The University of Texas at Arlington

More Complex Combinational Circuits

CSE 2340/2140 – Introduction to Digital Logic

Dr. Gergely Záruba

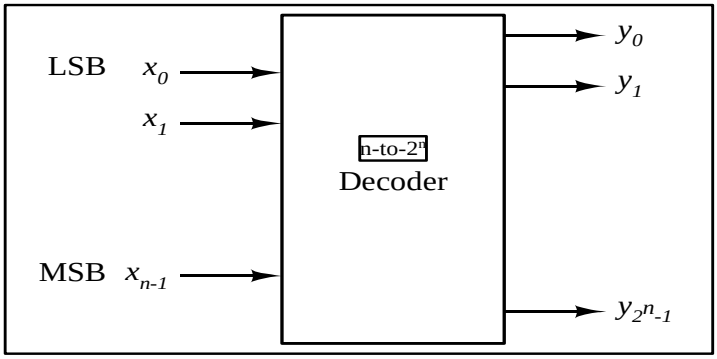
DECODERS

More Complex Combinational Devices

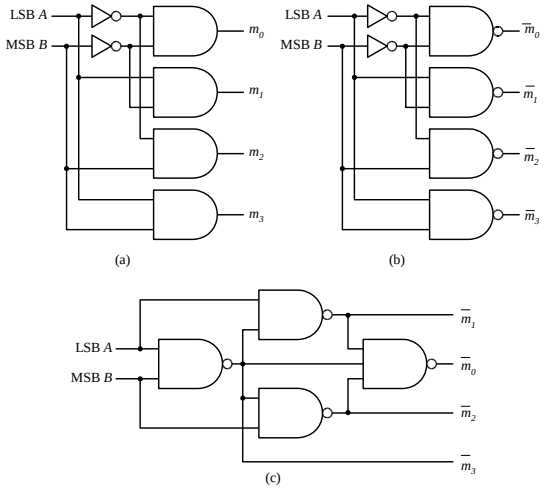
- Last lecture we have briefly seen a list of available combinational devices for circuit design engineers, many of which were not simple gates.
- By now, we know how we can use gates to create switching logic for switching functions; we can use several such functions for devices with more outputs.
- There is a natural question of what functions, i.e., what devices, could prove useful and thus could warrant single chip implementations. We can use such devices then as building blocks in our designs.
- We know from previous classes that ALUs in computers perform the arithmetic operations (i.e., combinational logic operations). What could be components of ALUs?
- We will look at:
  - Decoders (binary)
  - Encoders
  - Multiplexers
  - Demultiplexers (really, decoders)
  - Adders, subtractors
  - Comparators

Decoders

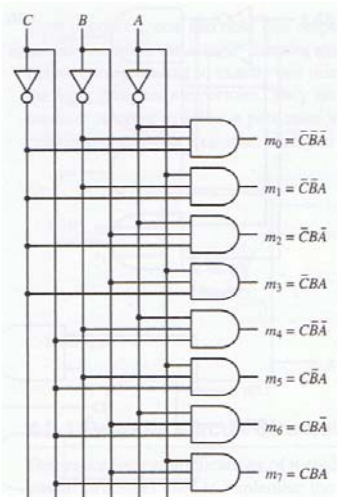
- An n-to-2^n decoder is a multiple-output combinational logic network with n input lines and 2^n output signals,
- For each possible input condition, one and only one output signal will be 'high'. Therefore, a decoder can be considered a minterm generator with each output corresponding to exactly one minterm.



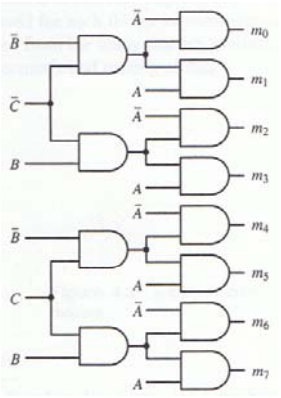
Decoder Realization



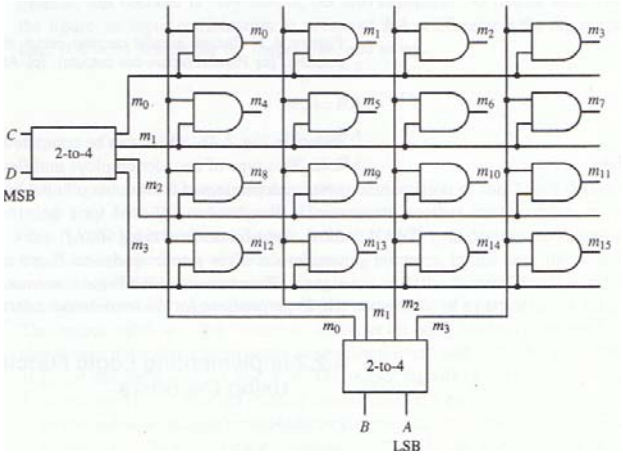
3-bit Parallel Decoder



3-bit Tree-Type Decoder

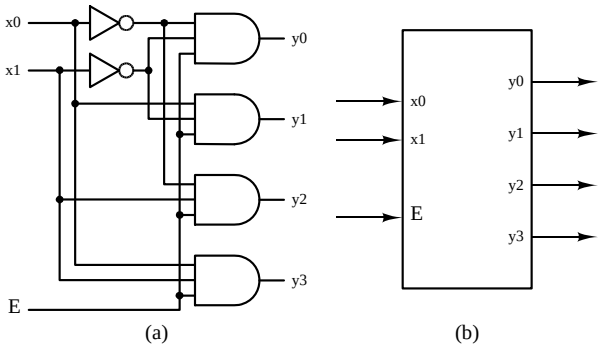


4-bit Dual Tre Decoder

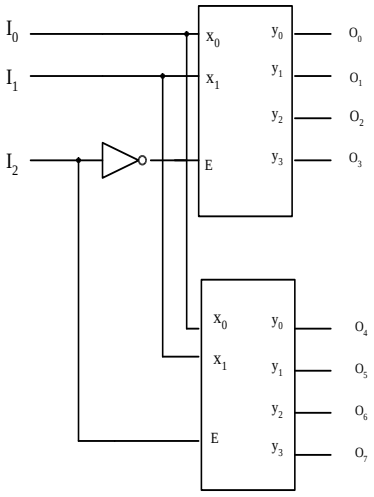


"Enable" Control Inputs

- Decoders and other functional modules often include one or more enable inputs, which can be used to either inhibit (disable) the designated function or allow (enable) it to be performed. The decoding function is inhibited by E=0 forcing all its outputs to the inactive '0' state. Setting E=1 "turns on" the decoder, (an output of 1 indicates the presence of corresponding minterm).

