

CSE 140 Lecture 12

Standard Combinational Modules

CK Cheng
CSE Dept.
UC San Diego

1

Part III. Standard Modules

Interconnect Modules:

1. Decoder, 2. Encoder
- 3. Multiplexer, 4. Demultiplexer**

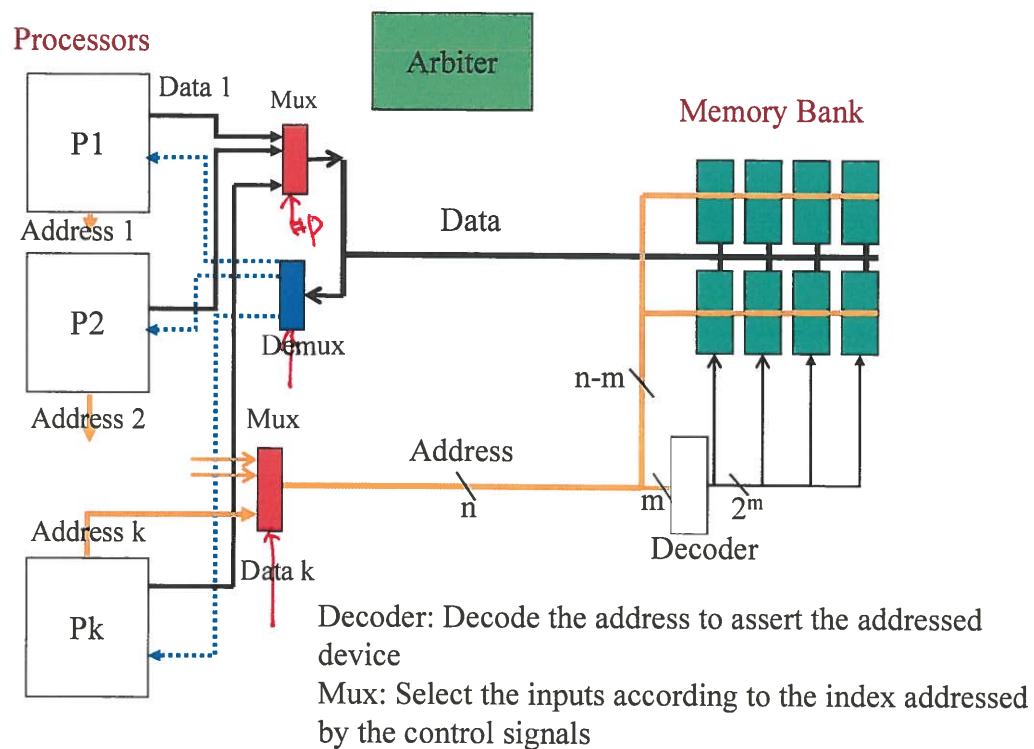
2

Multiplexer

- Definition
- Logic Diagram
- Application

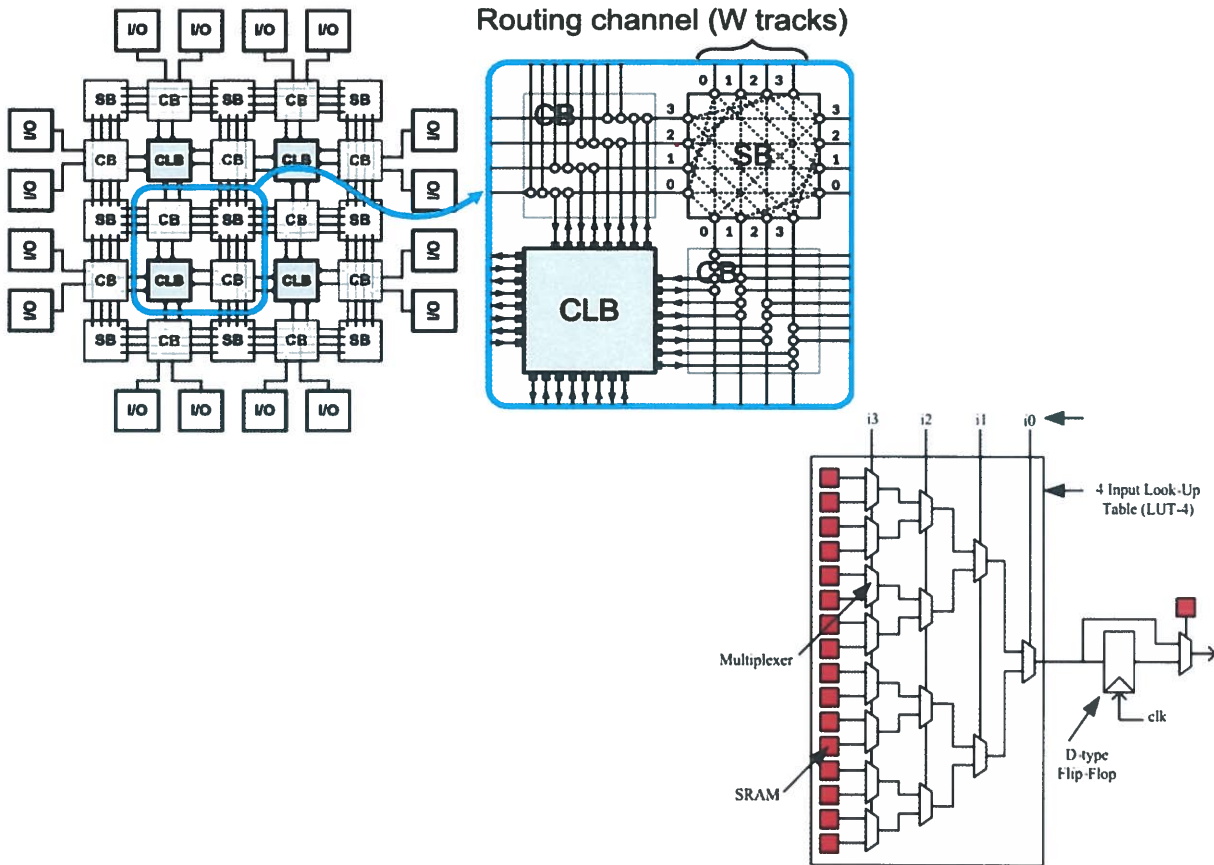
3

Interconnect: Decoder, Encoder, Mux, DeMux



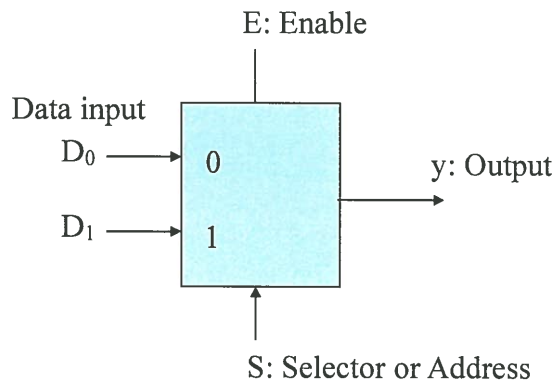
4

FPGA: Field Programmable Gate Array (Intel, AMD)



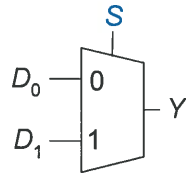
Multiplexer (Mux): Definition

- Selects between one of N inputs to connect to the output.
- $\log_2 N$ -bit select input – control input



Multiplexer (Mux): Definition

- Selects between one of N inputs to connect to the output.
- $\log_2 N$ -bit select input – control input
- Example: **2:1 Mux**



S	D_1	D_0	Y	S	Y
0	0	0	0	0	D_0
0	0	1	1	1	D_1
0	1	0	0		
0	1	1	1		
1	0	0	0		
1	0	1	0		
1	1	0	1		
1	1	1	1		

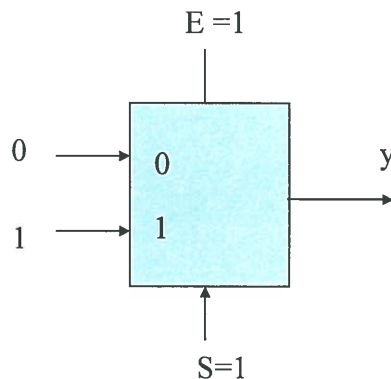
9

What is the output of the following MUX?

