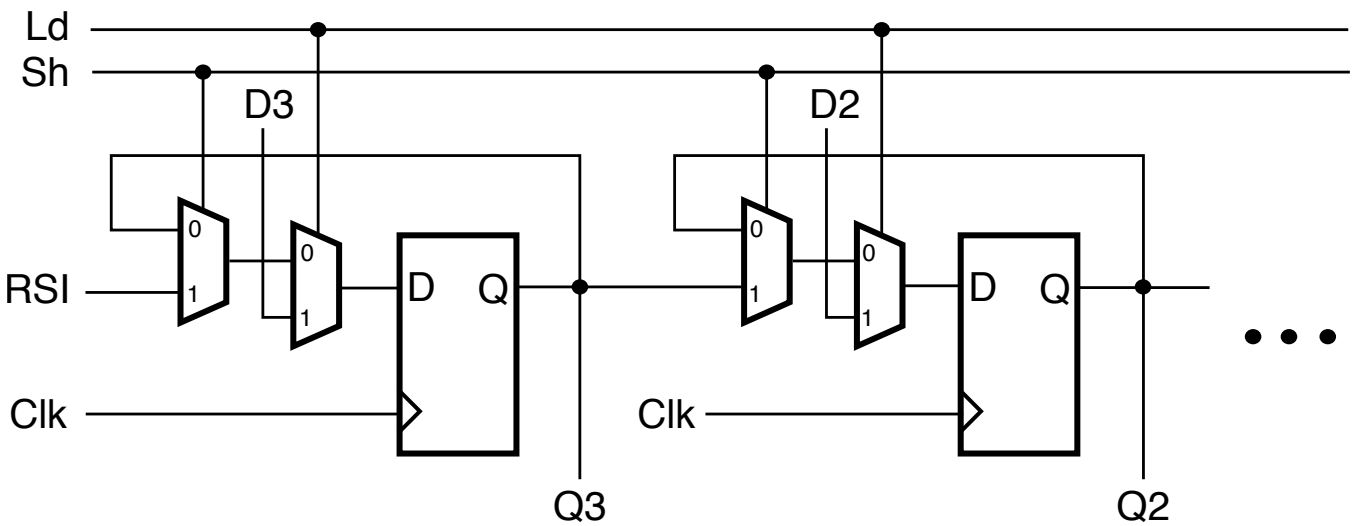


Building Blocks

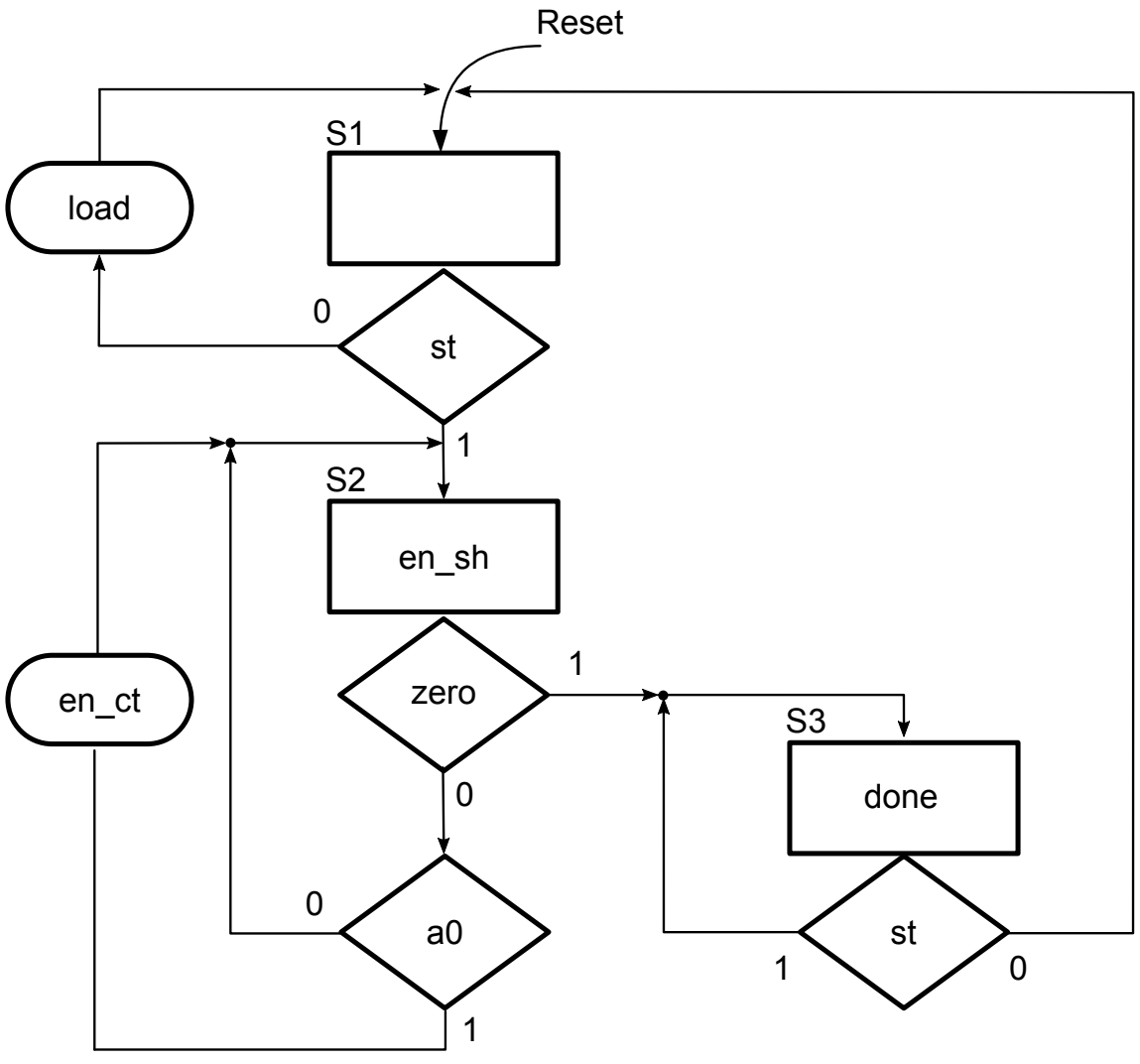
Some components must specially-designed.

- Up-counter with **clear** and (count) **enable** controls
