Optimization of
Finite State Machines (II)

State Minimization

State minimization **is the process of eliminating redundant states**

For a complex FSM, the designers initial choice of states will often result in more states than are necessary

By eliminating states, we can reduce the number of flip-flops required and simplify the Next State and Output Logic

State Minimization Definitions

Equivalent States: Suppose that A and B are two states in a FSM. States A and B are equivalent if for every possible input sequence, the same output sequence will be produced regardless of whether A or B was the initial state

k-Successor: Suppose a FSM is in state X. If the input is k and that input results in a transition to state Y, then Y is said to be a “K-successor” of X

0-Successor: If the FSM is in state X and an the application of the input of k =0 causes the FSM to transition to state Y, then State Y is a 0-Sucessor of State X

1-Successor: If the FSM is in state X and an the application of the input of k =1 causes the FSM to transition to state Y, then State Y is a 1-Sucessor of State X

State Minimization Procedure

If State A and State B are equivalent, then their corresponding k-successors must also be equivalent

Rather than trying to show which states are equivalent, we eliminate redundant states by determining which states are *not* equivalent

We do this by breaking the collection of states into *partitions*

Partition: A partition consists of a grouping states. The states in a partition may be equivalent, but the states in one grouping are definitely not equivalent to the states contained in another partition

State Minimization Example

Present state	Next state		Output z
	$w = 0$	$w = 1$	
A	B	C	1
B	D	F	1
C	F	E	0
D	B	G	1
E	F	C	0
F	E	D	0
G	F	G	0

Step 1: Create an initial partition that contains all of the states in a single grouping

$$P_1 = (A\ B\ C\ D\ E\ F\ G)$$

State Minimization Example

Present state	Next state		Output
	$w = 0$	$w = 1$	z
A	B	C	1
B	D	F	1
C	F	E	0
D	B	G	1
E	F	C	0
F	E	D	0
G	F	G	0

Step 2: Separate states that produces different outputs

$$P_1 = (A\ B\ C\ D\ E\ F\ G)$$

$$P_2 = (A\ B\ D)\ (C\ \ E\ F\ G)$$

***Note that its not possible for any state from the first group is equivalent to any state in the second group**

State Minimization Example

Present state	Next state		Output z
	$w = 0$	$w = 1$	
A	B	C	1
B	D	F	1
C	F	E	0
D	B	G	1
E	F	C	0
F	E	D	0
G	F	G	0

Step 3: Determine the 0,1 successors for each grouping

$$P_1 = (A\ B\ C\ D\ E\ F\ G)$$

$$P_2 = (A\ B\ D)\ \ (C\ \ E\ F\ G)$$

State Minimization Example

Present state	Next state		Output z
	$w = 0$	$w = 1$	
A	B	C	1
B	D	F	1
C	F	E	0
D	B	G	1
E	F	C	0
F	E	D	0
G	F	G	0

Step 4: Determine equivalent states

$P_1 = (A\ B\ C\ D\ E\ F\ G)$

$P_2 = (A\ B\ D)\ \ (C\ \ E\ F\ G)$

0 ↙ ↘ 1

(BDB) (C F G)

$P_3 =$

For the grouping (ABD), all of the 0-sucessor states (BDB) are also contained in a singular partition grouping within P_2

For the grouping (ABD), all of the 1 –successor states (CFG) are also contained in a singular partition grouping within P_2

Since all of the k-successors for the group (ABD) fall in a common group for each k, then we can assume that states A, B and D are equivalent, therefore states A, B and D should remained grouped together in the next partition P_3

State Minimization Example

Present state	Next state $w = 0$	Next state $w = 1$	Output z
A	B	C	1
B	D	F	1
C	F	E	0
D	B	G	1
E	F	C	0
F	E	D	0
G	F	G	0

Step 4: Determine equivalent states

P_1 = (ABCDEFG)

P_2 = (ABD) (CEFG)

0 → (FFEF)   1 → (ECDG)

P_3 =

For the grouping (CEFG), all of the 0-sucessor states (FFEF) are also contained in a single partition grouping within P_2

For the grouping (CEFG), all of the 1 –successor states (ECDG) are **NOT** contained in a single partition grouping within $P_{2.}$ This means that at least one of the states in the grouping (CEFG) is **NOT** equivalent to the others

State F must be different from States C, E and G because its 1 sucessor is D, which is in a different block than C, E, and G