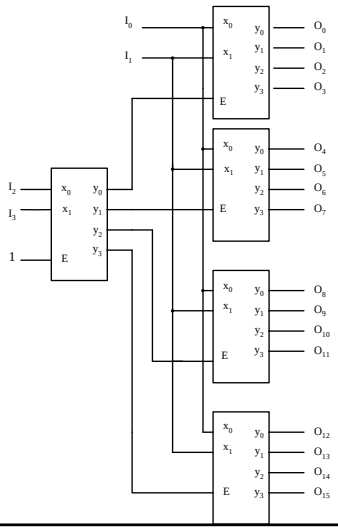


Cascading Decoders



Use of 2-to-4 decoder modules to realize a 3-8 decoder

Cascading Decoders (cont'd)



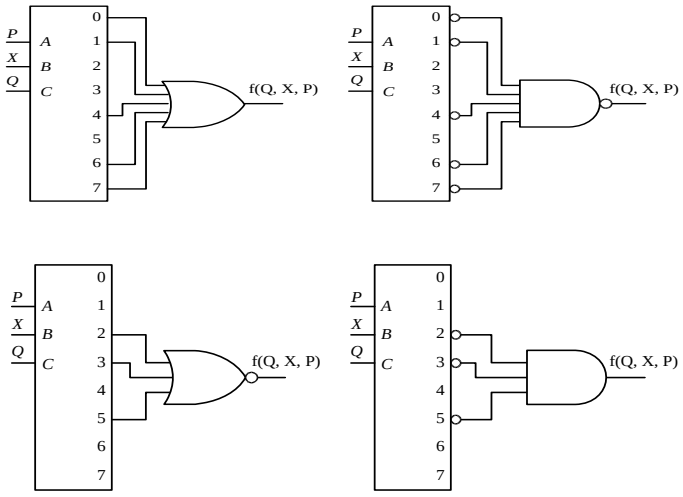
Use of 2-to-4 decoder modules to realize a 4-16 decoder

Decoders and SOP Switching Functions

- Decoders (and an additional gate) may be used to realize switching functions.
- For minterms and active high outputs (on the dfecoder) use an additional OR gate. For minterms and active low outputs use an additional NAND gate.
- For maxterms and active high outputs use an additional NOR gate. For maxterms and active low outputs use an additional AND gate.

12

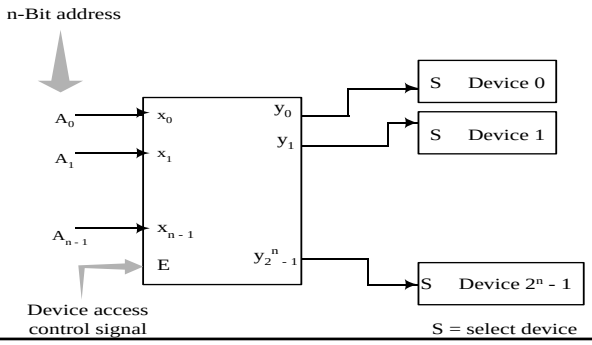
$f(P, Q, X) = m_0 + m_1 + m_4 + m_6 + m_7$



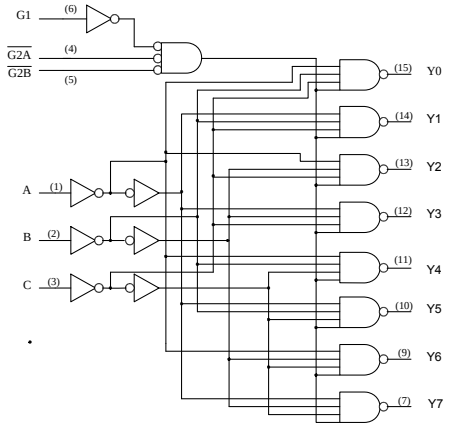
13

Decoders for Device Addresses

- The use of decoders as address decoders in computer memory is an important application.
  - In this application, each of the 2^n devices (memory cells or I/O ports) is assigned a unique n-bit binary number, or address. The decoder then decodes the address by activating one of 2^n select lines to access one of the devices.



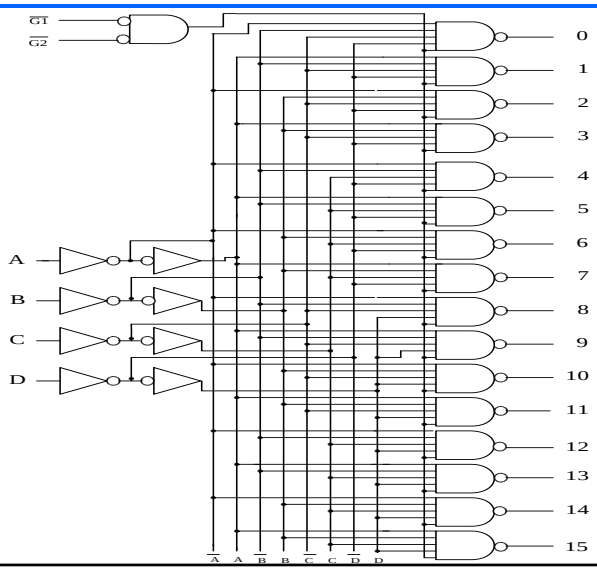
Standard 74138 Decoder Module



Inputs					Outputs							
Enable		Select										
G1	$\overline{G2}$*	C	B	A	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7
H	L	L	L	L	L	H	H	H	H	H	H	H
H	L	L	L	H	H	L	H	H	H	H	H	H
H	L	L	H	L	H	H	L	H	H	H	H	H
H	L	L	H	H	H	H	H	L	H	H	H	H
H	L	H	L	L	H	H	H	H	L	H	H	H
H	L	H	L	H	H	H	H	H	H	L	H	H
H	L	H	H	L	H	H	H	H	H	H	L	H
H	L	H	H	H	H	H	H	H	H	H	H	L
X	H	X	X	X	H	H	H	H	H	H	H	H
L	X	X	X	X	H	H	H	H	H	H	H	H