- Shift register with **load** and (shift) **enable** controls

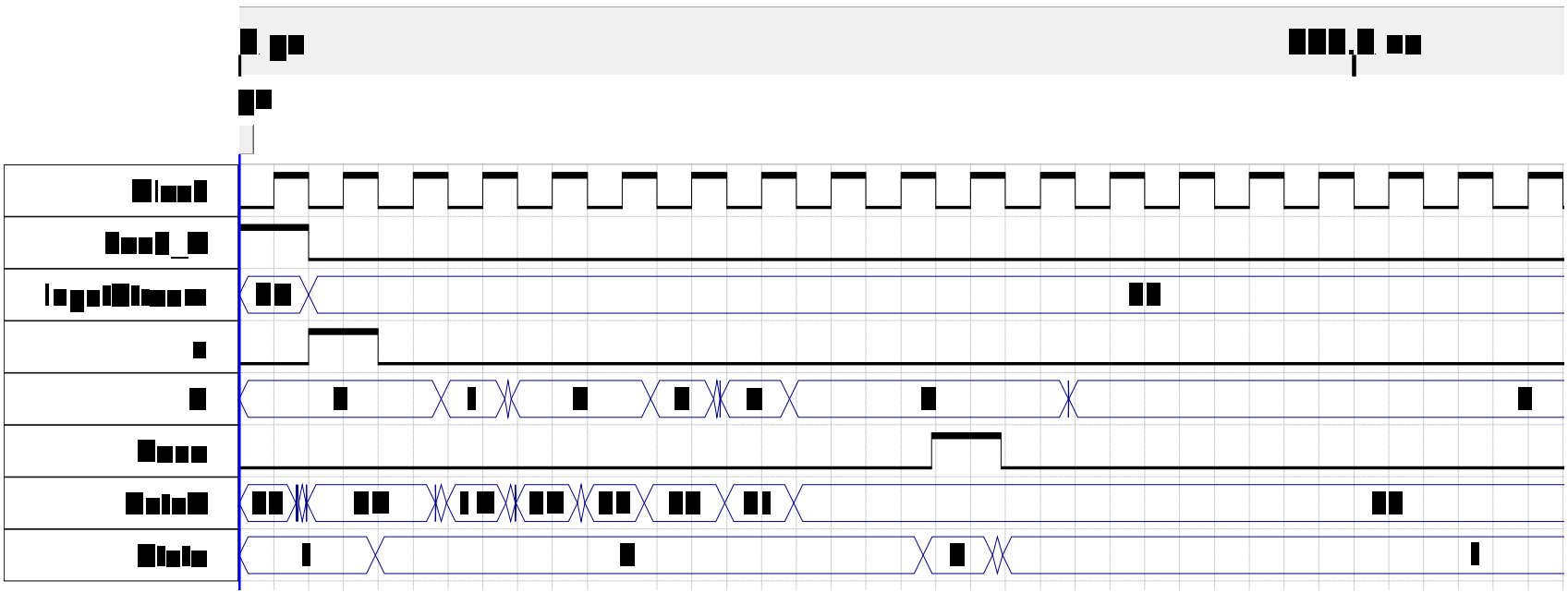
Test the function of each block before integrating into system.