State F MUST be **<u>unique</u>**, since it is in a block by itself

State Minimization Example

Present state	Next state		Output z
	$w = 0$	$w = 1$	
A	B	C	1
B	D	F	1
C	F	E	0
D	B	G	1
E	F	C	0
F	E	D	0
G	F	G	0

Step 4: Repeat partitioning process

$$P_3 = (ABD)(CEG)(F)$$

0 ↙ ↘ 1

(BDB) (C F G)

$$P_4 =$$

For the grouping (ABD), all of the 0-sucessor states (BDB) are also contained in a single partition grouping within P_3

For the grouping (ABD), all of the 1 –successor states (CFG) are **NOT** contained in a single partition grouping within $P_{3.}$ This means that at least one of the states in the grouping (ABD) is **NOT** equivalent to the others

State B must be different from States A, D as State Bs 1 sucessor is F, is in a different block than C and G

State B MUST be **<u>unique</u>**, since it is in a block by itself

State Minimization Example

Present state	Next state		Output
	$w = 0$	$w = 1$	z
A	B	C	1
B	D	F	1
C	F	E	0
D	B	G	1
E	F	C	0
F	E	D	0
G	F	G	0

Step 4: Repeat partitioning process

$$P_3 = (ABD)(CEG)(F)$$

0 ↙ ↘ 1

(FFF) (ECG)

$$P_4 =$$

The 0 and 1 successors of (CEG), (FFF) and (ECG) can both be found in single groupings within P3, therefore states C, E and G can still be assumed to be equivalent and can remain intact in the next partition P_4

State Minimization Example

Present state	Next state		Output
	$w = 0$	$w = 1$	z
A	B	C	1
B	D	F	1
C	F	E	0
D	B	G	1
E	F	C	0
F	E	D	0
G	F	G	0

Step 4: Repeat partitioning process

$$P_4 = (AD)(B)(CEG)(F)$$

$$P_5 =$$

The 0 and 1 successors of (AD), (BB) and (CG) can both be found in single groupings within P_4, therefore states AD can still be assumed to be equivalent and can remain intact in the next partition P_5

The 0 and 1 successors of (CEG), (FFF) and (ECG) can both be found in single groupings within P_4, therefore states CEG can still be assumed to be equivalent and can remain intact in the next partition P_5

Since $P_4 = P_{5,}$ and no new groupings can be formed, the states in each remaining grouping are equivalent

State Minimization Example

Present state	Next state		Output
	$w = 0$	$w = 1$	z
A	B	C	1
B	D	F	1
C	F	E	0
D	B	G	1
E	F	C	0
F	E	D	0
G	F	G	0