A.0

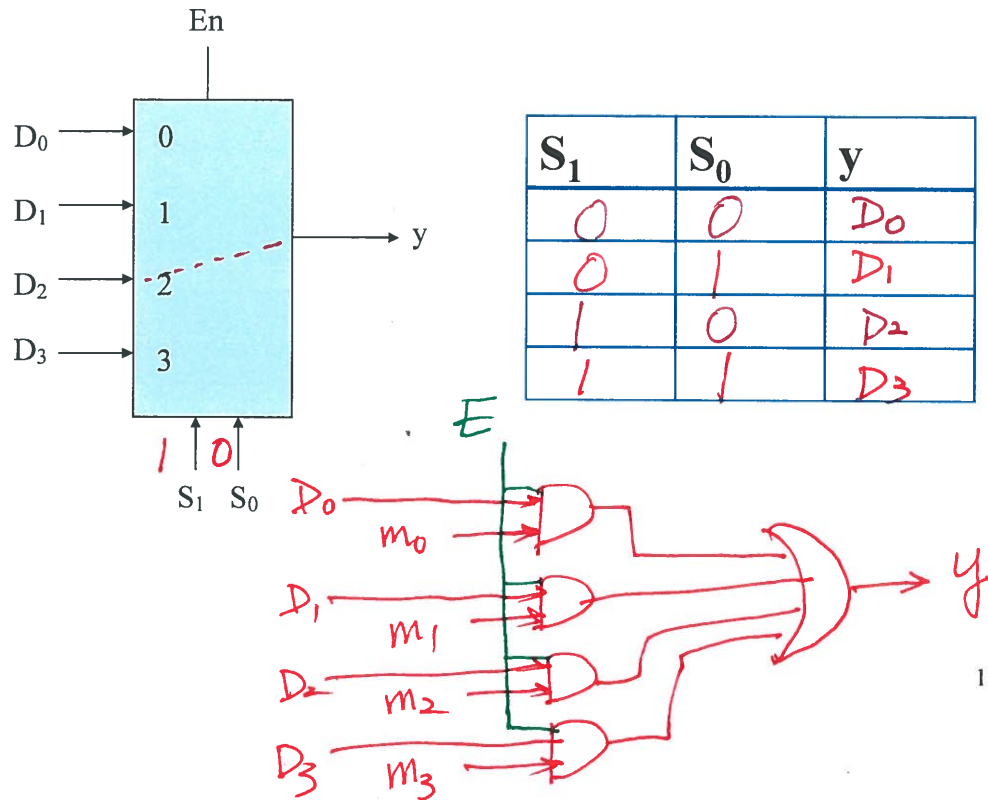
B.1

C.Can't say

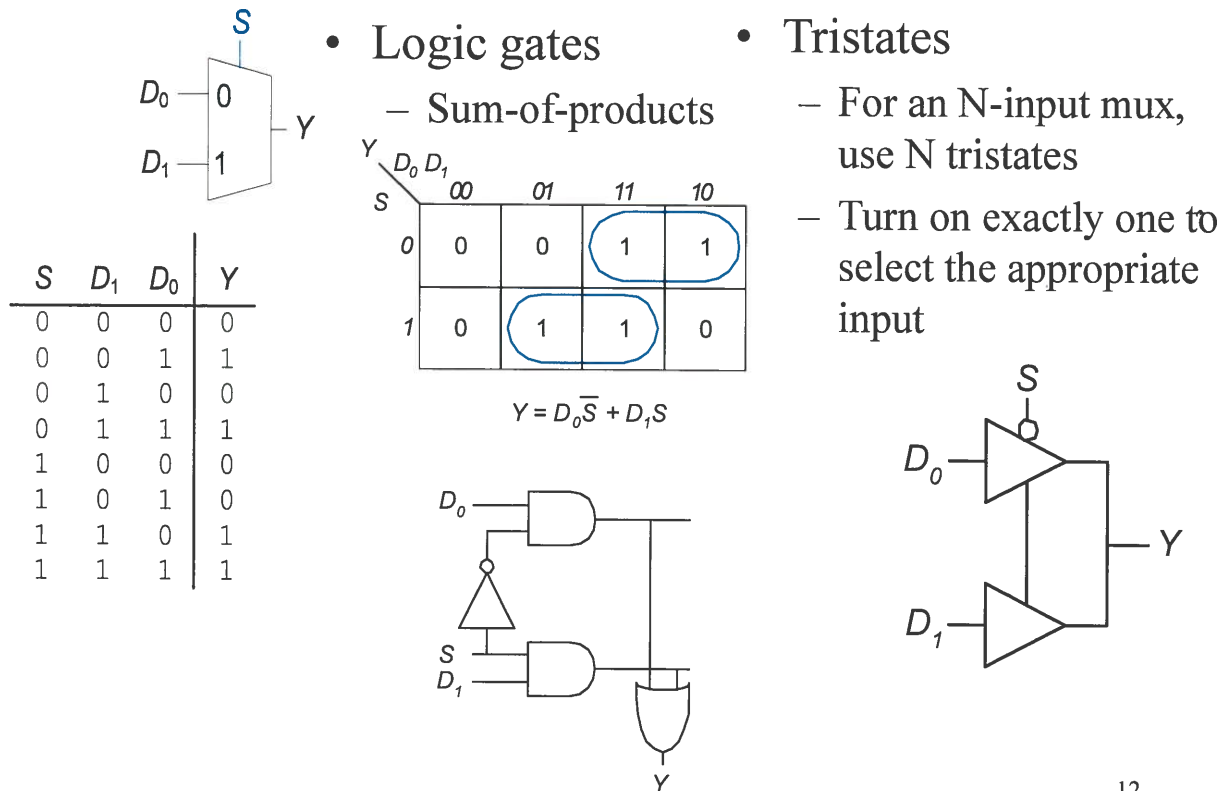


10

Multiplexer Definition: 4-input mux



Multiplexer: Logic Diagram



Multiplexer Application: universal set {Mux}

We use selector to decompose the function into smaller functions (less number of variables), which follows

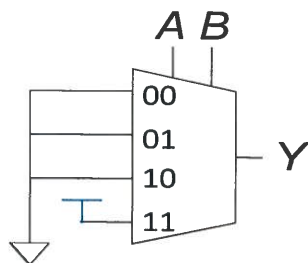
Shannon's expansion.

We simplify the decomposed functions using **K-map**, which follows **consensus theorem**.

