Physics 123 Lecture Notes

Peter York

Summer 2019

1.1 DC Circuits

Circuit = Ckt
Parallel = ||

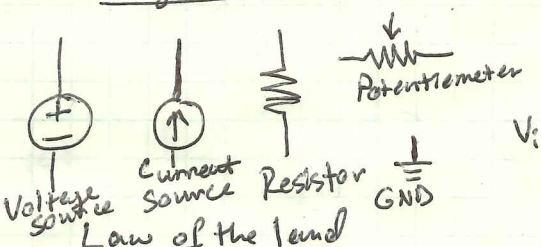
Circuit Sandbox

Key Terms "language of the land"
Voltage
Current
Resistance
Parallel & Series

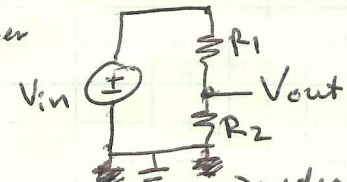
Combining Resistors
R in series add
R in || => $R_{eq} = \frac{1}{\frac{1}{R_1} + \frac{1}{R_2}}$

Rules for creating a function ckt
1) You need a closed path for current to flow
2) No voltage sources in parallel!

Toys (ckt element)



Key Circuits



Voltage Divider
Strong Suggestion
$R_{out} < 10R_{in}$

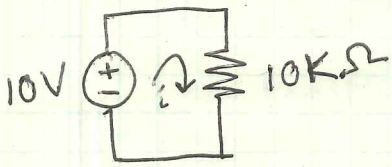
Law of the land
1) Ohm's Law $V = IR$
2) KCL current in = current out
3) KVL sum of voltage drops in ckt is zero

Voltage: A "Pressure" that causes current to flow $[V]$

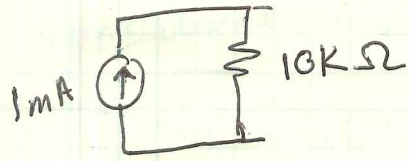
Current: Flow of charge past a point per unit time $[A] = \left[\frac{C}{s}\right] = \left[\frac{J}{C}\right]$

Resistance: A measure of the voltage needed to make a given current flow: $[\Omega] = \left[\frac{V}{A}\right]$

Example Circuits



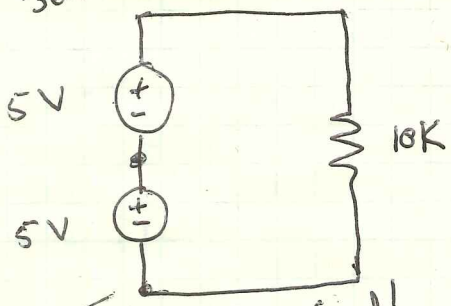
$$i = \frac{V}{R} = \frac{10V}{10K\Omega} = 1mA \quad 0.001A$$



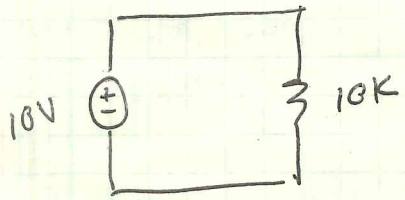
$$V = iR = (1mA)(10K) = 10V$$

Combining Sources

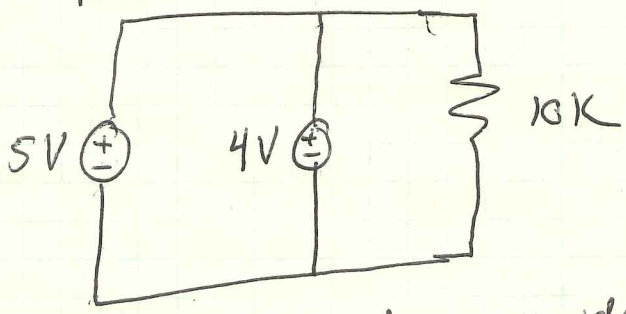
"series connection"



= Good!



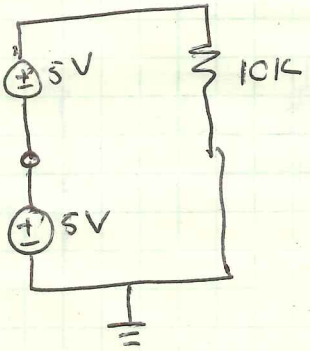
"Parallel connection"



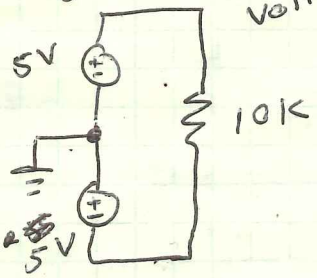
This is a dangerous idea!

Power dissipation in the resistor $P = 10V \cdot 1mA = 20mW$

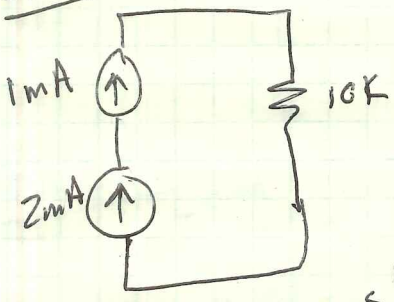
GND as Most Negative voltage



GND as Center voltage

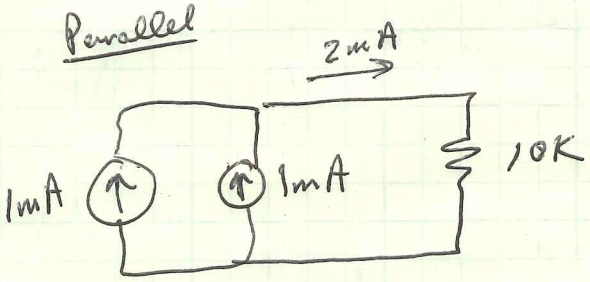


Series



This is dangerous!

Parallel



This is good

Considerations for Resistors

1) Tolerancing
2) Temperature
3) Power (Typical is 1/4 W resistor) 250 mW

$P = IV = I^2R$

Power

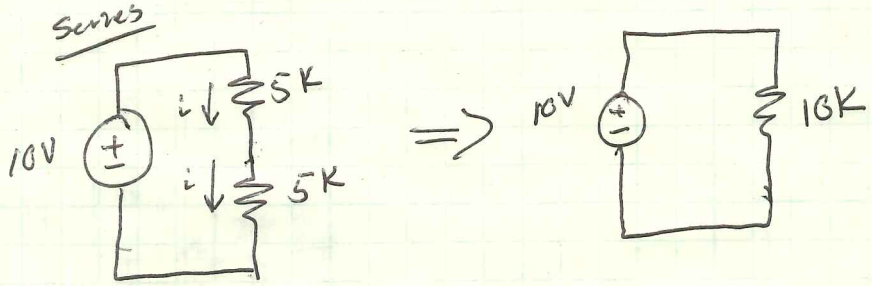
~~$P = [E][I]$~~ $P = I \cdot V$

$= \left[\frac{C}{S}\right]\left[\frac{J}{C}\right]$

$= \left[\frac{J}{S}\right] = [W]$

Combining Resistors

Series

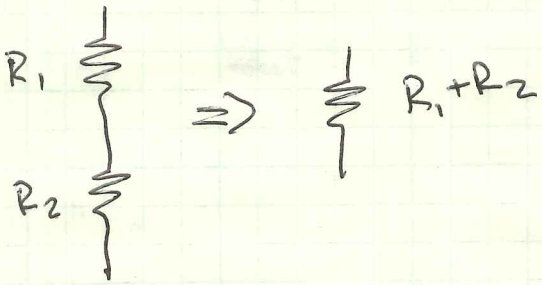


$10V = i5K + i5K$

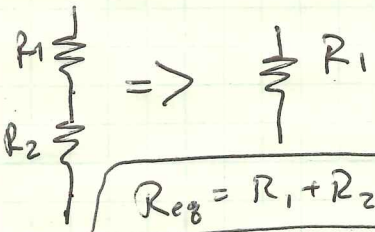
$10V = i(10K)$

~~Resistors~~

Extreme cases



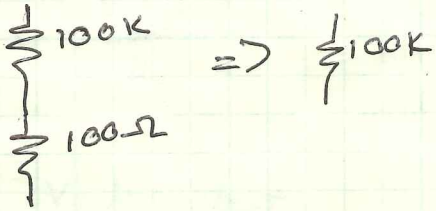
what if $R_1 >> R_2$



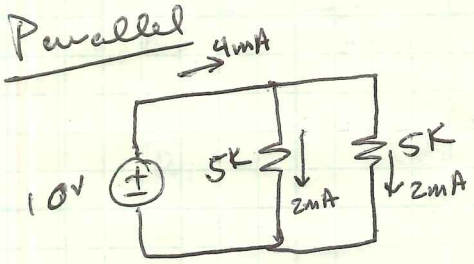
$R_{eq} = R_1 + R_2$

In series, big resistor dominates

For example



Parallel



$V = IR$

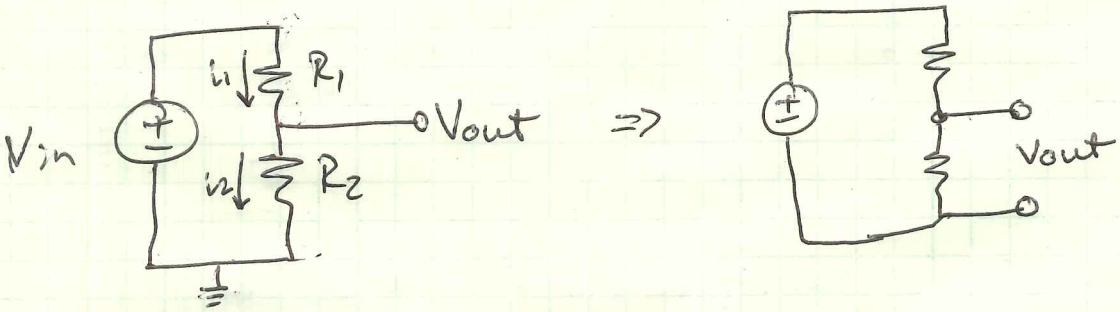
$$R_{eq} = \frac{10V}{i_{tot}} = \frac{10V}{4mA} = 2.5K$$

$$R_{eq} = \frac{10V}{\frac{10V}{5K} + \frac{10V}{5K}} = \frac{1}{\frac{1}{5K} + \frac{1}{5K}}$$

$$R_{eq} = \frac{1}{\frac{1}{R_1} + \frac{1}{R_2}}$$

In parallel, small resistor dominates

Voltage Dividers



Extreme Cases

$R_1 >> R_2$ $\quad V_{out} \approx 0$

$R_2 >> R_1$ $\quad V_{out} \approx V_{in}$

$R_2 = R_1$ $\quad V_{out} = \frac{V_{in}}{2}$

$i_1 = i_2$

$$\frac{V_{in} - V_{out}}{R_1} = \frac{V_{out}}{R_2}$$

$$\frac{V_{in}}{R_1} = \frac{V_{out}}{R_1} + \frac{V_{out}}{R_2}$$

$$\frac{V_{in}}{R_1} = V_{out}\left(\frac{1}{R_1} + \frac{1}{R_2}\right)$$

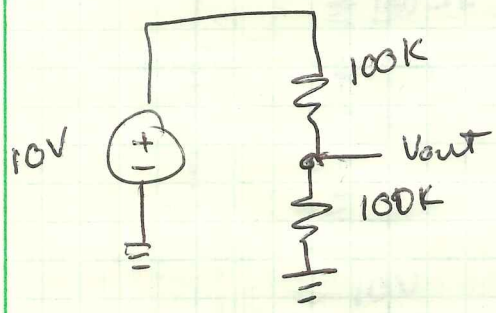
$$\frac{V_{in}}{R_1} = V_{out}\left(\frac{R_2 + R_1}{R_1 R_2}\right)$$

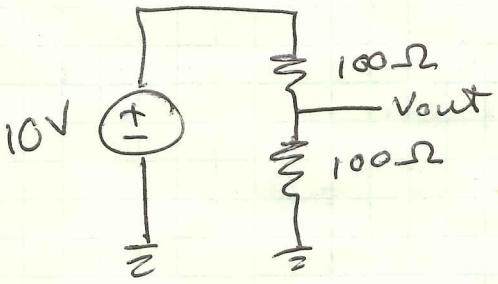
General Rule

$$\boxed{V_{out} = V_{in}\frac{R_2}{R_1 + R_2}}$$

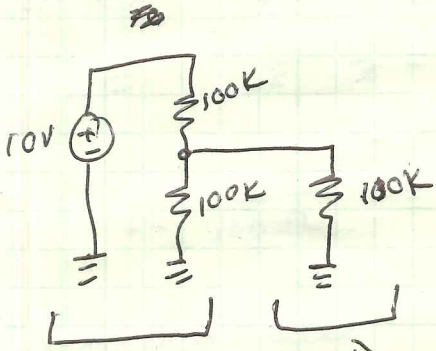
Extreme cases check out

What is the difference between:

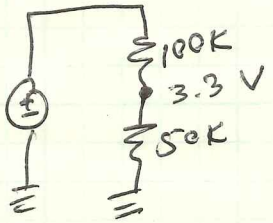




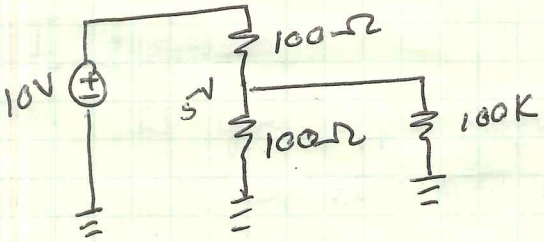
These ckts look different to the outside world!



⇒



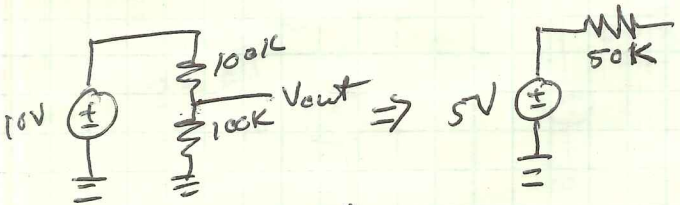
No longer 5V!



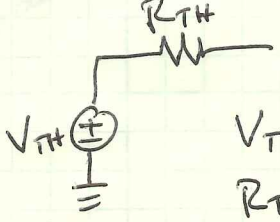
✓ Still delivers 5V to the load

Thevenin Equivalent Circuits

You can represent any network of voltage sources and resistors as a single voltage source in series with a resistor



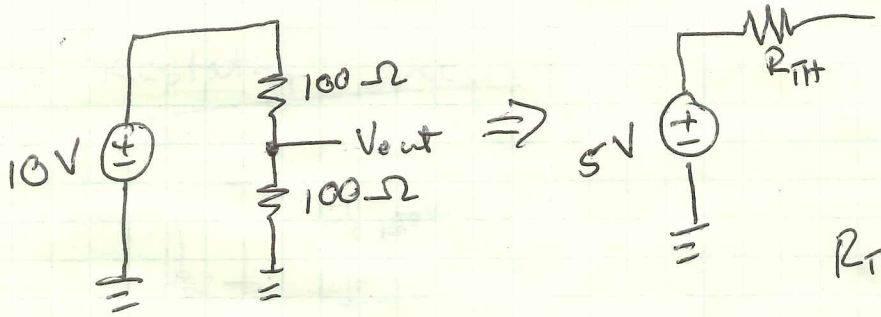
$i_{sc} = \frac{10V}{100K} = 0.1mA$



$V_{TH} = V_{open\ circuit}$

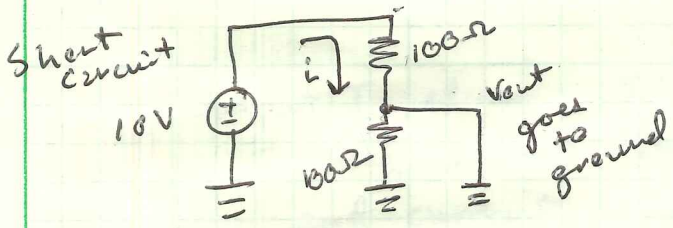
$R_{TH} = \frac{V_{open\ circuit}}{i_{short\ circuit}}$

$= \frac{5V}{0.1mA} = 50K$



$$R_{TH} = \frac{V_{oc}}{i_{sc}} = \frac{5V}{0.1A} = 50\Omega$$

$$i_{sc} = \frac{10V}{}$$



$$i_{sc} = \frac{10V}{100\Omega} = 0.1A$$

R_{TH}

General Rule: $R_{out} < 10 \cdot R_{LOAD}$ to not disrupt the behavior of the upstream circuit

Impedence:

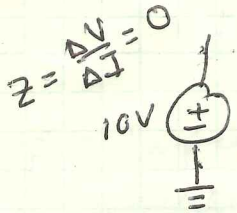
If you apply some wiggle ΔV, what change in current ΔI, will the circuit respond with?

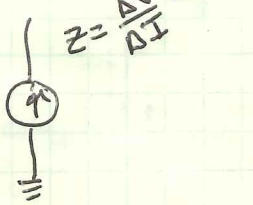
$Z = \frac{\Delta V}{\Delta I}$ is the impedence at that point

$\Delta V = \Delta I R$

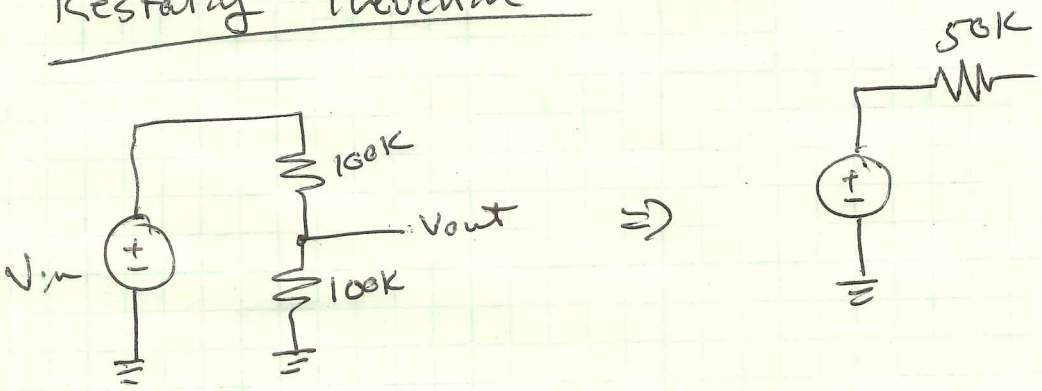
Example:

$$Z = \frac{\Delta V}{\Delta V/R} = R$$

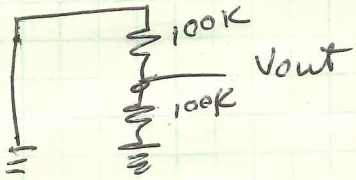


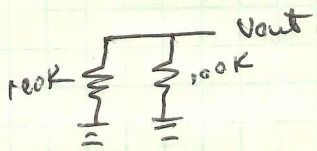


Restating Thevenin



Output Impedence: Resistance to ground with voltage sources short circuited (replaced by wires) and current sources open circuited)





$$R_{eq} = \frac{1}{\frac{1}{100K} + \frac{1}{100K}} = 50K$$

Equivalent to R_{TH}

Input Impedence: Resistance from input to ground
"LOAD Resistance"

Input Impedence should be 10x the output impedence

Capacitors!

$$Q = CV$$

$$\frac{dQ}{dt} = C\frac{dV}{dt}$$

$$\Rightarrow i = C\frac{dV}{dt}$$

In series: C_1, C_2 = C_{eq}

$$\frac{Q}{C_1} + \frac{Q}{C_2} = \frac{Q}{C_{eq}}$$

$$\frac{1}{C_1} + \frac{1}{C_2} = \frac{1}{C_{eq}}$$

In parallel: C_1, C_2 = C_{eq}

$$C_{eq} = C_1 + C_2$$

$$Q_1 + Q_2 = Q_{EQ}$$

$$C_1 V + C_2 V = C_{eq} V$$

$$C_1 + C_2 = C_{eq}$$

Parallel Plate Capacitor

$$C = \frac{\varepsilon_0 \varepsilon_r A}{d}$$

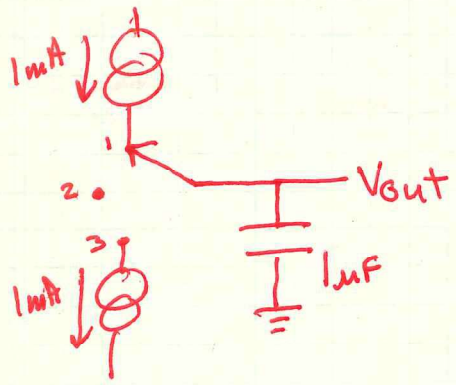
ε_0 ~ permittivity, ε_r ~ dielectric strength, A ~ area, d ~ plate separation

Nomenclature

		Example
Picofarads	10^{-12}	"101" = $10 \cdot 10^1$ pF
microfarads	10^{-6}	"0.001" = 0.001 μF = 1 nF
		$104 = 10 \cdot 10^4$ pF = 0.1 μF

Electrolytic >1μF ⇒ Polarity matters! Polarity matters! Negative side marked with a bar.

Time Domain Capacitor Behavior

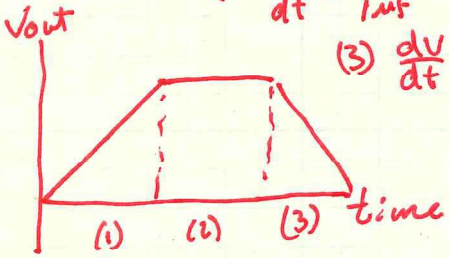


$$i = C\frac{dV}{dt}$$

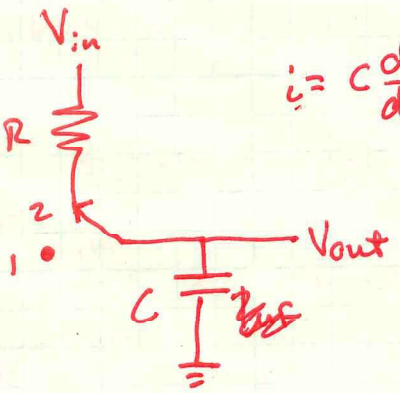
(1) $\frac{dV}{dt} = \frac{1mA}{1\mu F} = 1000\frac{V}{s}$

(2) $\frac{dV}{dt} = \frac{0}{1\mu F} = 0$

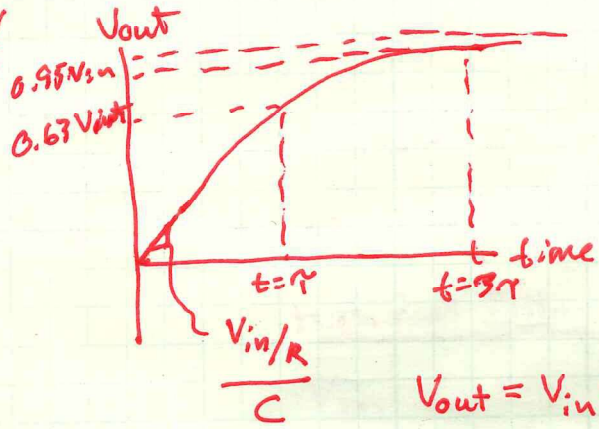
(3) $\frac{dV}{dt} = -1000\frac{V}{s}$



Replace the current source w/ a more useable one:



$i = C\frac{dV}{dt}$



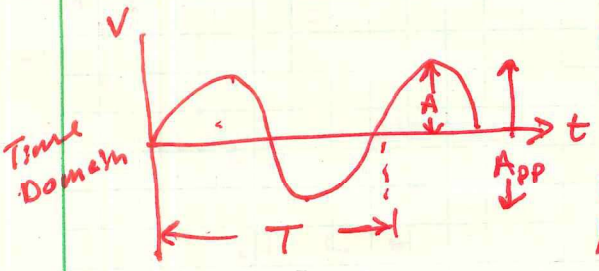
$V_{out} = V_{in}(1 - e^{-t/RC})$

Frequency Domain view of Capacitors

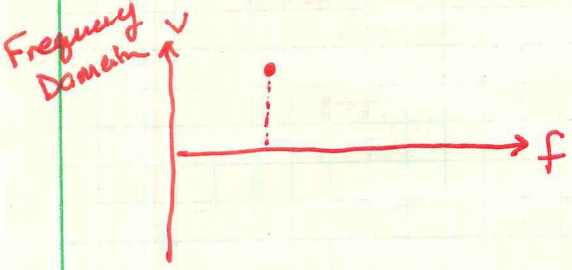
extreme cases:
- $t = 0 \Rightarrow V_{out} = 0$
- $t \to \infty$ $V_{out} = V_{in}$
- $t = RC$ $V_{out} = 0.63\, V_{in}$

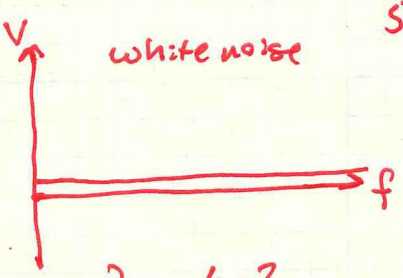
τ "time constant"

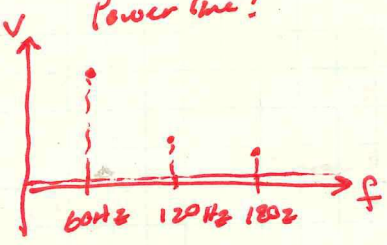
τ : 63%
3τ : 95%
5τ : 99%



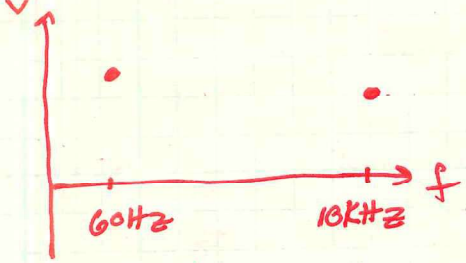
Amplitude (A)
Amplitude (App) (peak to peak)
Period (T)
Frequency ($f = 1/T$)







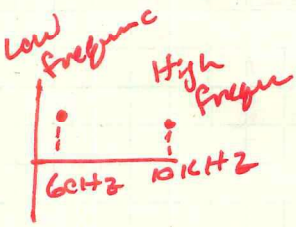
Question: If you are given the following signals:



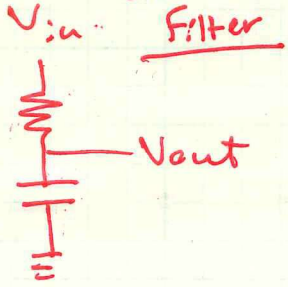
How do you keep one signal and attenuate the other?