Original State Table

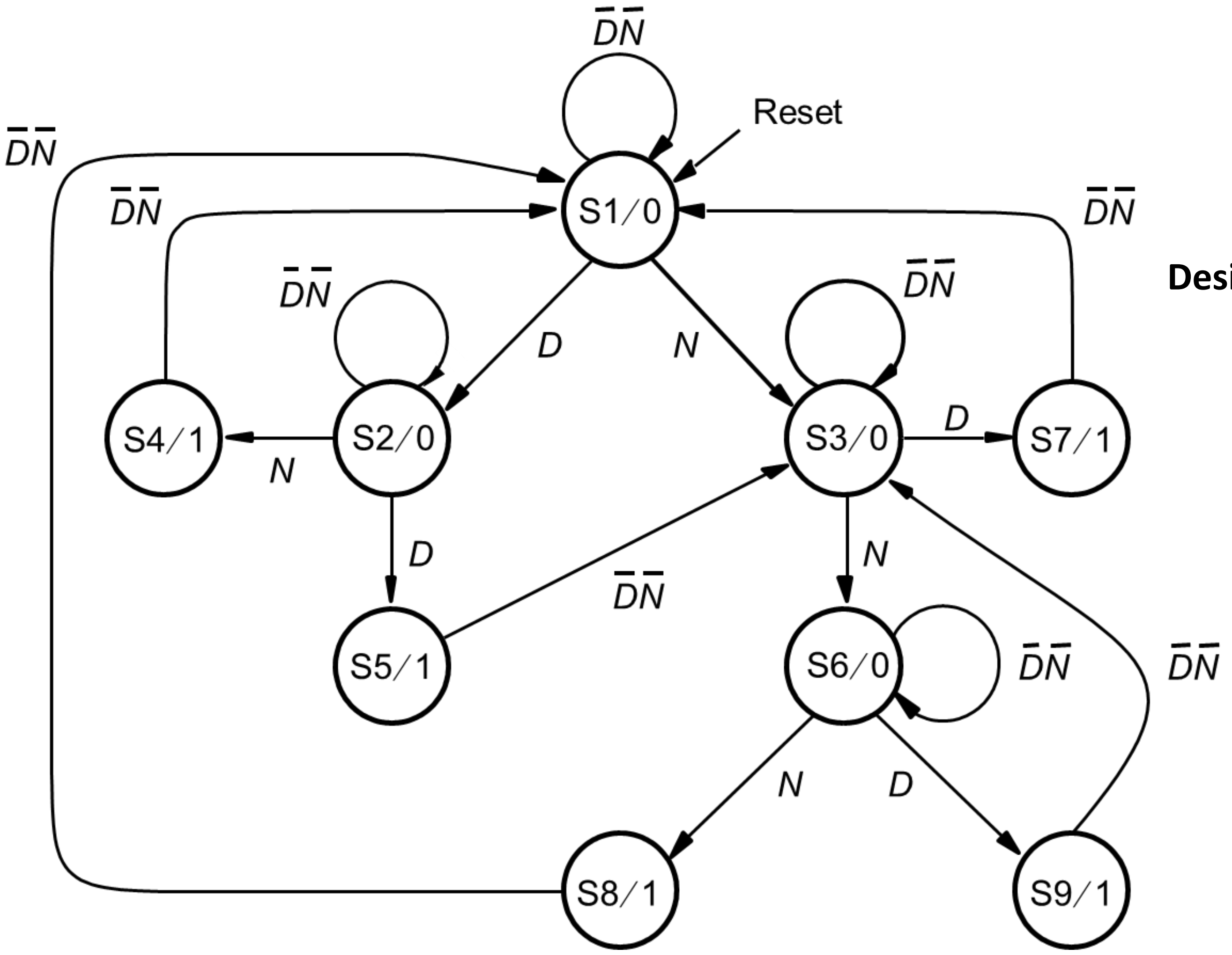
Present state	Nextstate		Output
	w = 0	w = 1	z

Minimized State Table

$$P_5 = (AD)(B)(CEG)(F)$$

The state transition Table can be re-written, using just one of the equivalent states to represent each grouping