13

Multiplexer Application

- Mux for a Boolean function with truth table as input
- Building blocks of FPGA (Field Programmable Gate Array).



For the logic diagram on left, output Y is

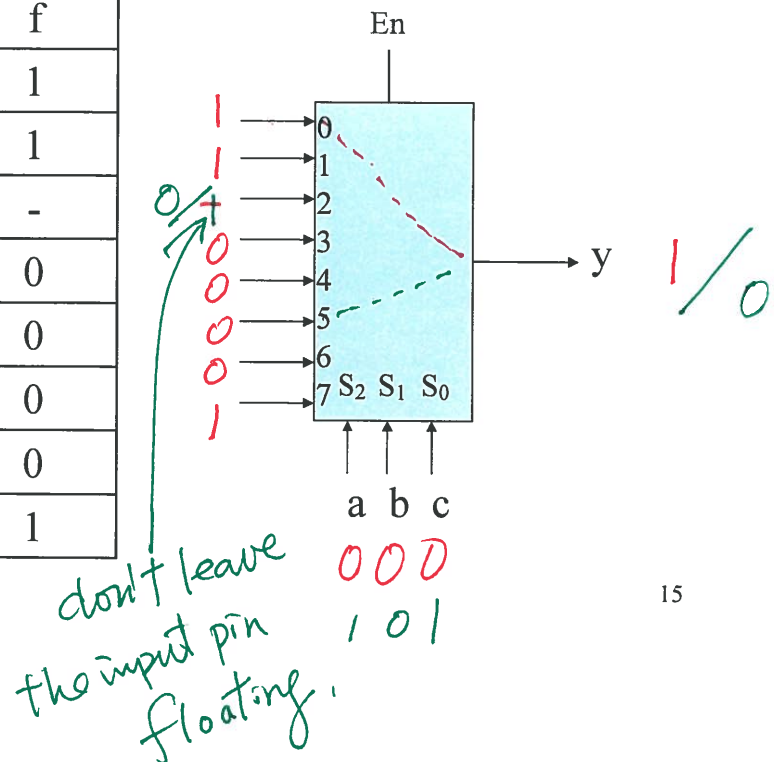
- A. AB
- B. $(AB)'$
- C. $A+B$
- D. $(A+B)'$
- E. None of the above

14

Multiplexer Application: universal set {Mux}

Example 1: Given $f(a,b,c) = \sum m(0,1,7) + \sum d(2)$, implement with an 8-input Mux.

