

WEEK 12-2015

Computer Design Fundamentals

ELEC2141: Digital Circuit Design

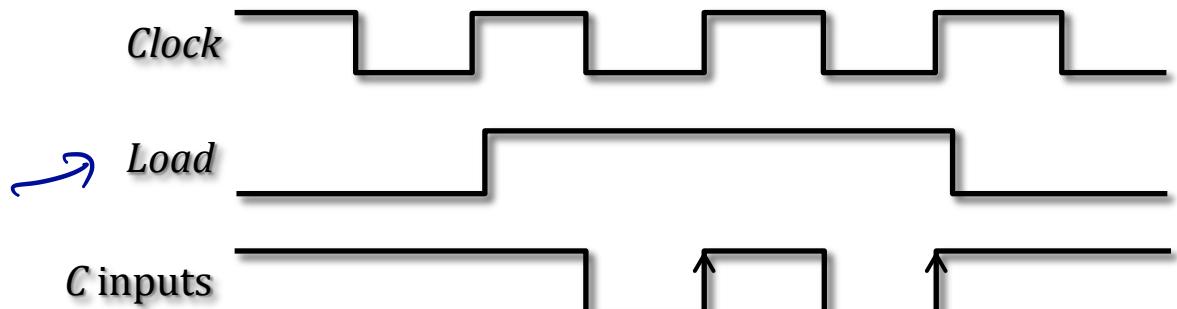
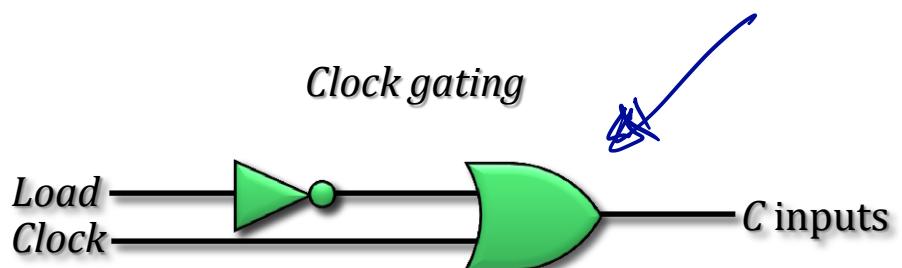
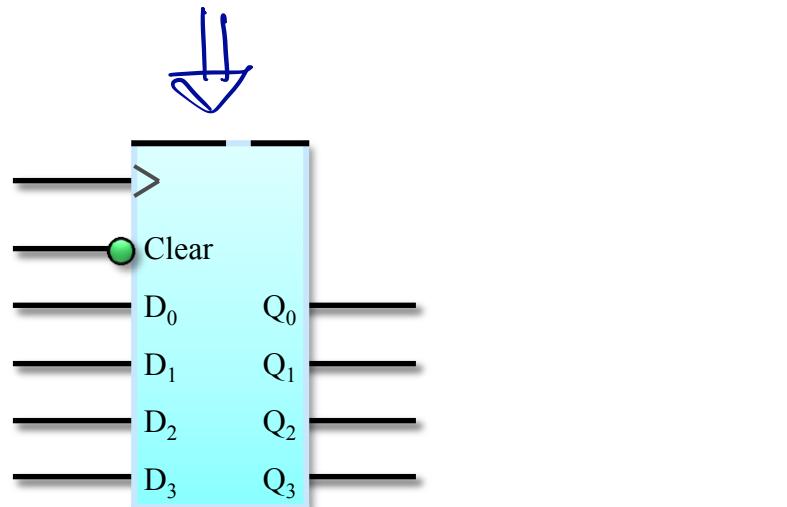
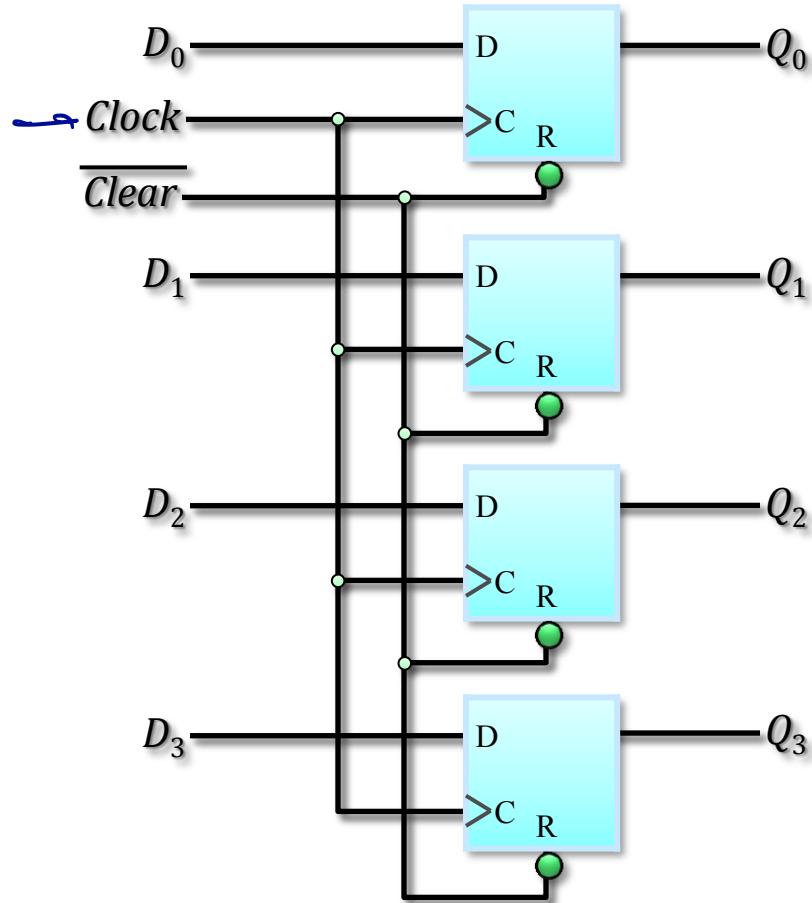


UNSW | ENGINEERING
THE UNIVERSITY OF NEW SOUTH WALES

Registers

- An n-bit register consists of n flip-flops and is capable of storing n-bits of binary information.
- May also have combinational circuit that implements state transitions of the flip-flops.
- The flip-flops hold data and the combinational circuit determine the new or transformed data that will transfer to the flip-flops.
- A counter is special type of register that goes through a predetermined sequence of states.
- Useful for storing and manipulating information in digital computers.