State Minimization Example



FSM Controller for coin operated vending machine

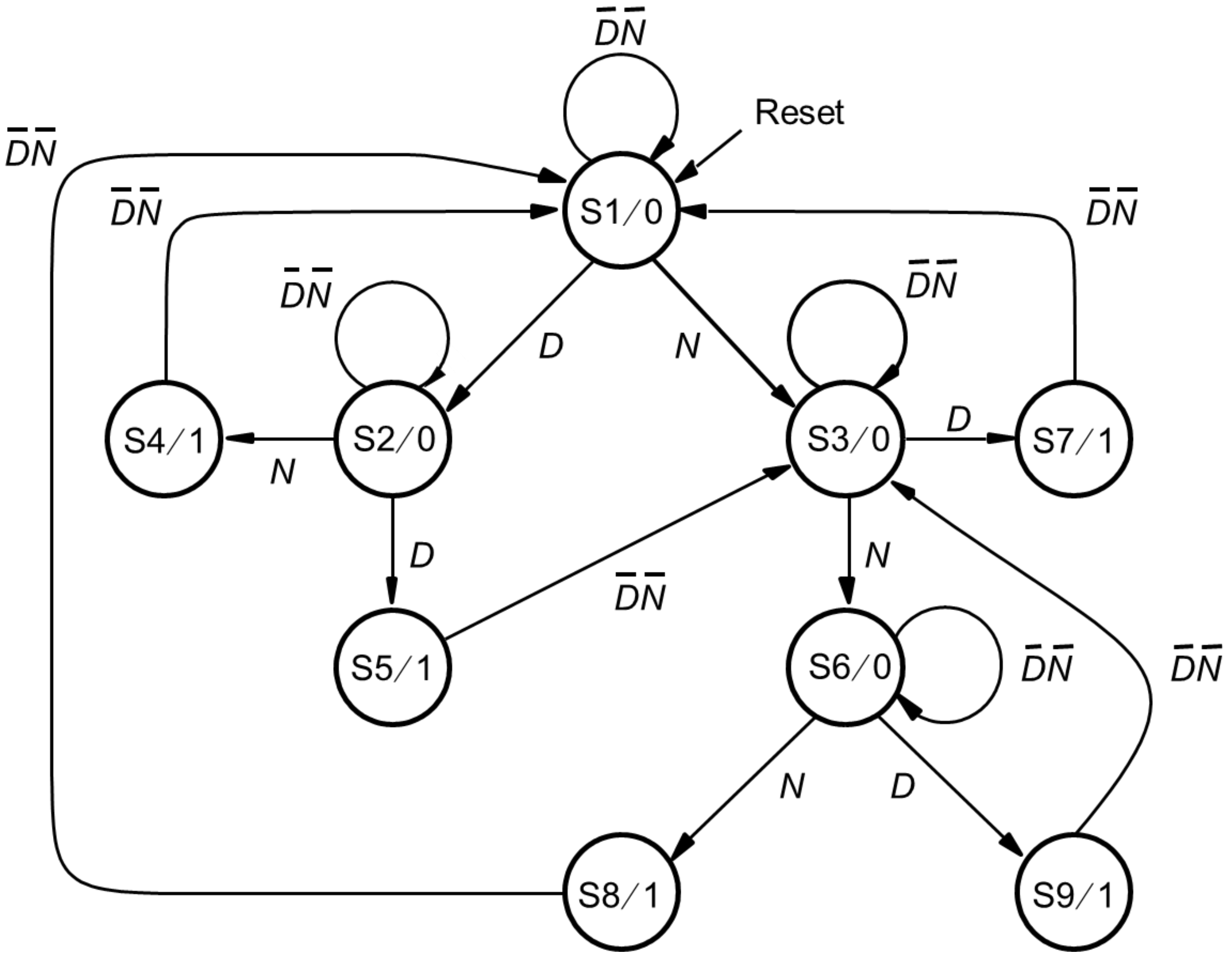
Design a sequential circuit to control coin-operated vending machine:

The machine accepts nickels and dimes (D, N)

It takes 15 cents for a piece of candy to be released from the machine (Output = 1)

If 20 cents is deposited, the machine will not return the change, but will credit the buyer with 5 cents and wait for the buyer to make a second purchase

State Minimization Example



FSM Controller for coin operated vending machine

Present state	Next state DN =00	01	10	11	Output z
S1	S1	S3	S2	–	0
S2	S2	S4	S5	–	0
S3	S3	S6	S7	–	0
S4	S1	–	–	–	1
S5	S3	–	–	–	1
S6	S6	S8	S9	–	0
S7	S1	–	–	–	1
S8	S1	–	–	–	1
S9	S3	–	–	–	1

State Transition Table with Don't Cares Included (-)

Note: Don't cares included in states S4,S5,S7,S8,S9 because there is no need to check for D and N because the machine is to another state in an amount of time that is too short for a new coin to have been inserted

State Minimization Example

Present state	Next state				Output z
	DN =00	01	10	11	
S1	S1	S3	S2	–	0
S2	S2	S4	S5	–	0
S3	S3	S6	S7	–	0
S4	S1	–	–	–	1
S5	S3	–	–	–	1
S6	S6	S8	S9	–	0
S7	S1	–	–	–	1
S8	S1	–	–	–	1
S9	S3	–	–	–	1