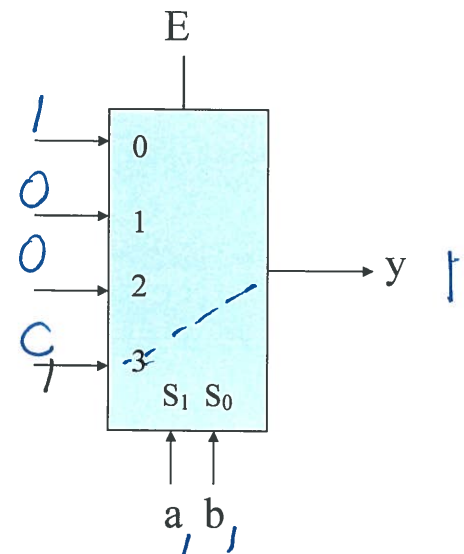
id	abc	f
0	000	1
1	001	1
2	010	-
3	011	0
4	100	0
5	101	0
6	110	0
7	111	1



15

Example 2: Given $f(a,b,c) = \sum m(0,1,7) + \sum d(2)$, implement with 4-input Muxes.

ab	c=0	c=1	$D(c)$
00	1	1	$D_0 = 1$
01	0	0	$D_1 = 0$
10	0	0	$D_2 = 0$
11	0	1	$D_3 = C$

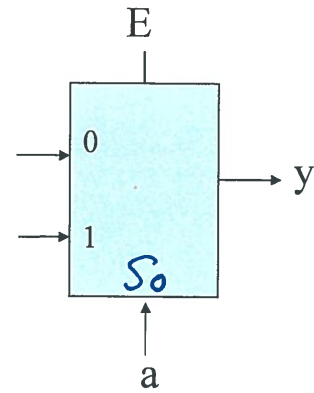


For $(abc) = (111)$
we have $f(1,1,1) = 1$

16

Example 3: Given $f(a,b,c) = \sum m(0,1,7) + \sum d(2)$, implement with 2-input Muxes.

a\bc	00	01	10	11	D(b,c)
0	⁰ 1	¹ 1	² -	³ 0	D_0
1	⁴ 0	⁵ 0	⁶ 0	⁷ 1	D_1



17

Example 3: Given $f(a,b,c) = \sum m(0,1,7) + \sum d(2)$, implement with 2-input Muxes.

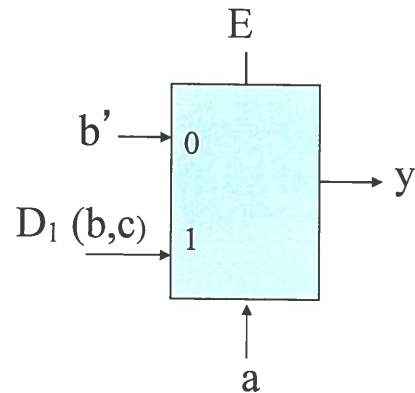
a\bc	00	01	10	11	D(b,c)
0	1	1	-	0	D_0
1	0	0	0	1	D_1

$$D_0(b,c) = b'$$

c=0	⁰ 1	² -
c=1	¹ 1	³ 0
	b=0	b=1

$$D_1(b,c) = bc$$

c=0	⁰ 0	² 0
c=1	¹ 0	³ 1
	b=0	b=1

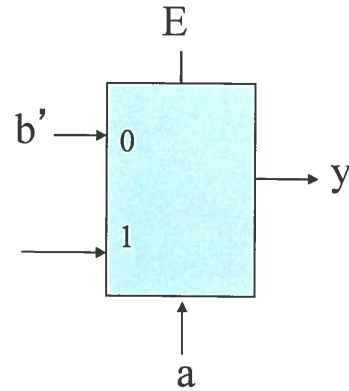


18

Example 3: Given $f(a,b,c) = \sum m(0,1,7) + \sum d(2)$, implement with 2-input Muxes.

$D_1(b,c)$

$b \backslash c$	0	1	D
0	0	0	$D_0 =$
1	0	1	$D_1 =$

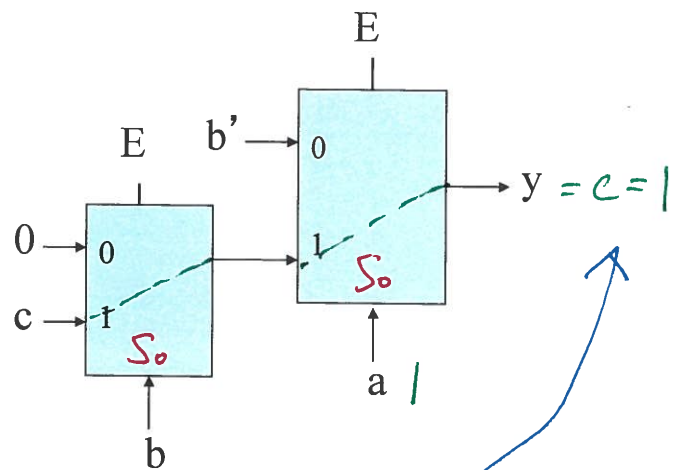


19

Example 3: Given $f(a,b,c) = \sum m(0,1,7) + \sum d(2)$, implement with 2-input Muxes.