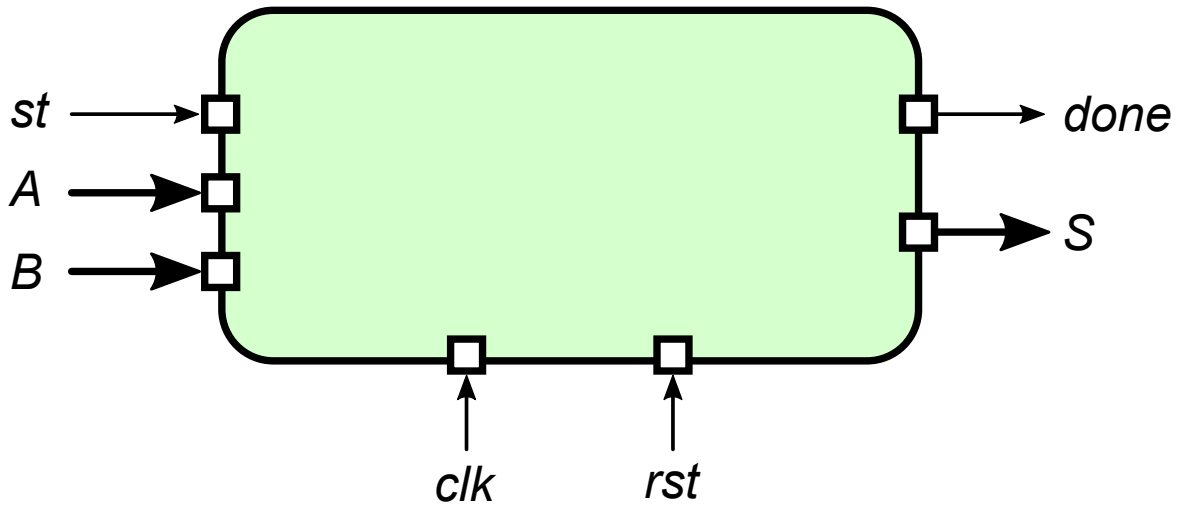
Low-Level ASM Chart



Ones Counter Simulation



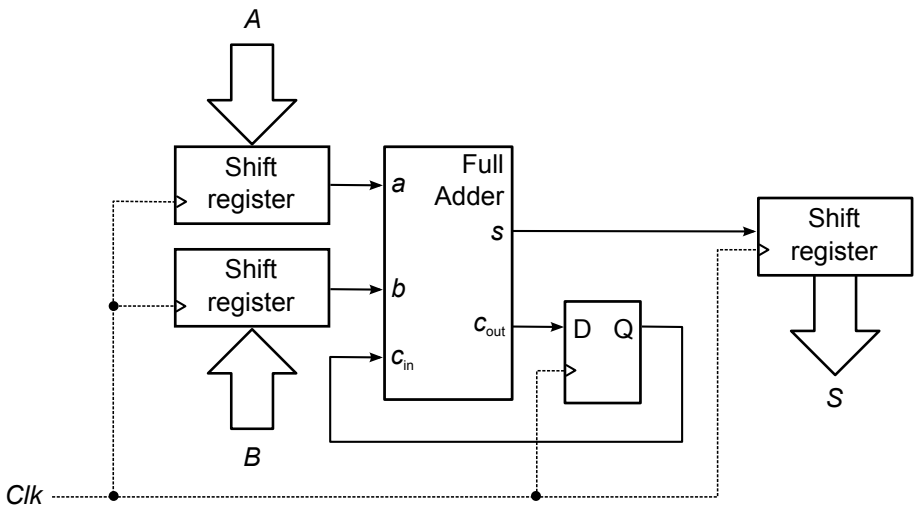
Serial Adder Entity



Serial Adder Specs

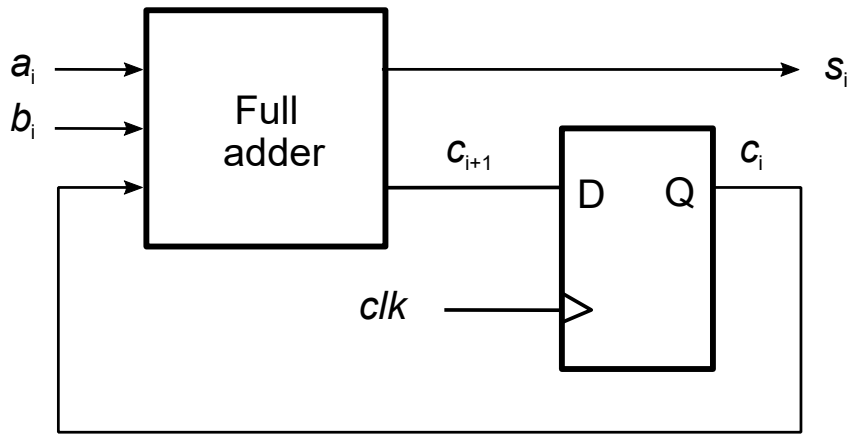
External Inputs	
Signal	**Function**
A	First n-bit Operand (Addend)
B	Second n-bit Operand (Augend)
st	Start signal which initiates the addition operation
rst	*Puts the controller into the initial state. This signal is used **only** on system power-up or system lock-up.*
clk	*Clock signal.*
External Outputs	
S	The n-bit sum of $S = A + B$
C_4	The carry from $S = A + B$
done	Active when the operation is complete

Serial Adder Basic Circuit



A	B	S	s_i	c_{out}
1011	0011	0000	0	1
0101	0001	1000	1	1
0010	0000	1100	1	0
0001	0000	1110	1	0