4-bit Register

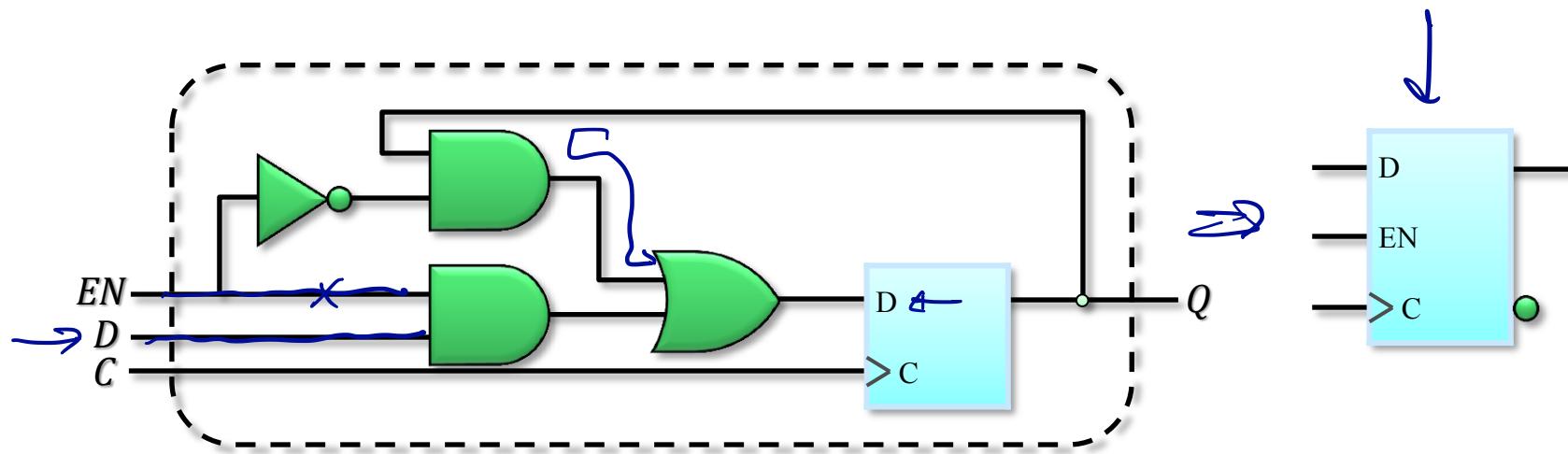


Clock skew

- Inserting additional logic in the clock path introduces a delay.
- Clock signals arrive at different flip-flops and registers at different times depending on the logic in their clock path.
- This is known as a *Clock Skew*
- In truly synchronous system, we must insure that all clock pulses arrive simultaneously throughout the system so that all flip-flops trigger at the same time.
- Therefore, control operation without clock gating is advisable.

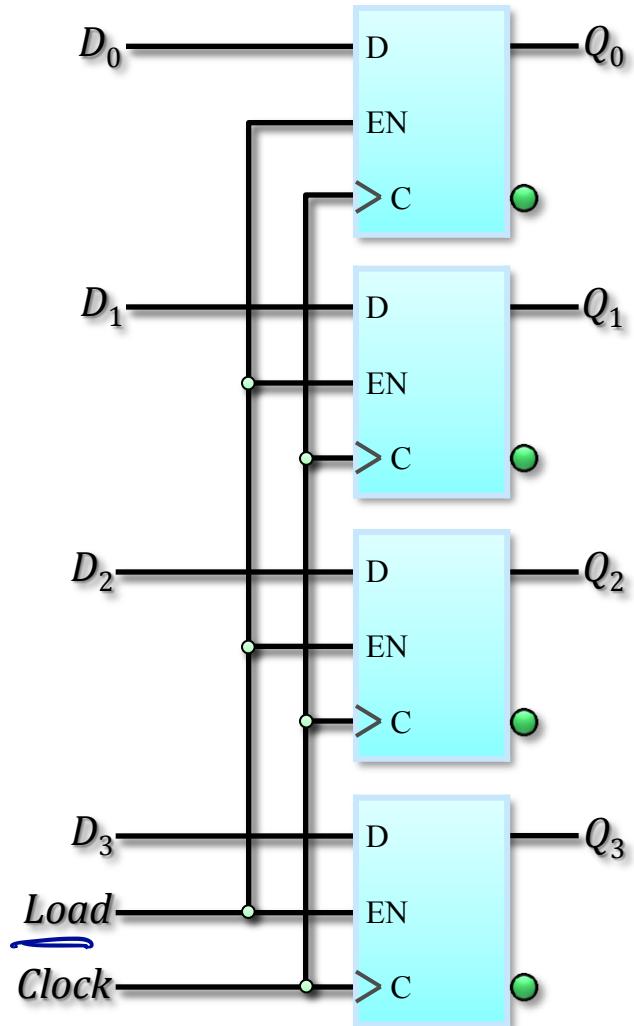
D Flip-flops with Enable

- A D flip-flop with Enable, with no clock gate, is a better option.
- It consists of a 2-to-1 multiplexer and a D flip-flop.



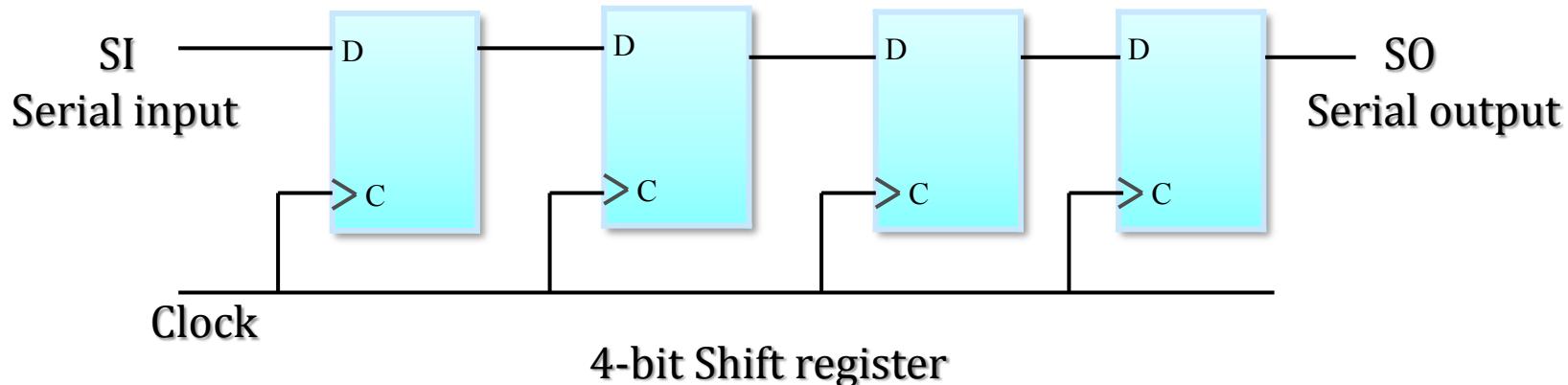
4 bit Register: D flip-flop with enable

- Use the D flip-flop with enable as the building block for the register.
- Connect a common Load signal to the EN inputs of the flip-flops
- When load is 1, the data from the inputs is transferred into the register with the next positive clock edge; otherwise, the current value in the register remains the same.