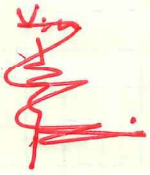
This motivates filtering!

filtering = keeping signals that you want and attenuating noise or otherwise unwanted signals

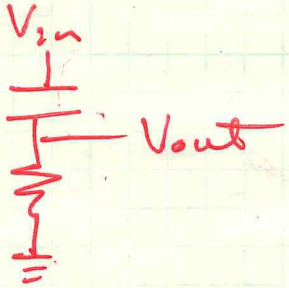


Low Pass Filter





High Pass filter

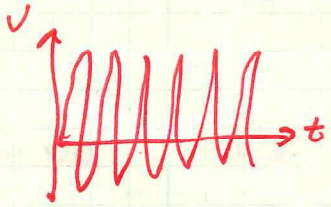


At high frequencies $\frac{1}{T} \Rightarrow$ |

At low frequencies $\frac{1}{T} \Rightarrow$ 0

$$i = C\frac{\Delta V}{\Delta t} \quad \text{or} \quad i = C\frac{dV}{dt}$$

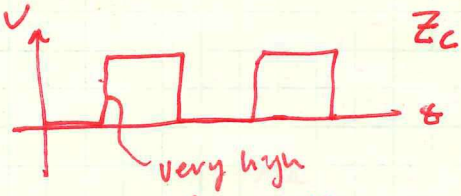
Think about a high frequency signal:



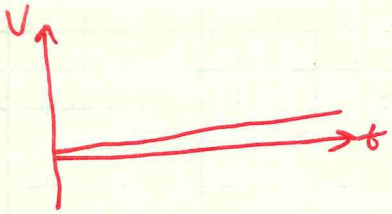
$\frac{dV}{dt} \Rightarrow$ very large $\Rightarrow$ i is also very large

Z_C is small

extreme case: Square wave



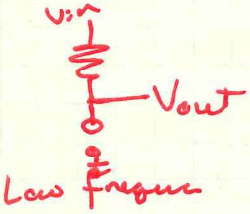
For a low frequency signal:

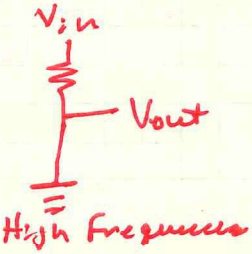


$\frac{dV}{dt} \Rightarrow$ very small $\Rightarrow$ i is also very small

Z_C is large

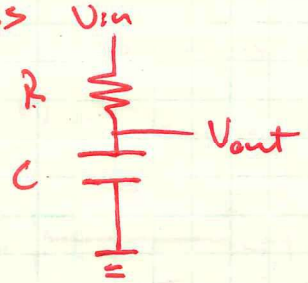
Redraw Low Pass for extreme cases





Quantifying Filter Behavior

Low Pass



$Z_C = \frac{1}{j\omega C}$ ← "Complex Impedance"

$j = \sqrt{-1}$ "i"

$\omega = 2\pi f$ $\left[\frac{rad}{s}\right]$

C: Capacitance in [F]

We can combine impedances just like resistors

$$\frac{V_{out}}{V_{in}} = \frac{\frac{1}{j\omega C}}{\frac{1}{j\omega C} + R}$$

Recall: $\frac{V_{out}}{V_{in}} = \frac{R_2}{R_2 + R_1}$

$$= \frac{1}{1 + j\omega \underbrace{RC}_{\tau}}$$

$$= \frac{1}{1 + j\omega\tau}$$

$\left|\frac{z_1}{z_2}\right| = \frac{|z_1|}{|z_2|}$

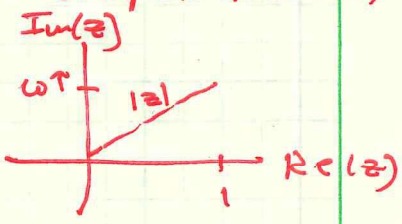
$$\left|\frac{V_{out}}{V_{in}}\right| = \frac{|1|}{|1 + j\omega\tau|} = \frac{1}{\sqrt{1 + \omega^2\tau^2}}$$

$j^2 = -1$

$$|z| = \sqrt{Re(z)^2 + Im(z)^2}$$

"Magnitude of a complex number"

$$\boxed{\left|\frac{V_{out}}{V_{in}}\right| = \frac{1}{\sqrt{1 + \omega^2\tau^2}}}$$



We care about trends. ← global

$$P_{dB} = 20\log_{10}\left(\left|\frac{V_{out}}{V_{in}}\right|\right)$$

$$= 20\log_{10}\left(\frac{1}{\sqrt{1 + \omega^2\tau^2}}\right)$$

Extreme cases

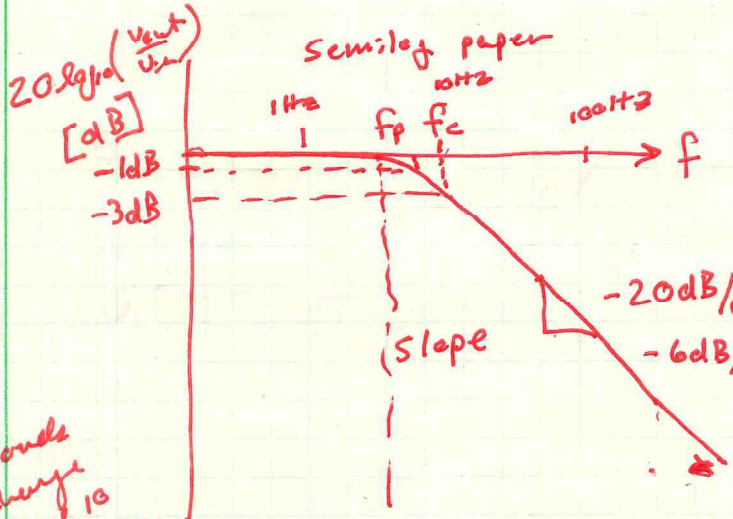
$\omega \to 0$ $\quad 20\log_{10} 1 = 0$ dB

$\omega \to \infty$ $\quad 20\log_{10}(\text{really small}) = -\infty$ dB

$\omega \to 1/\tau$ $\quad 20\log_{10}\left(\frac{\sqrt{2}}{2}\right) = -3$ dB

Use f_{3dB} and f_{-3dB} interchangably

$\omega_c \to 1/\tau$, $P_{dB} \to -3dB$ $\left|\frac{V_{out}}{V_{in}}\right| \to \frac{\sqrt{2}}{2} \approx 0.7$



"Bode plot"

$\omega = 2\pi f$

$\omega_c = 2\pi f_c = 1/\tau$

$f_c = \frac{1}{2\pi\tau} = \frac{1}{2\pi RC}$

$f_c = f_{3dB}$

Decade corresponds to one change in power of 10

10KHz → 100 KHz

or

100Hz → 1kHz

$f_p = \frac{f_c}{2} \Rightarrow \left|\frac{V_{out}}{V_{in}}\right| \approx 0.9 \Rightarrow -1dB$

p = pass

$\left|\frac{V_{out}}{V_{in}}\right| = \frac{1}{\sqrt{1+\omega^2\tau^2}}$

$\omega = \frac{1}{2\tau} \Rightarrow \frac{1}{\sqrt{1+\frac{1}{4\tau^2}\tau^2}} = \frac{1}{\sqrt{\frac{5}{4}}} = \frac{2}{\sqrt{5}} \approx 0.9$

Octave

Multiple of 2 of a signal

Check this!

Ideal filter would have very steep rolloff or slope

Slope: $\left|\frac{V_{out}}{V_{in}}\right| = \frac{1}{\sqrt{1+\tau^2\omega^2}}$ At high f, $\omega \gg 1/\tau$

$\approx \frac{1}{\sqrt{\omega^2}}$

$\approx \frac{1}{\omega}$

$P_{dB} = 20\log_{10}\left(\frac{1}{\omega}\right)$

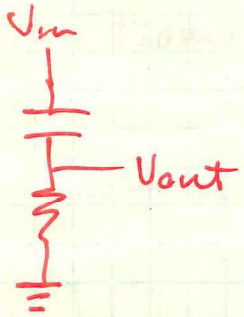
$= -20\log_{10}\omega$

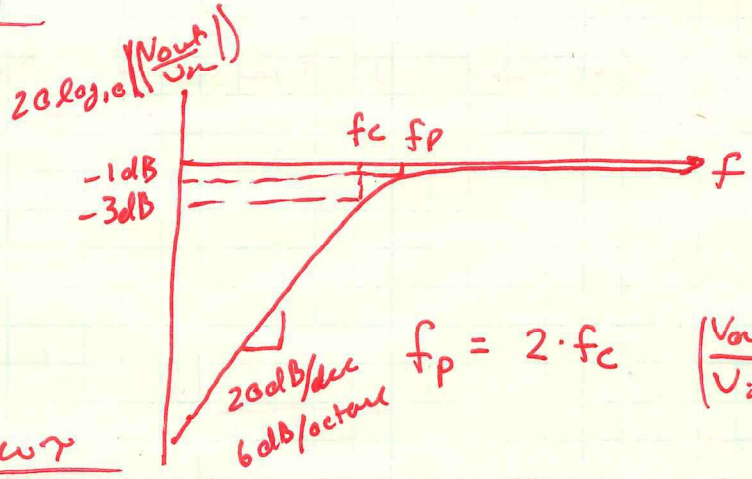
if $\omega = 10^2$ $P_{dB} = -40dB$

$\omega = 10^3$ $P_{dB} = -60dB$

$\omega = 10^4$ $P_{dB} = -80dB$

High Pass





$f_p = 2 \cdot f_c$ $\quad \left|\frac{V_{out}}{V_{in}}\right| \approx 0.9$

$$\left|\frac{V_{out}}{V_{in}}\right| = \frac{\omega\tau}{\sqrt{1+\omega^2\tau^2}}$$

$\omega \to 0$

$$20\log_{10}\left(\frac{\omega\tau}{\sqrt{1+\omega^2\tau^2}}\right) \Rightarrow -\infty$$

$\omega \to \infty$

$$20\log_{10}(1) \Rightarrow 0$$

$\omega \to 1/\tau$

$$20\log_{10}\left(\frac{\sqrt{2}}{2}\right) = -3dB$$

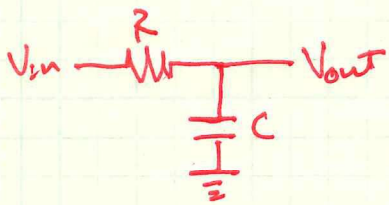
Impedances for filters

★ We want $Z_{in} > 10 Z_{out}$

By input impedance

★ Small output impedance

Low-Pass

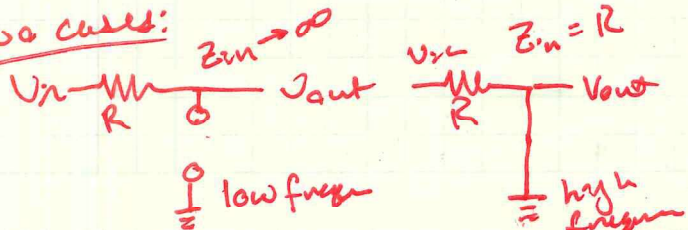


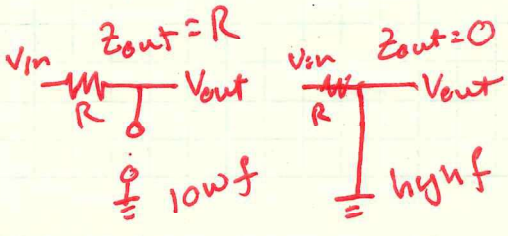
$Z_{in,\,worst\text{-}case} = R$

Recall $Z_C = \frac{1}{j\omega C} = \frac{1}{j2\pi f C}$

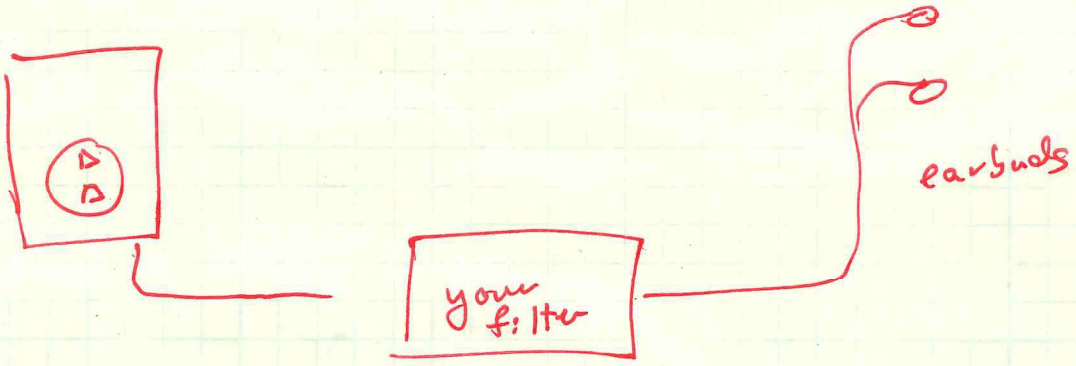
$Z_{out,\,worst\,case} = R$

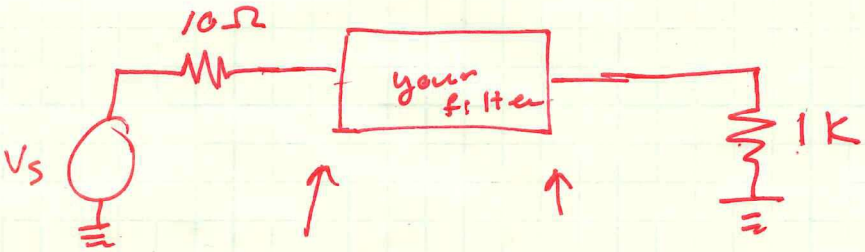
Two cases:





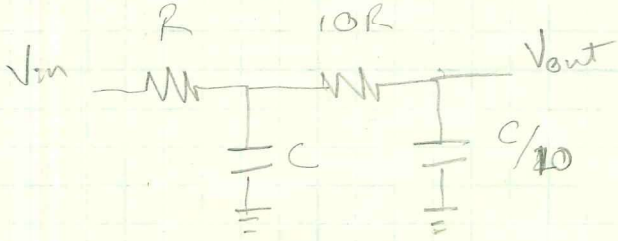
Homework 2 Q7 Guidance

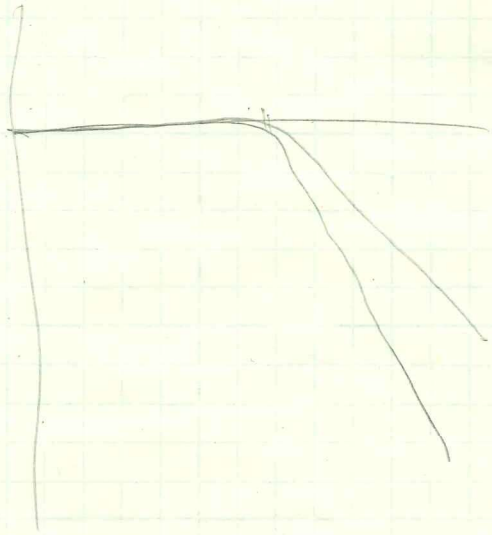




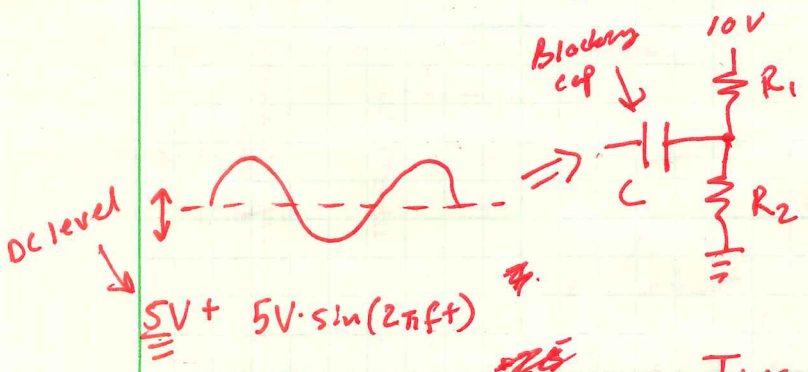
For both of these connections,

$Z_{in} > 10\, Z_{out}$





First, one more capacitor concept:



R_1 & R_2 are a voltage divider that set the new DC value of your signal

This is called "Biasing"

Important for transistor circuits!

To calculate the f_{3dB} of this circuit, use $R = R_1 \| R_2$

---

1) Inductors

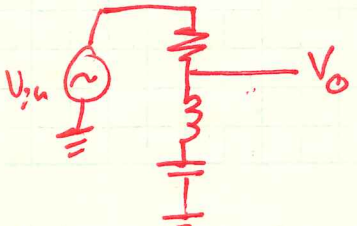
- Inductance : [H] μH and mH

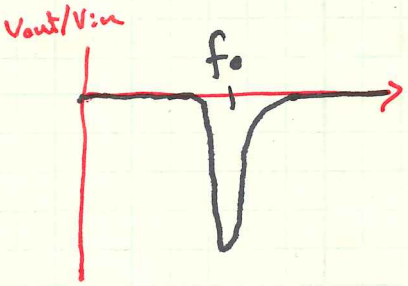
Capacitor vs Inductor

$i = C\frac{dV}{dt}$	$V = L\frac{di}{dt}$
can't change Voltage instantaneously	can't change current instantaneously
$Z_C = \frac{1}{jwC}$	$Z_L = jwL$ ~ inductance
High Frequency "wire"	"open circuit"
Low Frequency "open circuit"	"wire"

What can we do w/ inductors?

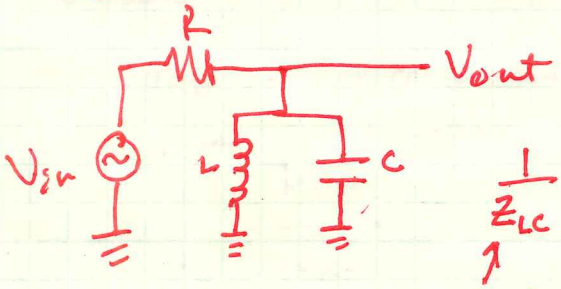
RLC Notch Filter





This filter attenuates a narrow range of frequencies

RLC Band Pass



$$\frac{1}{Z_{LC}} = \frac{1}{Z_L} + \frac{1}{Z_C} \Rightarrow Z_{LC} = \frac{j}{1/\omega L - \omega C}$$

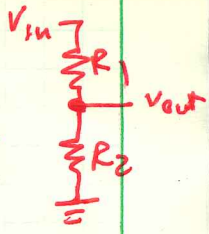
↑ equivalent impedance of cap and inductor in //

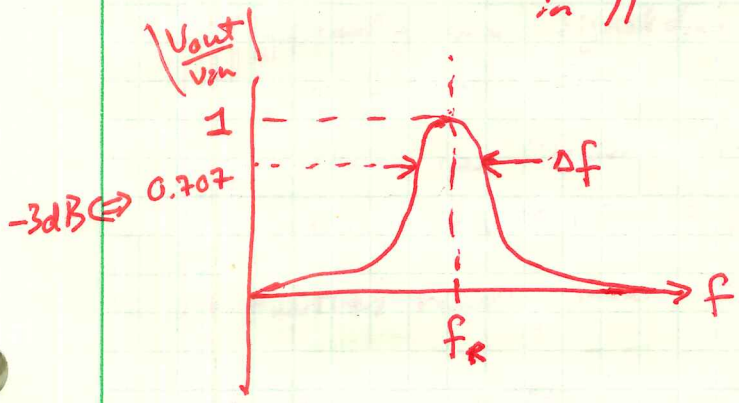
$Z_{LC} \to \infty$

$\omega = 2\pi f$

$$f_R \Rightarrow \frac{1}{2\pi\sqrt{LC}}$$

Recall: $\frac{V_{out}}{V_{in}} = \frac{R_2}{R_1 + R_2}$

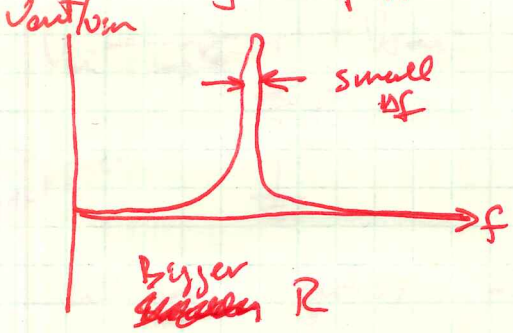




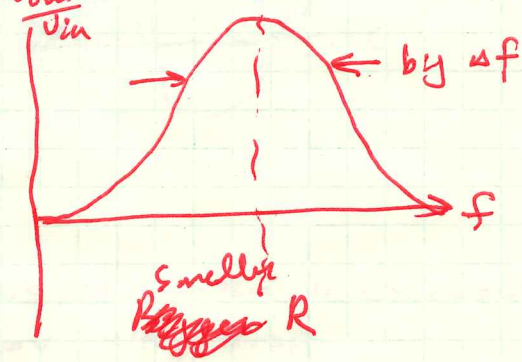
The shape of the response curve is captured by the "Quality Factor"

$$Q = \frac{f_R}{\Delta f}$$

High Q filter



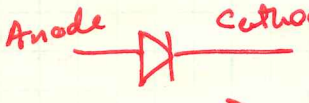
Low Q filter



$Q = 2\pi f_R RC$ for parallel RLC circuits

Diodes

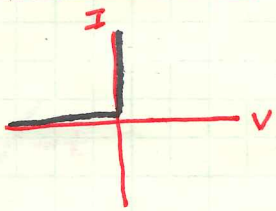
- Two terminal component



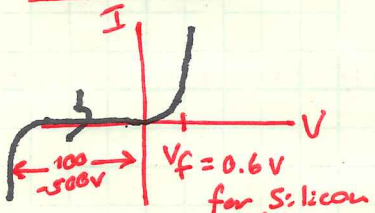
allows current flow in one direction!

- Typically use "signal diodes"
  Current rating ~100mA
- Sometimes use "power diodes"
  Current rating ~1A and up
- Further reading look PN junction

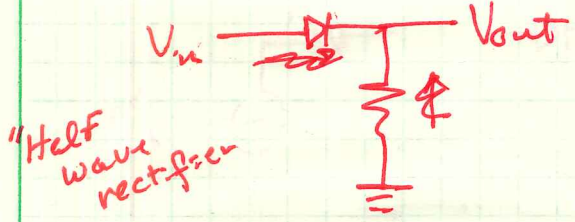
Perfect diode

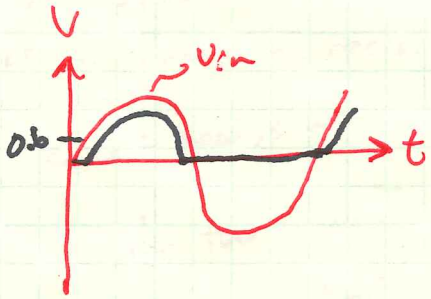


In Reality



Example Circuits





Whenever your diode is conducting, think about as a wire with a 0.6V voltage drop.

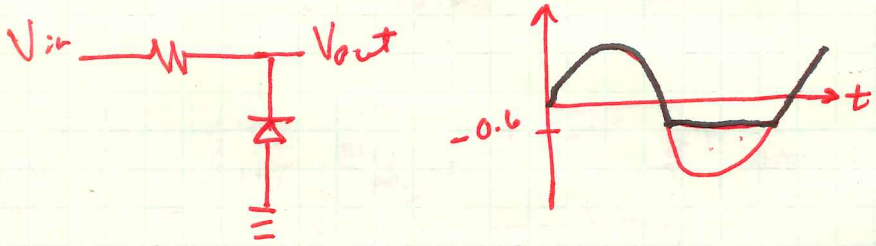
V_f

If $V_{in} = 0.5$ V, V_{out} would need to go to -0.1 V, and then we would violate Kirchoff's current Law

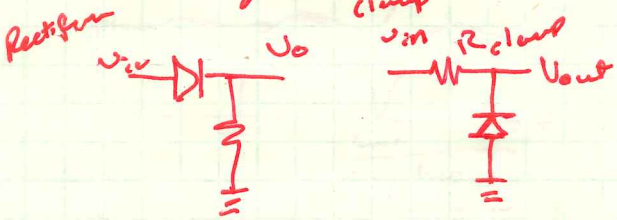
Transition point $\Rightarrow V_{in} = 0.6$ V

Think about as "Saturation".

"Clamp"



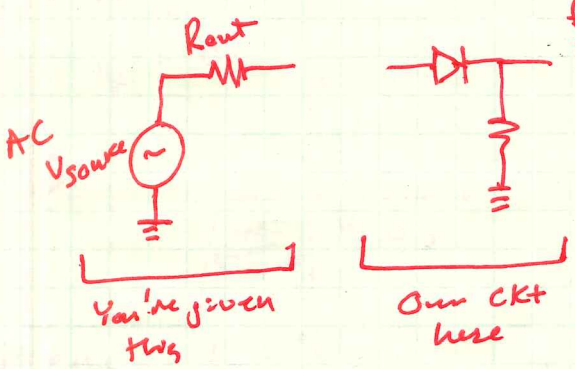
A subtle question: How to choose between rectifier & clamp?



We care about the time when V_{out} tracks V_{in}

Rectifier: conduction across a diode

Clamp: conduction across a resistor

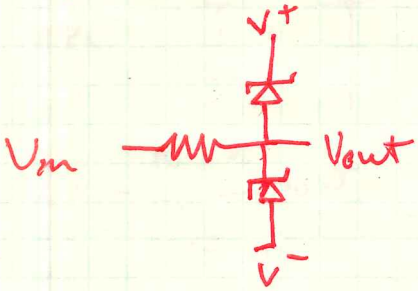


for rectifier
$R_{out} = R_{out,source}$

For the clamp,
$R_{out} = R_{out,source} + R_{clamp}$

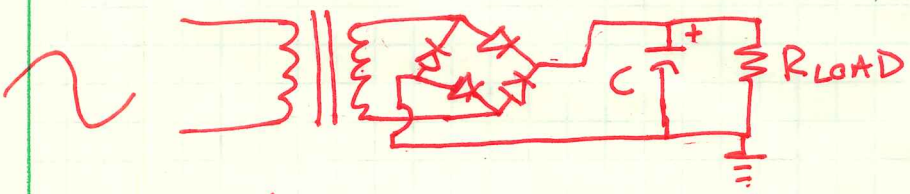
For clamp, can you just eliminate resistor and rely on R_{out} of source?

You can double clamp!

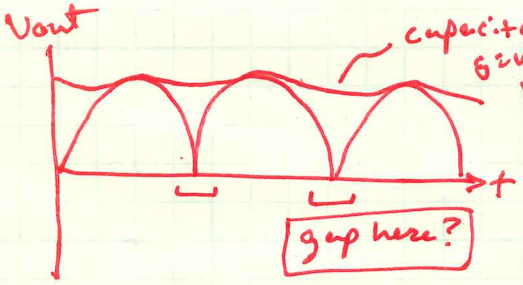


One last diode circuit

Full-wave bridge rectifier



- Trace full path for current
- conducts for both pos + neg inputs



→ This is a wavy but acceptable DC signal

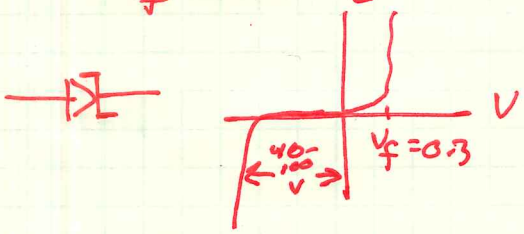
Waviness is called "ripple"

How to further reduce ripple?

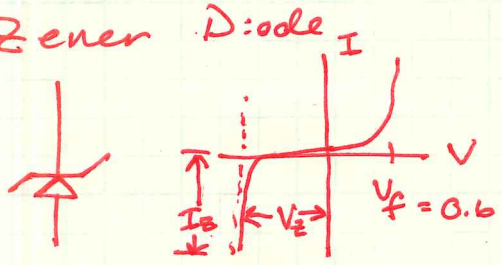
Other Diodes

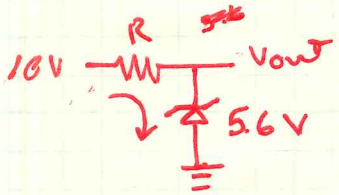
Schottky Diode

$V_f \approx 0.3\ V$



Zener Diode





$\frac{4.4\ V}{10\ mA} \Rightarrow R = 440\ \Omega$

Zener is a great voltage reference!

V_Z follows E12 & E24 series values

Typical: 5.6 V

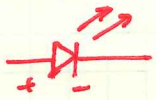
I_Z is current required to put zener into reverse conduction

I_Z typically 5 – 10 mA

If our goal is stable Vout, then Zener is the way to go!

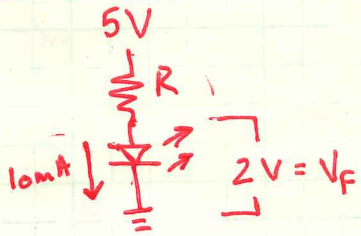
Hedges against V_{supply} / V_{BAT} Drop

For HW4: LEDs!