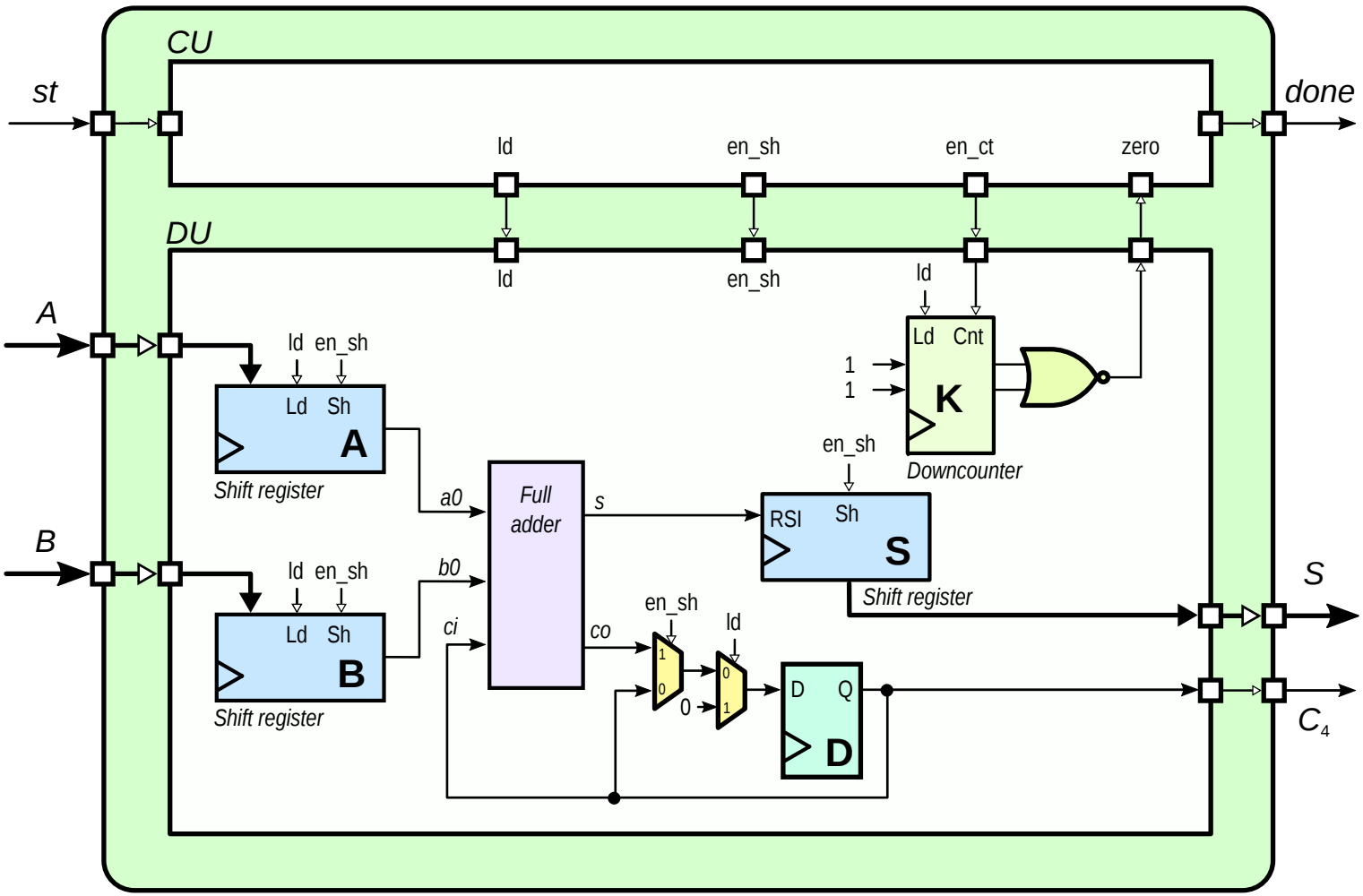
Serial Adder Basic Circuit



Must add extra circuits to:

- Clear carry flip-flop before adding first set of bits
- Stop clocking when all bits have added

Datapath


CU
st
done
ld
en_sh
en_ct
zero
DU
A
B
S
C_4
Ld Sh
A
Shift register
Ld Sh
B
Shift register
Full adder
a0
b0
ci
co
s
Ld Cnt
K
Downcounter
1
1
RSI Sh
S
Shift register
D Q
D
0

Signals to Control the Datapath

Signal	Function
ld	load the register/counter
en_sh	enable shifting of A, B, S & D
en_ct	enable counter K counting
zero	high when counter K = 0