Shift-register

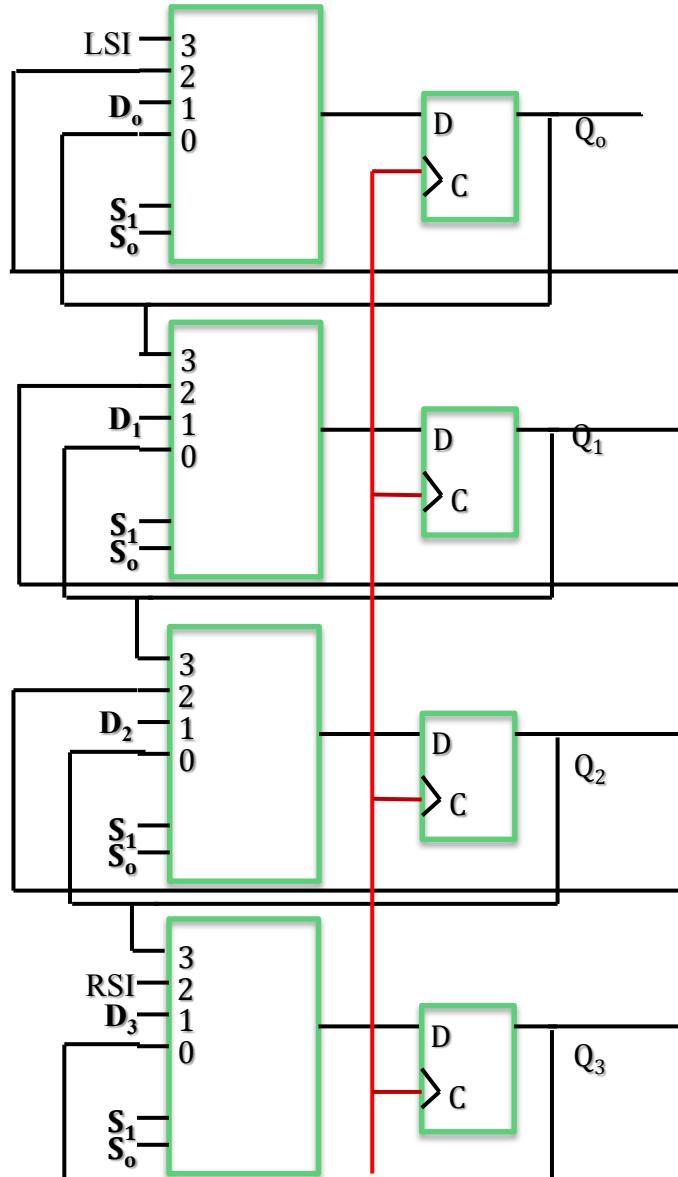
- A register capable of shifting its stored bits laterally in one or both directions.



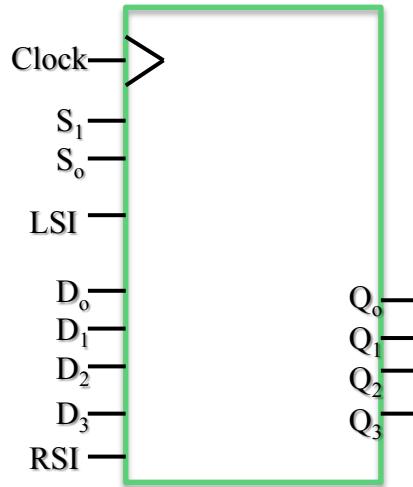
- Bidirectional shift register with parallel loading provides versatile operation.

Bidirectional Shift Register

- Capable of: left, right serial shift, parallel loading, and holding data.



Mode Control		Register Operation
<u>S_1</u>	<u>S_o</u>	
0	0	No change (Hold)
0	1	Parallel load
1	0	RSI (Shift up) ↗
1	1	LSI (Shift down) ↙



Computer Design Fundamentals

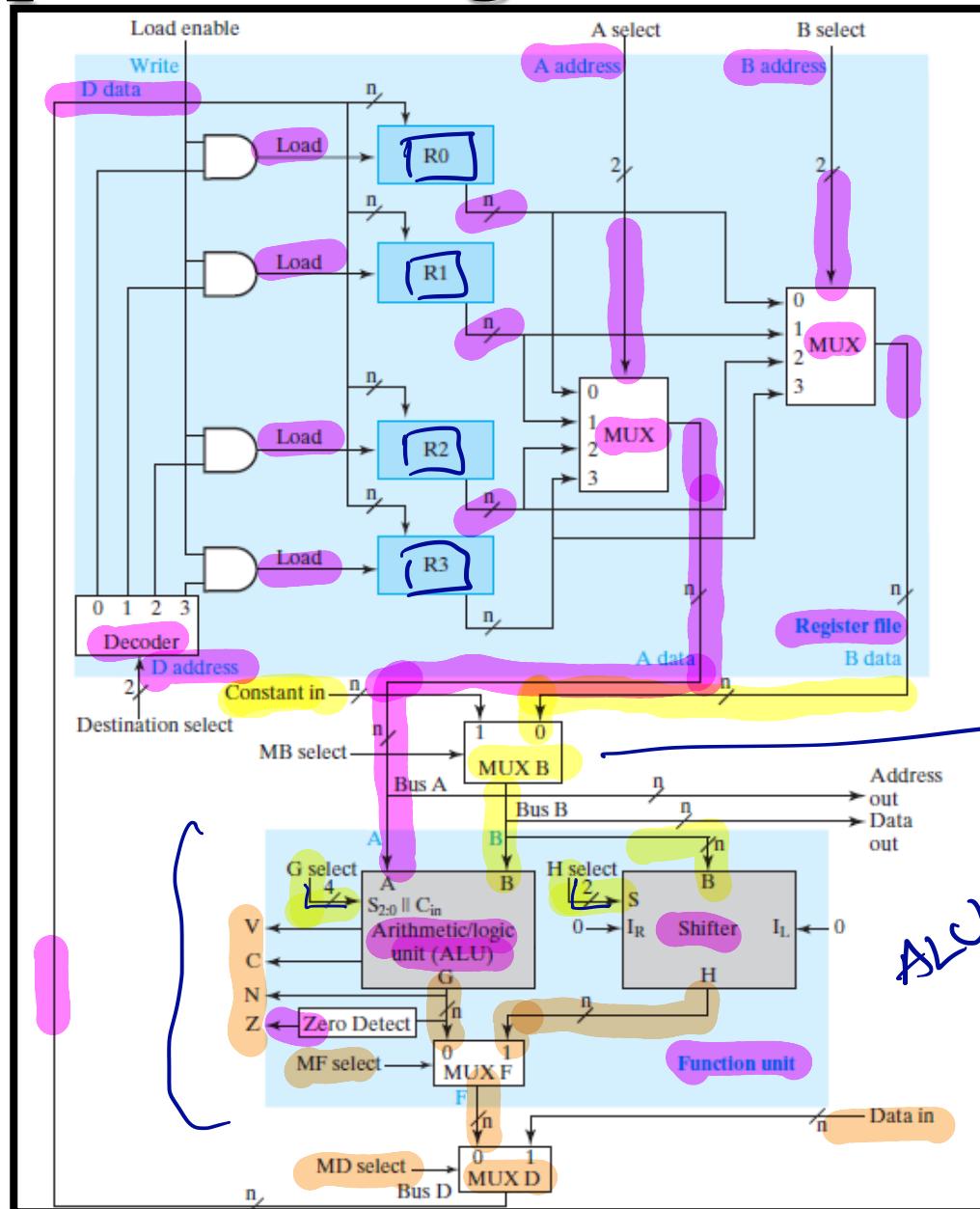
- A simple computer architecture can be divided into:
- Datapath – for performing operations. It is defined by three basic components:

-
- A set of registers
 - The microoperations performed on data stored in the registers
 - The control interface ← *bit level*

- Control Unit – for controlling datapath operations

Control Unit

Computer Design Fundamentals



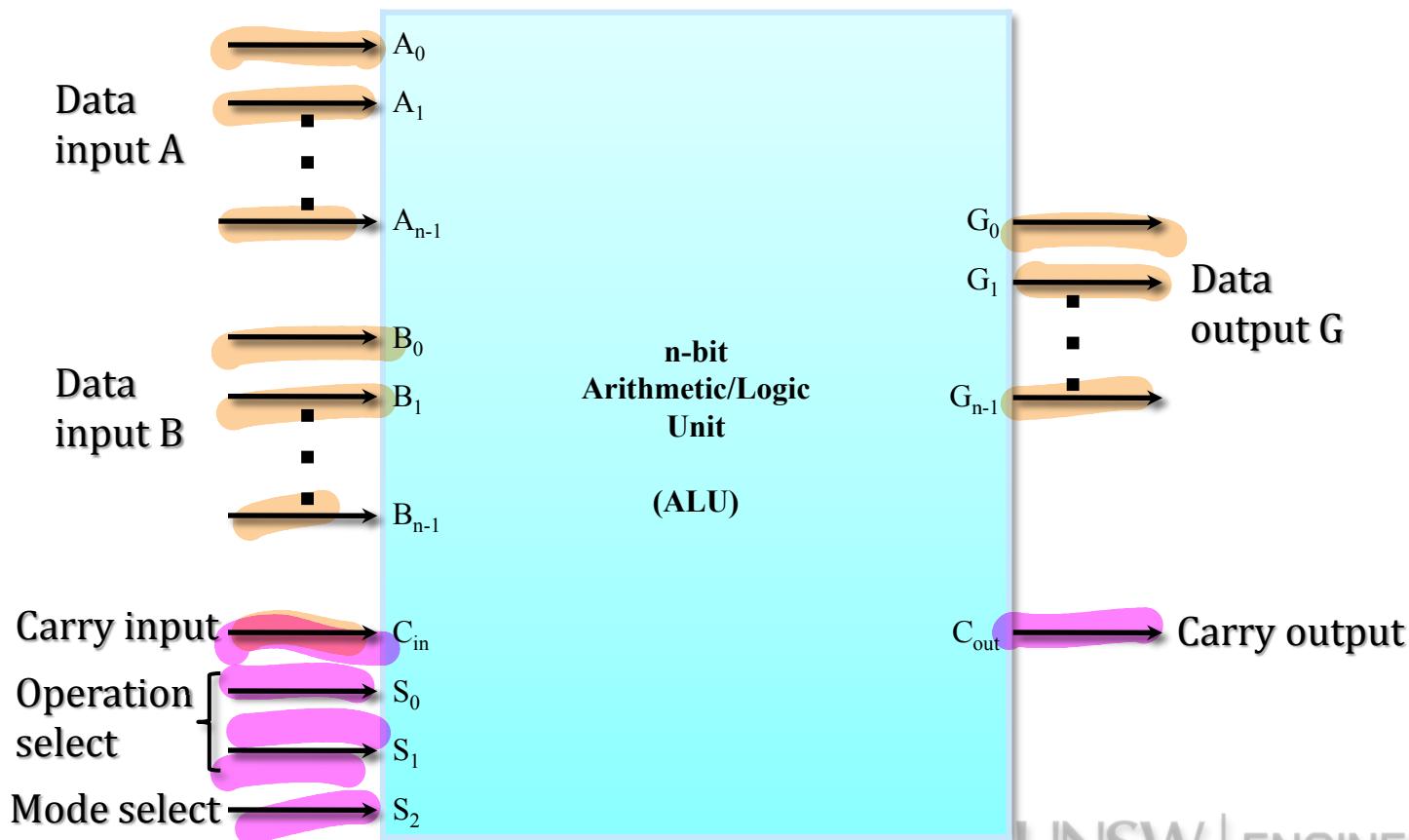
$R_0 + R_2$

processing const
from outside

ALU

Arithmetic/Logic Unit

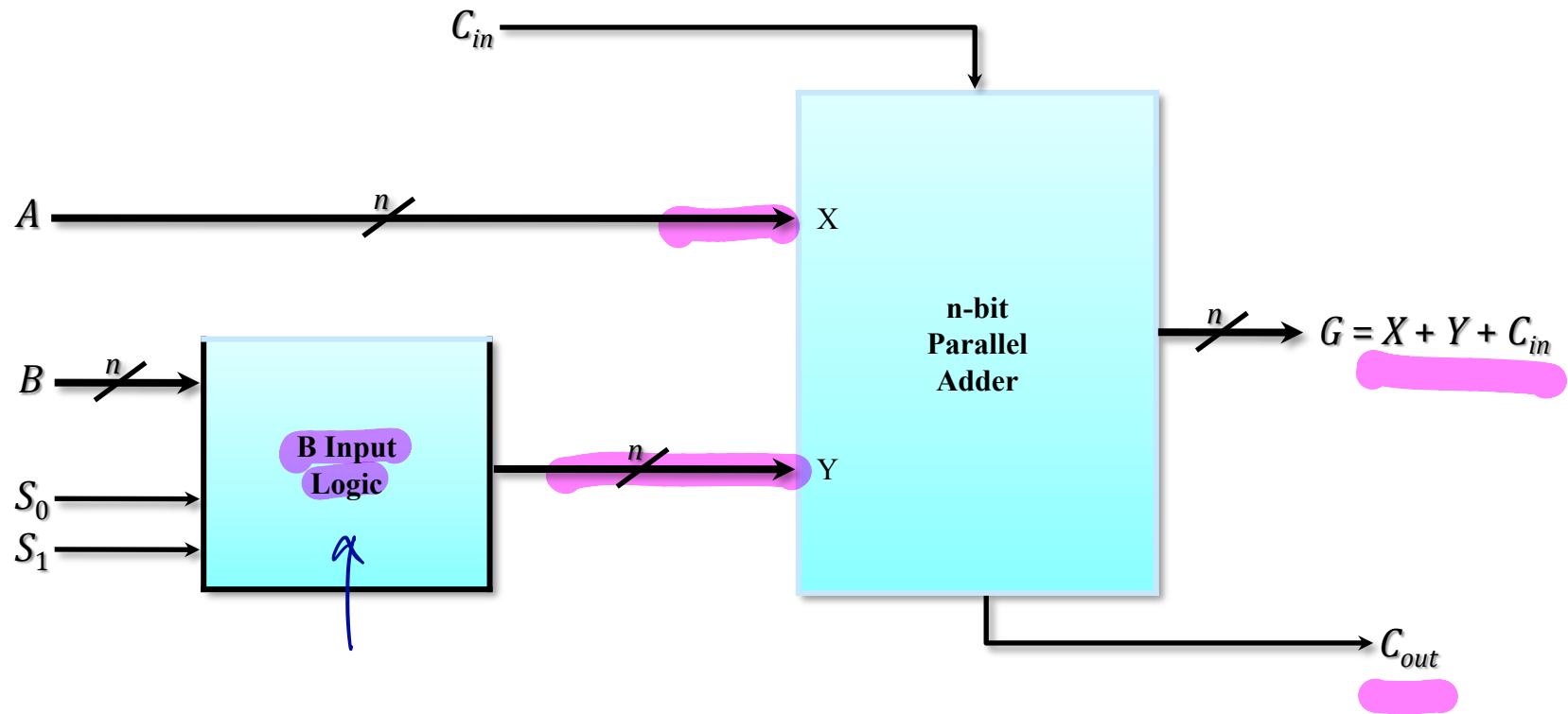
- A combinational circuit that performs basic arithmetic and logic operations



Arithmetic/Logic Unit

- Performs a set of basic arithmetic and logic microoperations.
- The n -bit result is output on vector G with carry-out on C_{out}
- S_2 selects between arithmetic or logic operations
- S_1 and S_0 select one of 4 operations within the arithmetic or logic operations
- Can be extended to 8 operations by using the C_{in} (carry-in) input

Arithmetic/Logic Unit



Arithmetic Unit

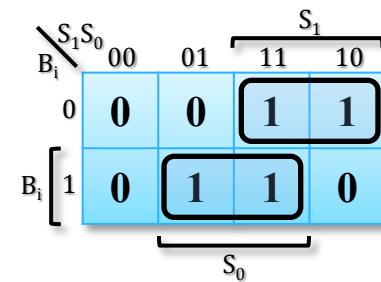
- The arithmetic circuit is based on the n -bit parallel adder
- By controlling the inputs to the adder, different types of arithmetic operations can be obtained
- Connect the X input of the adder directly to operand A
- Make the input Y some function of operand B

Select	Input	$G = A + Y + C_{in}$	
$S_1 \quad S_0$	Y	$C_{in} = 0$	$C_{in} = 1$
0 0	All 0s	$G = A$ (transfer)	$G = A + 1$ (increment)
0 1	B	$G = A + B$ (add)	$G = A + B + 1$
1 0	\bar{B}	$G = A + \bar{B}$	$G = A + \bar{B} + 1$ (subtract)
1 1	All 1s	$G = A - 1$ (decrement)	$G = A$ (transfer)