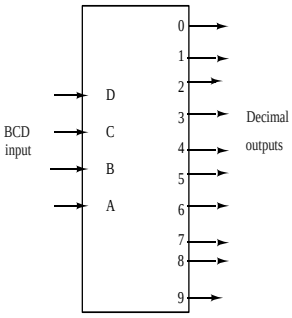
$\overline{G2}$* = $\overline{G2A}$ + $\overline{G2B}$

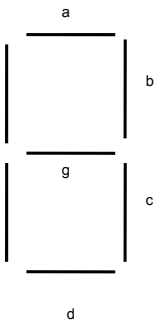
Standard 74154 Decoder Module



More Decoders

- There are other decoders than binary
- E.g., BCD decoder (similar),
- E.g., 7-segment display decoder





17

ENCODERS

18

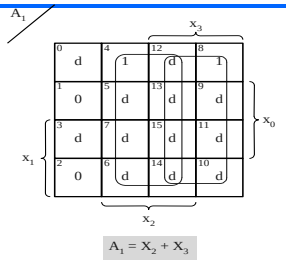
Encoders

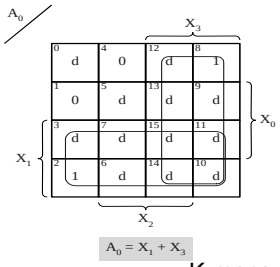
- An encoder is a combinational logic module that assigns a unique output code for each input signal.
- If an encoder has n inputs, the number of outputs s must satisfy the expression s >= $\log_2 n$
- If the inputs are mutually exclusive then this is trivial.
- If not there are choices: if input is invalid, set output to zero, signal that it is invalid, or pick the smallest or largest of the inputs (priority encoder) to be encoded, or just do something unpredictable (easiest, as some minterms become don't cares).

Simple 4-to-2 Encoder

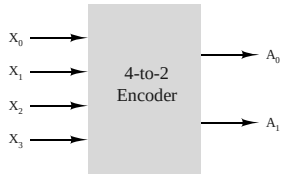
X_3	X_2	X_1	X_0	A_1	A_0
0	0	0	0	d	d
0	0	0	1	0	0
0	0	1	0	0	1
0	0	1	1	d	d
0	1	0	0	1	0
0	1	0	1	d	d
0	1	1	0	d	d
0	1	1	1	d	d
1	0	0	0	1	1
1	0	0	1	d	d
1	0	1	0	d	d
1	0	1	1	d	d
1	1	0	0	d	d
1	1	0	1	d	d
1	1	1	0	d	d
1	1	1	1	d	d

Truth tables

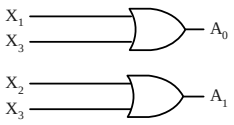




K-maps



Functional Diagram

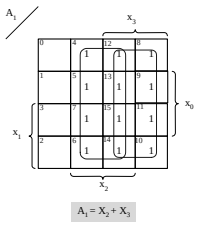


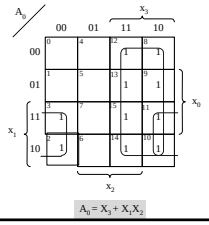
Logic Diagram

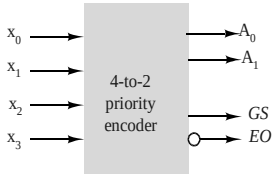
Priority Encoders

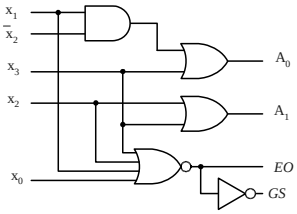
- Pick highest value input line.
- EO =1 indicates no input line is active; GS =1 indicates one or more inputs are active

Inputs				Outputs			
X_3	X_2	X_1	X_0	A_1	A_0	*GS*	*EO*
0	0	0	0	0	0	0	1
0	0	0	1	0	0	1	0
0	0	1	0	0	1	1	0
0	0	1	1	0	1	1	0
0	1	0	0	1	0	1	0
0	1	0	1	1	0	1	0
0	1	1	0	1	0	1	0
0	1	1	1	1	0	1	0
1	0	0	0	1	1	1	0
1	0	0	1	1	1	1	0
1	0	1	0	1	1	1	0
1	0	1	1	1	1	1	0
1	1	0	0	1	1	1	0
1	1	0	1	1	1	1	0
1	1	1	0	1	1	1	0
1	1	1	1	1	1	1	0



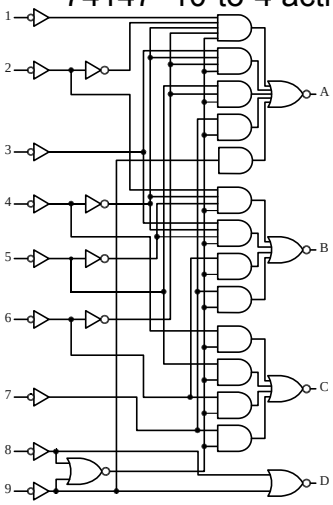


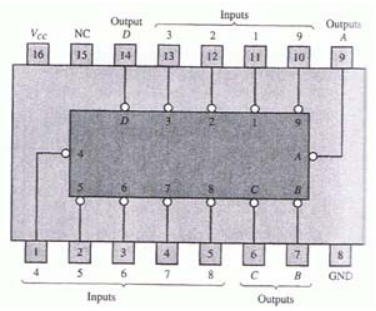




74147 Standard MSI Encoders

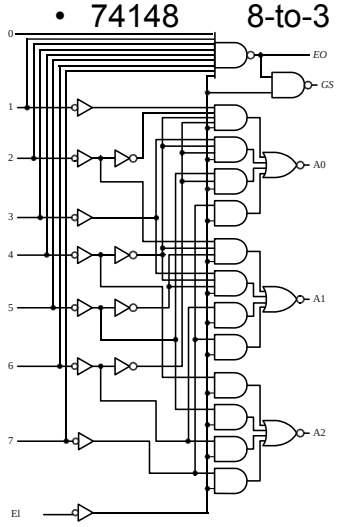
- 74147 10-to-4 active-low priority encoder