- Polarity matters
- V_F typically around 2 V
- I_F maximum typically around 20mA

Typical circuit:

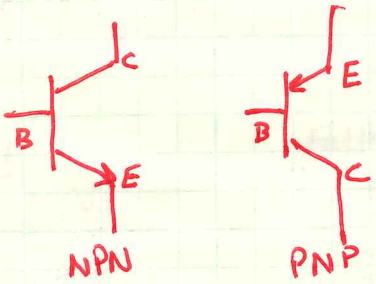


$R = \frac{5V - 2V}{10mA} = 300\,\Omega$

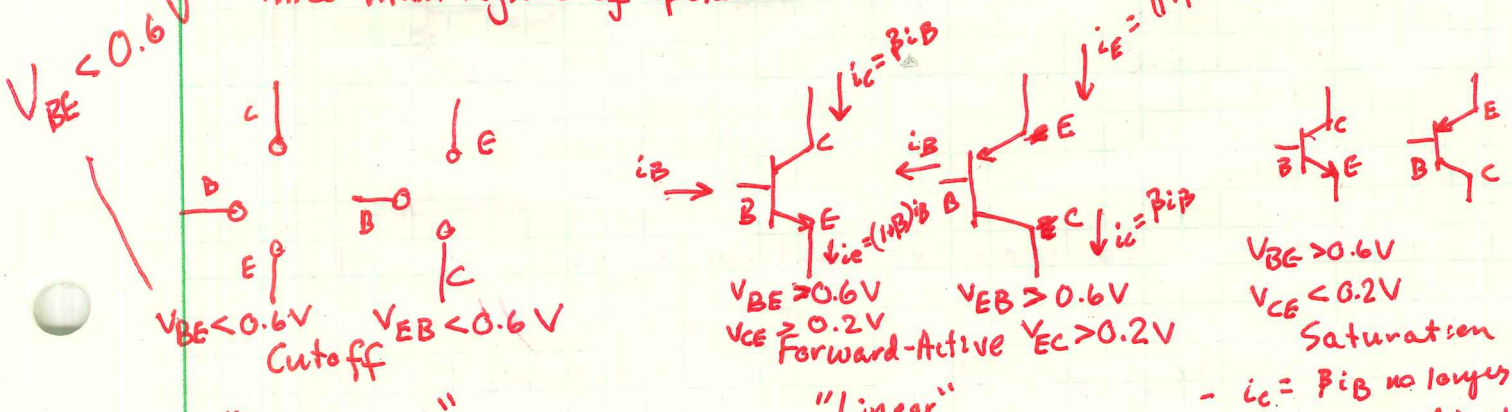
$R = 330\,\Omega$

Transistors - BJTs

- Bipolar Junction Transistor
- Labeled "Q" circuit diagrams
- Two types: NPN and PNP



- Three main regions of operation

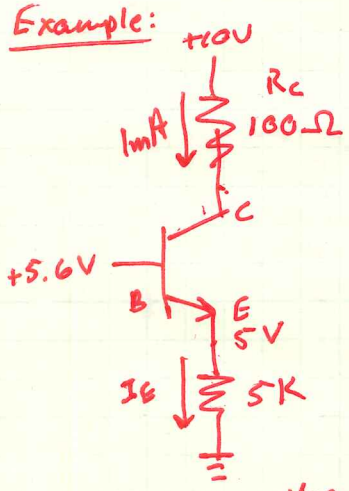


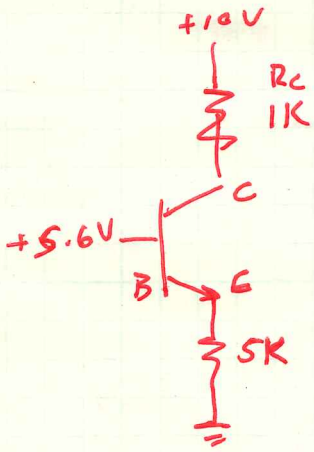
Constant Current Source

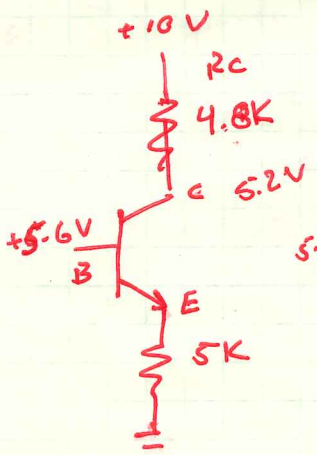
max load before saturation

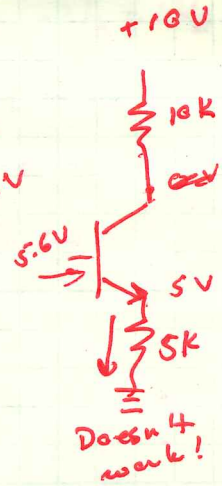
"saturation"

Example:









V_{BE}

Eq. $V_E = 5.6V - 0.6V$

$= 5V$

$I_E = \frac{5V}{5K} = 1mA$

$I_C \approx I_E = 1mA$

$V_C = 10V - (100\Omega)(1mA)$

$= 9.9V$

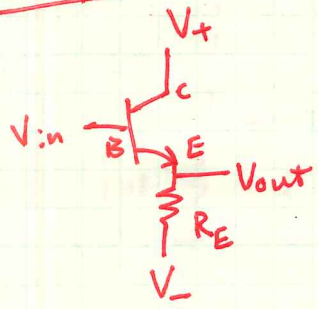
Same, but $V_C = 9V$

Same, but $V_C = 10V - (4.8K)(1mA)$

$= 5.2V$

Output voltage compliance: Range of voltages that are supported by the current source

Example:



Follower

If V_{in} is small enough such that the transistor stays in the forward active region, then

$V_{out} = V_{in} - 0.6V$

Input Impedance

(1) Apply ΔV_B at the base. The change at the emitter will also be ΔV_B. So $\Delta V_E = \Delta V_B$

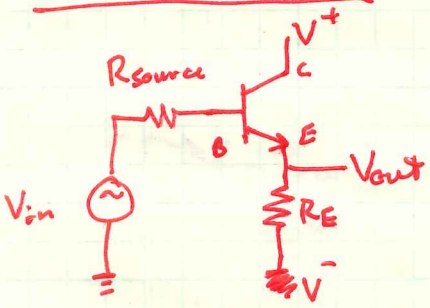
(2) $\Delta I_E = \frac{\Delta V_E}{R_E} = \frac{\Delta V_B}{R_E}$

(3) $\Delta I_E = (1+\beta)\Delta I_B$

(4) $\frac{\Delta V_B}{R_E} = (1+\beta)\Delta I_B$

(5) $\boxed{R_{IN} = \frac{\Delta V_B}{\Delta I_B} = (1+\beta)R_E}$

Output Impedance



(1) Apply ΔV_E. This yields $\Delta V_B = \Delta V_E$

(2) Now, $\Delta I_B = \frac{\Delta V_B}{R_{source}} = \frac{\Delta V_E}{R_{source}}$

(3) We know that $\Delta I_E = (1+\beta)\Delta I_B + \frac{\Delta V_E}{R_E}$

(4) Substitute:

$$\Delta I_E = (1+\beta)\frac{\Delta V_E}{R_{source}} + \frac{\Delta V_E}{R_E}$$

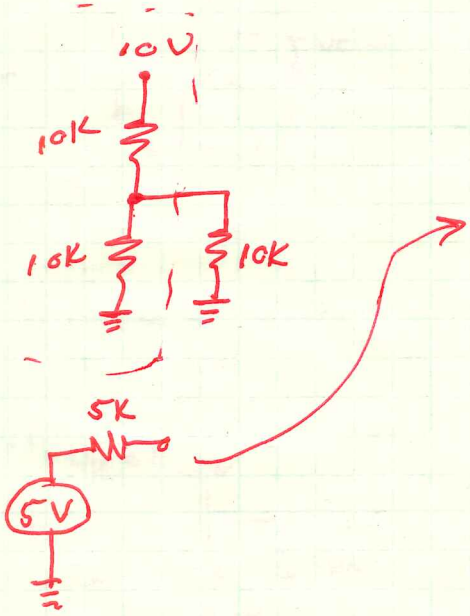
$$\Delta I_E = \left(\frac{1+\beta}{R_{source}} + \frac{1}{R_E}\right)\Delta V_E$$

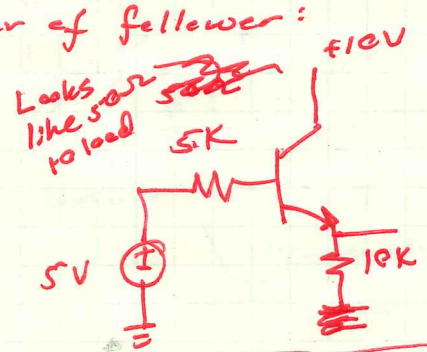
$$R_{out} = \frac{\Delta V_E}{\Delta I_E} = \frac{1}{\frac{1+\beta}{R_{source}} + \frac{1}{R_E}}$$

($\frac{1}{R_E}$ → small)

$$\boxed{R_{out} \approx \frac{R_{source}}{1+\beta}}$$

Example to illustrate power of follower:



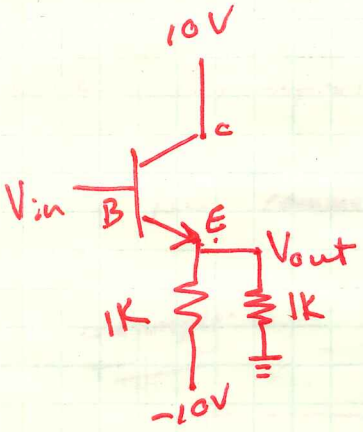


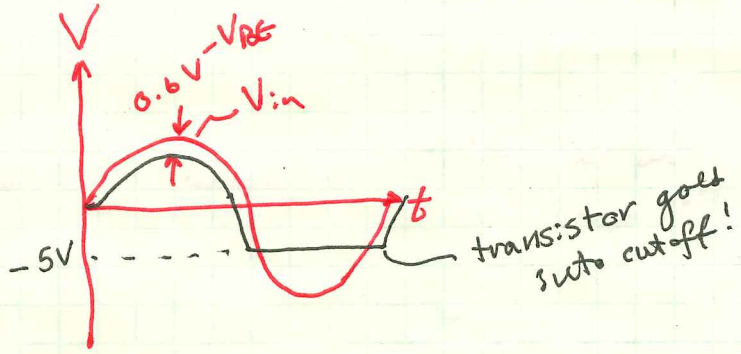
Now we can drive heavier load

5V models diode drop

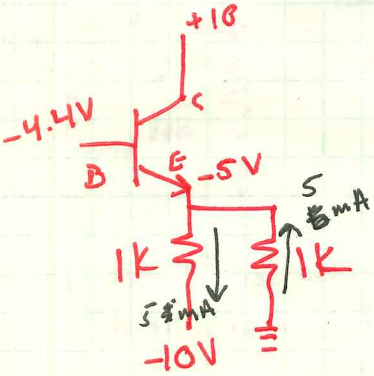
Mixing R_E & "R_{LOAD}" ★

Be careful!

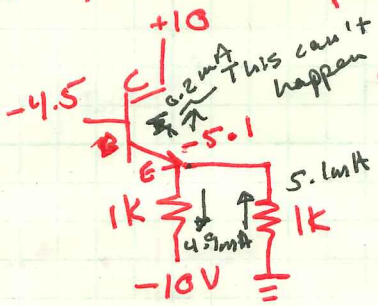




Problem appears when V_E hits 5V



If input decreases further

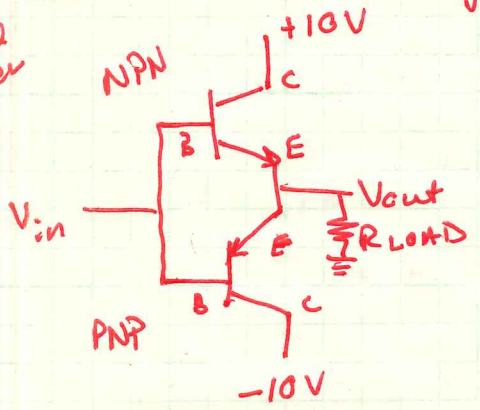


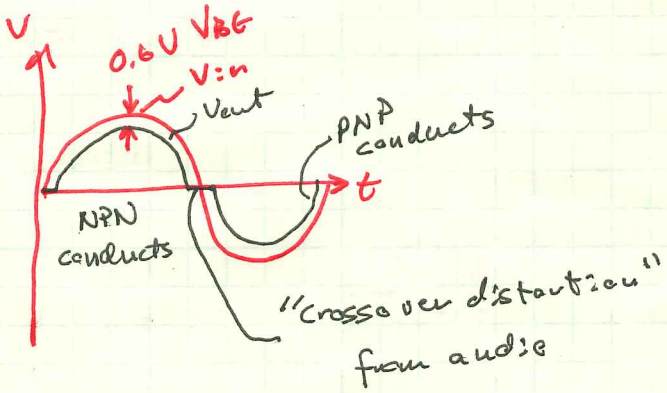
currents at E emitter need to sum to zero

NPN only wants to source current

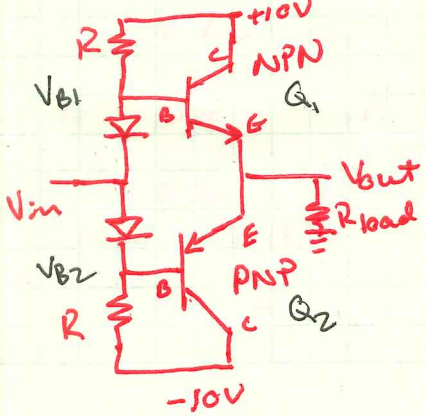
Solution to this asymmetry problem:

Push-Pull Follower



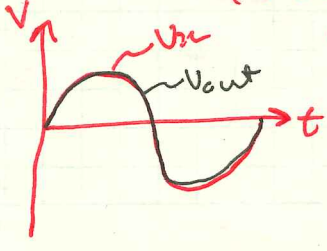


Improved Push-Pull



(1) Resistor bring diodes into forward conduction

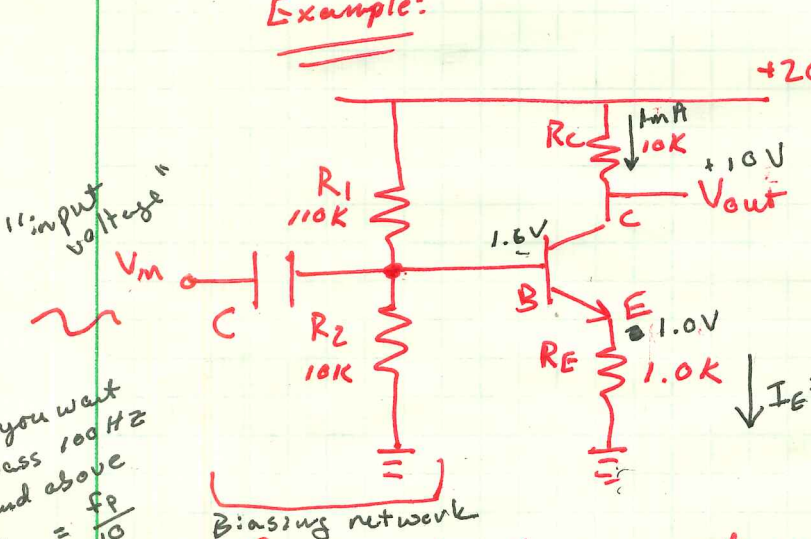
(2) Resistors provide base currents to transistors

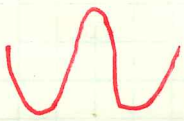


One last, important circuit:

AC common-emitter amplifier. Now we are amplifying voltage

Example:





"inverting amplifier"

Say you want to pass 100Hz and above

$f_{3dB} = \frac{f_P}{10}$ or $\frac{f_P}{2}$

First, look at quiescent conditions ($V_{in} = 0$):

$$V_B = \frac{R_2}{R_1+R_2} V_s = \frac{10K}{100K+10K} \cdot 20V = 1.6\,V$$

$$V_E = V_B - 0.6V = 1.0\,V$$

$$I_E = \frac{1V}{1K} = 1mA$$

$$I_C = I_E = 1mA$$

$$V_{out} = 20 - (10K)(1mA) = 10\,V$$ "centered output"

Now, let's see what happens when we apply some input:

ΔV_{in}.

$$\Delta V_E = \Delta V_{in}$$

$$\Delta I_E = \frac{\Delta V_E}{1K} = \frac{\Delta V_{in}}{1K}$$

$$\Delta I_C = \Delta I_E = \frac{\Delta V_{in}}{1K}$$

$$\Delta V_{OUT} = -(10K)\cdot\frac{\Delta V_{in}}{1K}$$

$$\frac{\Delta V_{out}}{\Delta V_{in}} = -10$$

(-) indicates inversion

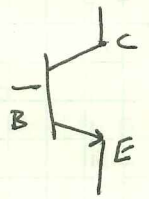
$$G = -\frac{R_C}{R_E}$$

"Gain"

what happens w/ small R_E?

Refined Transister Model

Yesterday:



$i_c = \beta i_\beta$

$V_{BE} = 0.6\ V$

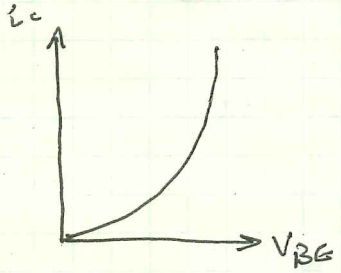
Today:

"Ebers-Moll" Model

V_{BE} can vary slightly: 0.2 - 0.8 V

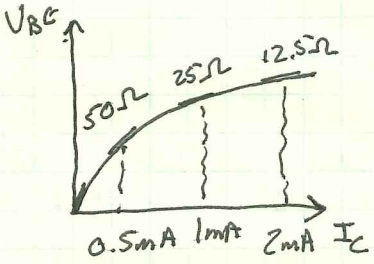
V_{BE} determines i_c according to $i_c = i_s\left(e^{\frac{V_{BE}}{kT/q}} - 1\right)$

$i_c \approx i_s e^{V_{BE}/25mV}$



- $kT/q \approx 25mV$ at room temp
- -1 is small relative to exponential term
- i_s: saturation current [10^{-12} to 10^{-15} A]
  Temperature dependent

Flip the axes:



Take log of both sides

$$\ln(i_c) \approx \ln(i_s) + \frac{V_{BE}}{25mV}$$

$$r_e = \frac{\Delta V_{BE}}{\Delta I_C} = \frac{25}{I_C(\text{in mA})}\ \Omega$$

$$V_{BE} = 25mV \cdot \ln(i_c)$$

$$\frac{dV_{BE}}{di_c} = \frac{25}{(i_c)_{mA}}\ \Omega$$

The upshot: - We lump the complex Ebers-Moll relation in one simple parameter => r_e

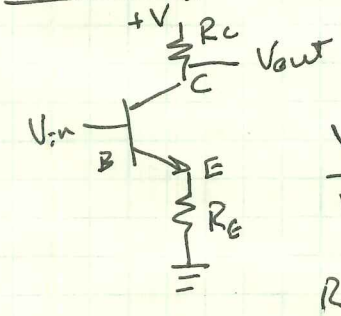
- We only need to use this refined model when r_e is of a similar size to the other resistors in your circuit.

=> Or, as a heuristic, if something is too good to be true - i.e. $Z_{out} = 0$ or $G_{ain} = \infty$

Two cases in which r_e is needed:

Grounded Emitter Amplifier

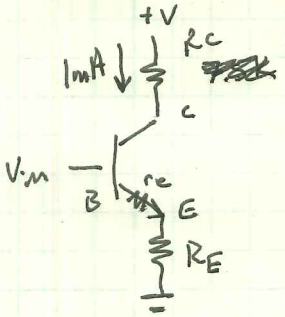
Yesterday:



$$\frac{V_{out}}{V_{in}} = -\frac{R_C}{R_E}$$

$R_E \to 0 \quad \frac{V_{out}}{V_{in}} \to \infty$

Today:



r_e is "intrinsic emitter resistance"

$$r_e = \frac{25}{I_{C(mA)}} = \frac{25}{1} = 25\Omega$$

$$G = \frac{V_{out}}{V_{in}} = -\frac{R_C}{r_e + R_E}$$

if $R_E \to 0$

$$\frac{V_{out}}{V_{in}} = -\frac{R_C}{r_e}$$

When to care about r_e?

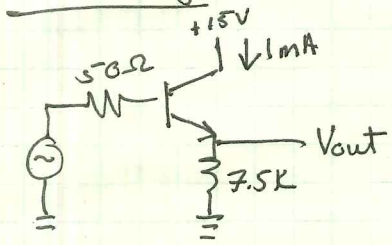
$r_e = 25\Omega$

if $R_E > 250\Omega$

r_e doesn't matter

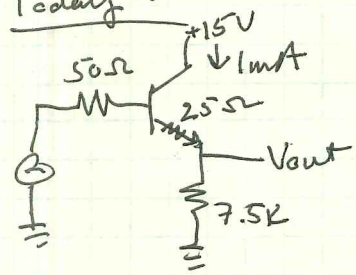
Follower

Yesterday:



$$R_{out} = 7.5K \parallel \frac{50\Omega}{\beta} \approx \frac{50}{\beta} \approx 0.5\Omega$$

Today:



$$R_{out} = 7.5K \parallel \frac{50\Omega}{\beta} + r_e \sim 25\Omega$$

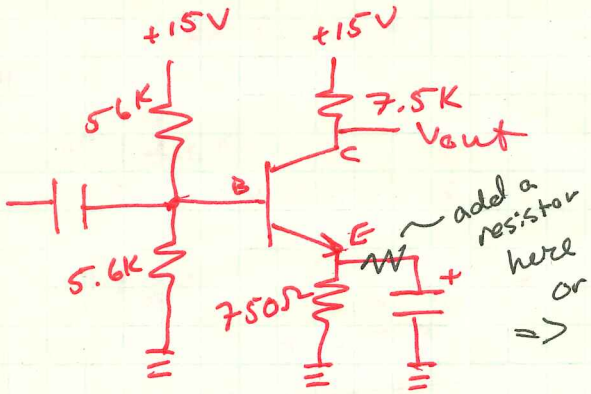
$$\approx 25\Omega$$

r_e puts a floor on follower output impedance.

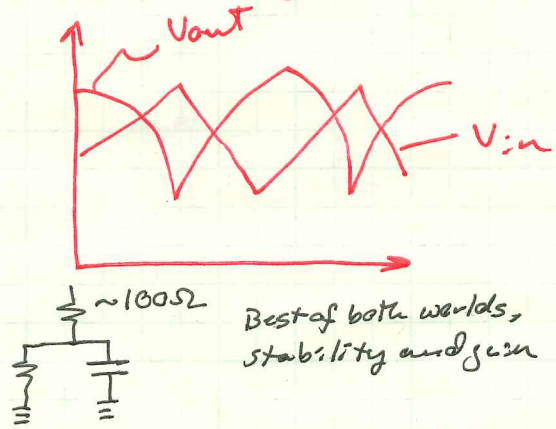
Today's Topics
- Distortion explained by r_e
- Temperature (in)stability of transistors
- Difference Amplifier

Distortion

In lab you had this circuit:



Qualitatively

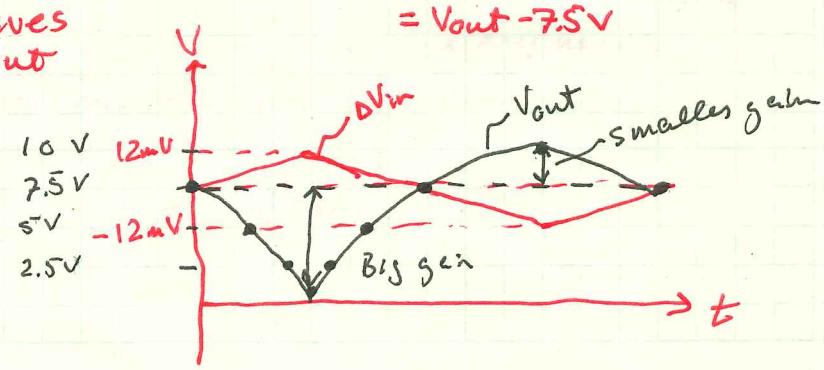


Vout	I_c (mA)	r_e (Ω)	Gain	ΔVout	ΔVin (V)
14.8	0.03	940	8	7.3	-0.9
12.5	0.33	75	100	5.0	-0.05
10	0.66	38	200	2.5	-0.012
7.5	1	25	300	0	0
5	1.33	19	400	-2.5	0.006
2.5	1.66	15	500	-5.0	0.01
0.2	2	12.5	592	-7.3	0.012

$$I_c = \frac{15 - V_{out}}{7.5k} \qquad r_e = \frac{25}{I_c(mA)} \qquad G = \frac{7.5k}{r_e} \qquad \Delta V_{in} = \frac{V_{out}}{G}$$

$$\Delta V_{out} = V_{out} - 7.5V$$

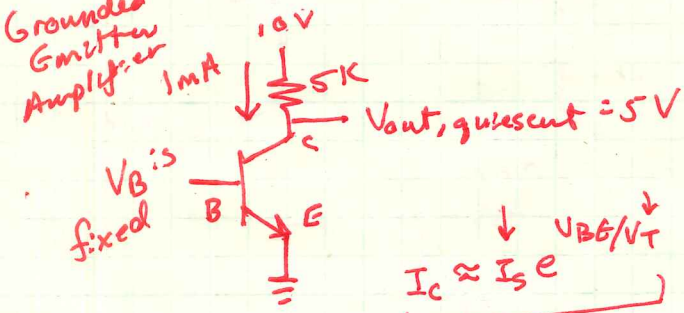
- Symmetric input gives asymmetric output
- Input shape is distorted
- To correct, sacrifice gain for stability
  => add R_E



Temperature instability:

- Transistors are very sensitive to temperature changes!
- Quantify:
  - I_C grows at about 9% per °C, if you hold V_{BE} constant
  - V_{BE} falls at 2mV/°C, if you hold I_C constant

Grounded Emitter Amplifier



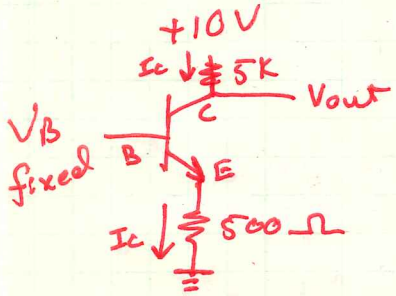
$$I_C \approx I_S e^{V_{BE}/V_T}$$

I_C (mA)	T (°C)
1	25
1.09	26
⋮	⋮
2	33

The quiescent voltage is changing drametically – this is bad!

→ Only 8°C resulted in $V_{out,quiescent} \approx 0V$

Simple remedy: Add an emitter resistor



(1) Increase in Temp raise I_C

(2) Increasing I_C increases V_E

(3) Since V_B is fixed, V_{BE} shrinks

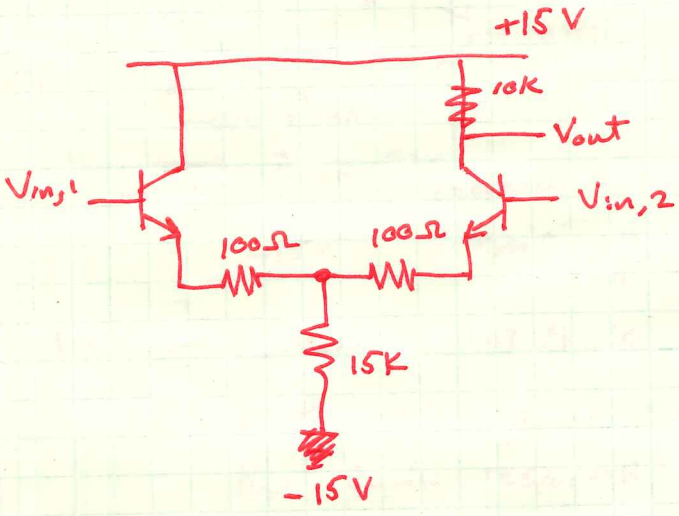
(4) Since V_{BE} shrinks, I_C decreases.