Arithmetic Unit

- Use a truth table and K-maps to implement the select logic on input B

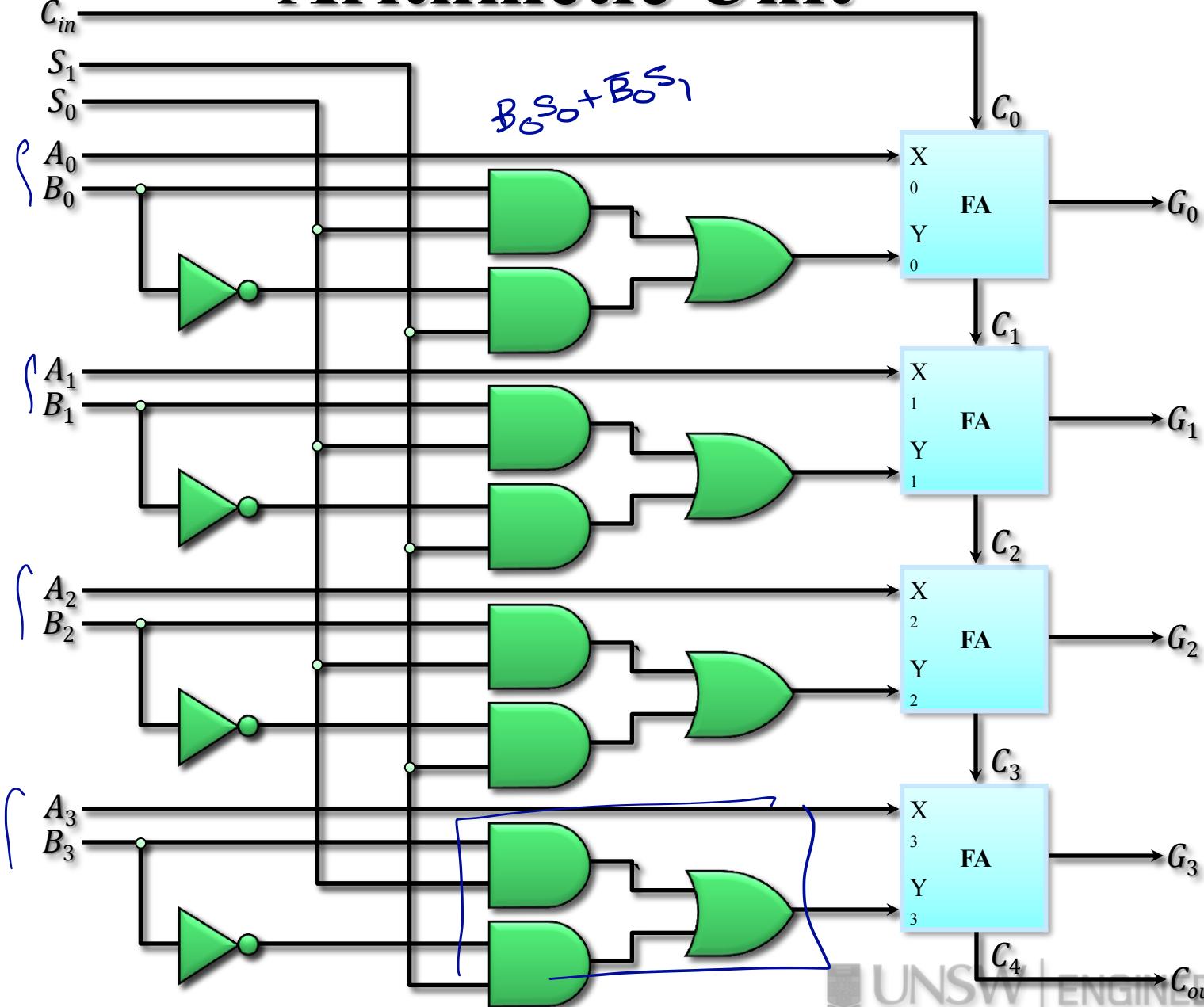
Inputs			Output	Function
S_1	S_0	B_i	Y_i	
0	0	0	0	$Y_i = 0$
0	0	1	0	
0	1	0	0	$Y_i = B_i$
0	1	1	1	
1	0	0	1	$Y_i = \bar{B}_i$
1	0	1	0	
1	1	0	1	$Y_i = 1$
1	1	1	1	