$D_1(b,c)$

$b \backslash c$	0	1	D(c)
0	0	0	$D_0 = 0$
1	0	1	$D_1 = c$

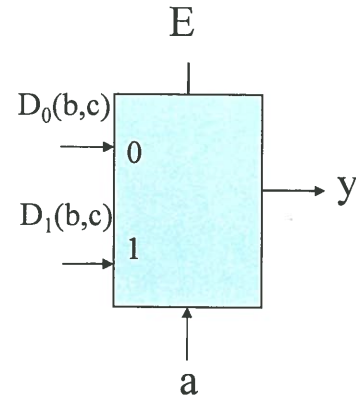


$$f(a,b,c) = 1$$

20

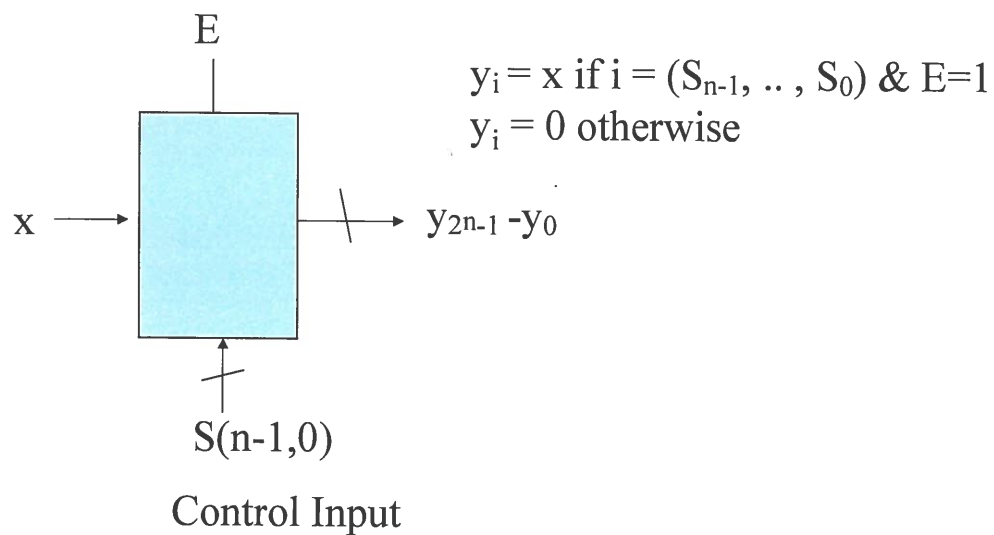
Example 4: Given $f(a,b,c) = \sum m(0,2,4,7) + \sum d(3,5)$, implement with 2-input Muxes.

a\bc	00	01	10	11	D
0	1	0	1	-	D_0
1	1	-	0	1	D_1



21

4. Demultiplexers



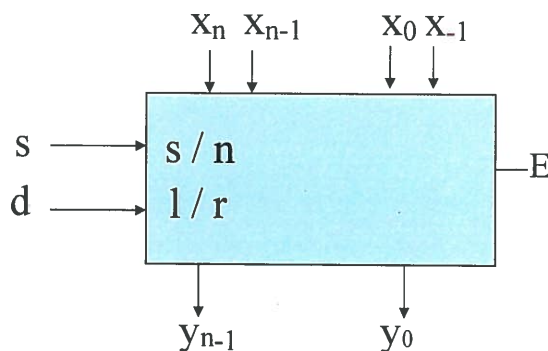
22

Shifters

- **Logical shifter:** shifts value to left or right and fills empty spaces with 0's
 - Ex: $11001 \gg 2 = 00110$
 - Ex: $11001 \ll 2 = 00100$
- **Arithmetic shifter:** same as logical shifter, but on right shift, fills empty spaces with the old most significant bit (msb).
 - Ex: $11001 \ggg 2 = 11110$
 - Ex: $11001 \lll 2 = 00100$
- **Rotator:** rotates bits in a circle, such that bits shifted off one end are shifted into the other end
 - Ex: $11001 \text{ ROR } 2 = 01110$
 - Ex: $11001 \text{ ROL } 2 = 00111$

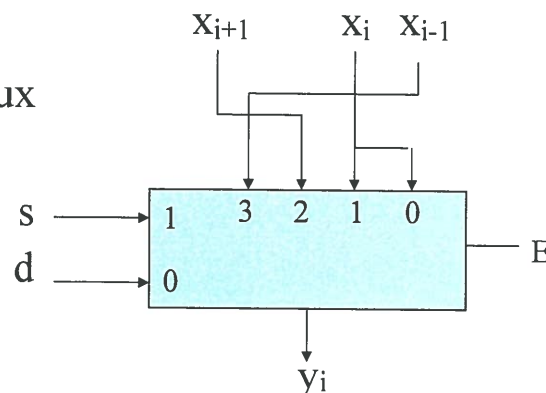
23

Shifter

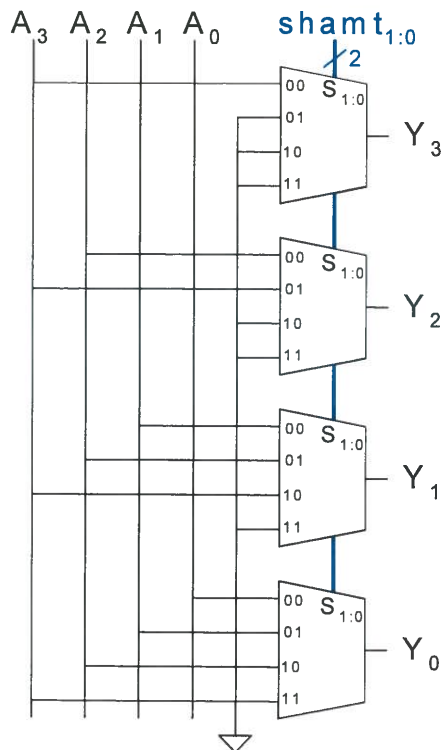
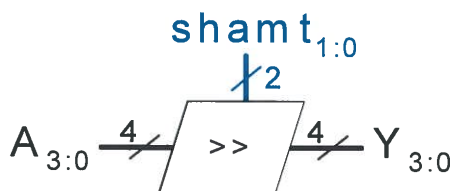


$$\begin{aligned}
 y_i &= x_{i-1} \text{ if } E = 1, s = 1, \text{ and } d = L \\
 &= x_{i+1} \text{ if } E = 1, s = 1, \text{ and } d = R \\
 &= x_i \text{ if } E = 1, s = 0 \\
 &= 0 \text{ if } E = 0
 \end{aligned}$$