Original State Table

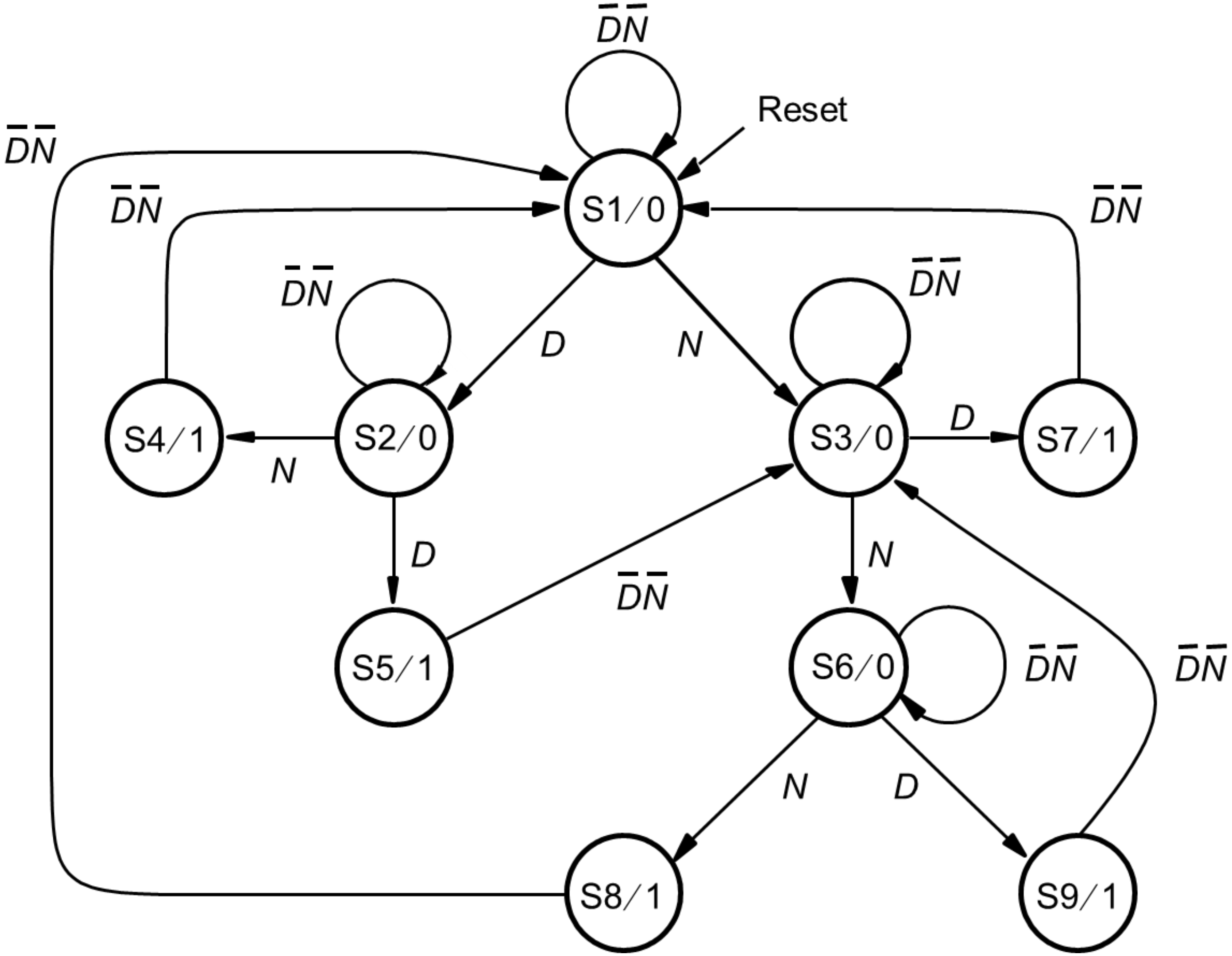
Present state	Next state				Output z
	DN =00	01	10	11	
S1	S1	S3	S2	–	0
S2	S2	S4	S5	–	0
S3	S3	S2	S4	–	0
S4	S1	–	–	–	1
S5	S3	–	–	–	1