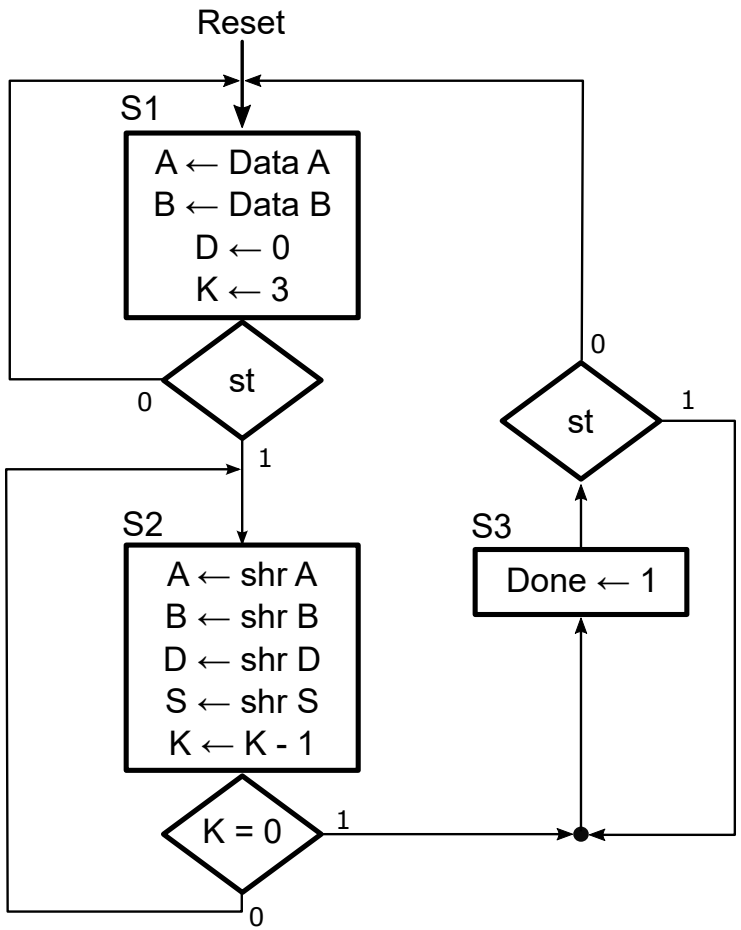
Datapath					**Control Signals**					
A	B	S	s_i	c_{out}	st	ld	en_sh	en_ct	zero	done
1011	0011	xxxx	0	1	0	1	0	0	0	0
1011	0011	xxxx	0	1	1	1	0	0	0	0
0101	0001	0xxx	1	1	1	0	1	1	0	0
0010	0000	10xx	1	0	1	0	1	1	0	0
0001	0000	110x	1	0	1	0	1	1	1	0
0000	0000	1110	0	0	1	0	0	0	0	1

Datapath Operation

- On reset, flip-flop D is cleared
- Idle state, waiting for *st* (start) signal: A, B registers and counter C are continously loaded. Flip-flop D remains cleared as it is not enabled.
- When start signal is received, repeat the following 4 times:
  - Shift registers A, B, S and flip-flop D
  - Decrement counter C
- After the bits have been shifted 4 times (signaled by Zero signal from downcounter), stop shifting and output the *done* signal. The 5-bit sum is now stable and can be used by other circuits.