Can be implemented with a mux

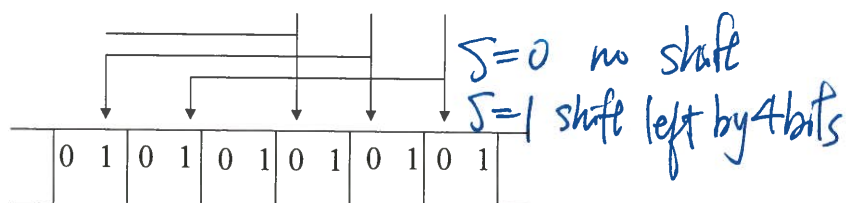
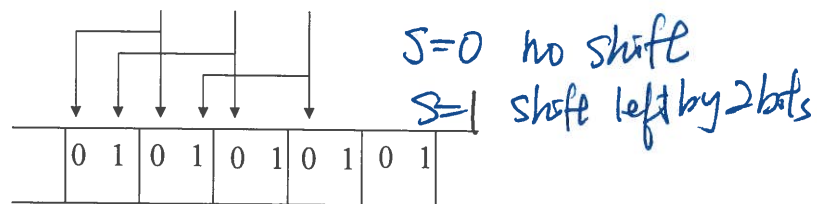
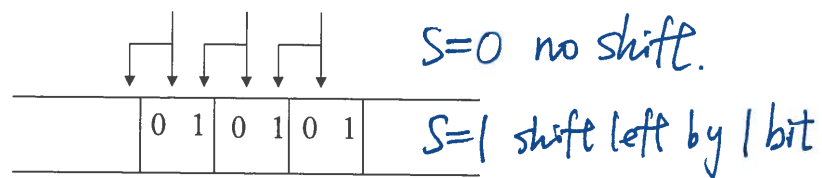
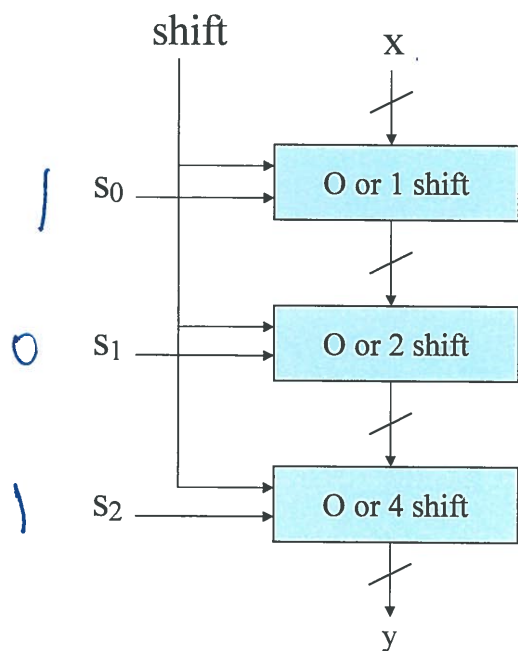


Shifter Design



25

Barrel Shifter



Shifters as Multipliers and Dividers

- A left shift by N bits multiplies a number by 2^N
 - Ex: $00001 \ll 2 = 00100$ ($1 \times 2^2 = 4$)
 - Ex: $11101 \ll 2 = 10100$ ($-3 \times 2^2 = -12$)
- The arithmetic right shift by N divides a number by 2^N
 - Ex: $01000 \ggg 2 = 00010$ ($8 \div 2^2 = 2$)
 - Ex: $10000 \ggg 2 = 11100$ ($-16 \div 2^2 = -4$)

CSE 140 Lecture 16

System Designs Methodology.

GPU
CPU
Video image proc

CK Cheng

CSE Dept.

UC San Diego

1

System Designs

- Introduction
 - Methodology and Framework
- Components
- Specification
- Implementation