Minimized State Table

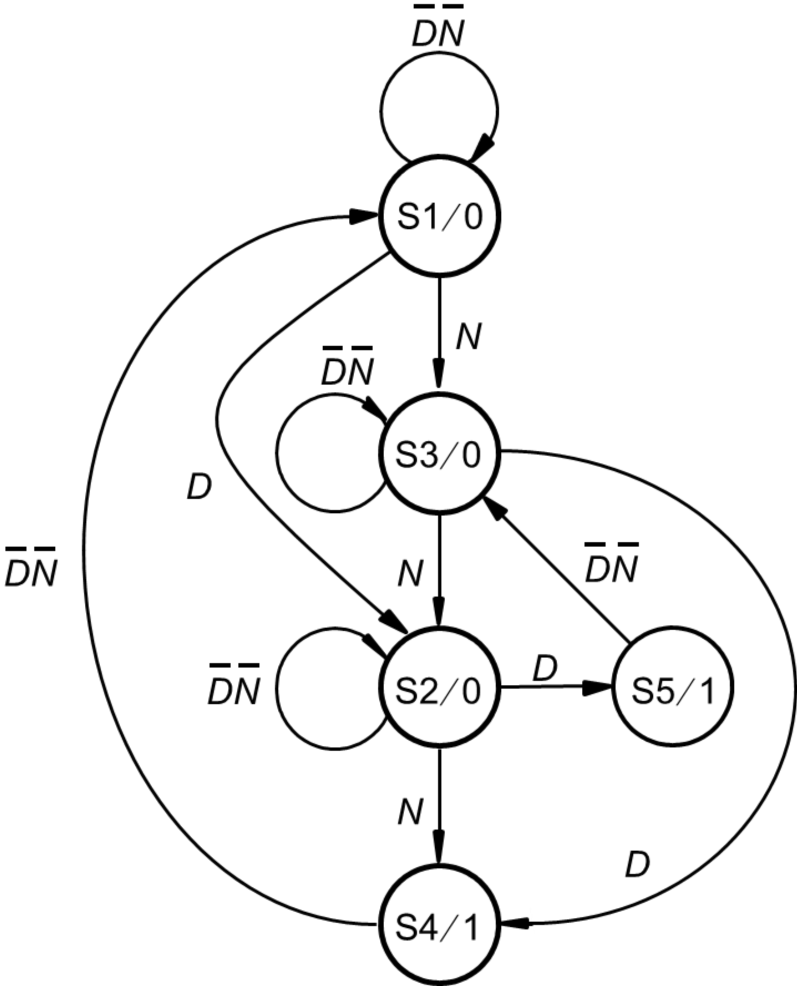
$$P_5 = (S1)(S3)(S2,S6)(S4,S7,S8)(S5,S9)$$

Extra Credit Homework Problem: Use the minimization procedure to show how to get from the original state table to P5 and the minimized state table. Extra credit is to be turned separately from HW 9 and is due by the evening of Friday April 24th. Score on the extra credit will be used to replace your worst quiz score. You MUST show all work to receive full credit and explain how you decided which states were redundant.

State Minimization Example



Original State Diagram



Minimized State Diagram

Course Wrap up: What Did You Learn?