Stable! Negative Feeback is the key here.

- Other Temp Stability methods rely on putting multiple transistors on a single dye, so that they all see the same temp change.

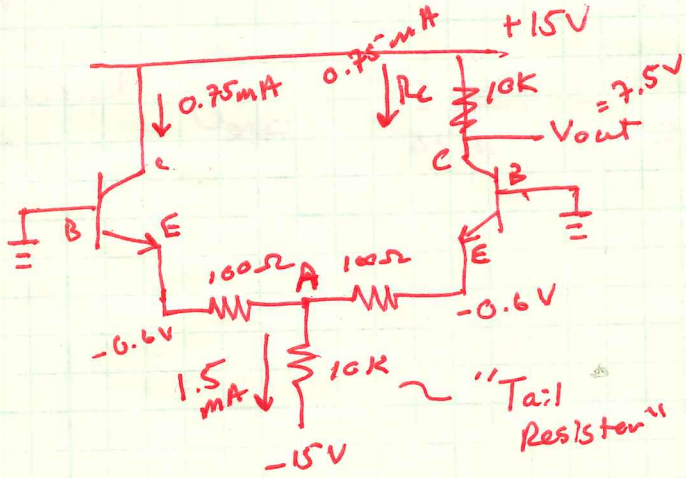
Difference Amplifier

- Main idea – amplify the difference between two signals and attenuate the common component of the signals.
- Implementation:



- First step in analysis is to look at quiescent behavior:

$$V_{in,1} = V_{in,2} = 0\,V$$



$V_E = -0.6\ V$

$V_A \approx -1\ V$

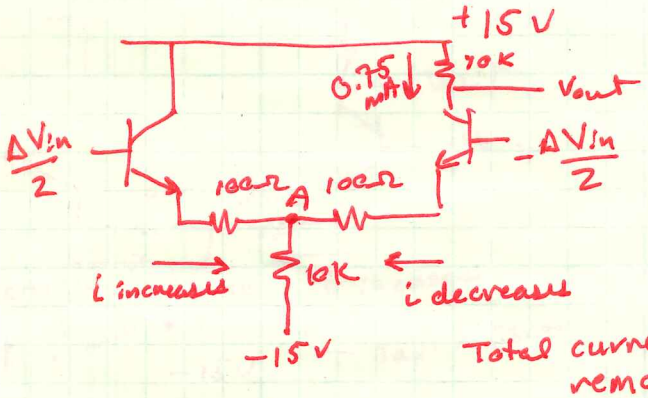
$$I_{TAIL} = \frac{-1-(-15)}{10K} \approx 1.5mA$$

Because R_C doesn't affect current, then the current must be shared equally between the two branches

$$V_{out} = 15 - 0.75mA \cdot 10K$$
$$= 7.5\,V$$

sits at center of possible output swing.

- Now look at differential gain
  - ↳ apply opposite signals at the inputs
    "raise one, lower the other"

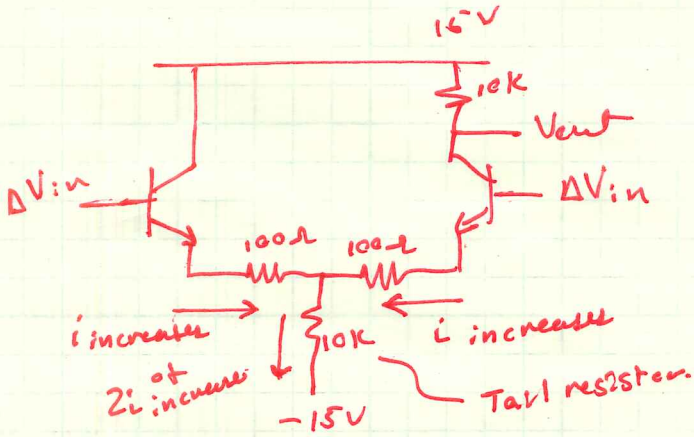


  - ↳ Key insight: Point A is at a fixed voltage, so all the change in current must occur across the 100Ω resistors! This will evoke a change in output voltage given by:

Familiar "Gain equation" $\Rightarrow \quad \frac{\Delta V_{out}}{(\Delta V_{in}/2)} = \frac{10K}{100\Omega + r_e \approx 33\Omega}$ $\qquad r_e = \frac{25}{.75} \approx 33\Omega$

Differential Gain $\quad G_{DIFF} = \frac{\Delta V_{out}}{\Delta V_{in}} = \frac{10K}{2(100\Omega + 33\Omega)} \approx 37.5$

- Common Mode Gain
  - → same signal to both inputs



- Now, the change in current will occur across r_e, 100Ω, and the 10k tail resistor.

This evokes a change in voltage at the collector

$$G_{CM} = \frac{\Delta V_{out}}{\Delta V_{in}} = \frac{10K}{100\Omega + r_e + 2(10K)} \approx \frac{1}{2}$$

↑ Because tail current is shared by the two branches, this R looks twice as big

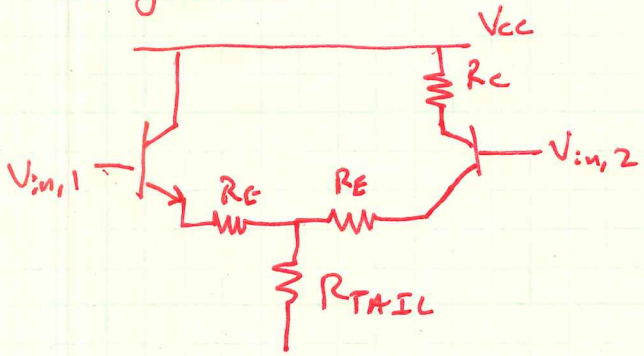
- Common Mode Rejection Ratio (CMRR) is a performance metric for differential amplifiers

$$CMRR = \frac{G_{DIFF}}{G_{CM}}$$

For our design $CMRR = \frac{37.5}{1/2} = 75$

$$CMRR_{dB} = 20\log_{10}\left(\frac{G_{DIFF}}{G_{CM}}\right)$$

- In general:



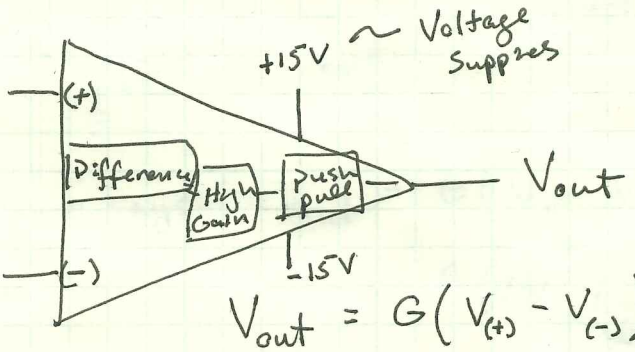
$$G_{DIFF} = \frac{R_C}{2(R_E + r_e)}$$

$$G_{CM} = \frac{R_C}{R_E + r_e + 2R_{TAIL}}$$

$$CMRR \approx \frac{R_{TAIL}}{R_E + r_e}$$

- To increase CMRR, increase R_{TAIL}, decrease R_E.
- Use a current source (high R_{out}) for R_{TAIL}

Operational Amplifier - "Op Amps"



$$V_{out} = G(V_{(+)} - V_{(-)})$$

For our op-amp:

$G \approx 600-700$

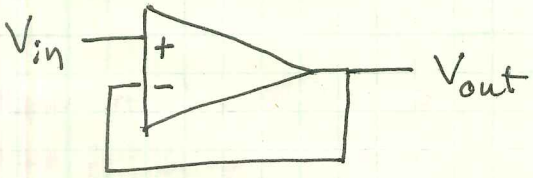
For op-amp ICs:

$G \approx 10^5$ to 10^6

Define (+) terminal as: non-inverting input

(-) terminal as: inverting input

Circuit #1: Follower



$$V_{out} = G(V_{in} - V_{out})$$

$$(1+G)V_{out} = G(V_{in})$$

$$\frac{V_{out}}{V_{in}} = \frac{G}{1+G}$$

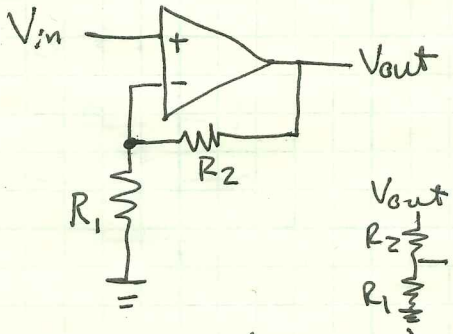
if $G \gg 1$,

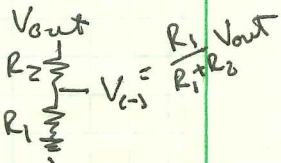
$$\boxed{\frac{V_{out}}{V_{in}} \approx 1}$$

Negative Feedback

The output is fed back to the (-) input.

Circuit #2: Non-inverting Amplifier





$$V_{out} = G(V_{(+)} - V_{(-)})$$

$$= G\left(V_{in} - \frac{R_1}{R_1+R_2}V_{out}\right)$$

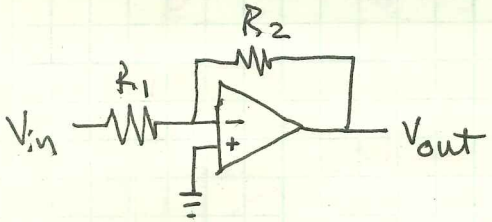
$$\left(1 + \frac{GR_1}{R_1+R_2}\right)V_{out} = G V_{in}$$

$$\frac{V_{out}}{V_{in}} = \frac{G}{1+G\frac{R_1}{R_1+R_2}}$$

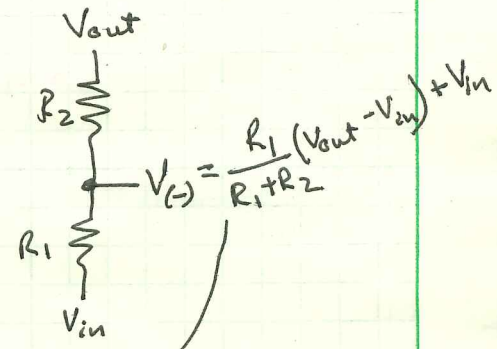
if $G \gg 1$

$$\boxed{\frac{V_{out}}{V_{in}} = \frac{1}{\frac{R_1}{R_1+R_2}} = 1 + \frac{R_2}{R_1}}$$

Circuit #3: Inverting Op Amp



No current into op-amp



op-amp says: $V_{out} = G(V_{(+)} - V_{(-)})$

$$= G\left(0V - \left(\frac{R_1}{R_1+R_2}(V_{out} - V_{in}) + V_{in}\right)\right)$$

$$V_{out} + G\frac{R_1}{R_1+R_2}V_{out} = -GV_{in} + G\frac{R_1}{R_1+R_2}V_{in}$$

$$V_{out}\left(1 + G\frac{R_1}{R_1+R_2}\right) = V_{in}\,G\left(\frac{R_1}{R_1+R_2} - 1\right)$$

$$\frac{V_{out}}{V_{in}} = \frac{G\left(\frac{R_1}{R_1+R_2} - 1\right)}{1 + G\frac{R_1}{R_1+R_2}}$$

Tedious Algebra Motivates Need for simpler rules!

Golden Rules:

When you employ negative feedback:

#1 output drives the input terminals to the same voltage

#2 the inputs draw no current.

$G \gg 1$

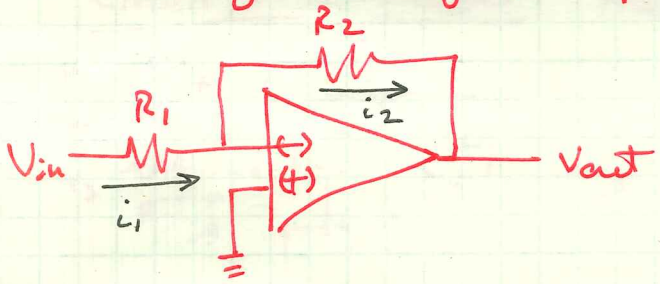
$$\approx \frac{\frac{R_1}{R_1+R_2} - 1}{\frac{R_1}{R_1+R_2}}$$

$$= 1 - \frac{R_1+R_2}{R_1}$$

$$= 1 - 1 - \frac{R_2}{R_1}$$

$$\boxed{\frac{V_{out}}{V_{in}} = -\frac{R_2}{R_1}}$$

Revisiting Inverting Op Amp w/ golden rules:



Fraction fed back

GR #1: Voltage at (-) terminal will be ≈ 0V

GR #2: (-) and (+) terminals draw no current

KCL: $i_1 = i_2$

$$\frac{V_{in}}{R_1} = \frac{0V - V_{out}}{R_2}$$

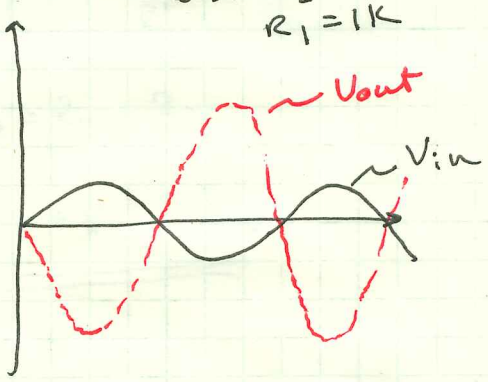
$$\frac{V_{in}}{R_1} = -\frac{V_{out}}{R_2}$$

$$\boxed{\frac{V_{out}}{V_{in}} = -\frac{R_2}{R_1}}$$

Example

Choose $R_2 = 3K$

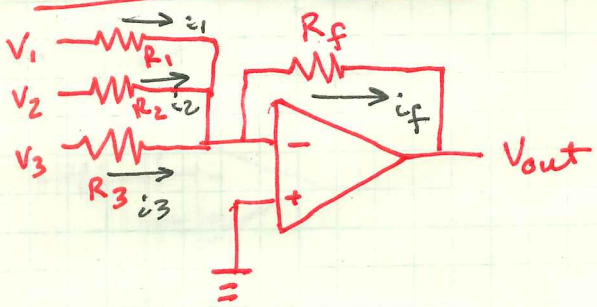
$R_1 = 1K$



Analagous in purpose to the common emitter amp but much more user friendly

Output Range is set by the voltage supplies.

Circuit #4 : Summer "Adder"



$$i_1 + i_2 + i_3 = i_f \quad \text{(KCL)}$$

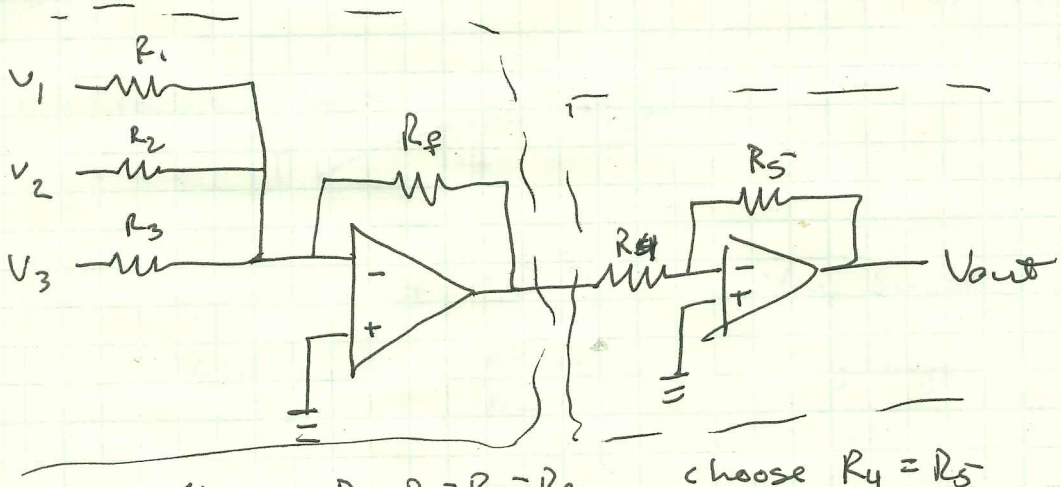
$$\frac{V_1}{R_1} + \frac{V_2}{R_2} + \frac{V_3}{R_3} = -\frac{V_{out}}{R_f}$$

$$\Rightarrow \boxed{V_{out} = -R_f\left(\frac{V_1}{R_1} + \frac{V_2}{R_2} + \frac{V_3}{R_3}\right)}$$

if $R_1 = R_2 = R_3 = R_f$

$$V_{out} = -(V_1 + V_2 + V_3)$$

What if:

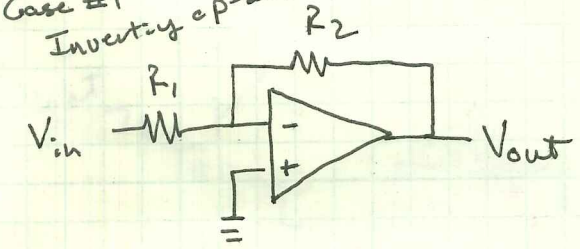


Choose $R_1 = R_2 = R_3 = R_f$ choose $R_4 = R_5$

$$\boxed{V_{out} = V_1 + V_2 + V_3}$$

Input and Output Impedances of Op Amps

Case #1
Inverting op-amp



GR #1: (−) and (+) inputs are driven to the same voltage.

Apply ΔV at input, $\Delta i = \frac{\Delta V}{R_1}$

$$R_{in} = \frac{\Delta V}{\Delta i} = \frac{\Delta V}{\Delta V / R_1} = R_1$$

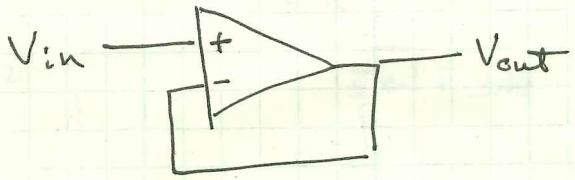
Try to apply ΔV at output, negative feedback fights back and squelches the ΔV wiggle.

$$R_{out} = \frac{\Delta V}{\Delta i} \approx \frac{0}{\Delta i} = 0\ \Omega$$

Case #2:
Follower

a.k.a. "buffer"

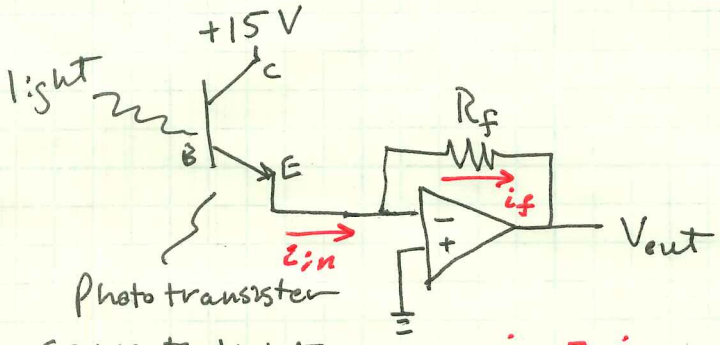
GR #2: Inputs draw no current



As before, $R_{out} \approx 0\ \Omega$ Very Small

Now, $R_{in} = \frac{\Delta V}{\Delta i} \approx \infty$ Very Large

Circuit #5 : Current to Voltage converter

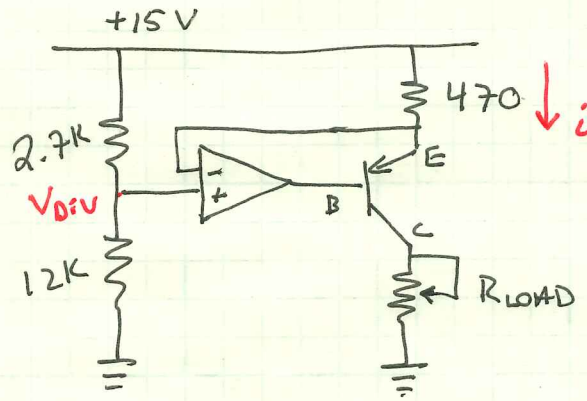


$$i_{in} = i_f$$

$$i_{in} = \frac{-V_{out}}{R_f}$$

$$\boxed{V_{out} = -i_{in} R_f}$$

Circuit # 6 : Improved Current Source

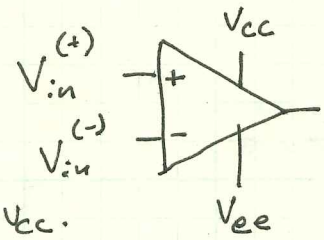


For this ckt

$$\downarrow i_{const} = \frac{V_{DIV}}{R_E} \simeq 0.6\,mA$$

Small imperfections in current sources related to "Early Effect" that negative feedback remedies.

Op-Amp Characteristics (roughly in order of importance)



$$V_{out} = A_{VOL}\left(V_{in}^{(+)} - V_{in}^{(-)}\right)$$

(1) Output Voltage Range

- Most of the time, the output voltage can't go from V_{ee} to V_{cc}.
- For LF411, w ±15 V supplies, output range is ±13.5 V
- Note diode drops
- Two special types
  - "Single-Supply Op Amps"
    - ↳ Used in battery-driven circuits
    - ↳ output can go all the way down to V_{ee}
    - ↳ Usually choose V_{ee} to be Gnd
    - ↳ Example: LM358
  - "Rail-to-Rail opamps"
    - ↳ Output swings all the way from V_{ee} to V_{cc}
    - ↳ Example is: LMH6645.

(2) Input Voltage Range

- Most of the time, the inputs can't go down to V_{ee} or up to V_{cc}
- LF411 w/ ±15 V supplies, input range is ±11 V
- a.k.a. "Common-mode input range"
- Note can also get rail-to-rail inputs
- Single supply op-amps accept input voltages down to V_{ee}.

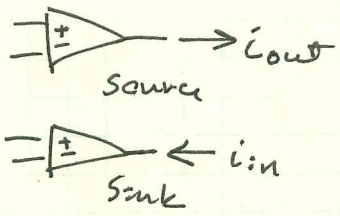
(3) Supply Voltage Range

- Op Amps only properly operate for some range of V_{ee} & V_{cc}
- LF411, $3.5 < V_{cc} < 18$ V
  $-18 < V_{ee} < -3.5$ V

What limits ~~output~~ $V_{cc,max}$?

(4) Output Current Limit

- Op Amps usually can't source or sink large amounts of current
- LF411 source 20mA (min)
  sink 10mA (min)



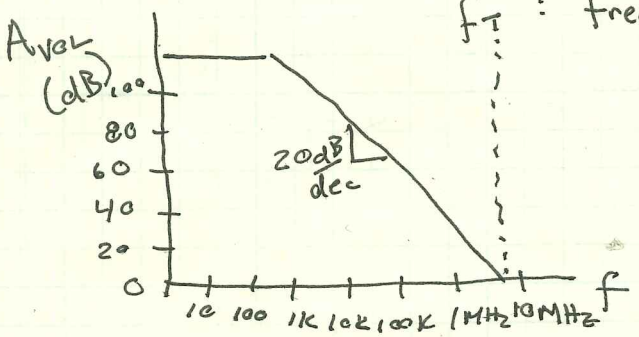
- Current-limiting resistors protect push-pull output of op-amp
-

(5) Gain Rolloff

- Op-Amp gain decreases w/ frequency.
- LF411 : 4 MHz

f_T : frequency at which gain = 1
a.k.a. "Gain bandwidth product"
"unity-gain bandwidth"

LF411
$A_{VCL} = 200 \frac{V}{mV}$
DC level



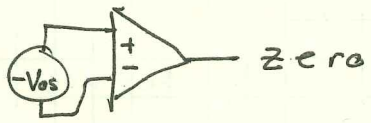
- The real issue is effectiveness of negative feedback degrades with decrease A_{VCL}.

(6) Slew Rate

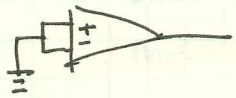
- Defined as the max $\frac{\Delta V}{\Delta t}$ of the op-amp output
- LF411: 15 V/μs
- Also due to compensation capaciter.

(7) V_{offset} : The voltage diffference in input voltage neccessary to bring the output to zero.

Pictorially:



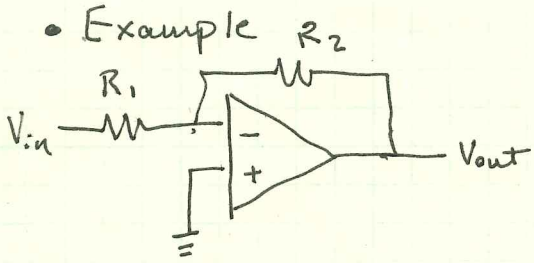
If:



This would saturate

(go to V_{cc} or V_{ee})

- Due to imperfect matching behavior between the two sides of the differential stage.
- LF411, typical is 0.8 mV
- Example



Because of V_{offset}, $V_{in}^{(-)}$ won't sit at 0V, but will sit at $-V_{offset}$ instead.

$\Rightarrow$ error $\sim \left(\frac{R_2}{R_1} V_{offset}\right)$

(8) I_{Bias} : The very small amount of current flowing into or out of the input terminals

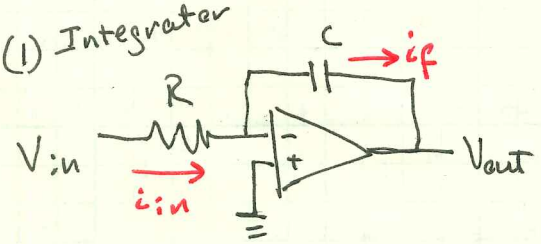
- LF411, 50pA typical at 25 °C
- I_{Bias} is average of two input currents
- Only matters in two edge cases
  - ↳ You use really big resisters (~10 MΩ)
  - ↳ Can cause errors if you use mismatched resistive paths.

(9) I_{offset} : Difference in input currents.

- LF411, 25 pA (typical)
- Can cause errors w/ big resisters.

Some More Op-Amp Circuits:

(1) Integrater



$i_{in} = \frac{V_{in}}{R}$ $\quad i_f = -C\frac{dV_{out}}{dt}$

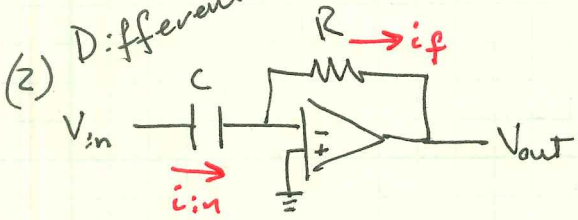
$i_{in} = i_f$ (KCL)

$$\frac{V_{in}}{R} = -C\frac{dV_{out}}{dt}$$

$$\Rightarrow \frac{dV_{out}}{dt} = -\frac{1}{RC}V_{in}$$

$$\boxed{V_{out} = -\frac{1}{RC}\int V_{in}\,dt}$$

(2) Differentiater



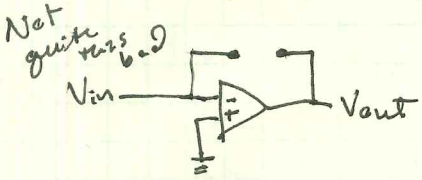
$i_{in} = C\frac{dV_{in}}{dt}$ $\quad i_f = \frac{-V_{out}}{R}$

$i_{in} = i_f$ (KCL)

$$C\frac{dV_{in}}{dt} = -\frac{V_{out}}{R}$$

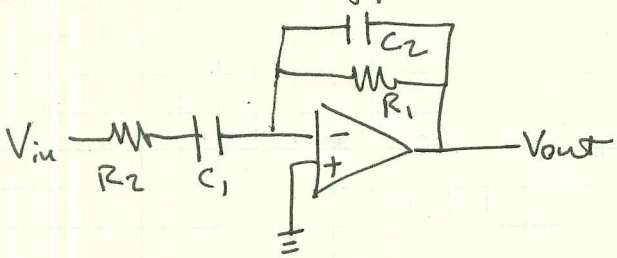
$$\Rightarrow \boxed{V_{out} = -RC\frac{dV_{in}}{dt}}$$

This circuit has issues at high frequencies



→ Feedback is feeble.