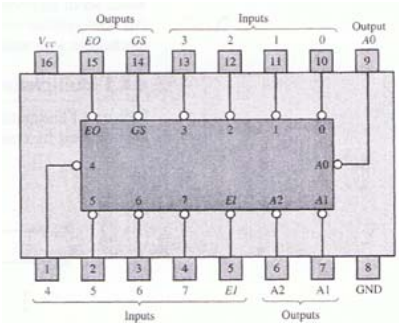




74148 Standard MSI Encoders

- 74148 8-to-3 active-low priority encoder





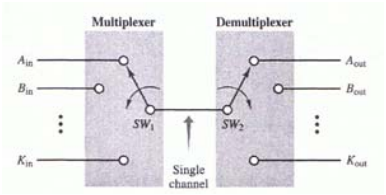
23

MULTIPLEXERS (DATA SELECTORS)

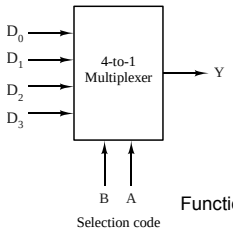
24

Multiplexers/Data Selectors

- A multiplexer, also called a data selector, is a modular device that selects one of many input lines to appear on a single output line.
- In a n-to-1 line multiplexer, one of the n input data lines (D_{n-1}, D_{n-2},……..D_0) is designated for connection to the single output line (Y) by a selection code.



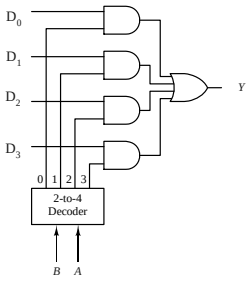
4-to-1 Multiplexer



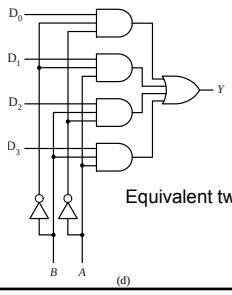
Functional diagram

B	A	Y
0	0	D_0
0	1	D_1
1	0	D_2
1	1	D_3

Truth table



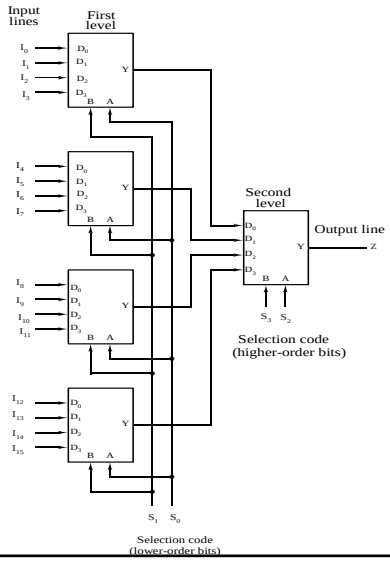
Logic diagram



Equivalent two-level circuit

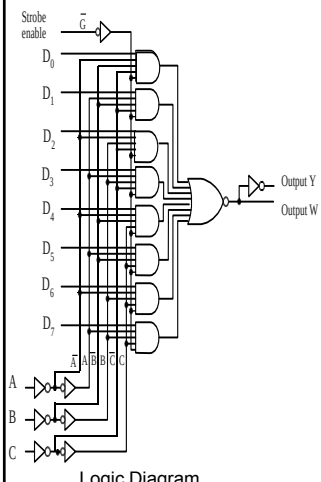
26

Tree Type 16-to-1 Multiplexer



27

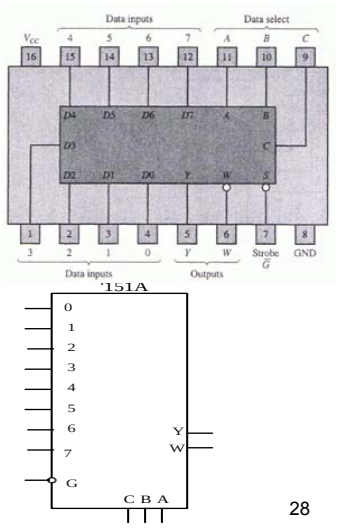
74151A 8-to-1 Multiplexer with Strobe



Logic Diagram

Inputs				Outputs	
Select			Strobe		
C	B	A	$\overline{G}$	Y	W
X	X	X	H	L	H
L	L	L	L	D0	$\overline{D0}$
L	L	H	L	D1	$\overline{D1}$
L	H	L	L	D2	$\overline{D2}$
L	H	H	L	D3	$\overline{D3}$
H	L	L	L	D4	$\overline{D4}$
H	L	H	L	D5	$\overline{D5}$
H	H	L	L	D6	$\overline{D6}$
H	H	H	L	D7	$\overline{D7}$