Binary Numbers

Logic Gates

Boolean Algebra

Synthesis of Logic Circuits

Implementation Technology

Digital Design CAD Tools

Optimization of Logic Functions

Combinational Building Blocks

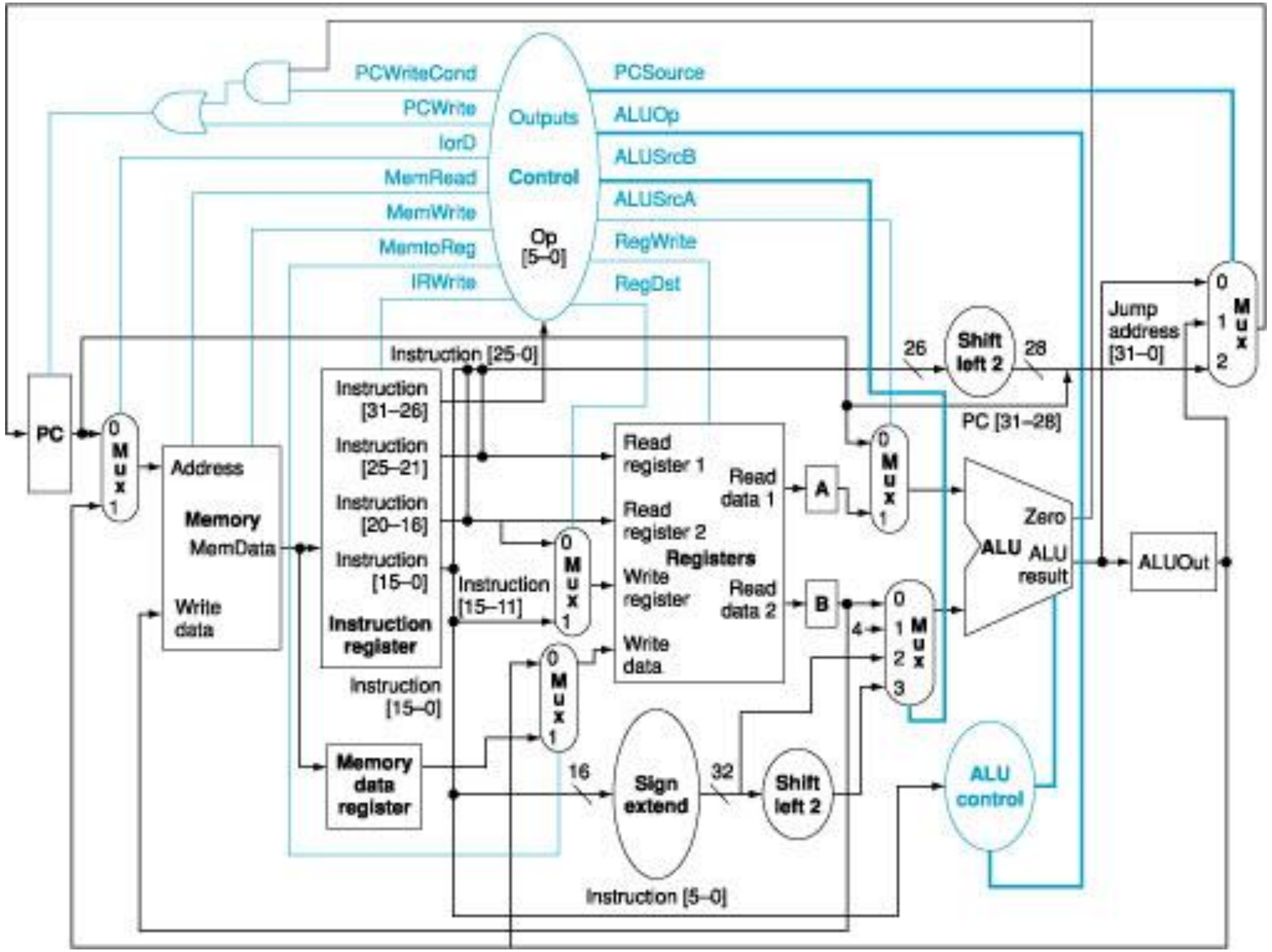
Arithmetic Circuits

Sequential Logic Circuits

Finite State Machines

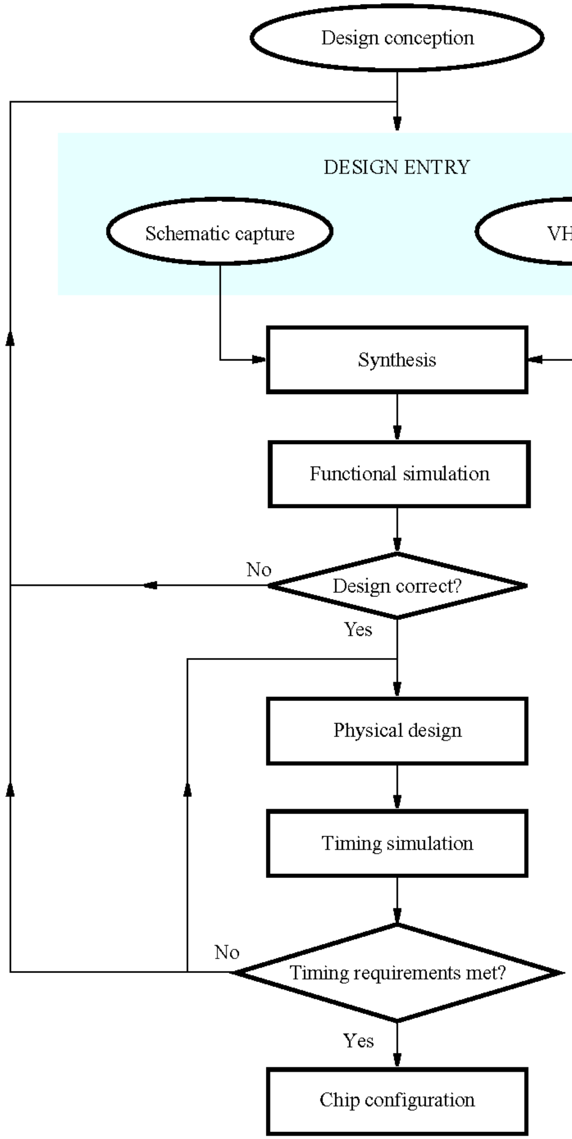
Optimization of FSMs

Course Wrap up: What Can You Do With What You Learned?



MIPS 32 bit Multicycle CPU Datapath

Course Wrap up: What Will You Learn Next?



Digital Design Flow