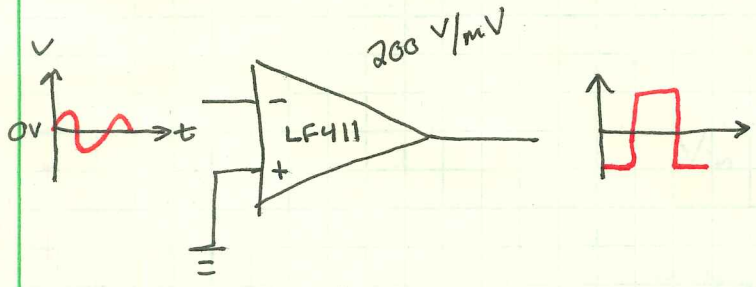
(3) Practical Differentiater



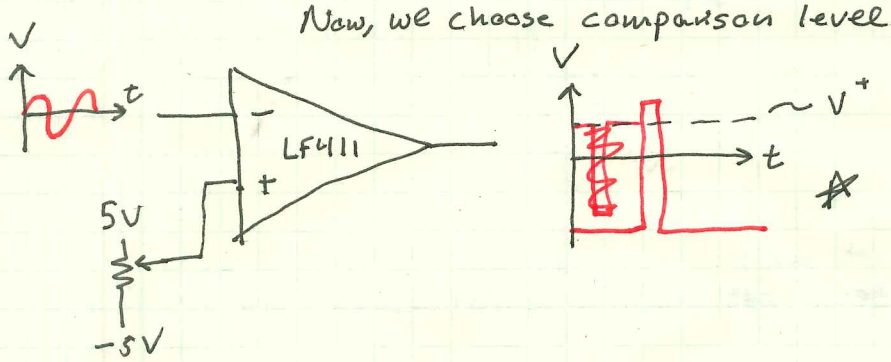
ECE 2035 Summer 2011 Op-Amps III: Nice Positive Feedback

7/11

Question: What does this do? Compares input to zero

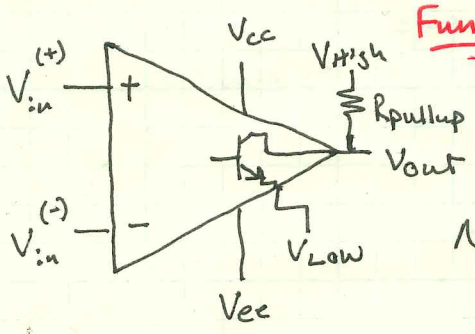


Now, we choose comparison level



Dedicated ICs to do this comparison task.

- LM311 "Open Collector output" - output is collector of npn transistor.



Functionality:

Transistor turns on when $V_{in}^{(+)} < V_{in}^{(-)}$

Note: Assume biasing is taken care of inside ~~opamp~~ comparator.

Typical Chaices:

$V_{high} = 5V$

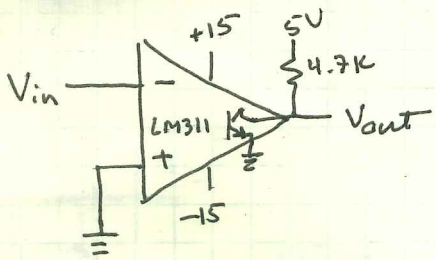
$V_{Low} = 0V$ GnD

$R_{pullup} = 4.7K$

Transistor on: $V_{out} = V_{Low}$

Transistor off: $V_{out} = V_{high}$

Example:

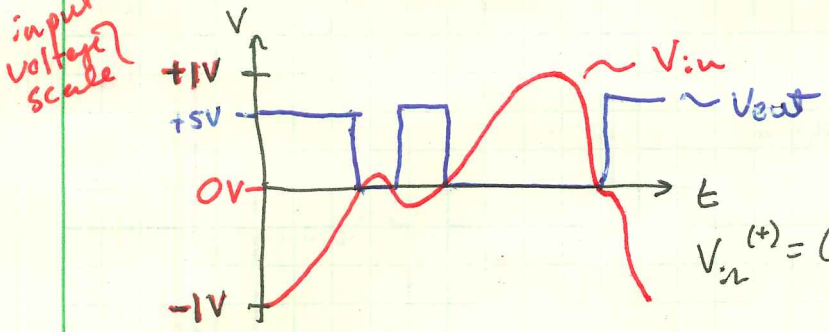


Transistors turns on when

$$V_{in}^{(+)} < V_{in}^{(-)}$$

If $V_{in}^{(-)} > V_{in}^{(+)}$, $V_{out} = 0V$

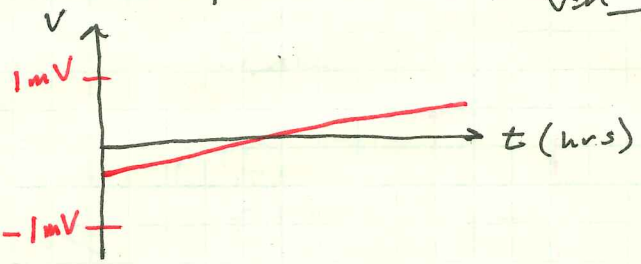
$V_{in}^{(-)} < V_{in}^{(+)}$, $V_{out} = 5V$



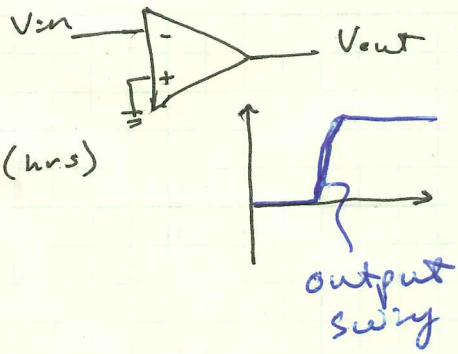
Two key challenges with ~~comparators~~ these type of comparisons

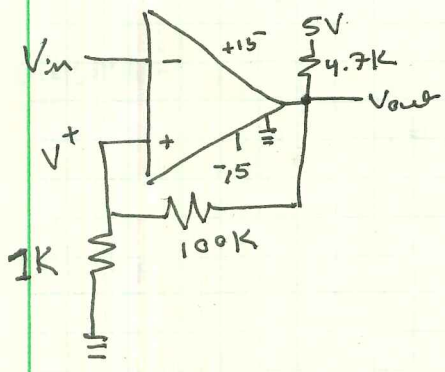
- Noise can create multiple zero crossings.
  - ↳ Hysteresis solves this!
- Slowly changing inputs can result in slow output swings.

Example:



Example w/hysteresis

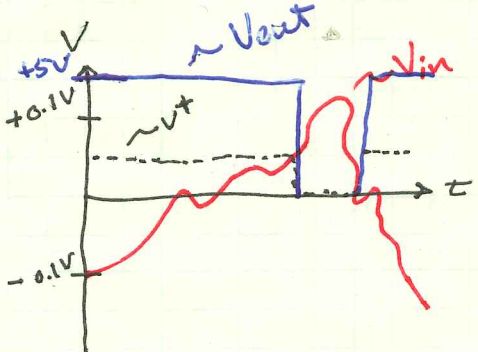




$V_{out} = 5V$

$$V^+ = \frac{1k}{100k + 1k} 5V$$

$V^+ = 0.05V$ 50mV

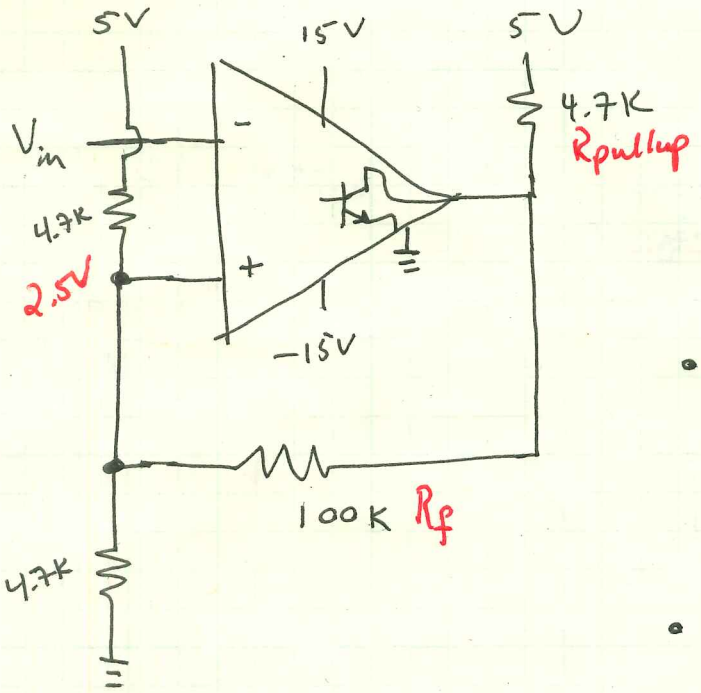


$V_{out} = 0V$

$V^+ = 0V$

- What if V_{in} is centered at 50mV?
- Center your switching levels about center of V_{in}.

Another Example



- Choose $R_f >> R_{pullup}$

- This is for case of Vin centered about 2.5V

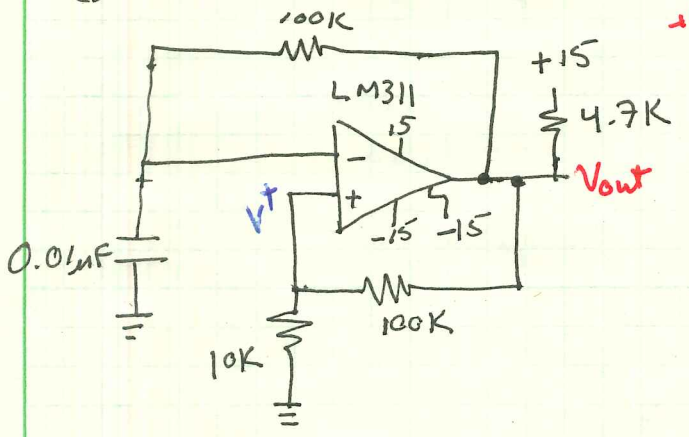
- Most of the time, center of hysteresis will be at nominal switching threshold.

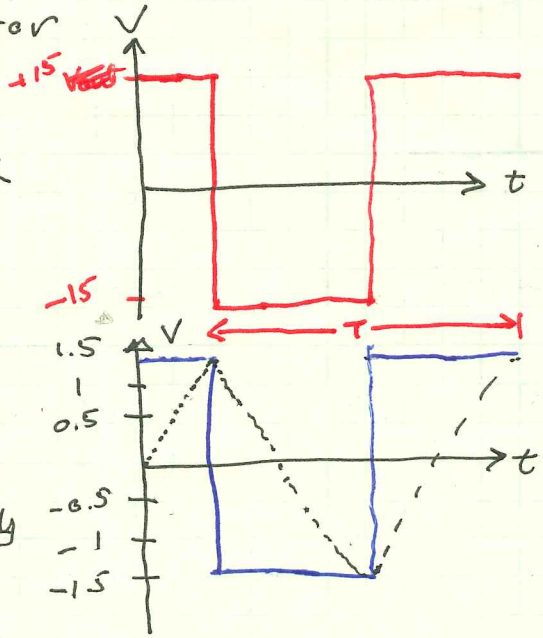
- Lots of ways to build positive feedback to give desired hysteresis

- Positive feedback speeds up output transitions.

Oscillators

(1) RC Relaxation Oscillator





- Assume Current is approximately constant

$$I = C \frac{\Delta V}{\Delta t}$$

$$\frac{15V}{100K} = 0.01\mu F \frac{2.8V}{\Delta t}$$

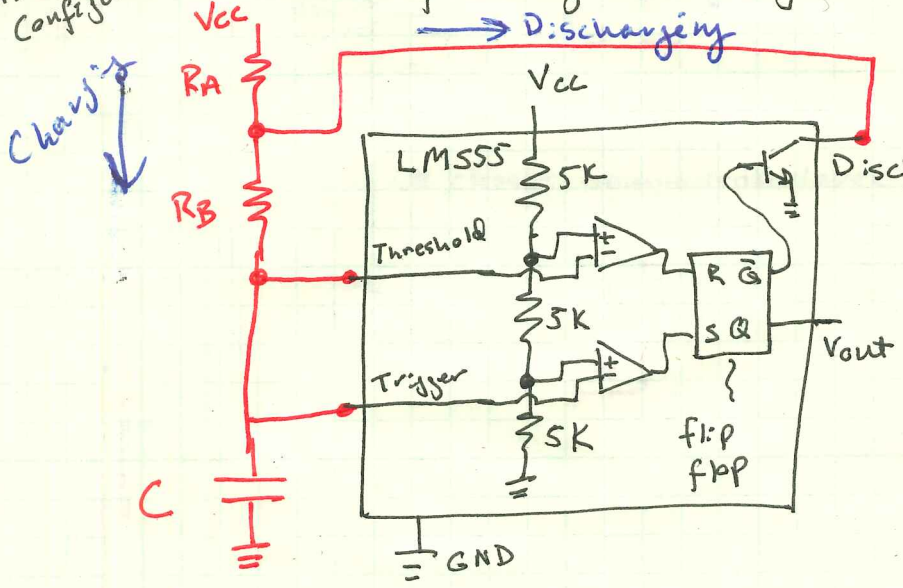
$$T = 2\Delta t$$

$$f = 1/T$$

(2) '555 RC Oscilleter Timer

- Versatile and Ubiquitous IC for oscilleter circuits
- Mix of analog and digital components

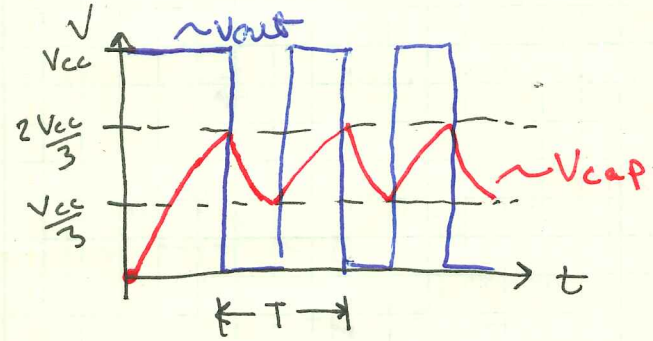
"Astable configuration"



Set	Reset	Q	Q̄
0	1	0	1
1	0	1	0

0 => 0V

1 => 5V



$$T = 0.7(R_A + 2R_B)C$$

$$f = \frac{1}{T} \approx \frac{1.4}{(R_A + 2R_B)C}$$

- Ratio of R_A & R_B is related to time high and time low

Duty Cycle: % of time that the output is high.

$$D = \frac{R_A + R_B}{R_A + 2R_B}$$

Metal Oxide Semiconducting Field Effect Transistor (MOSFETs)

Basic Understanding: Voltage Controlled Switches

Key Attribute: Very high input impedence (10^{12} to $10^{14}\ \Omega$)

Key Application: Foundation of modern digital logic (logic switching)

Other Applications: Analog switching, Power switching

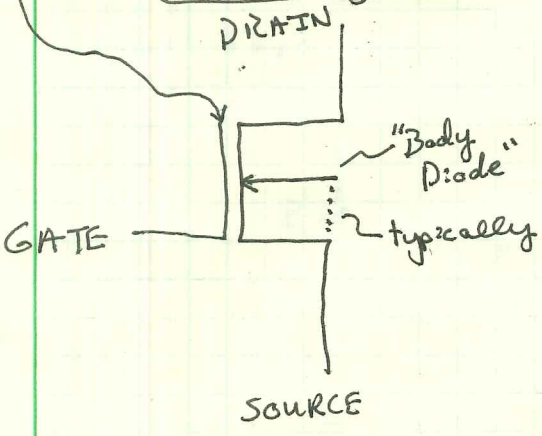
Key Types: NMOS Enhancement Mode (NMOS)

PMOS Enhancement Mode (PMOS)

Analogous to NPN & PNP Bipolar Transistors

gap reminds you of high Z in

NMOS Symbol



V_T: "Threshold Voltage"

Example

NMOS 3N170

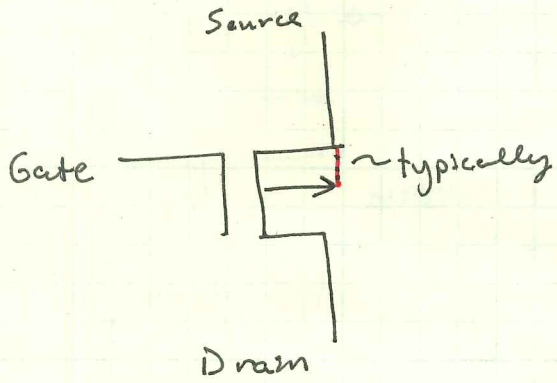
$V_{T,min} = 1\ V$

$V_{T,max} = 2V$

Normal operation

- Gate is the input, drain is output
- Drain is more positive than Source
- Gate to source voltage (V_{GS}) controls the switch state.
  - ↳ $V_{GS} > V_T$, switch is closed
  - ↳ $V_{GS} < V_T$, switch is open
- Body diode typically connected to negative most voltage in circuit
  - ↳ typically source, but sometimes a negative supply
  - ↳ We don't want body diode to conduct, typically

PMOS Symbolically



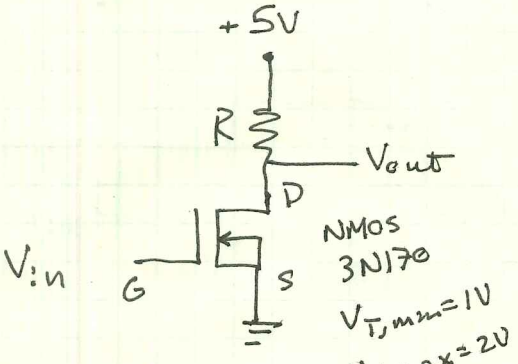
V_T typically
1 to 2.5 V

Normal Operation

- Source is higher voltage than drain
- V_{GS} controls switch state.
  - ↳ $V_{SG} < V_T$ switch is off
  - ↳ $V_{SG} > V_T$ switch is on
- Body diode is connected to most positive voltage in ckt.

Basic Logic Gate

NMOS Inverter



When $V_{in} = 0V$, $V_{GS} = 0V$, $V_{GS} < V_T$,
so switch is off, $V_{out} = 5V$

When $V_{in} = 5V$, $V_{GS} = 5V$, $V_{GS} > V_T$,
so switch is on, $V_{out} \approx 0V$

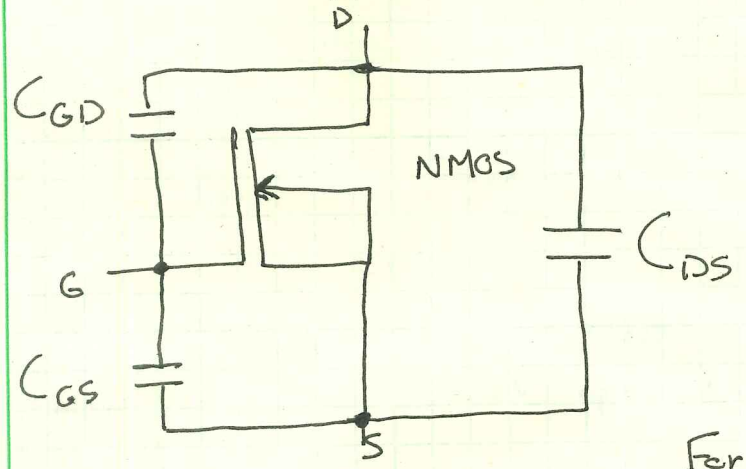
Two issues w/ this approach

(1) Power Dissipation in the resistor when the switch is on.

(2) Output rise is slow for high levels because stray capacitance at the output must be driven by the resistor.

MOSFETS

Aside: ~~Transistors~~ have pesky capacitances that can slow down switching!



On Spec Sheets

C_{iss}	$C_{GD}+C_{GS}$	Input capacitance
C_{oss}	$C_{GD}+C_{DS}$	Output capacitance
C_{rss}	C_{GD}	Feedback capacitance

For 3N170:

$C_{iss} = 5pF$

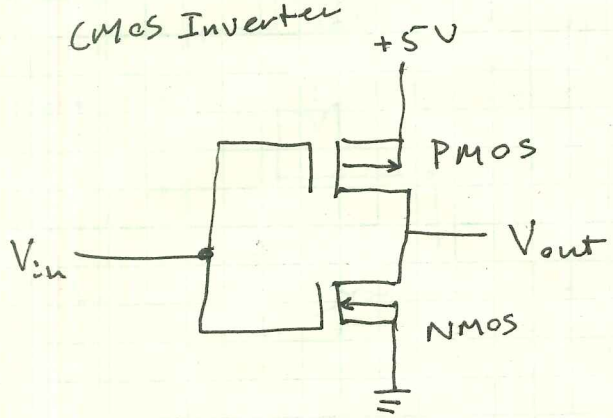
$C_{rss} = 1.3pF$

$C_{DB} = 5pF$ (C_{DS}) → "Body"

Better Approach to Logic

CMOS! "Complimentary" MOS

CMOS Inverter



When

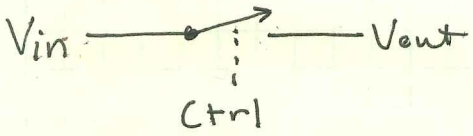
$V_{in} = 0V$ { NMOS OFF, PMOS ON } $V_{out} = 5V$

$V_{in} = 5V$ { NMOS ON, PMOS OFF } $V_{out} = 0V$

- Very little power dissipation
- Fast output transitions.

Analog Switching

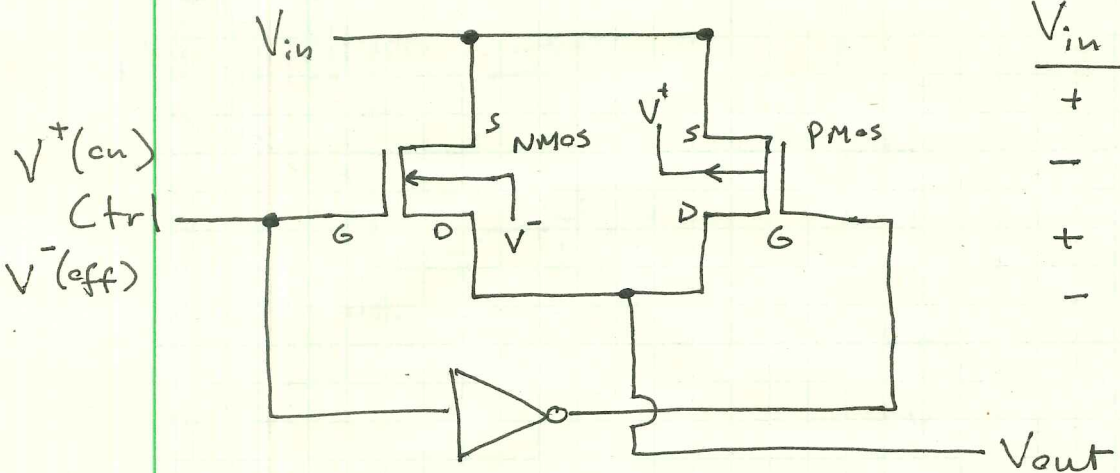
- Use a digital signal to control the passing or blocking of an analog signal.



Let Ctrl = 5V close the switch.
Ctrl = 0V open the switch.

Tricky because we need to handle +/- inputs and outputs.

↳ indicates we need a CMOS approach



"Logic Inversion"

V_{in}	Ctrl	NMOS	PMOS
+	V^+	off	on
−	V^+	on	off
+	V^-	off	off
−	V^-	off	off

DG403

S: Input
D: Output
IN1: Ctrl

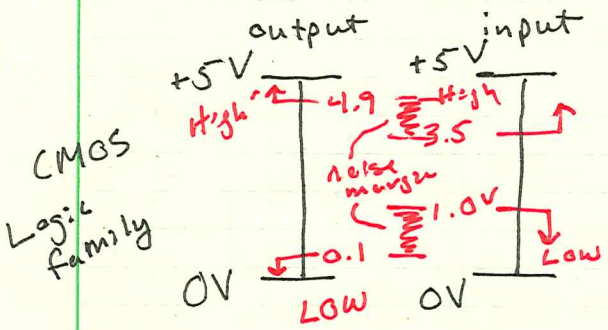
Power supplies:
- V_- = −15V
- V_+ = +15V
- V_L = 5V → "Logic"
- GND

12 L.2.2 - Chopper Circuits
Sample & hold

Then loop to beginning if you have time.

- Analog = Continuous Waveforms (e.g. sine waves)
- Digital = Discrete Values

How does this work? We still deal with voltage



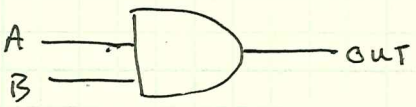
* With CMOS, we gaurantee our output will be between 4.9 to 5V or 0 to 0.1 V
* We ask that inputs lie between 0 and 1V or 3.5 and 5V

What do we do with digital signals?

Today: Combinatorial logic with logic gates.

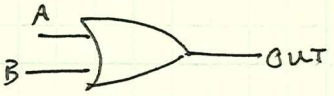
Basic Logic Gates

AND



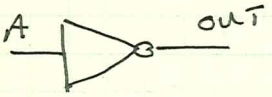
A	B	OUT
0	0	0
0	1	0
1	0	0
1	1	1

OR



A	B	OUT
0	0	0
0	1	1
1	0	1
1	1	1

NOT "inverter"



A	OUT
0	1
1	0

High: 1
Low: 0

True: High
False: Low

"Truth Table"

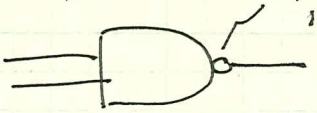
"Universal Logic Gates"

↓

Either lets us build any digital device.