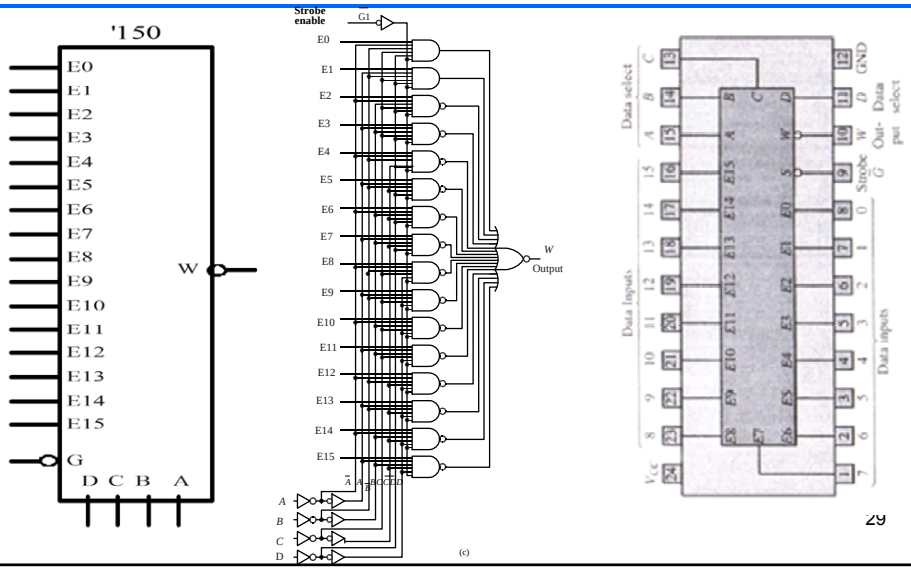
Truth Table



Generic Logic Symbol

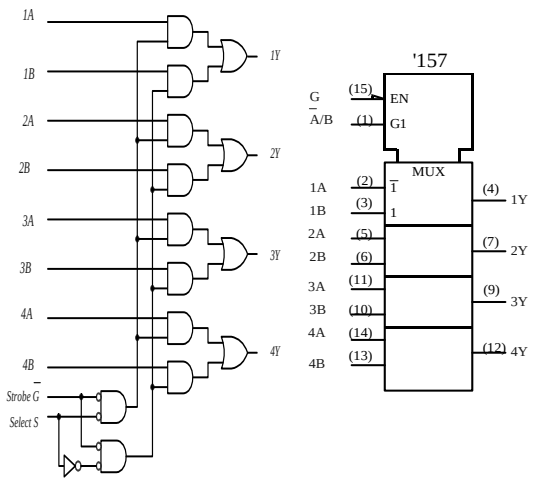
28

74150 16-to-1 Multiplexer

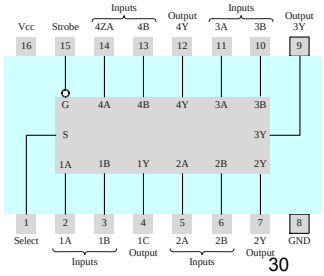


29

74157 Quadruple 2-to-1 Multiplexer

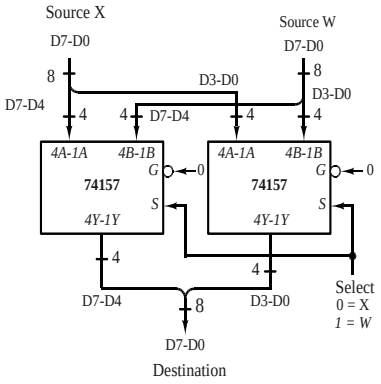


Inputs				Output
Strobe $\overline{G}$	Select S	Data A	Data B	Y
H	X	X	X	L
L	L	L	X	L
L	L	H	X	H
L	H	X	L	L
L	H	X	H	H

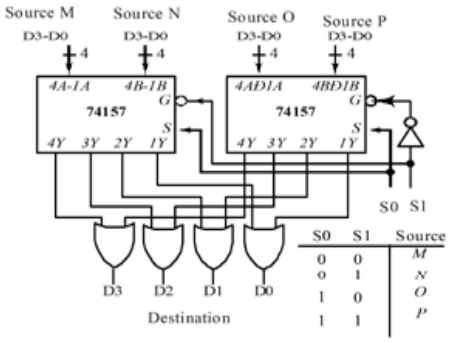


30

Using Multiple 74157-s



8-bit two-input multiplexer



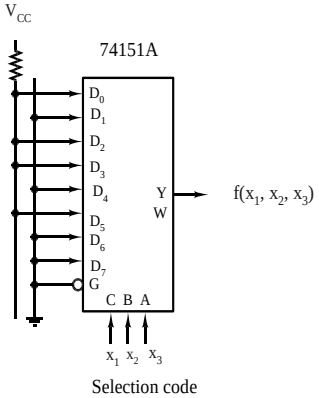
4-bit four-input multiplexer

31

Multiplexers for Switching Function Implementation

- The multiplexers presented so far can be used to implement switching functions. The idea is to use the selection code to generate the minterms of the function, and to use the data lines D_i to enable the minterms present in a specific case.
- E.g., $f(x_1, x_2, x_3) = \sum m(0, 2, 3, 5)$

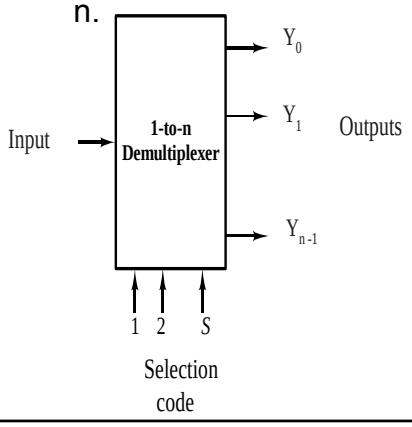
i	C	B	A	Y
	x_1	x_2	x_3	f
0	0	0	0	$1 D_0 = 1$
1	0	0	1	$0 D_1 = 0$
2	0	1	0	$1 D_2 = 1$
3	0	1	1	$1 D_3 = 1$
4	1	0	0	$0 D_4 = 0$
5	1	0	1	$1 D_5 = 1$
6	1	1	0	$0 D_6 = 0$
7	1	1	1	$0 D_7 = 0$

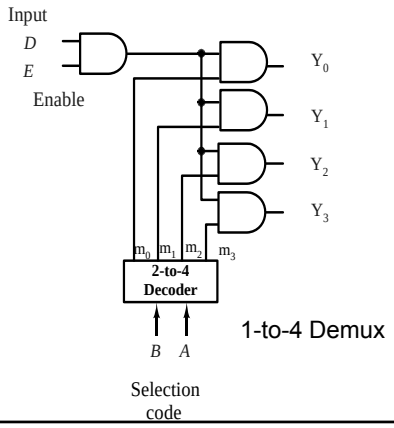


DEMULTIPLEXERS (DATA DISTRIBUTORS)