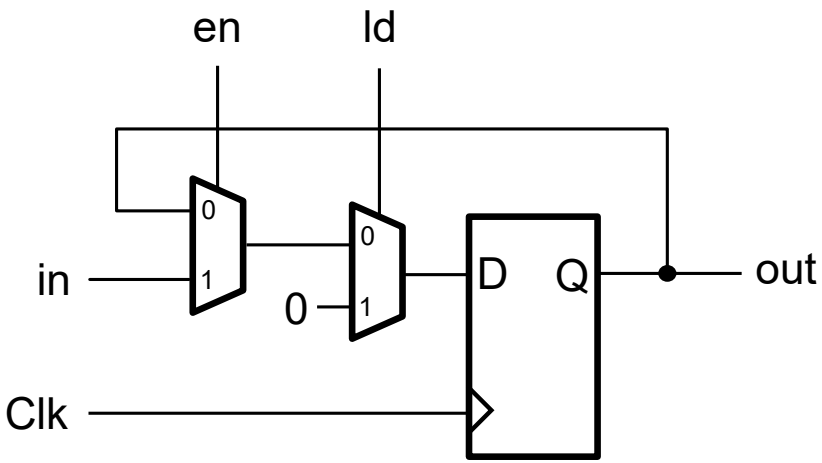
High-Level ASM Chart

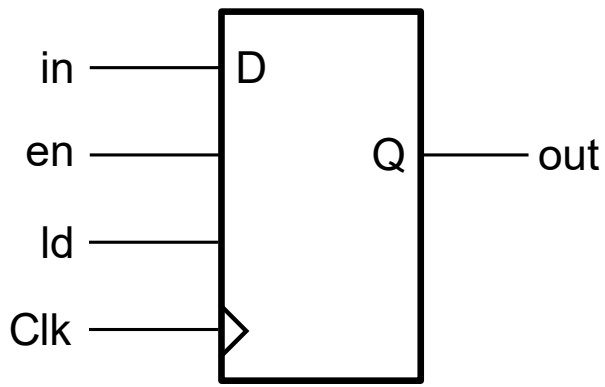
Describes what the circuit is doing



Building Blocks: DFF with enable & load