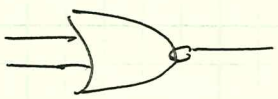
NAND

"not and"



A	B	OUT
0	0	1
0	1	1
1	0	1
1	1	0

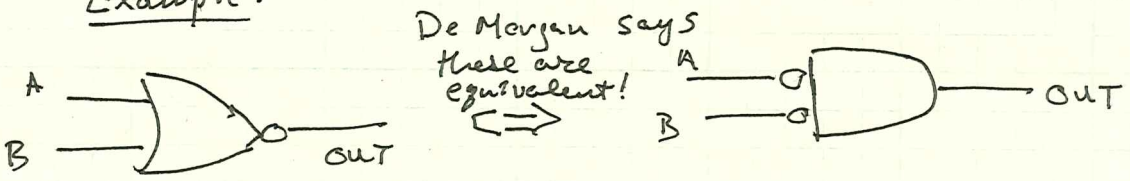
NOR



A	B	OUT
0	0	1
0	1	0
1	0	0
1	1	0

De Morgan's Theorem: You can swap shapes if at the same time, you ~~swap~~ invert all inputs and outputs:

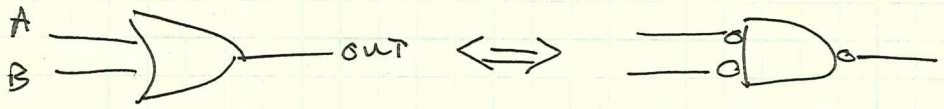
Example:



A	B	OUT
0	0	1
0	1	0
1	0	0
1	1	0

A	B	OUT
0	0	1
0	1	0
1	0	0
1	1	0

Example:

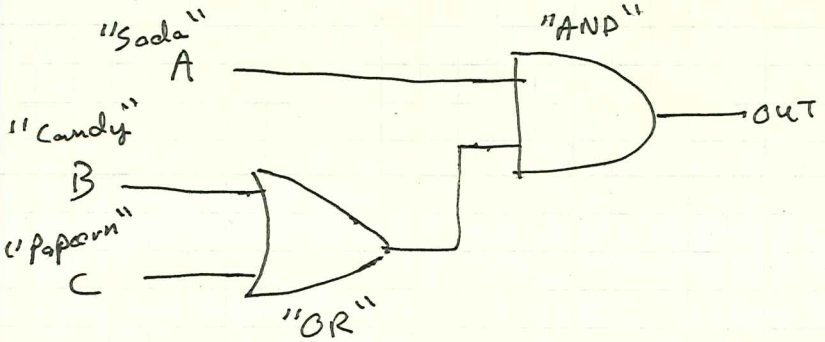


A	B	OUT
0	0	0
0	1	1
1	0	1
1	1	1

A	B	OUT
0	0	0
0	1	1
1	0	1
1	1	1

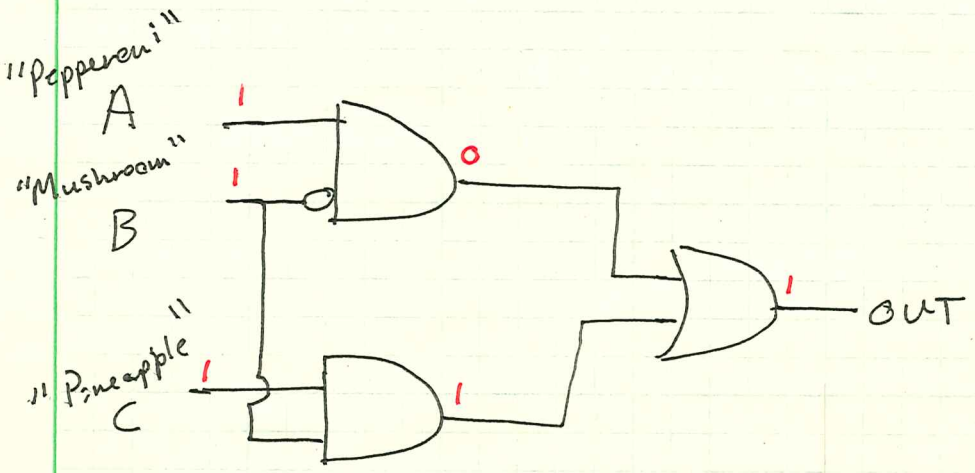
We can combine Gates together to do more complex operations.

Example #1: "Movie Theater"



A	B	C	OUT
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

Example #2 "Pizza Selector"



A	B	C	OUT	
0	0	0	0	
0	0	1	0	
0	1	0	0	
0	1	1	1	←
1	0	0	1	←
1	0	1	1	←
1	1	0	0	
1	1	1	1	←

Converting Logic Table to Logic Circuit

* "Sum of Products"
  (1) Look at all the rows for which the output is high
  (2) Take the sum of the products of the inputs, inverting inputs that are low.

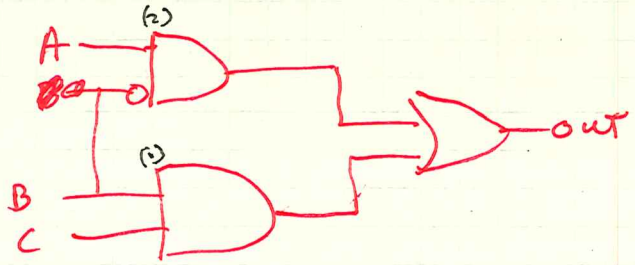
For Ex #2:

$$OUT = \underbrace{\bar{A}\cdot B\cdot C}_{\text{product}} + \underbrace{A\cdot\bar{B}\cdot\bar{C}}_{A\cdot\bar{B}\cdot(C+\bar{C})} + A\bar{B}C + ABC$$

$$\underbrace{\qquad\qquad\qquad\qquad\qquad\qquad}_{B\cdot C\cdot(A+\bar{A})}$$

$$= B\cdot C\cdot \underbrace{(A+\bar{A})}_{1} + A\bar{B}\underbrace{(C+\bar{C})}_{1}$$

$$= \overset{(1)}{B\cdot C} + \overset{(2)}{A\bar{B}}$$



Notation

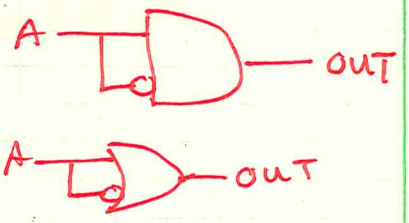
"—" means inverted
"·" mean "and"
"+" means "or"

Note:

$A + \bar{A} = 1$

$A\cdot\bar{A} = 0$

For all cases



Implement the following truth table:

Ex #3

A	B	OUT	
0	0	1	←
0	1	0	
1	0	1	←
1	1	0	

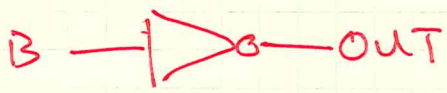
Ex #4

A	B	OUT	
0	0	1	←
0	1	0	
1	0	1	←
1	1	1	←

$OUT = \bar{A}\bar{B} + A\bar{B}$

$= (\bar{A} + A)\bar{B}$

$= \bar{B}$

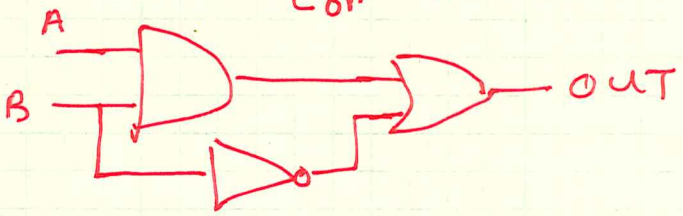


$OUT = \bar{A}\bar{B} + A\bar{B} + AB$

$= \bar{B}\cdot(A + \bar{A}) + AB$

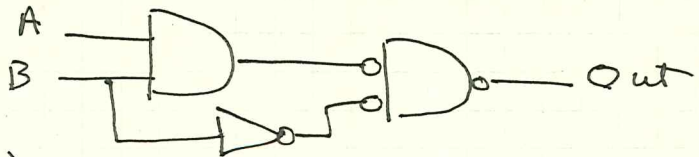
$= \bar{B} + A\cdot B$

(or — and)

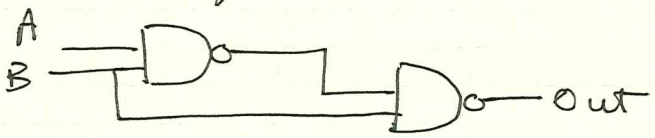


What if we want to build the circuit from Ex#4 with only NAND and NOT Gates? Use DeMorgan!

Step #1) Convert all "OR" Gates into "ANDS" Apply DeMorgan



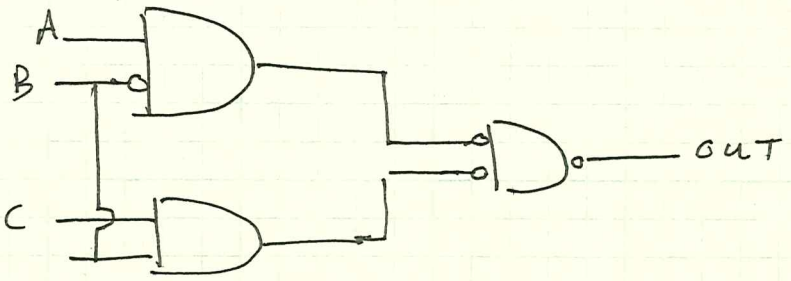
Step # 2) Simplify

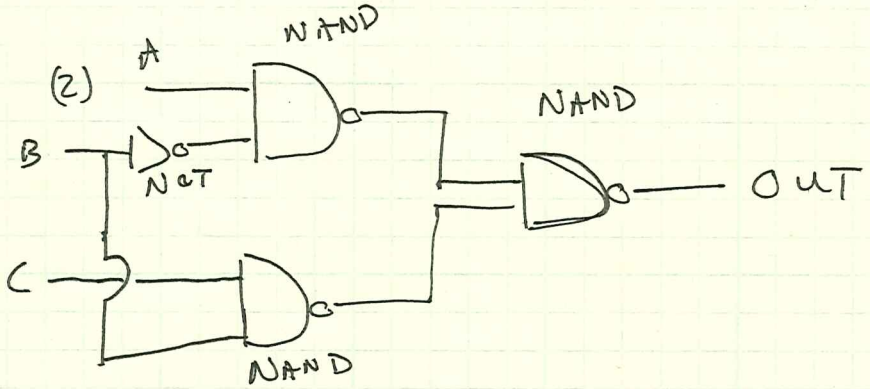


Example

Redraw Pizza Selection w/ only NANDs & Nots

(1) Apply DeMorgan to ORs

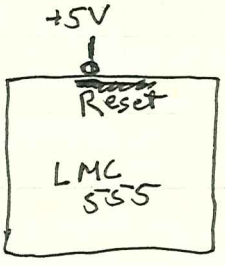




Active-Low Signals

- Action happens when signal goes low
- Signal is normally high
- Common in digital electronics due to historical reasons.
  - ↳ Assymetries in TTL
    - ↳ BJT based logic
  - ↳ Read More about 14N.5

Example



$\overline{\text{Reset}}$ was an active low signal.

Doesn't reset until we apply 0V.

"$\overline{\text{Reset}}$ is asserted" asserted = active

"$\overline{\text{Reset}}$ is not asserted"

Binary and Hexidecimal Representations

* Unsigned Binary (base-two)

Analogous to decimal (base ten)

"Most significant Bit"

MSB ... 2^4 2^3 2^2 2^1 2^0 LSB

"Least significant Bit"

Each digit is one bit

0-1 0-1

10^3 10^2 10^1 10^0

0-9 0-9

Example

Binary: 100

$1 \cdot 2^2 + 0 \cdot 2^1 + 0 \cdot 2^0$

$= 4$

Binary: 10010

$1 \cdot 2^4 + 0 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0$

$16 + 0 + 0 + 2 + 0$

$= 18$

Use 0b to denote binary

0b100 or 100_2 } denotes binary

100_{10} } decimal

* Signed Binary

- Two's Compliment
- Treat the MSB as negative and add to the rest of the number.

Example

0b1011 in 2's comp

$-2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0$

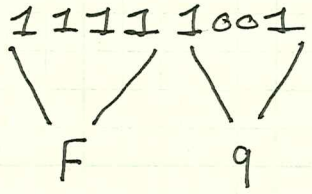
$-8 + 0 + 2 + 1$

$= -5$

Hexadecimal (base 16)

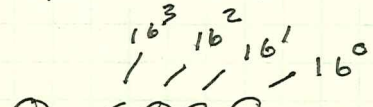
- Easier to read for large numbers.

Example:



Write as F9h or 0xF9

Note: A = 10, B = 11, C = 12, D = 13, E = 14, F = 15



Example: What is 0xC000 in binary?

1100 0000 0000 0000

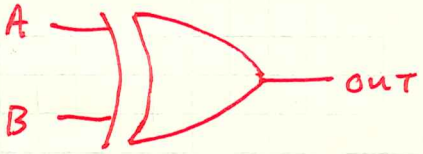
What is 0xC000 in decimal? (using unsigned binary)

$$2^{15} + 2^{14} = 49,152_{10}$$

What is 0xC000 in Two's comp? (decimal way using)

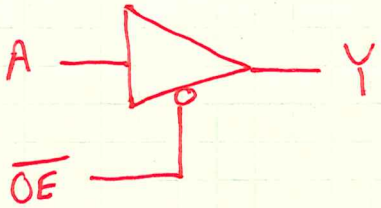
$$-2^{15} + 2^{14} = -16384_{10}$$

(1) XOR Gate (74HC86) — "exclusive OR"



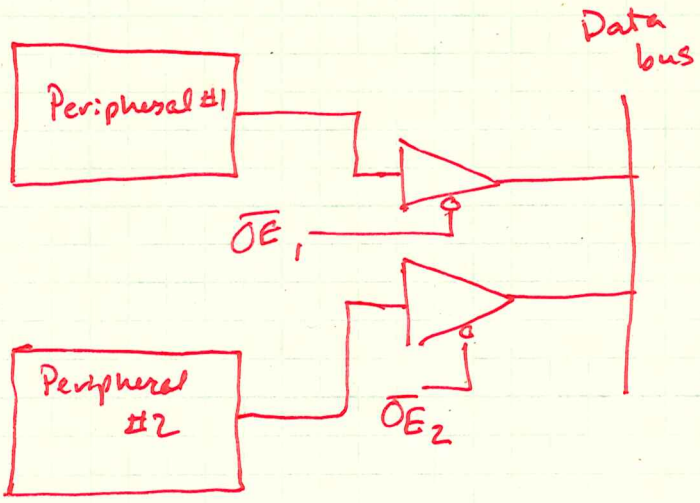
A	B	OUT
0	0	0
0	1	1
1	0	1
1	1	0

(2) Three-State Buffer (74HC125) a.k.a. "Tri-State Buffer"



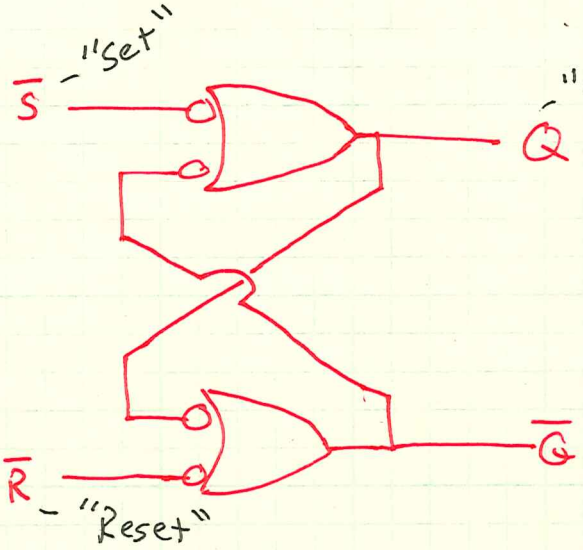
A	$\overline{OE}$	Y
1	0	1
0	0	0
1	1	Hi-Z — "High Impedance" open-circuit
0	1	Hi-Z

Example: ~~Class~~ Use case: When digital devices share an output



(3) S-R Latch (74HC279) "Set-Reset Latch"

- Simplest "flip-flop"
- Flip-flop: A digital circuit with two stable states
  Can be used to store information
  Foundation for memories, counters, processors, etc.



Output fed back to input: sequential logic

$\bar{S}$	$\bar{R}$	Q	$\bar{Q}$
1	1	Q_p	$\bar{Q}_p$
0	1	1	0
1	0	0	1
0	0	invalid	* avoid

- Interesting Case: $\bar{S}, \bar{R} = 1$

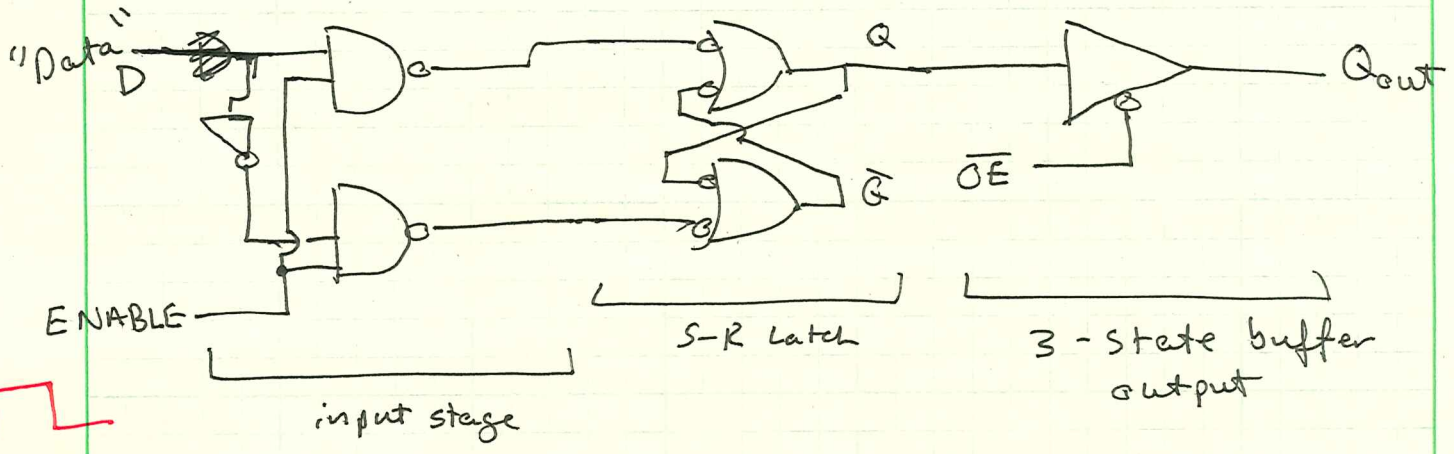
Q_p: "Q previous"

$$\begin{matrix} Q=0 \\ \bar{Q}=1 \end{matrix} \Rightarrow \begin{matrix} Q=0 \\ \bar{Q}=1 \end{matrix}$$

$$\begin{matrix} Q=1 \\ \bar{Q}=0 \end{matrix} \Rightarrow \begin{matrix} Q=1 \\ \bar{Q}=0 \end{matrix}$$

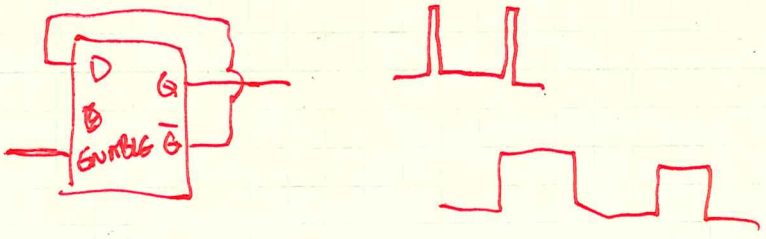
- We can set some value into Q and $\bar{Q}$ and then keep it latched by disasserting $\bar{S}$ and $\bar{R}$. "Memory Effect"

(9) Transparent D-Latch (HC573)



D	ENABLE	$\overline{OE}$	Q_{out}
1	1	0	1
0	1	0	0
1	0	0	$Q_{previous}$
Memory State: 0	0	0	$Q_{previous}$
X	X	1	Hi-Z

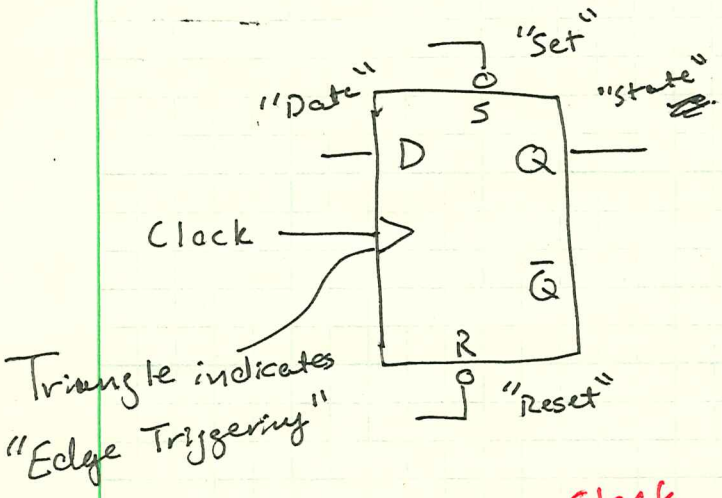
- Note: This device takes action when ENABLE goes HIGH. This is an example of "Level triggering"

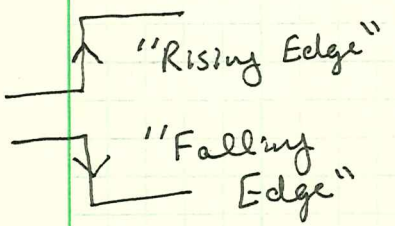


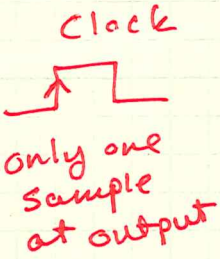
(5) D flip-flop (74HC74)

Money

Note: Ch. 8 pg. 508



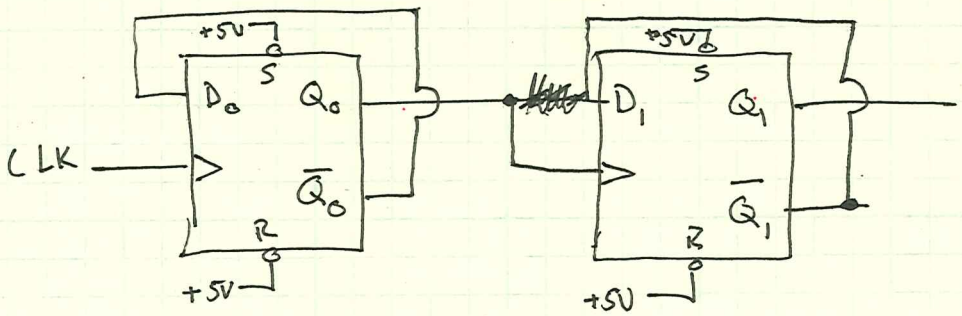




How it works

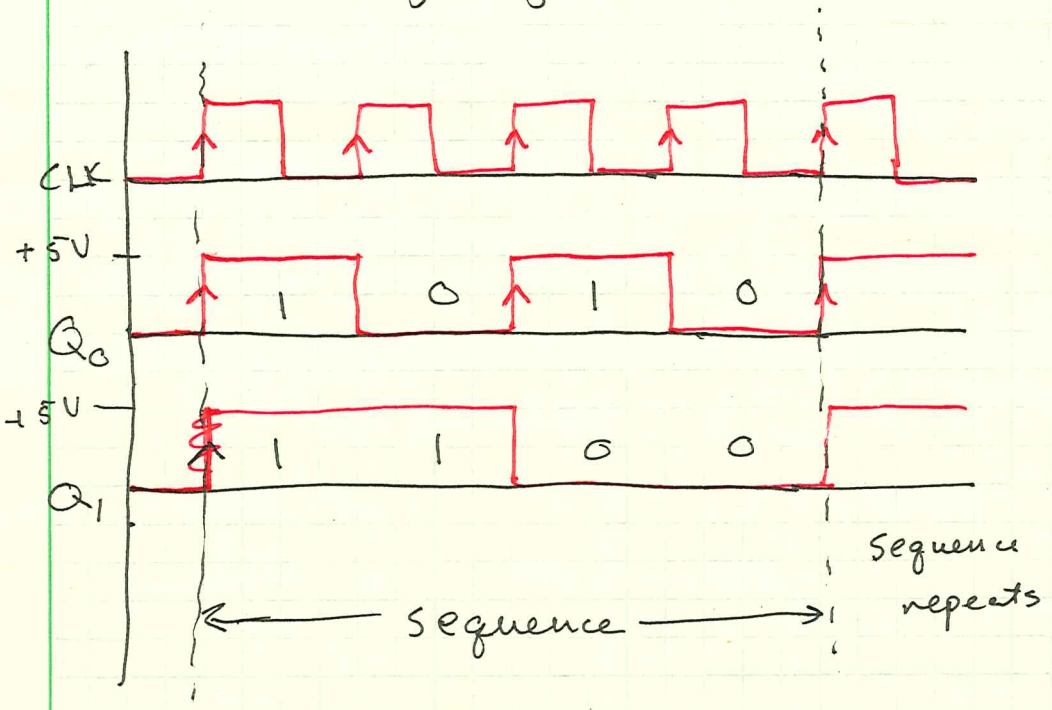
- At a rising clock edge, Data gets passed through to Q (the output)
- $\bar{Q}$ is always the inverse of Q
- Asserting Set drives Q high
- Asserting Reset drives Q low

(6) Ripple Counter



- Clock: Anything with positive (rising) edges
- Assume Q_0 and Q_1 start at 0. What happens

Draw a timing diagram:



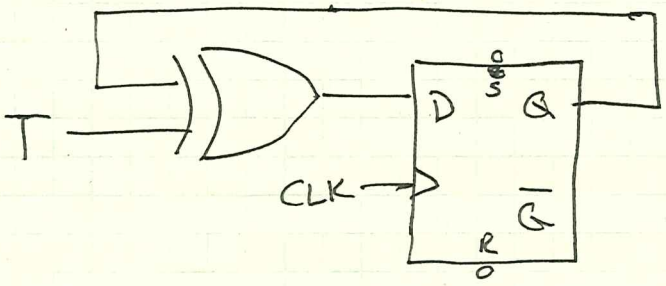
Another View

	Q_1	Q_0
3	1	1
2	1	0
1	0	1
0	0	0

t ↓

- This circuit counts to 4!
- This circuit divides frequency by 4!
- The issue w/ this ckt is that Q_1 doesn't change until Q_0 does. This is called "ripple delay"
  - ↳ Slows us down (i.e. limits speed)
  - ↳ Produces bad transient (in-between) states.

(7) T-flop "Toggle flop"



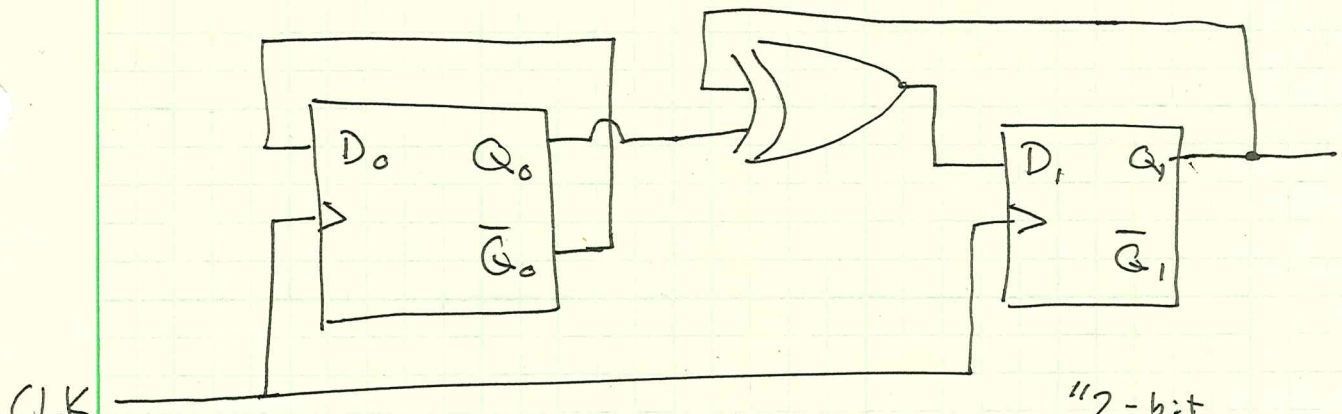
How does it work?