Demultiplexers

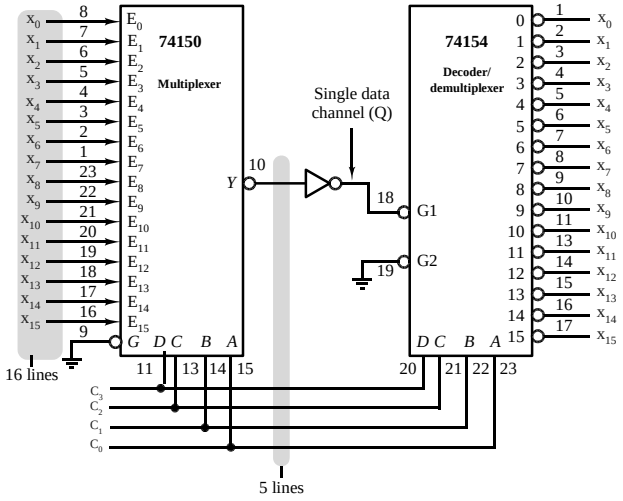
- A demultiplexer (Data Distributor) connects a single input line to one of n output lines, the specific output line being determined by an s-bit selection code where 2_s >= n.





1-to-4 Demux

Multiplexer-Demultiplexer System



35

ADDERS

36

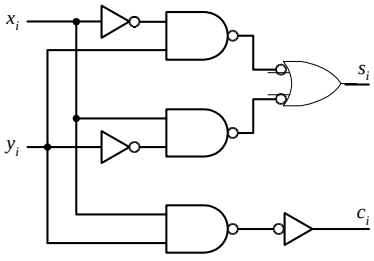
Adding Binary (encoded) Numbers

- Addition can be performed with the same old algorithm we learned in elementary schools. Go from LSB to MSB and keep track of the carry.
- If we need to add more than two numbers we can divide that up into adding two and then adding the third to the sum of the two, etc.
- If we are adding two binary numbers, then the carry can only be 0 or 1.

Binary Half-adder

- When adding the LSBs, no carry is taken as input. The circuit realizing this is a half-adder.

x_i	y_i	c_i	s_i
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0



Half adder propagation delays I nthe above configuration are not uniform

$t_{add} = 3\ t_{gate}$

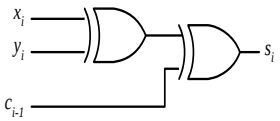
$t_{carry} = 2\ t_{gate}$

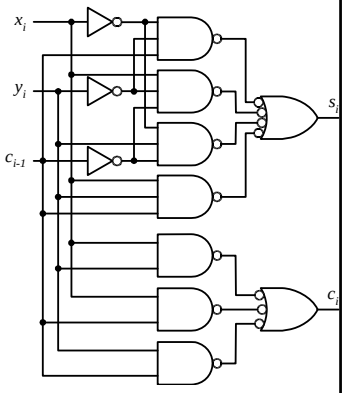
Full-Adder

- In the next bits, we need to have a third input: the carry-in.
- From the truth table (shown in fig. 4.35e) we can show that $s_i = x_i \oplus y_i \oplus c_{i-1}$

$c_i = x_i y_i + x_i c_{i-1} + y_i c_{i-1}$

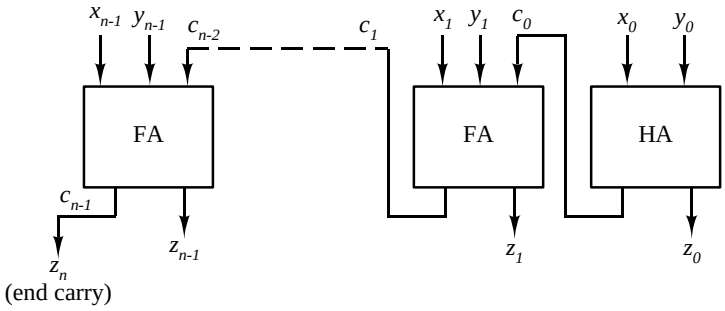
x_i	y_i	c_{i-1}	c_i	s_i
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1





Ripple Carry Adder

- Here is our "algorithm":



Half adder propagation delays

$t_{add} = 3\ t_{gate}$

$t_{carry} = 2\ t_{gate}$

Full adder propagation delays

$t_{add} = 3\ t_{gate}$

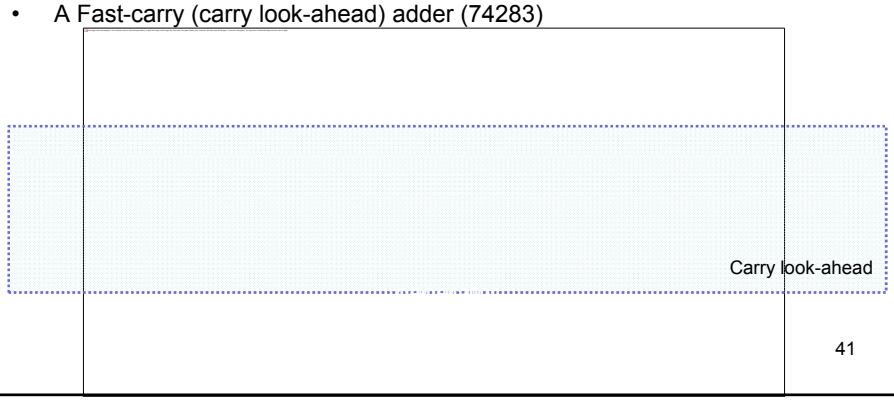
$t_{carry} = 2\ t_{gate}$

Ripple-Carry Adder (n-bits)

$t_{add} = (n - 1)2\ t_{gate} + 3\ t_{gate} = (2n + 1)\ t_{gate}$

Parallel Adders and Fast-Carry

- If we are not using our modular algorithm, faster adders can be created (parallel adders). The fastest adder design would be strictly parallel. That is, all the inputs would be applied simultaneously and propagate through two levels of logic to obtain the result. However, this approach is unrealistic for large number of bits because of the fan-in requirements.
- A Fast-carry (carry look-ahead) adder (74283)