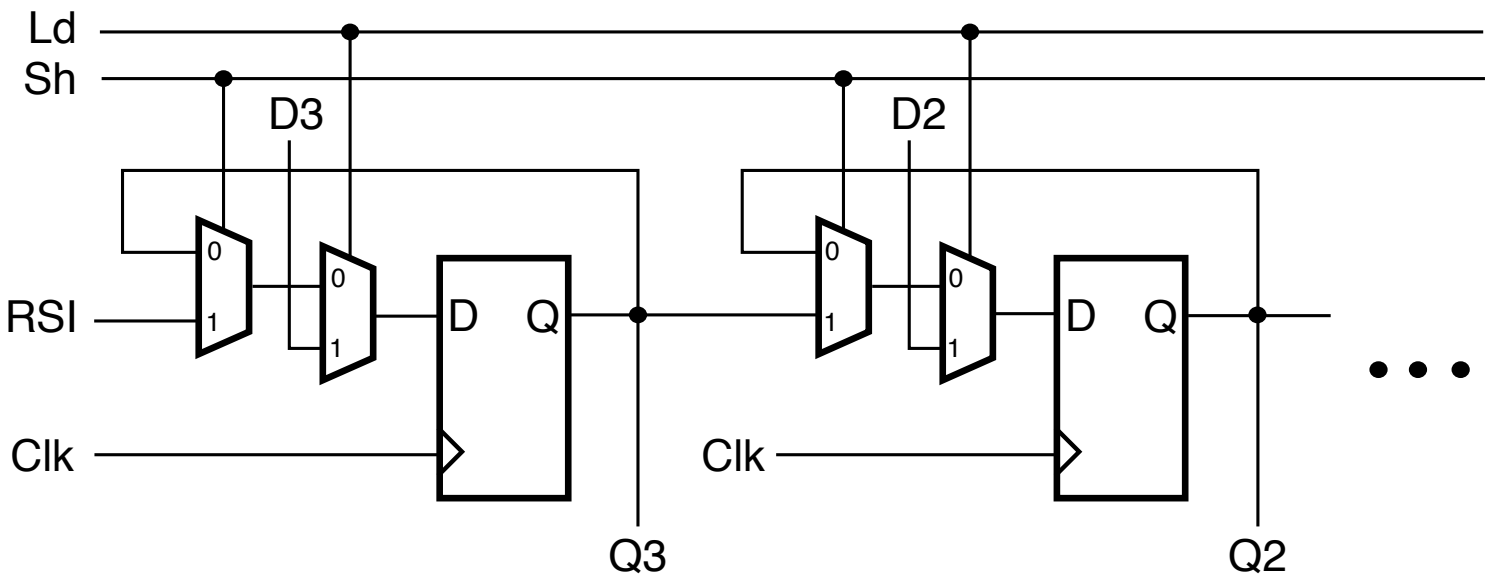
Special circuits designed at low level





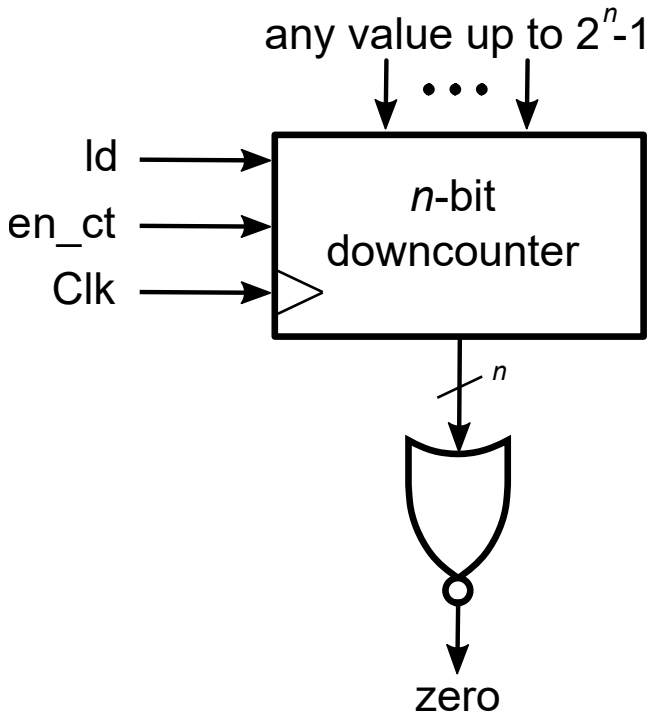
Building Blocks: SRG with Parallel Load

Special circuits designed at low level