When we get a positive Clock edge:

- If T is Low, Q remains same
- If T is High, Q Toggles
  - (switches state)

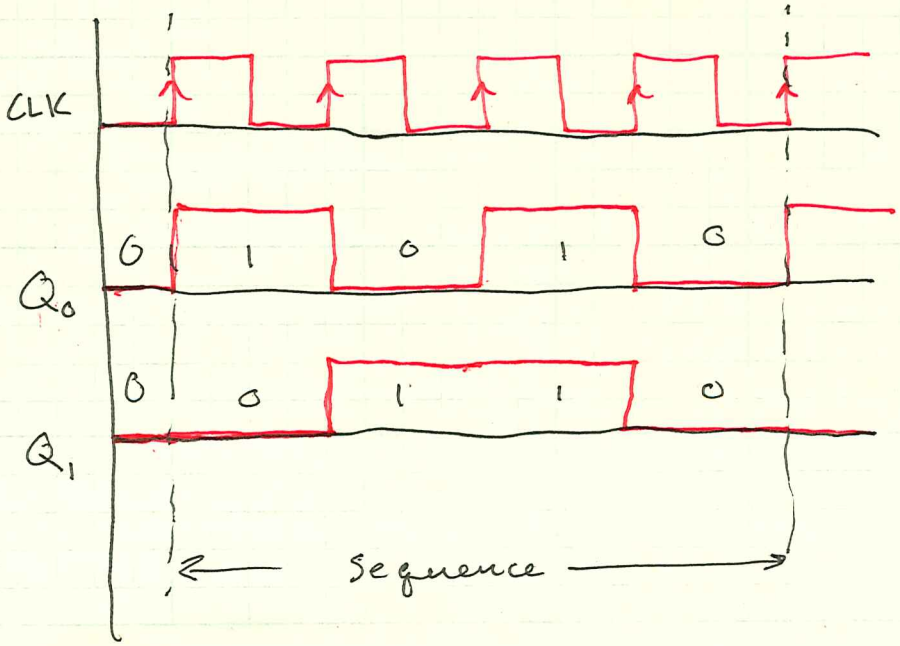
(8) Synchrous Counter

- everything happens ~~at~~ with the same clock signal



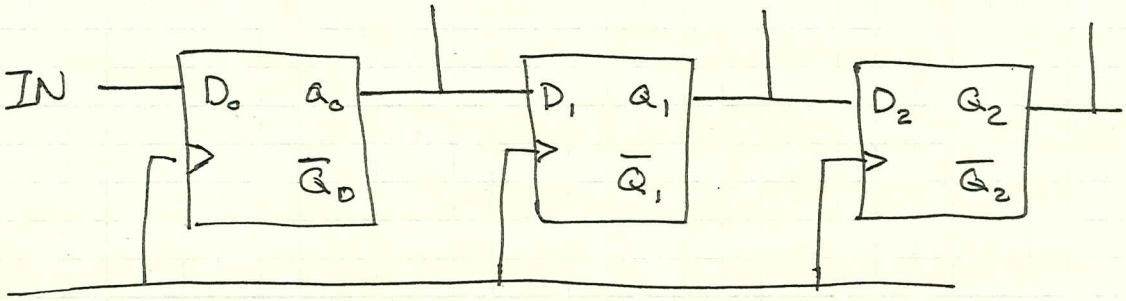
"2-bit Synchronous Up Counter"

Q_1	Q_0
0	0
0	1
1	0
1	1

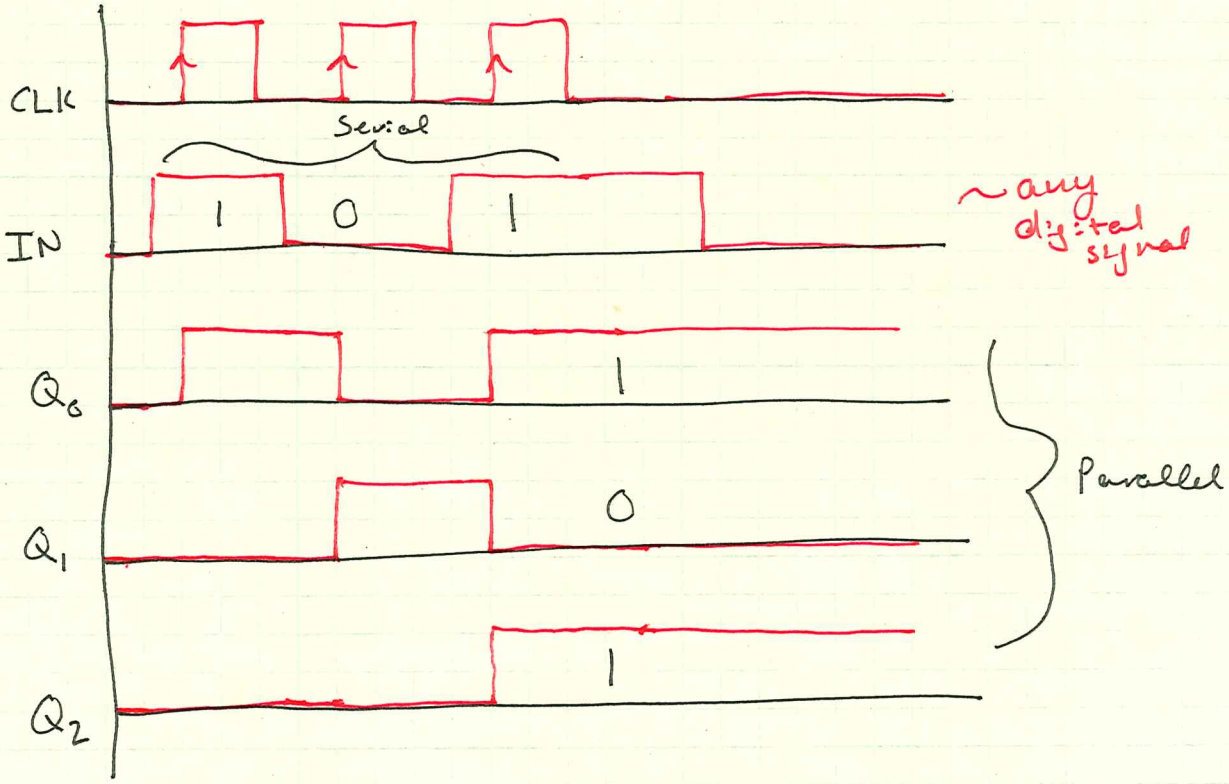


- Note, we looked slightly back in time to get value for D that will be passed on to Q
- This is called "setup time"
  $t_{setup} \approx 16$ ns for 74HCT4

(9) Shift Register



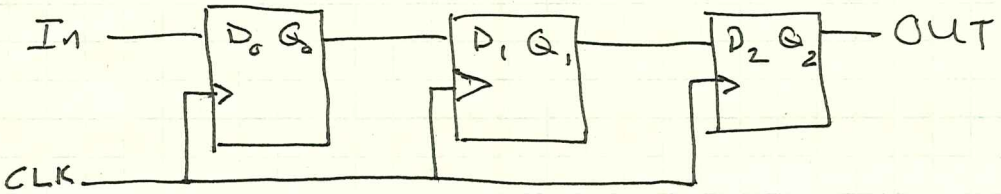
- Lets you do basic "serial to parallel" conversion and "parallel to serial" conversion



Counters Ch.16 | Physics 123 | 7/22

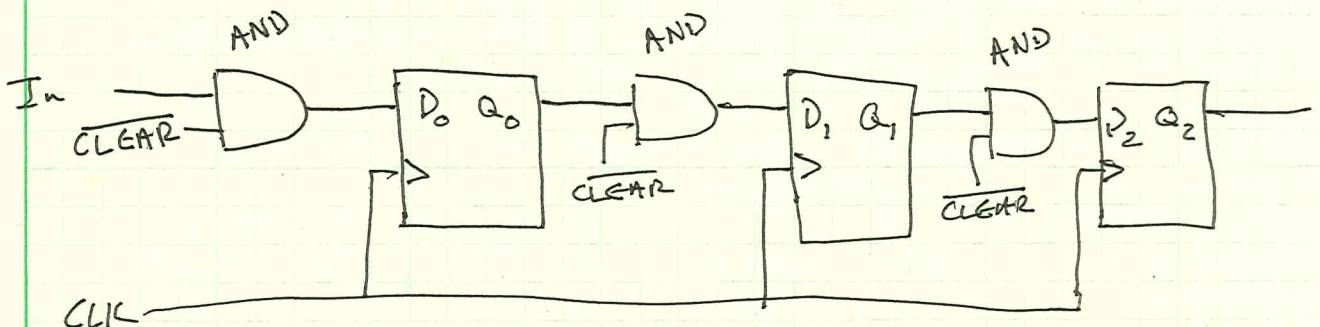
Warm-Up

(1) Given a 3-bit shift register:

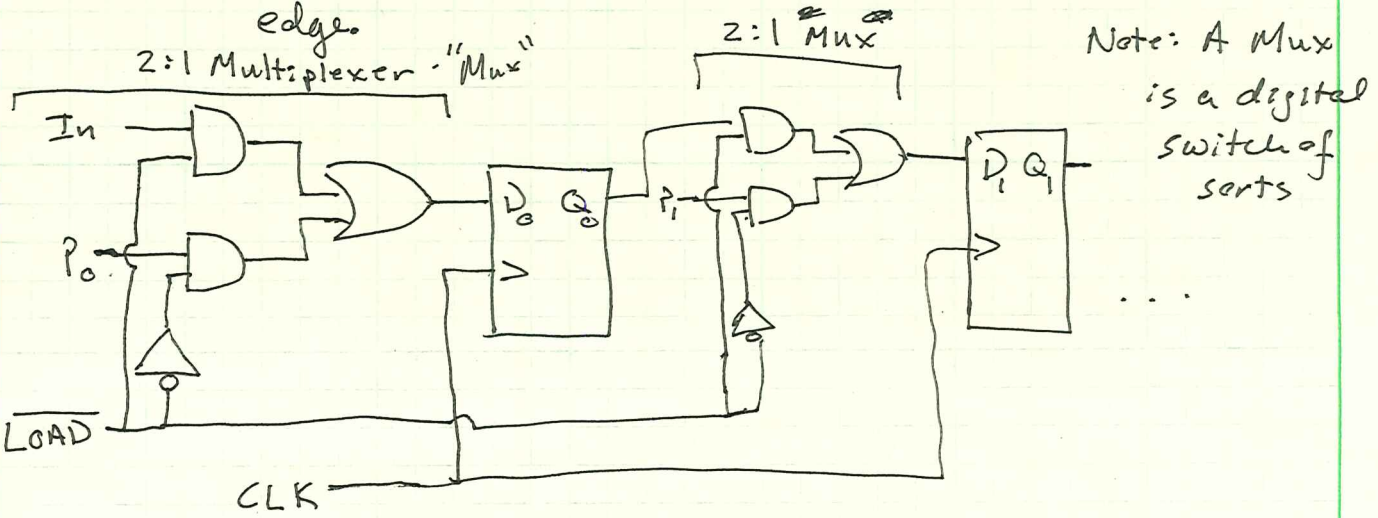


(a) Add a synchronous CLEAR function controlled by an active-low signal $\overline{\text{CLEAR}}$

⇒ On next positive clock edge, all Q's go to LOW



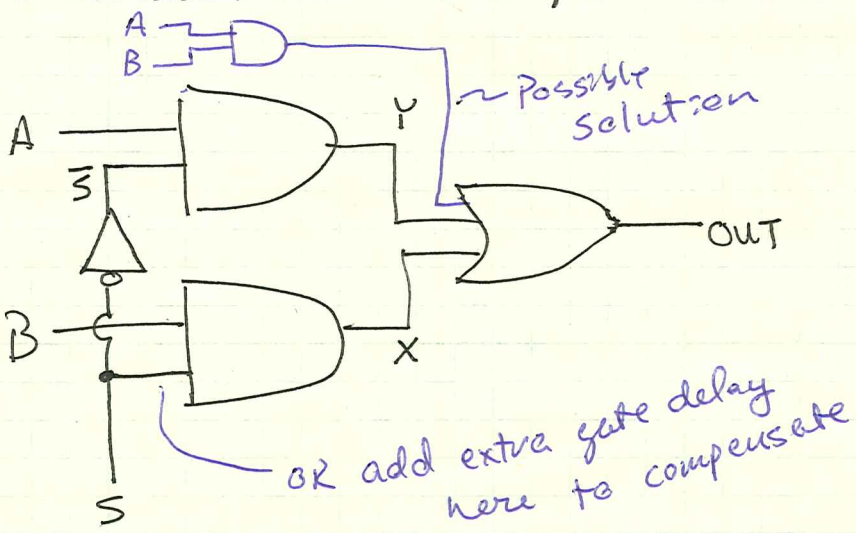
(b) Add a synchronous LOAD Function controlled by active Low signal $\overline{\text{LOAD}}$, that loads values P_0, P_1, and P_2 at the next positive clock edge.



Note: A Mux is a digital switch of sorts

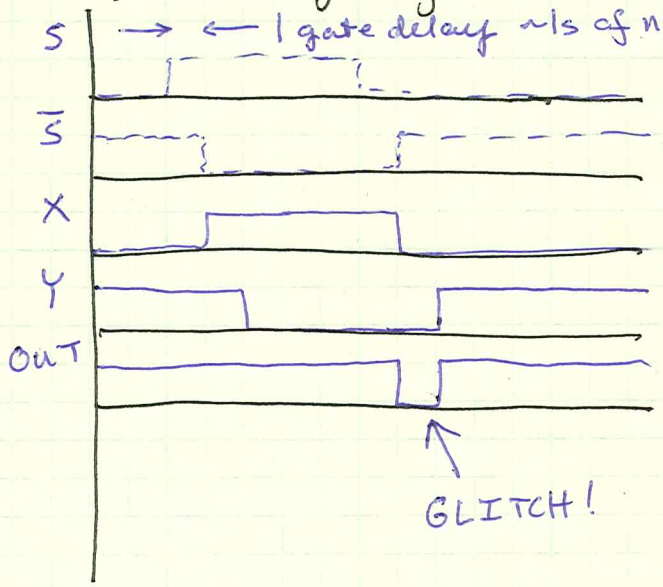
(3) Timing Glitches in Combinatorial Logic

Consider the 2:1 Multiplexer:



Assume A = HIGH, B = HIGH ⇒ OUT should be HIGH

Draw Timing Diagram



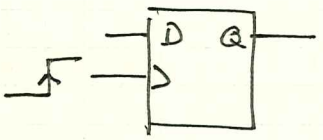
Note: Gate delay due to
- wave propagation in wires
- RC charging in gates

Takeaways:

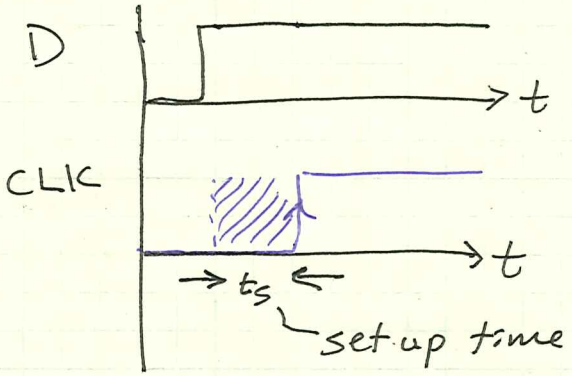
- If transients (glitches) at output could be an issue, then it's important to look at dynamics of behavior, not just combinatorics
- Timing Diagram is a great debugging tool!

Edge-Triggered ✓

(4) Flip Flop Subtlety: Set up Time

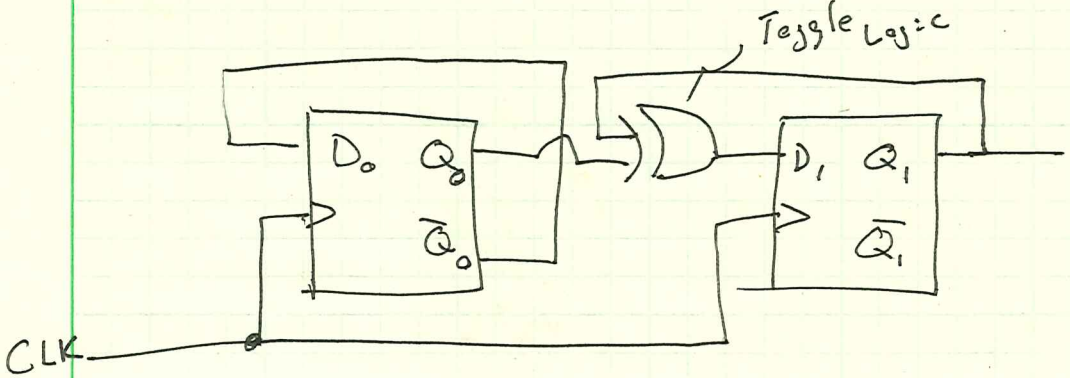


- The time required for a change in D level to work its way into the flop
  - ↳ comes from gate delay inside the flop



- D must be constant during the set up time!
- 74HC74 ~20-30 ns
- Set up Time sets a limit on frequency of operation

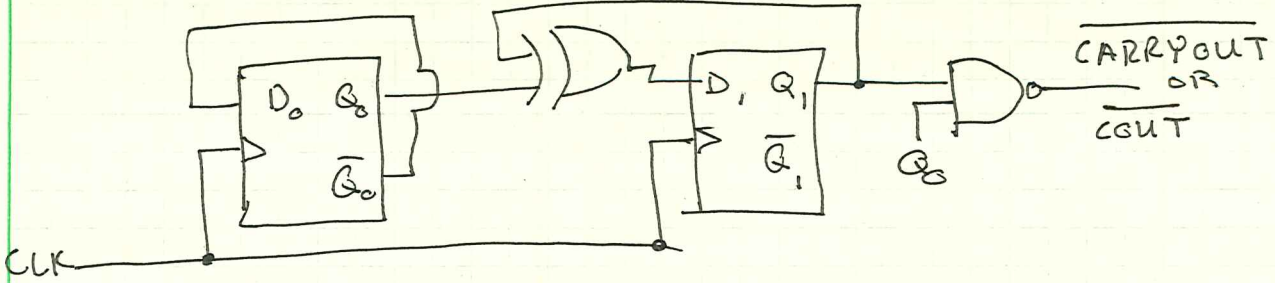
(5) From last time: 2-bit Synchronous Up Counter



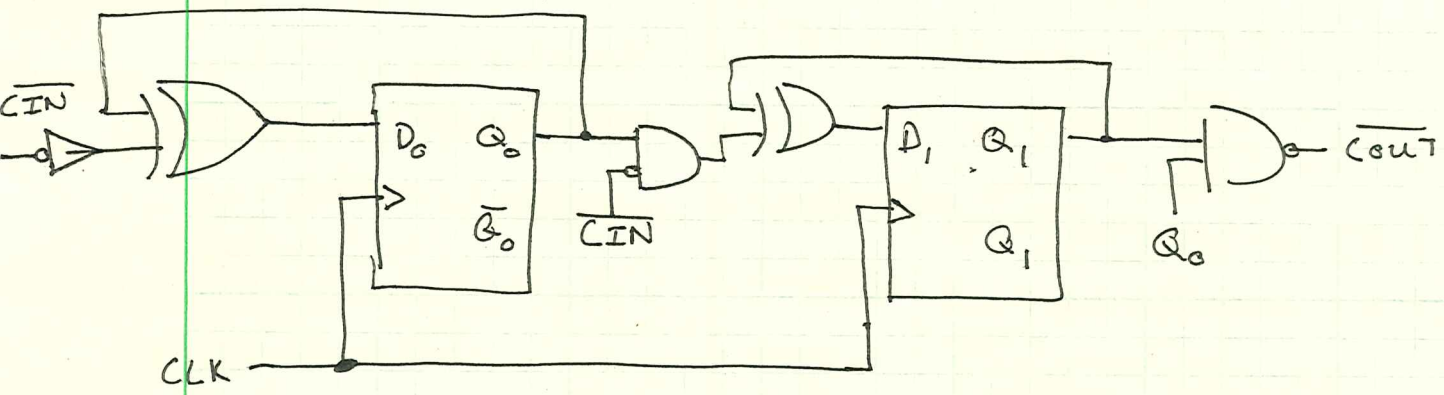
- 2 bits ⇒ 2 flip flops
- Counts from 0 to 3
- Q_0 is ÷ 2
- Q_1 is ÷ 4

First Evolution: Output to tell us when the counter is full! "CARRY OUT"

* Counter is full means Q_0, Q_1 = HIGH

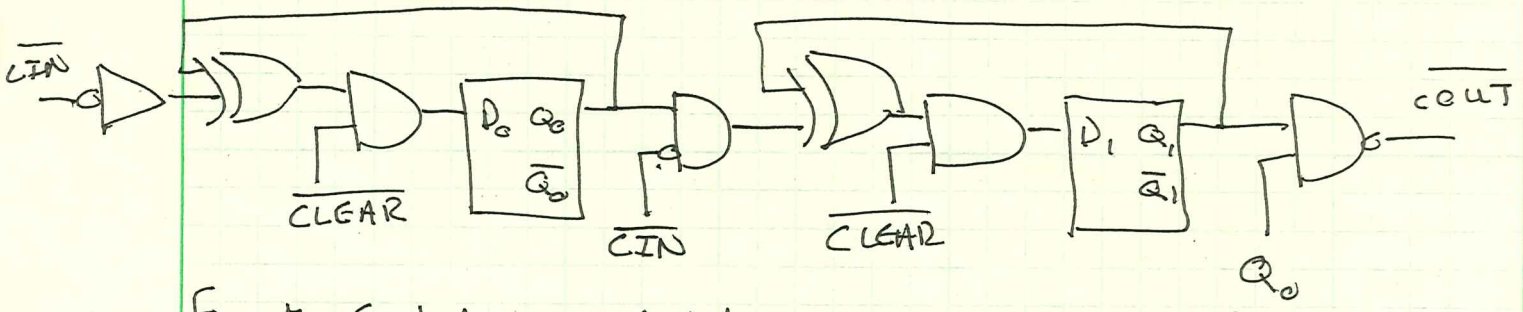


Second Evolution: Control input to tell us to start counting: "Carry In"

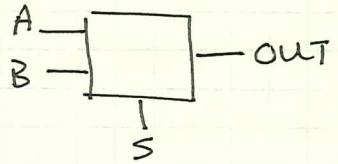


Third Evolution: Add a way to synchronously clear all the states.

* Use active low $\overline{CLEAR}$

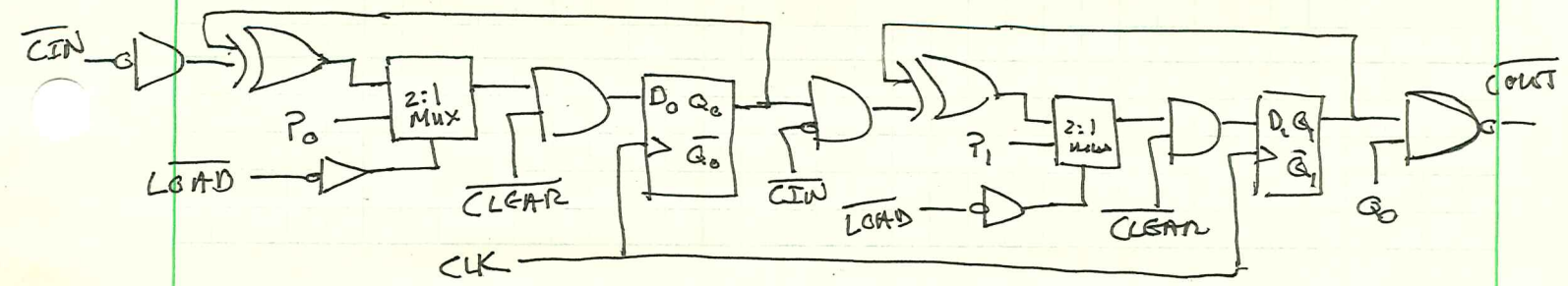


Fourth Evolution: Add a synchronous load function controlled by $\overline{LOAD}$ that loads values P_0 & P_1 into Q_0 & Q_1, respectively.



let $S : B$

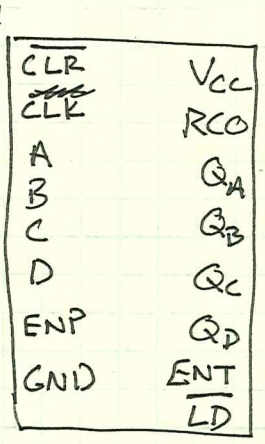
$\overline{S} : A$



Fifth Evolution: Add more flip flops (bits)

- Make sure only to allow toggle after lower bits are full
- Add all bits to the carry out logic

We don't build this from raw components, we use 74HC163, which implements these functions.



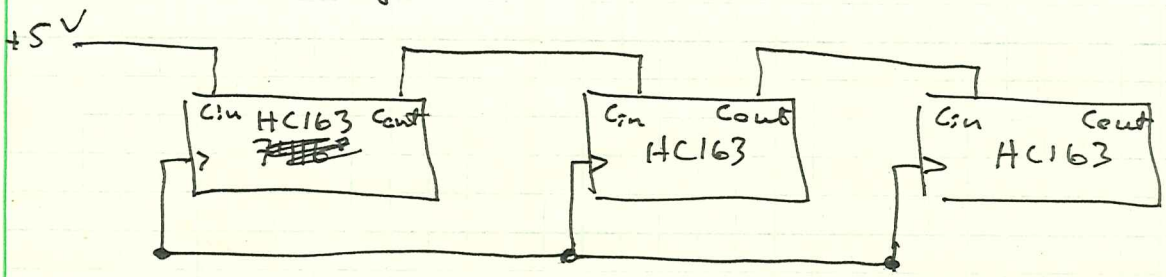
- 4 bit counter
- Counts to 16!

- $\overline{CLR}$: Synchronous Clear
  - => Drives all Q's LOW at next pos. clock edge.
- CLK: Clock
- A, B, C, D: Parallel load inputs
  - => Will get loaded into Q_A, Q_B, Q_C, Q_D on next positive clock edge after $\overline{LD}$ is asserted
- RCO: Carry Out
  - => Goes HIGH immediately after Q_A, Q_B, Q_C, Q_D all are HIGH
  - "Ripple Carry Out"
- ENP, ENT: Carry In
- GND: Ground
- V_{CC}: +5 V
- Q_A, Q_B, Q_C, Q_D: Output Bits
  - Q_A → LSB
  - Q_D → MSB
- $\overline{LD}$: Synchronous LOAD

Note: 74HC161: Identical except has an asynchronous clear

Counter Examples:

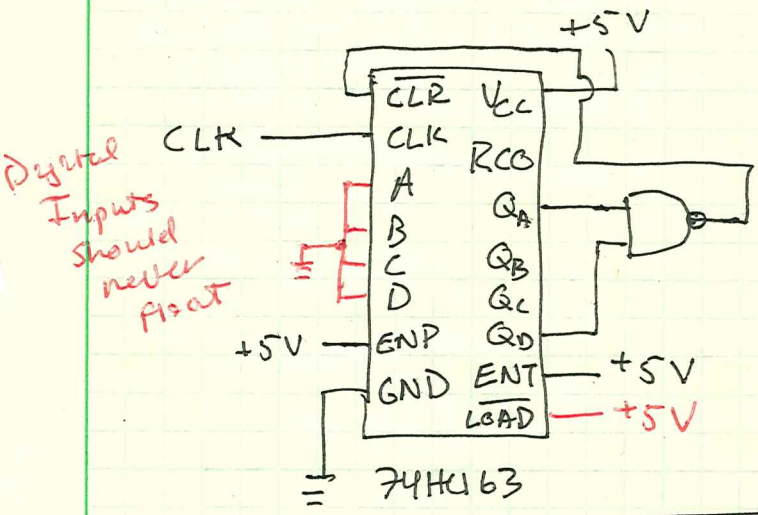
(a) Cascading Counters



4 bit + 4 bit + 4 bit = 12 bits

Counts to $2^{12} = 4096$

(b) Counting to strange numbers



Can you make it count to ten?

START: 0_{10} : 0000

END: 9_{10} : 1001

$Q_D Q_C Q_B Q_A$

START: 0000

0001

0010

0011

0100

0101

0110

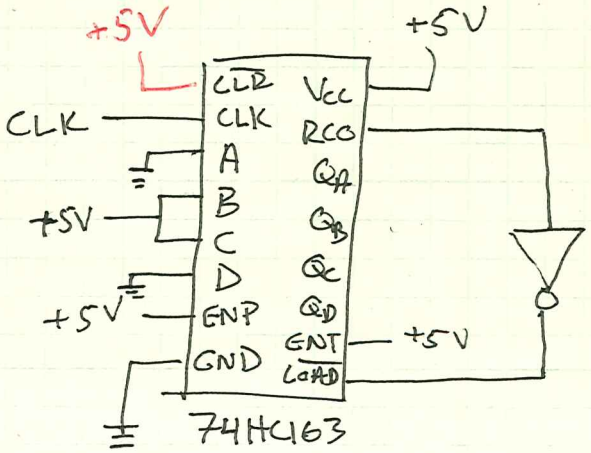
0111

1000

1001 → ASSERT $\overline{CLR}$

ON NEXT CLOCK EDGE

(c) Another method to count to ten:



START: 6_{10}

END: 15_{10}

$Q_D Q_C Q_B Q_A$

0110 6_{10}

0111

1000

1001

1010

1011

1100

1101

1110

1111 15_{10}

ON NEXT CLOCK EDGE

$\overline{LOAD}$ asserted

RCO HIGH

Today in Lab: 16-bit counter

* Will be "address counter" for your computer
* Made from Programmable Array Logic – not from HC163's
* A few extra features & more than the '163:
  - 3-state outputs (controlled by $\overline{OE}$ "output enable")
  - Asynchrous LOAD (controlled by $\overline{LD}$, contrast with $\overline{SYNC\ LD}$)
  - UP/Down Counting Control (Controlled $\overline{UP}$/Down)

(1) Finite State Machine

- A device or program that walks through a predictable sequence of states
- Gives us a formalized way to design and build complex sequential circuits
- In hardware, built from flip-flops and combinatorial logic (logic gates)
  - ↳ Q's of the flip flops combine to form state
  - ↳ n flip flops allows 2^n states
- A great tool for computer programming as well!