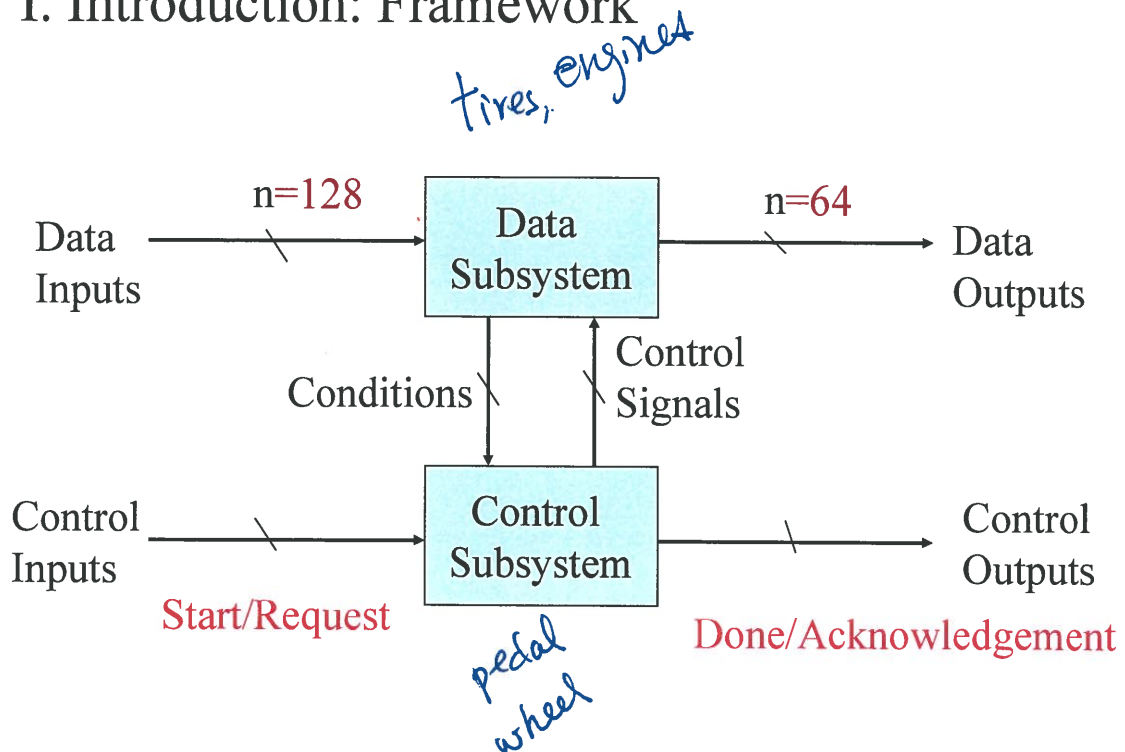
2

Introduction

- Methodology
 - Approach with success stories.
 - Hierarchical designs with interface between the modules (BSV).
- Data Subsystem and Control Subsystem
 - For n-bit data, each operation takes n times or more in hardware complexity.
 - Data subsystem carries out the data operations and transports.
 - Control system sequences the data subsystem and itself.

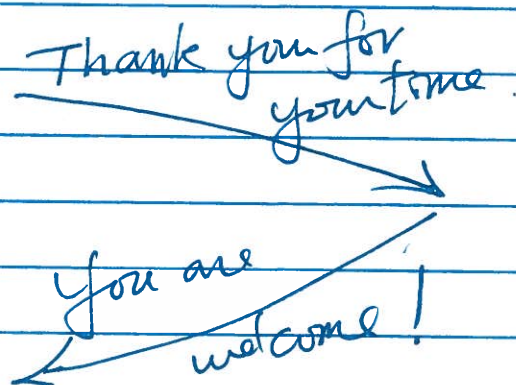
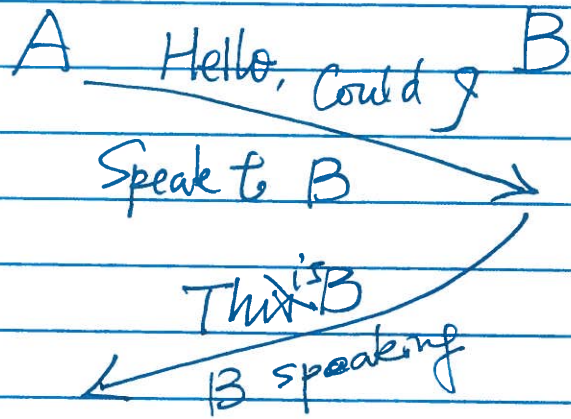
3