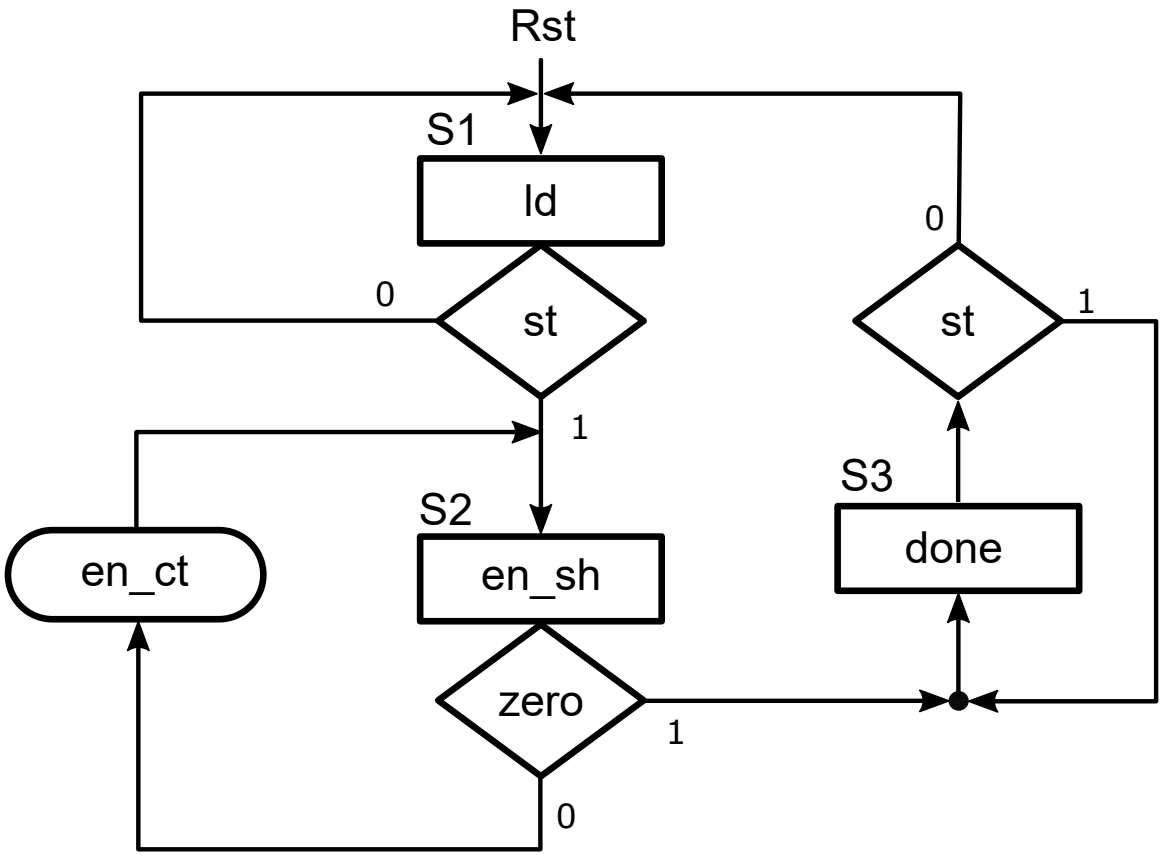
Building Blocks: Loop Control Downcounter

Special circuits designed at low level



Low-Level ASM Chart

Sequences the control signals



Serial Adder Simulation