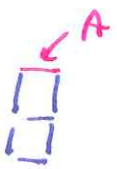
Circle blocks of 2^N 1s & Xs until every 1 covered.
The bigger the blocks the better.

Eg: A segment of seven segment display



A	B	C	D	a
0	0	0	0	1
0	0	0	1	0
0	0	1	0	1
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	1 ← sometimes 0
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
above				X

Gray Code order

CD \ AB	00	01	11	10
00	1	0	X	1
01	0	1	X	1
11	1	1	X	X
10	1	1	X	X

$\bar{B}\bar{D} + A + C + DB$

Remark: $\bar{B}\bar{D} + BD = \overline{B \oplus D}$
for which an IC exists

$\bar{a} = \bar{A}\bar{B}\bar{C}D + \bar{A}\bar{B}C\bar{D}$
$a = (A+B+C+\bar{D})(A+B+\bar{C}+\bar{D})$ ← maxterm

Eg segment b of seven segment

always ON except for 5 & 6

ABCD

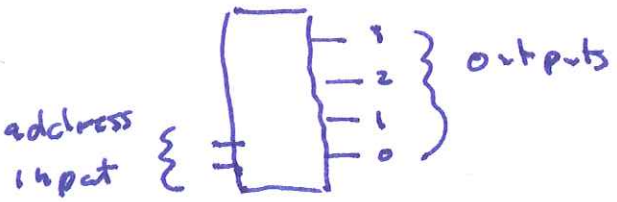
CD \ AB	00	01	11	10
00	1	1	X	1
01	1	0	X	1
11	1	1	X	X
10	1	0	X	X

$\bar{B} + CD + \bar{C}\bar{D}$

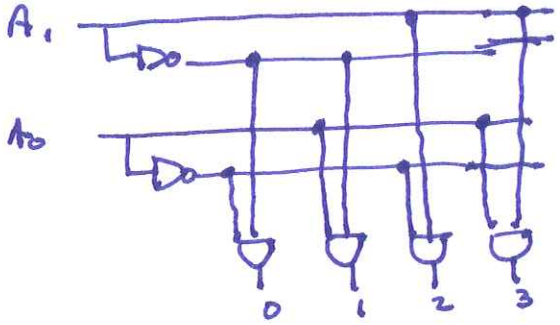
Standard Packaged Functions

→ note previously discussed adders; there are other types of arithmetic chips that would compare magnitude of two integers & more complex chips that could add/subtract/multiply/divide numbers in scientific notation ("floats") → Floating Point Processors

Decoder: given a parallel binary number "address" select (eg produce a H) the output that corresponds to that number. Eg a 2 bit address could select one of four outputs

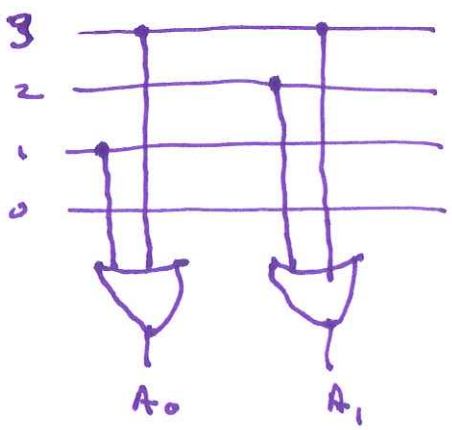


A_1 A_0	0	1	2	3
0 0	1	0	0	0
0 1	0	1	0	0
1 0	0	0	1	0
1 1	0	0	0	1



Remark: a decoder might be "active low" i.e. normal (unselected) output would be H and the selected output would be L

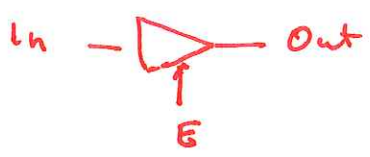
Encoder: given 2^N input lines for which it is known only one will be H, produce an output address (N bits) for the single H input [Truth Table is exactly as Decoder but with inputs/outputs swapped]



Remark: "Priority Encoder" – deals with case where more than one input might be H: produces the address of highest number input that is H

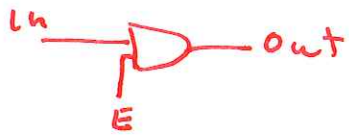
Multiplexer (aka Mux, input selector): given 2^N inputs and an N bit parallel binary number ("address") connect the input named by the address and connect it to the single output: Given many in and one out control which input controls the output

Needed: an enabled buffer → when enable is H output follows input; when enable is L output is L (independent of input)



In	E	out
x	0	0
0	1	0
1	1	1

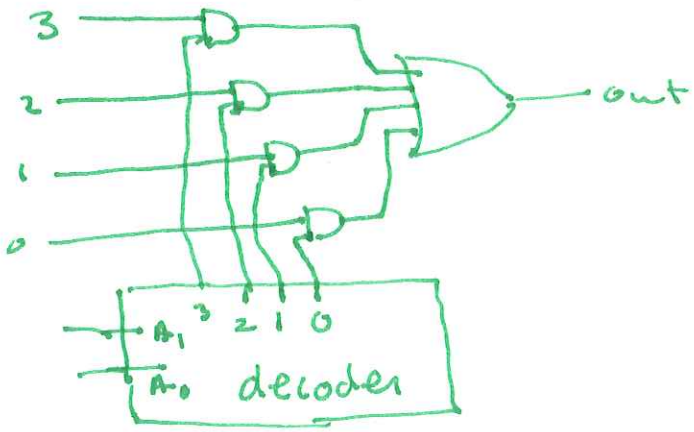
=



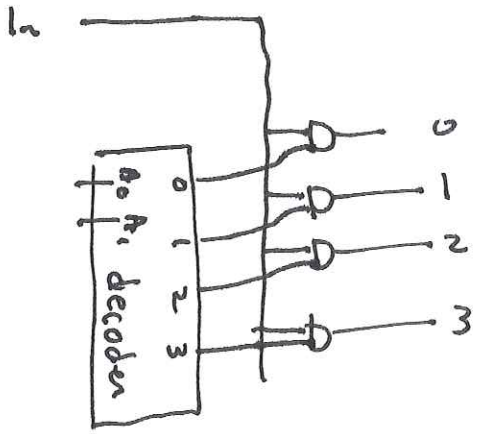
Remark: other simple gates may be considered enabled buffers/inverters. Eg NOR = active low enabled inverter

XOR: buffer or inverter as you command

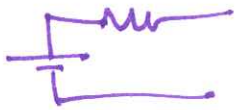
Remark2: For analog signals there are devices that act like a digitally controlled switch —/o— called "transmission gates" or "analog switch" they often have more limited function compared to physical switch (eg: only work with ⊕ voltages, have resistances >10Ω etc)



De-Mux: send the one input to a selected output



A few words about input/output characteristics of real gates. In Analog Electronics we'll prove "Thevenin's Thm" which says (exactly for "linear" components like ideal LRC and approximately for more complex circuits) that an output always acts as if it were a battery in series with a resistor. (The resistance of the equivalent resistor is called the output impedance - it shows how much voltage will be "lost" as you pull increasing current from the output)

So a H output looks like 

and a L output looks like 

The values of these Rs depend on the circuit used to implement the logic - that is reported in the letters following 74 as in 74LS32, 74F138, 74AC08 etc. Think about Rs ~ 1kΩ so only a few mA of current can be controlled by a simple gate - not enough to do much real work

the input to TTL looks approximately like:

As a result if an input is held L it will source a current (typically a fraction of a mA) whereas if held H much smaller currents will flow. Note that an unconnected input (0 current flow) is much like (will act like) a H input.