$$Y_i = B_i S_0 + \bar{B}_i S_1$$

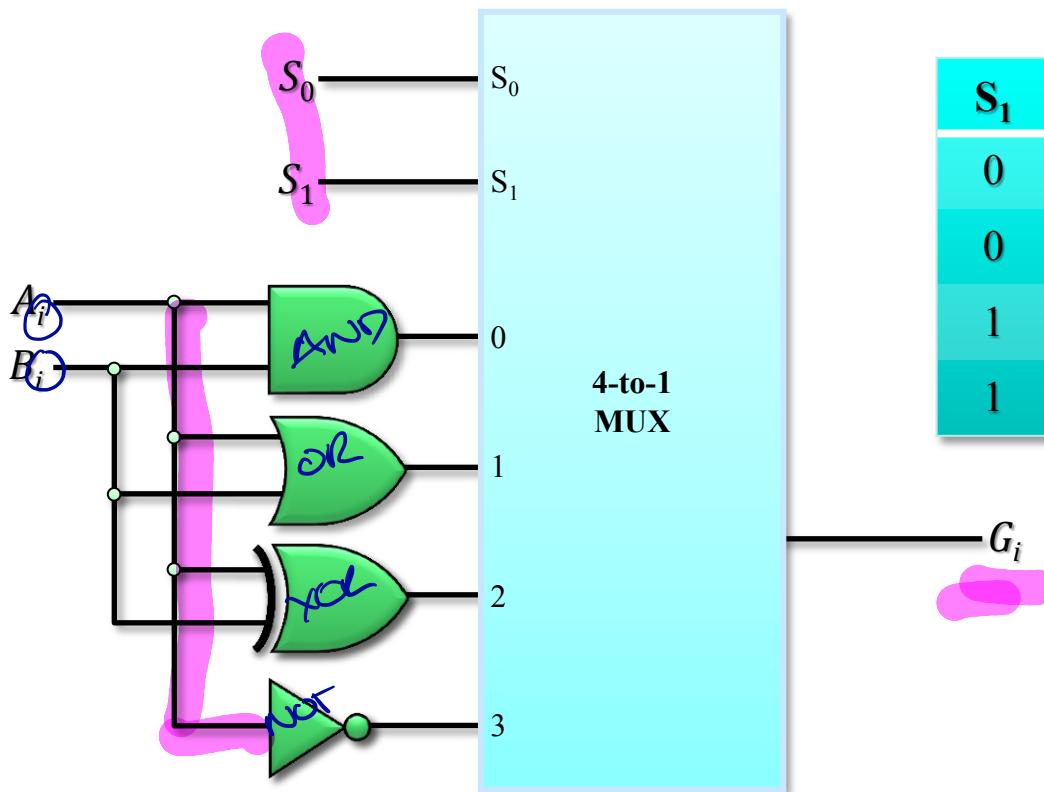
control unit

Arithmetic Unit



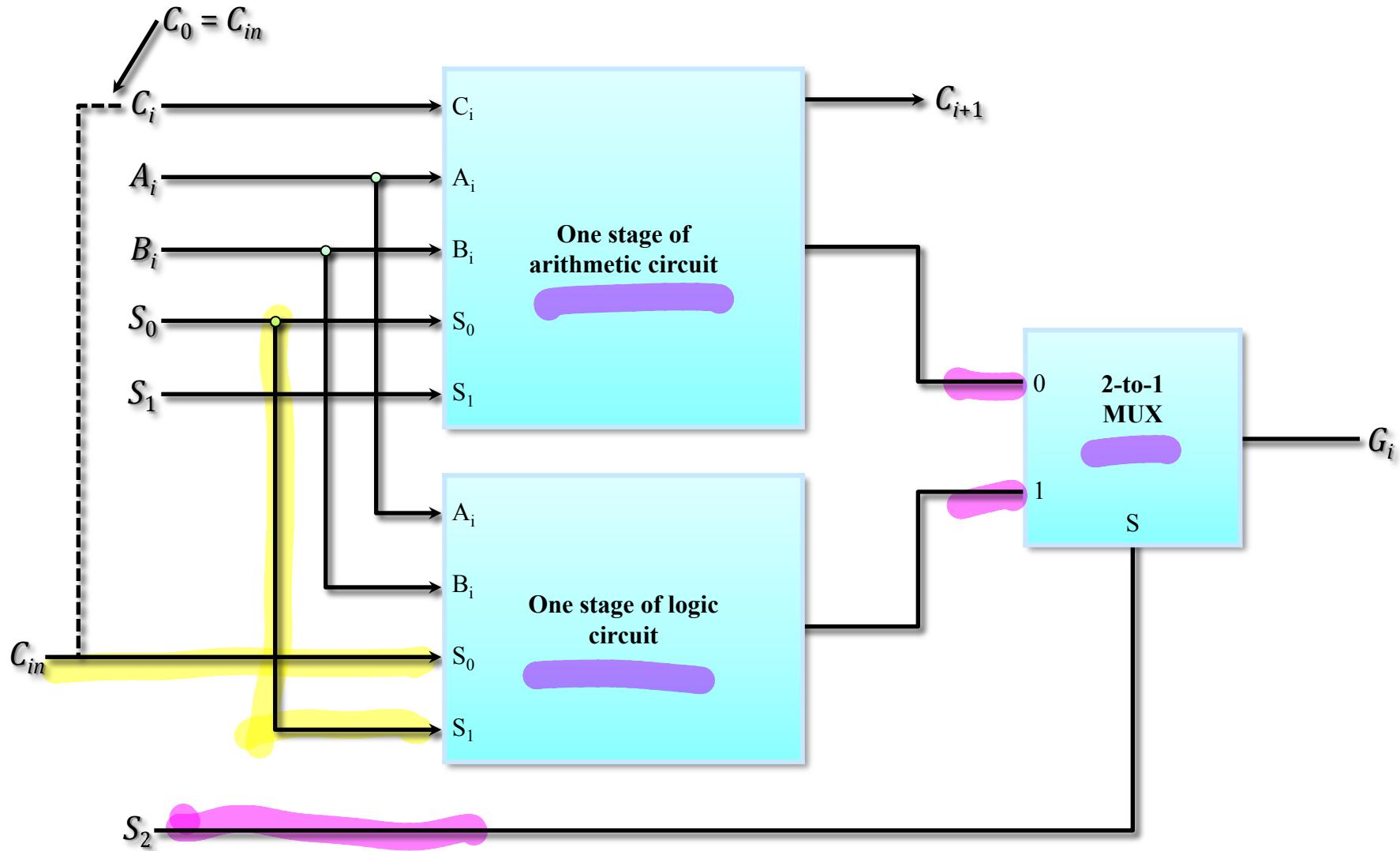
Logic Unit

- Performs bitwise logical operations on the bits of the operands.
- There are four commonly used logic operations- AND, OR, XOR, and NOT
- One stage logic unit: (as many as n-bit are required)



S_1	S_0	Output	Operation
0	0	$G = A \& B$	AND
0	1	$G = A B$	OR
1	0	$G = A \oplus B$	XOR
1	1	$G = \bar{A}$	NOT