I. Introduction: Framework

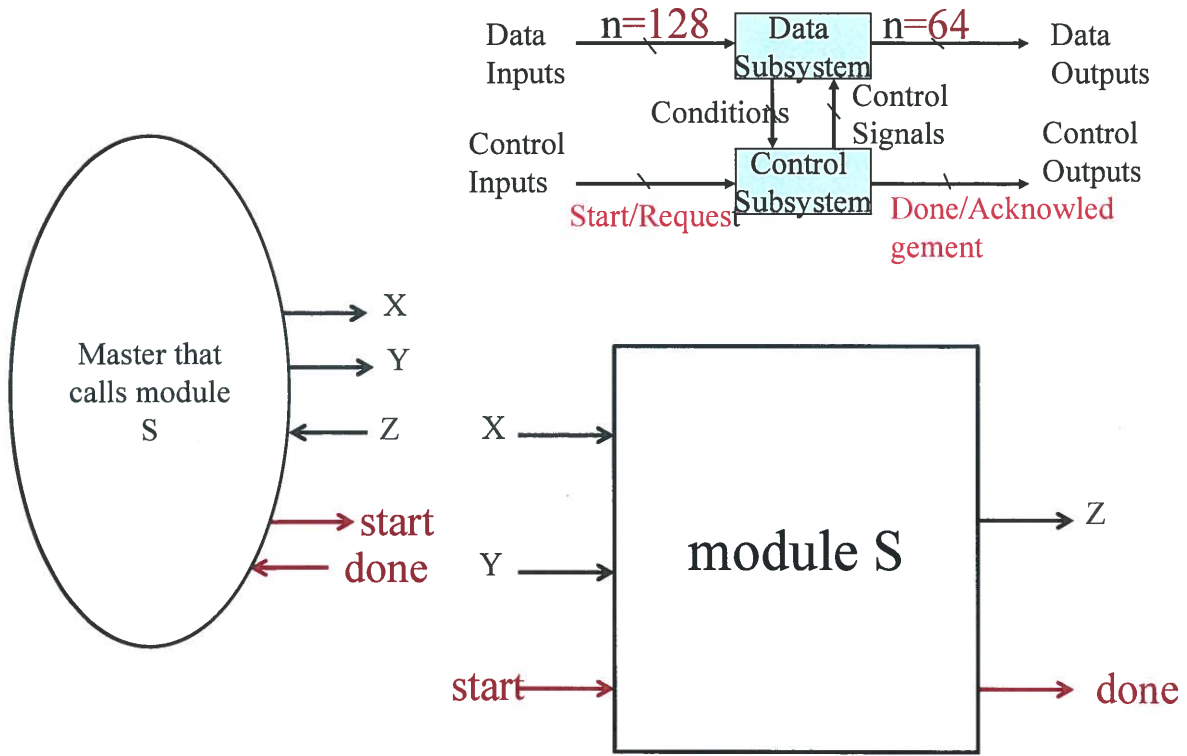


4

4-Phase Handshaking

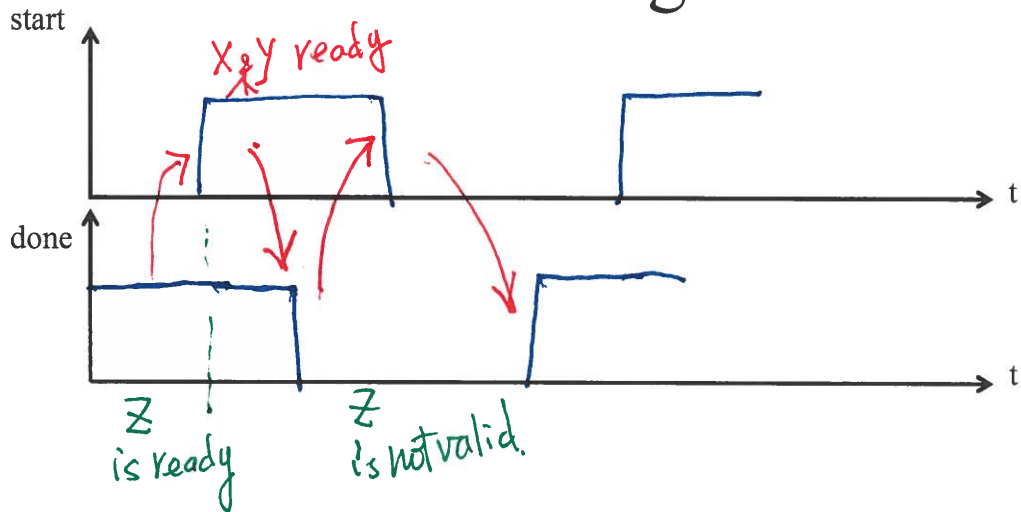


I. Introduction: Handshaking



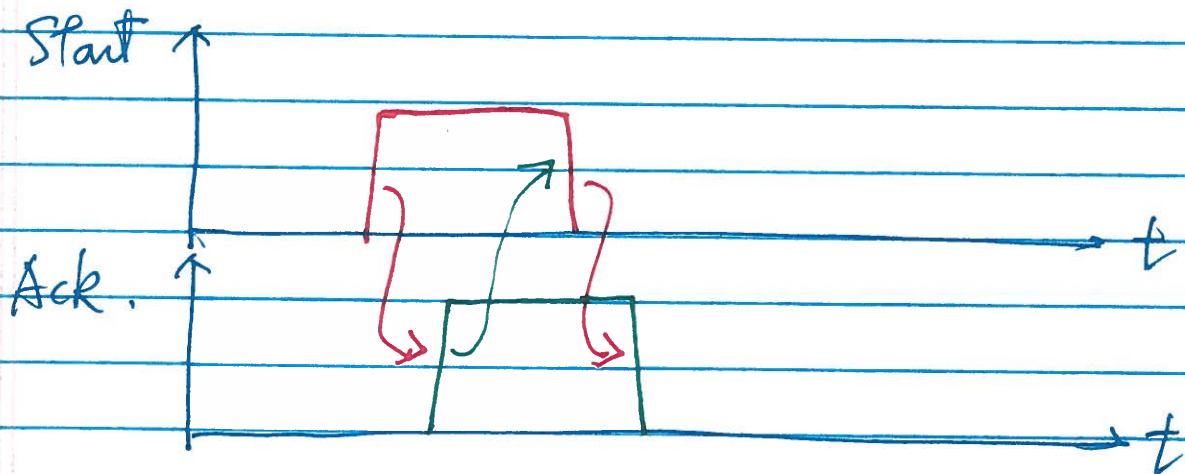
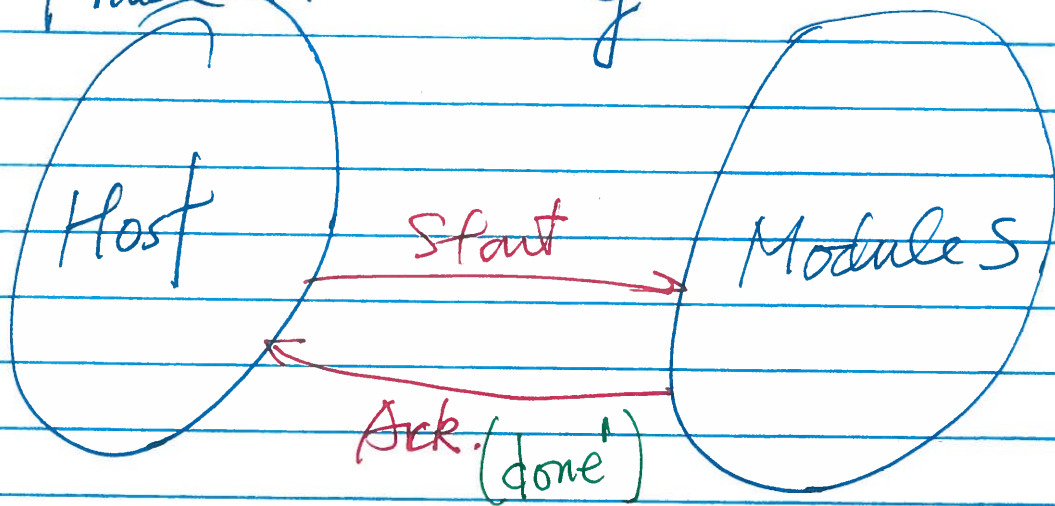
5

Handshaking



6

4-Phase Handshaking



Host Sender wants to start the operation

raises $Start = 1$

Mod.S Receiver detects $Start = 1$,

raises $Ack = 1$ $done = 0$

Host Sender detects $Ack = 1$

lowers $Start = 0$

Mod.S Receiver detects $Start = 0$

lowers $Ack = 0$