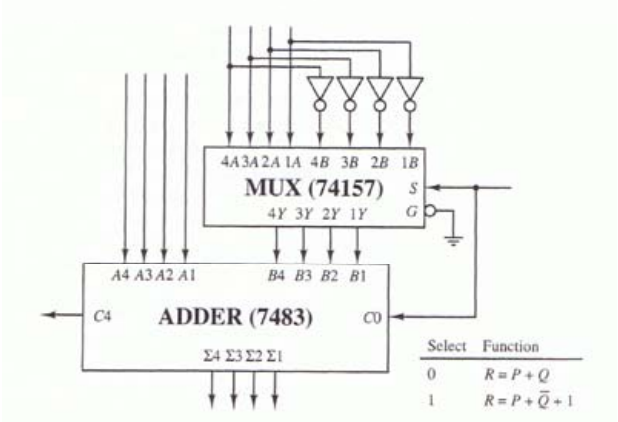
41

Subtraction

- Recall: If we encode negative numbers in the right way (using the overflow of adders) then subtraction becomes the addition of a number and a negative number.
- Two-s complement is the solution. Two-s complement can be easily generated by flipping each bit and adding one to the result.

42

4-bit Adder/Subtractor Circuit



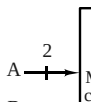
43

COMPARATORS

44

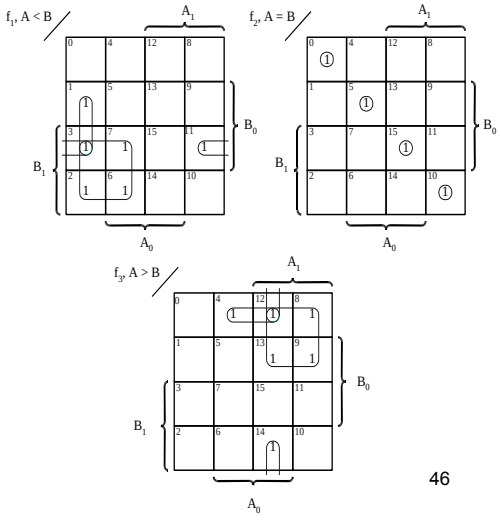
Comparators

- In general, a comparator can perform a magnitude comparison of two words A and B in either straight binary or BCD codes
- Thus, a comparator will generate three output signals:
  - $f_1 = 1$ if A < B
  - $f_2 = 1$ if A = B
  - $f_3 = 1$ if A > B
- Therefore, a comparator is a 2n-input, 3-output combinational logic module.

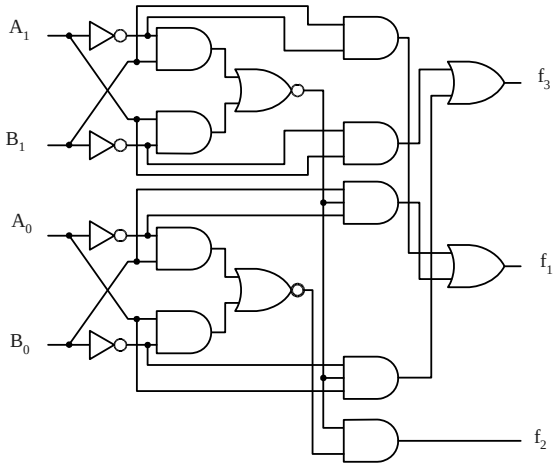


Designing a 2-bit Comparator

i	A_1	A_0	B_1	B_0	f_1	f_2	f_3
0	0	0	0	0	0	1	0
1	0	0	0	1	1	0	0
2	0	0	1	0	1	0	0
3	0	0	1	1	1	0	0
4	0	1	0	0	0	0	1
5	0	1	0	1	0	1	0
6	0	1	1	0	1	0	0
7	0	1	1	1	1	0	0
8	1	0	0	0	0	0	1
9	1	0	0	1	0	0	1
10	1	0	1	0	0	1	0
11	1	0	1	1	1	0	0
12	1	1	0	0	0	0	1
13	1	1	0	1	0	0	1
14	1	1	1	0	0	0	1
15	1	1	1	1	0	1	0



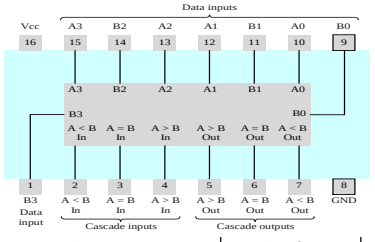
Realizing a 2-bit Comparator



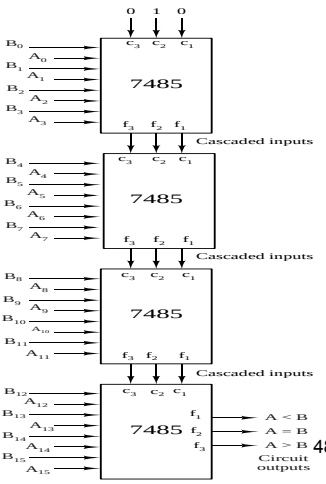
47

7485 MSI Comparator

- The 7485 module is a 4-bit magnitude comparator (can be used to cascade higher order, e.g., 16-bit magnitude comparators)



Comparing inputs				Cascading inputs			Outputs		
A3, B3	A2, B2	A1, B1	A0, B0	A > B	A < B	A = B	A > B	A < B	A = B
A3 > B3	×	×	×	×	×	×	H	L	L
A3 < B3	×	×	×	×	×	×	L	H	L
A3 = B3	A2 > B2	×	×	×	×	×	H	L	L
A3 = B3	A2 < B2	×	×	×	×	×	L	H	L
A3 = B3	A2 = B2	A1 > B1	×	×	×	×	H	L	L
A3 = B3	A2 = B2	A1 < B1	×	×	×	×	L	H	L
A3 = B3	A2 = B2	A1 = B1	A0 > B0	×	×	×	H	L	L
A3 = B3	A2 = B2	A1 = B1	A0 < B0	×	×	×	L	H	L
A3 = B3	A2 = B2	A1 = B1	A0 = B0	H	L	L	H	L	L
A3 = B3	A2 = B2	A1 = B1	A0 = B0	L	H	L	L	H	L
A3 = B3	A2 = B2	A1 = B1	A0 = B0	L	L	H	L	L	H
A3 = B3	A2 = B2	A1 = B1	A0 = B0	×	×	H	L	L	H
A3 = B3	A2 = B2	A1 = B1	A0 = B0	H	H	L	L	L	L
A3 = B3	A2 = B2	A1 = B1	A0 = B0	L	L	L	H	H	L