Arithmetic/Logic Unit



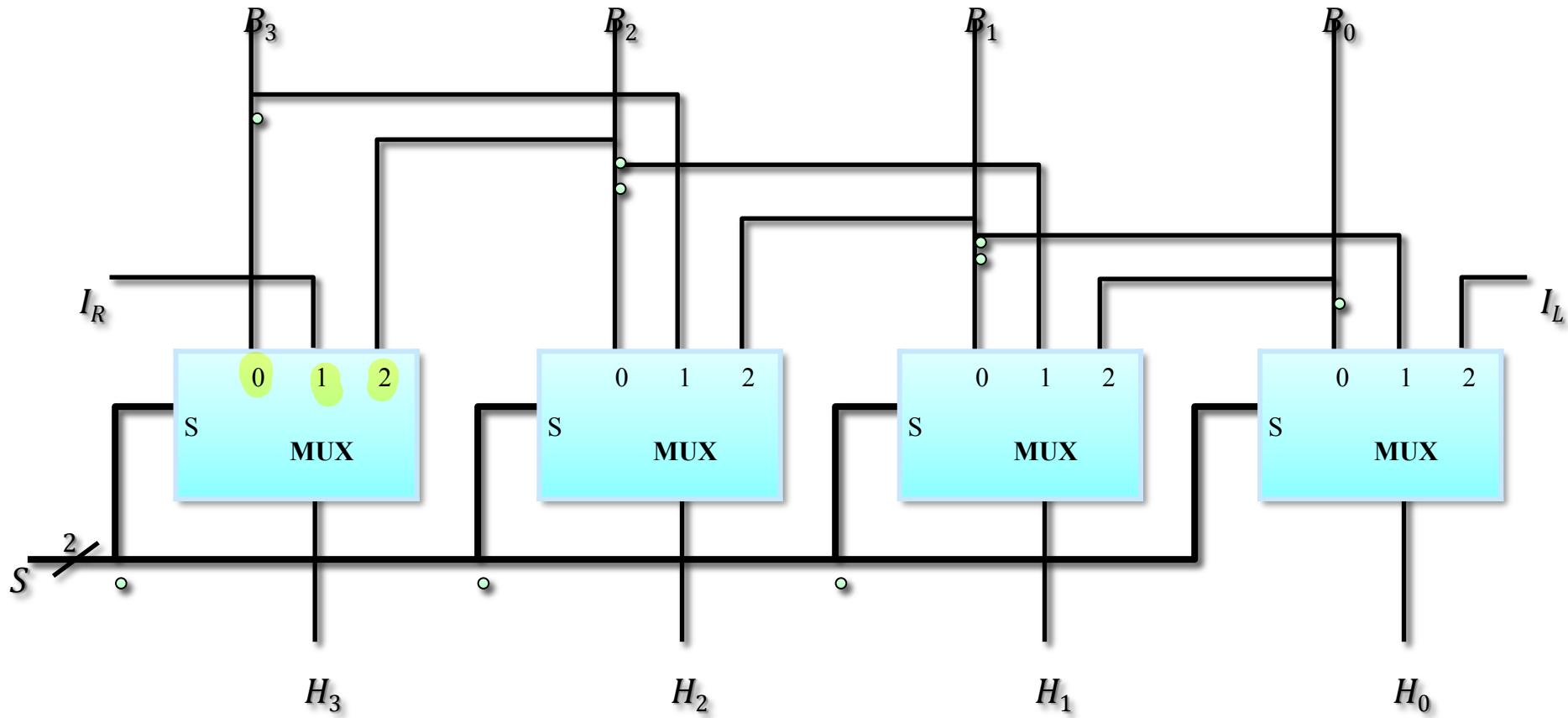
Arithmetic/Logic Unit

Operation Select				Operation	Function
S_2	S_1	S_0	C_{in}		
0	0	0	0	$G = A$	Transfer A
0	0	0	1	$G = A + 1$	Increment A
0	0	1	0	$G = A + B$	Addition
0	0	1	1	$G = A + B + 1$	Add with carry input of 1
0	1	0	0	$G = A + \bar{B}$	A plus 1s complement of B
0	1	0	1	$G = A + \bar{B} + 1$	Subtraction
0	1	1	0	$G = A - 1$	Decrement A
0	1	1	1	$G = A$	Transfer A
1	X	0	0	$G = A \& B$	AND
1	X	0	1	$G = A B$	OR
1	X	1	0	$G = A \oplus B$	XOR
1	X	1	1	$G = \bar{A}$	NOT (1s complement)

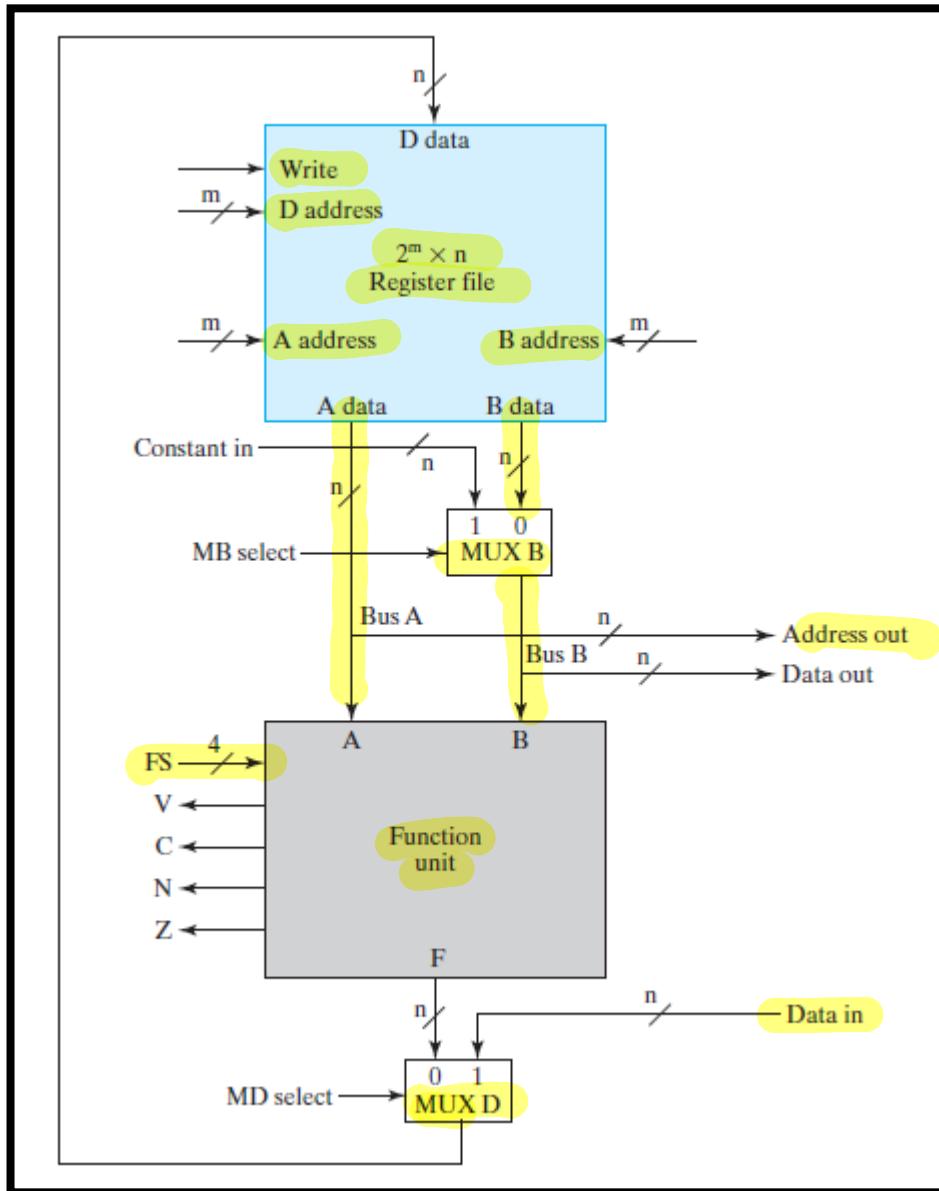
The Shifter

- The *Shifter* shifts the value on Bus B by one bit to the left or by one bit to the right
- Constructed using a combinational circuit so not dependent on the system clock
- The shifter can be designed using n multiplexers with two select lines:
 - $S = 00$ – no shift
 - $S = 01$ – right shift
 - $S = 10$ – left shift

4-bit shifter



Datapath Representation



Datapath representation

- Reduce the apparent of the datapath by enclosing different functions into modules