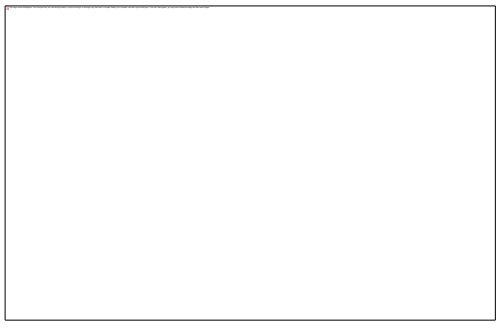
48

ARITHMETIC LOGIC UNITS

49

ALUs

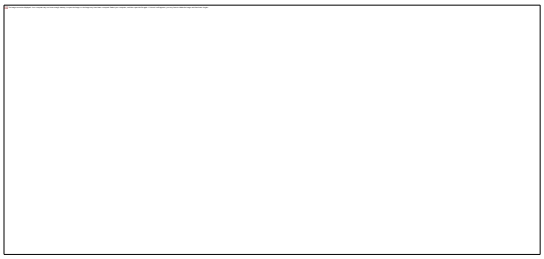
- An Arithmetic Logic Unit is used in most processors to carry out logic and arithmetic operations on two numbers.



50

Building a 32-bit ALU

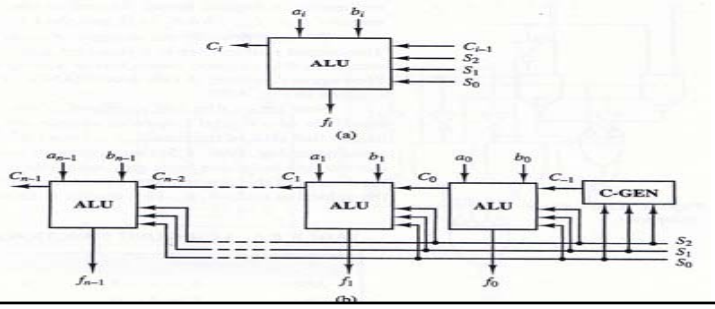
- Let us design a simple, modular 32-bit ALU with four logic and four arithmetic operators. Thus we need 3 bits for operation selection.



51

Building a 32-bit ALU

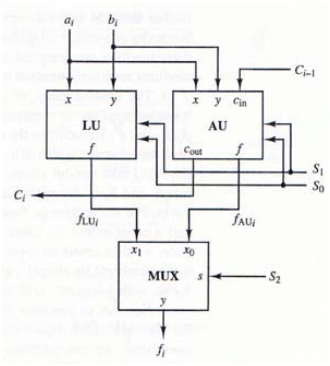
- We can make this ALU modular in several ways. Let us make it modular by focusing on the i^{th} digit in the operand and make a module for that (then copy this 32 times).



52

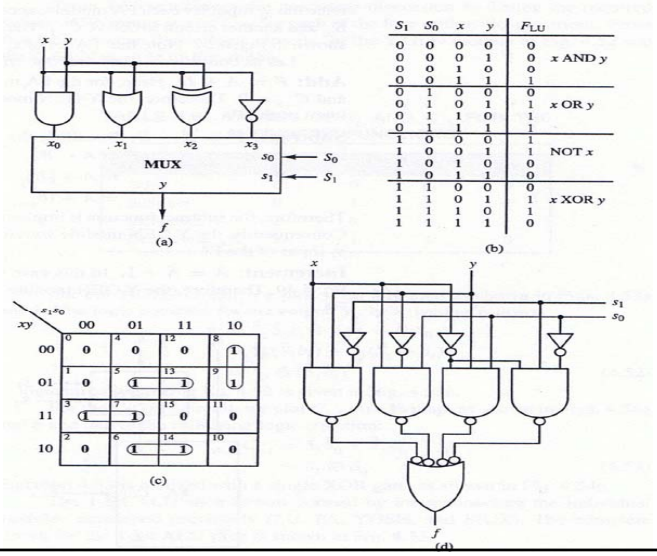
Inside the 1-bit ALU

- Let us divide up the operations into a logic unit (LU) and into an arithmetic unit (AU).



53

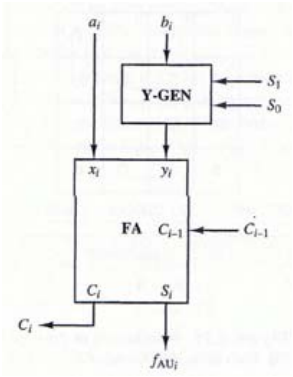
Inside the 1-bit LU



54

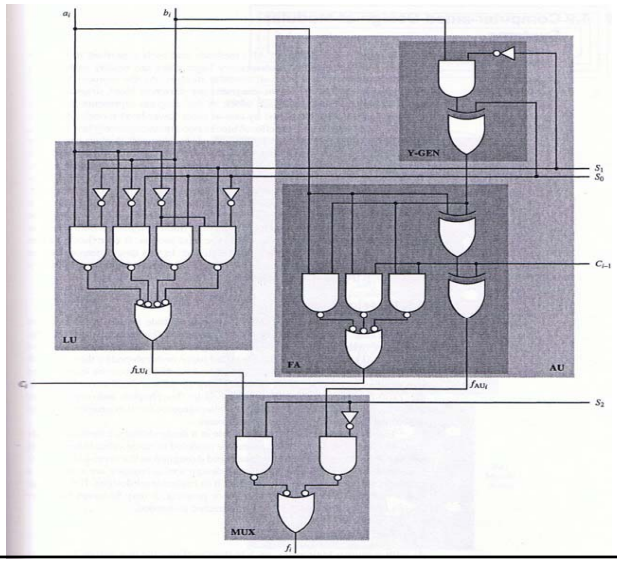
Inside the 1-bit AU

- Recall our discussion on subtraction



55

The Complete 1-bit ALU



56