- Different implementations of the modules can be interchanged without redesigning the entire datapath
- The size of the register file is $2^m \times n$ where m is the number of register address bits and n is the number of bits per register
- The required operation of the Function Unit is selected using the 4-bit signal FS (function select)

Data representation

FS[3:0]	MF Select	G[3:0] Select	H[1:0] Select	Operation
0000	0	0000	XX	$F = A$
0001	0	0001	XX	$F = A + 1$
0010	0	0010	XX	$F = A + B$
0011	0	0011	XX	$F = A + B + 1$
0100	0	0100	XX	$F = A + \bar{B}$
0101	0	0101	XX	$F = A + \bar{B} + 1$
0110	0	0110	XX	$F = A - 1$
0111	0	0111	XX	$F = A$
1000	0	1X00	XX	$F = A \& B$
1001	0	1X01	XX	$F = A B$
1010	0	1X10	XX	$F = A \oplus B$
1011	0	1X11	XX	$F = \bar{A}$
1100	1	XXXX	00	$F = B$
1101	1	XXXX	01	$F = B \gg 1$
1110	1	XXXX	10	$F = B \ll 1$

Data representation

- From the function table, can derive the internal inputs in the Function Unit in terms of the *FS* external input:

$$MF = FS[3] \cdot FS[2]$$

$$G[3:0] = FS[3:0]$$

$$H[1:0] = FS[1:0]$$