Example: • Sketch a design for a 2-bit sync UP/$\overline{\text{DOWN}}$ Counter.

If UP/$\overline{\text{DOWN}}$ = 1, should count up
= 0, should count down

- Should include an async clear
- Should include $\overline{\text{CARRYOUT}}$ that is asserted at highest state when counting up and lowest state counting down.

(a) How many states do we need?

Count to 4 => four states

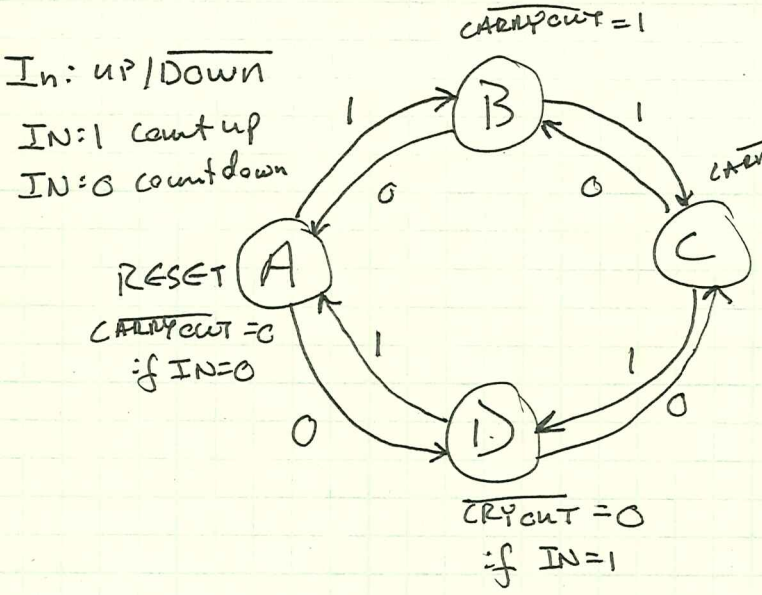
How many flip-flops?

$2^n \geq 4$ states

$n = 2$ flip flops

(b) Draw a directed graph for the FSM. Mark the reset state with a note. Show outputs for each state.

Note: * graphs are an important concept in discrete mathematics.
* graphs have nodes and edges
* in directed graphs, edges have direction



(c) Put Directed Graph into Present State / Next State Table

PS	NS IN=1	NS IN=0	CARRY OUT
A	B	D	0 : if IN = 0
B	C	A	1
C	D	B	1
D	A	C	0 : if IN = 1

(d) Assign Binary Values to the States

	Q_1	Q_0
A	0	0
B	0	1
C	1	0
D	1	1

- Somewhat left to your choosing but considers choices made previously (Reset and output choices)

Redraw PS/NS Table in terms of binary values:

PS	NS	
	In = 1	In = 0
$Q_1\ Q_0$	$D_1\ D_0$	$D_1\ D_0$
0 0	0 1	1 1
0 1	1 0	0 0
1 0	1 1	0 1
1 1	0 0	1 0

(e) Find D_0 & D_1 in terms of Q_1, Q_0, and IN.

- Use Sum of Products

$$D_0 = \overline{Q_0}$$

$$D_1 = \overline{Q_1}Q_0\,IN + Q_1\overline{Q_0}\,IN + \overline{Q_1}\,\overline{Q_0}\,\overline{IN} + Q_1Q_0\,\overline{IN}$$

$$= Q_1\left(\overline{Q_0}\,In + Q_0\overline{IN}\right) + \overline{Q_1}\left(Q_0\,IN + \overline{Q_0}\,\overline{IN}\right)$$

($Q_0\overline{IN}$: XOR ≠ ; $\overline{Q_0}\,\overline{IN}$: NOT XOR ≠)

Note: $A\overline{B} + \overline{A}B = \overline{A}\overline{B} + AB$

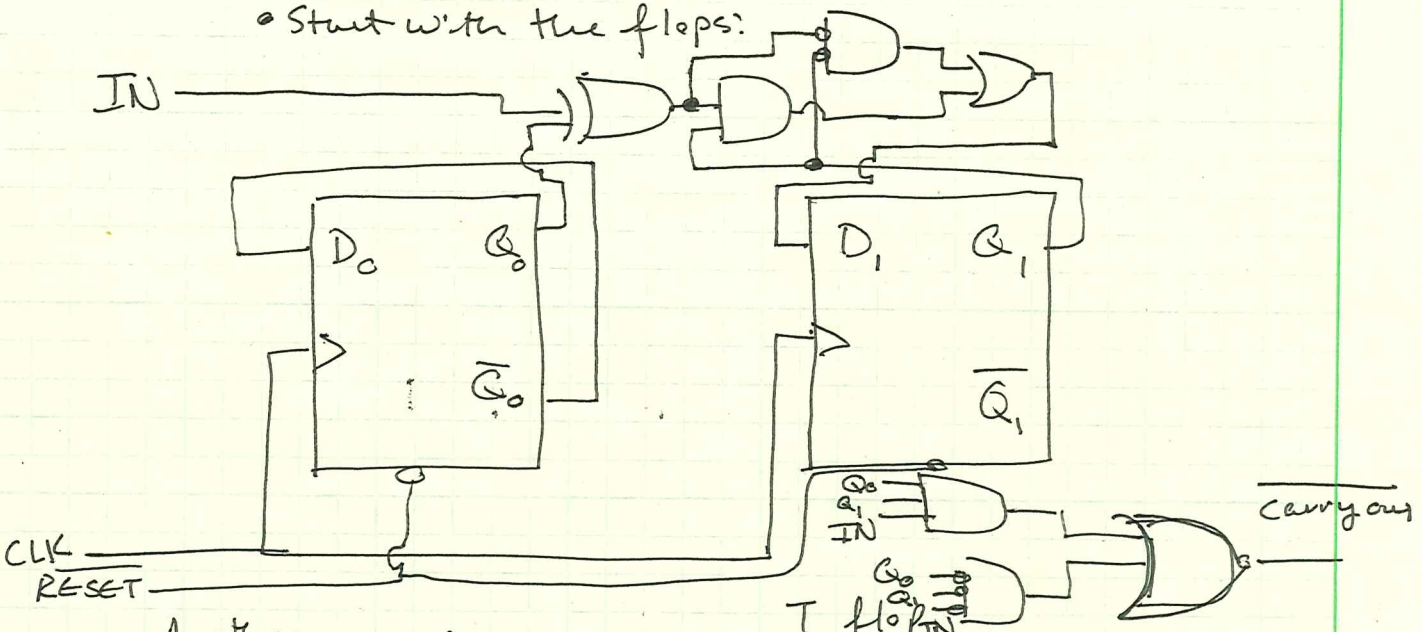
Q_1 XOR'D with XOR of IN & Q_0

(f) Show the combinatorial function for the output:

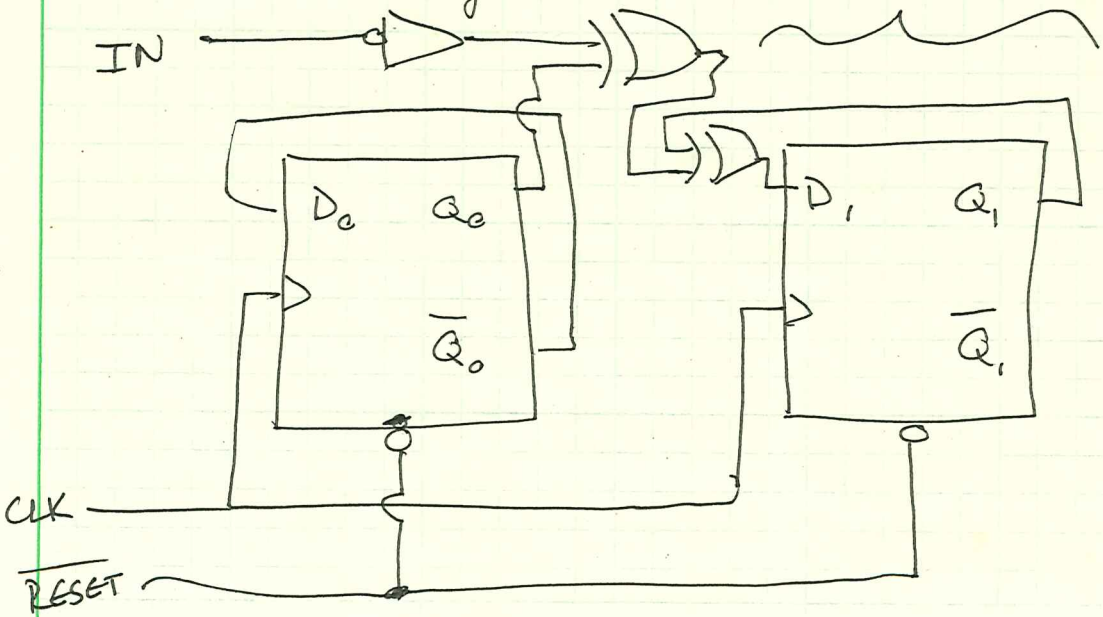
$$\overline{CARRY_{OUT}} = \overline{Q_0 Q_1 IN + \overline{Q_0}\,\overline{Q_1}\,\overline{IN}}$$

(g) Draw the circuit!

- Start with the flops:



Another way:

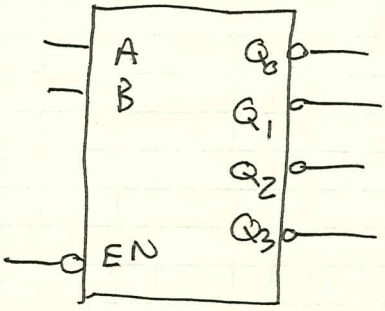


(2) Combinatorial Devices

(2:) Decoders! ⇒ A binary value is the code

n-digits can represent 2^n numbers

Chooses its output based on the input code



74HC 139

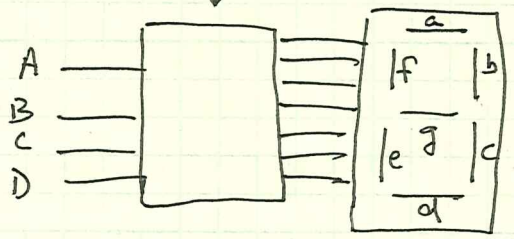
2:4 encoder

(code)

$\overline{EN}$	A	B	Q_0	Q_1	Q_2	Q_3
0	0	0	0	1	1	1
0	0	1	1	0	1	1
0	1	0	1	1	0	1
0	1	1	1	1	1	0
1	X	X	1	1	1	1

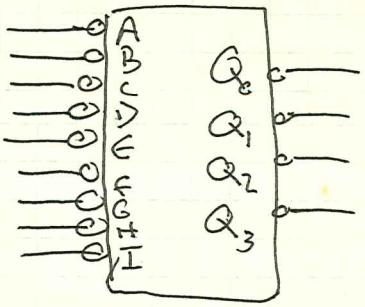
Another Decoder:

74LS247 : Binary to Seven-Segment Display Decoder



inputs from 0_{10} to 9_{10}

(2:) Encoders: Set an output code based on which input is asserted



74HC147

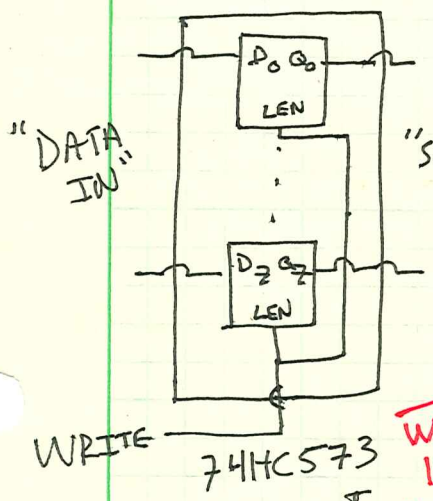
10:4 encoder

- Output is 1111 if no input is asserted
- Inputs have ranking or priority system. "I" has highest priority.

(3) Memory: • An array of flip flops (or other device that can store a bit of information)

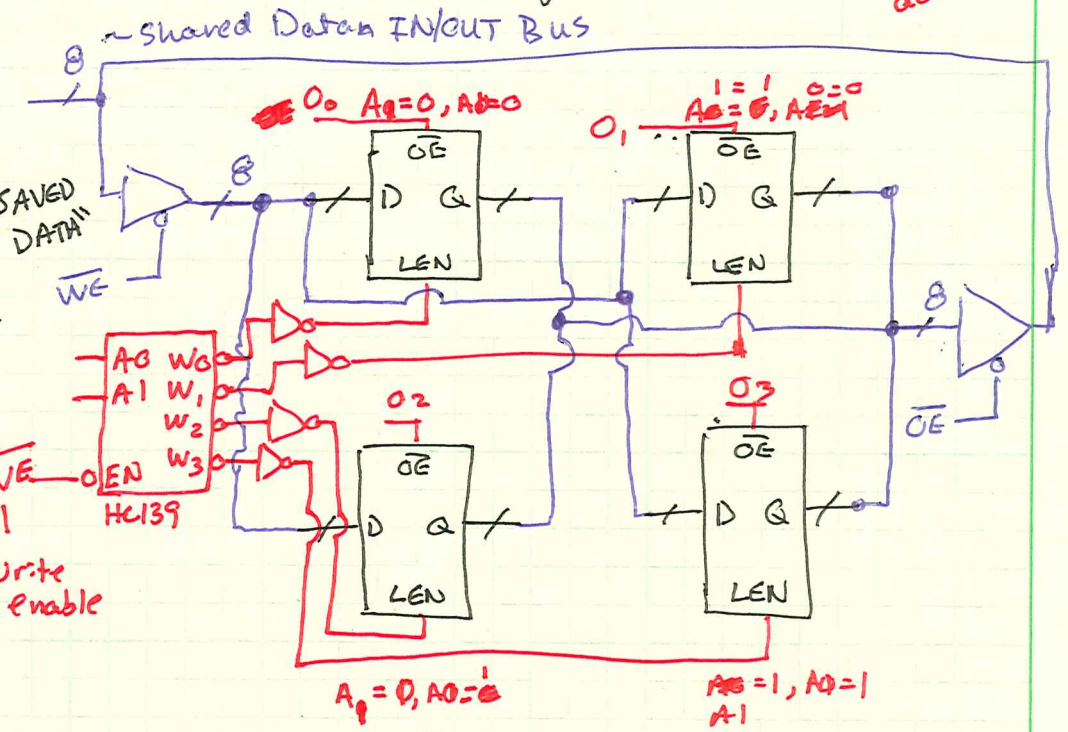
• Organized in a way to limit the number of physical lines needed to store and access the information.

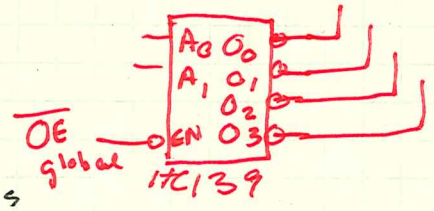
(a) Building Block "Register"



(b) Tile the building blocks

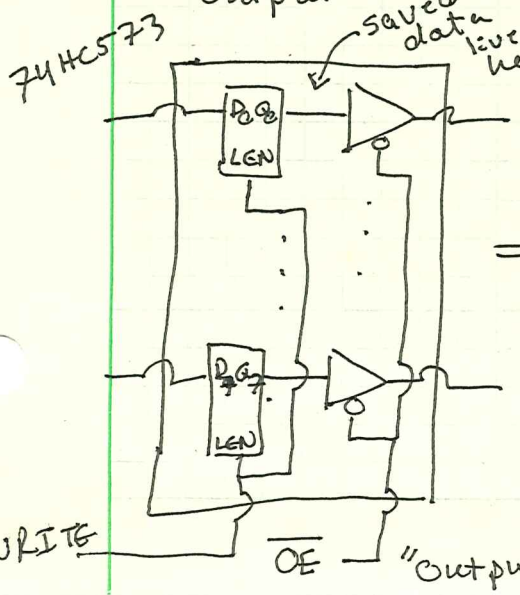
• Each register has an address!

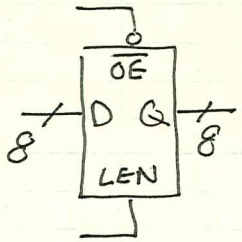




• 8 flip flops so 8 bits of memory

• Has 3-state output





• Only one register outputs to the data line at a given time.

• Internal logic prevents mutual assertion of $\overline{OE}$ & $\overline{WE}$

• Only write or read — can't do both at same time.

Summarize:

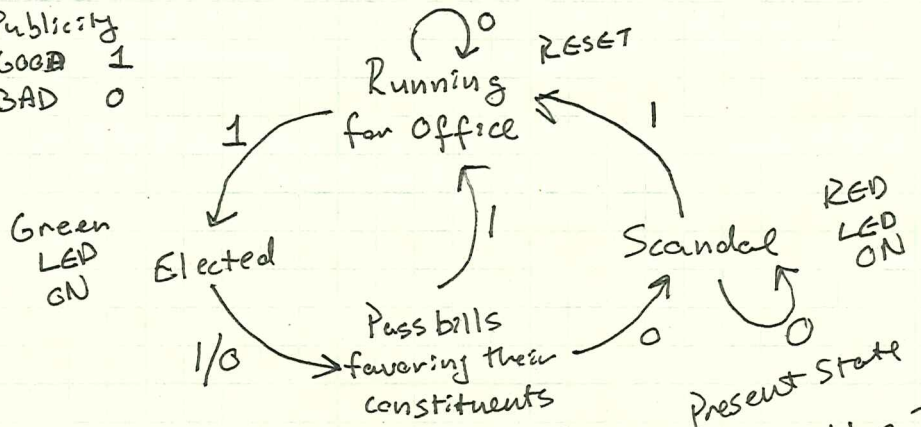
- 2 rows and 2 columns
- Each register has 8 bits of memory
- 4 x 8 bit memory = 32 bits!
- 2 address lines (A_0 & A_1)
- 8 data lines
- WE* controls writing, OE* controls output

In lab today

- 64 rows and 512 columns of 8-bit registers
  - ↳ 32K x 8 bit memory
- 15 address lines ($A_0, A_1, \ldots A_{14}$)
- 8 data lines
- OE* Output enable
  CS* Chip Select
  WE* Write enable.

State Machine Example:

IN: Publicity
GOOD 1
BAD 0



	$Q_1 Q_0$
A: RESET	00
B: ELECTED	11
C: PASS BILLS	01
D: SCANDAL	10

(P.S. = present state, N.S. = next state)

P.S.	N.S. IN=1	N.S. IN=0
A	B	A
B	C	C
C	A	D
D	A	D

P.S. $Q_1 Q_0$	N.S. IN=1 $D_1 D_0$	N.S. IN=0 $D_1 D_0$
00	11	00
11	01	01
01	00	10
10	00	10

Find D_0 & D_1 in terms of Q_0, Q_1 & IN

$$D_0 = IN \cdot \bar{Q}_1 \bar{Q}_0 + IN\, Q_1 Q_0 + \overline{IN} \cdot Q_1 Q_0$$

$$= IN\bar{Q}_1\bar{Q}_0 + Q_1 Q_0 (IN + \overline{IN}) \quad (IN + \overline{IN} = 1)$$

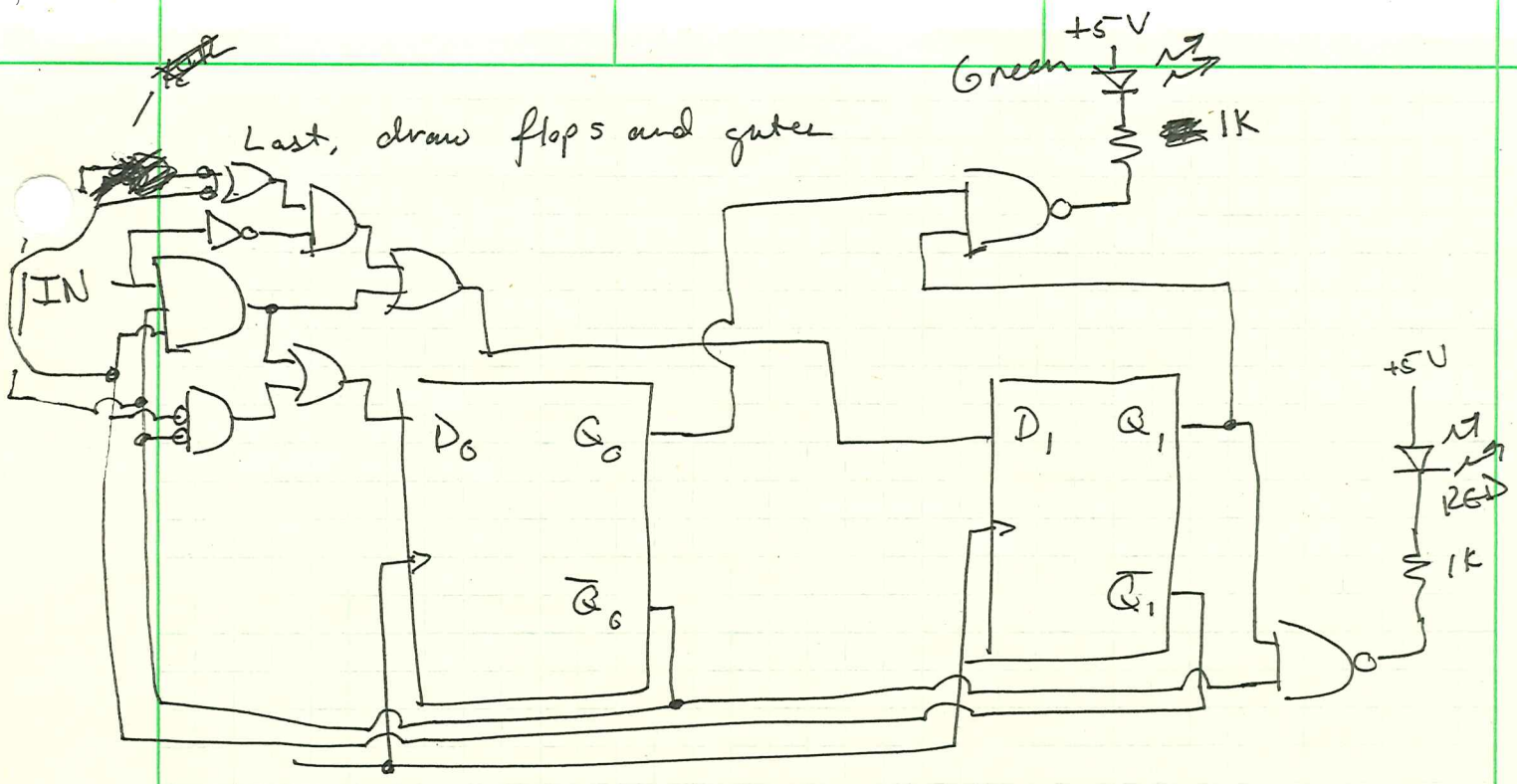
$$= IN\bar{Q}_1\bar{Q}_0 + Q_1 Q_0$$

$$D_1 = IN\,\bar{Q}_1\bar{Q}_0 + \overline{IN}\,\bar{Q}_1 Q_0 + \overline{IN}\,Q_1\bar{Q}_0$$

$$= IN\bar{Q}_1\bar{Q}_0 + \overline{IN}(\bar{Q}_1 Q_0 + Q_1\bar{Q}_0)$$

$\bar{Q}_1 Q_0 + Q_1\bar{Q}_0 = Q_1 \oplus Q_0$ → XOR

Last, draw flops and gates



(1) Digital to Analog Conversion

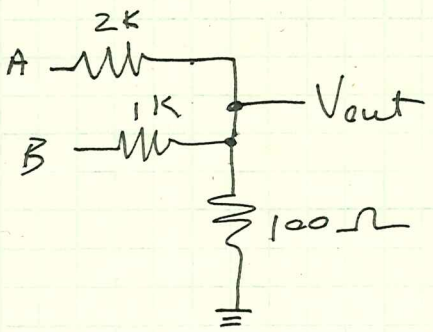
- The goal: Create an analog signal proportional to a digital code.
- The ideal: 2-bit 10V full scale DAC (digital to analog converter)

Input Code	Output Analog
00	0V
01	3.33V
10	6.66V
11	10V

For illustration, 3-bit 10V full scale

Input Code	Output Analog	
000	0	note rounding
001	1.4	
010	2.8	
011	4.2	
100	5.6	
101	7	
110	8.4	
111	10V	

- Method 1: Binary Weighted Resistor DAC

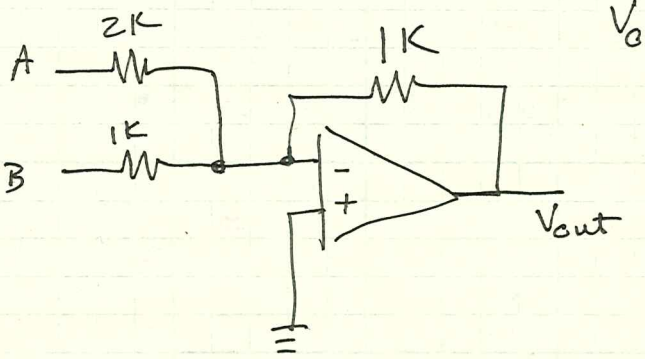


A	B	Vout
0	0	0
+5V	∄	$\frac{1}{21}5V \approx 0.24V$
∄	+5V	$\frac{1}{11}5V \approx 0.45V$
+5V	+5V	$\frac{3}{23}5V \approx 0.65V$

$$V_{out} = \frac{100\Omega}{2/3K + 100\Omega} 5V$$

Issues: Non-linearity & 3-state requirement

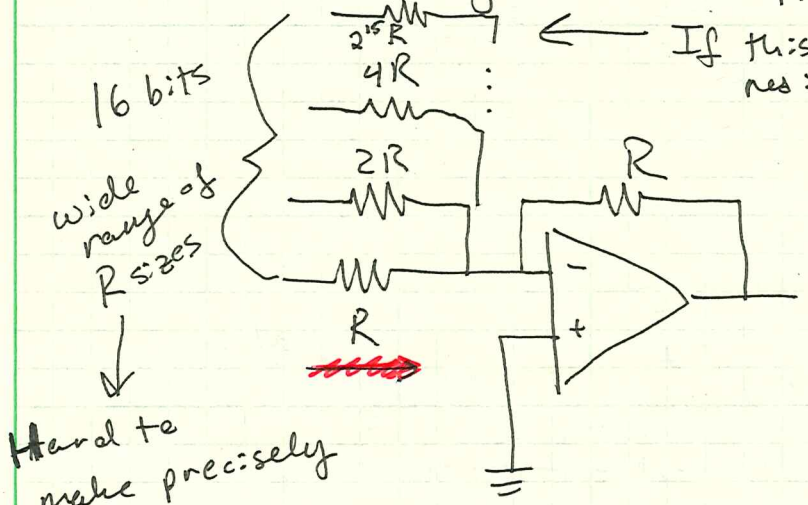
- Method 2: Op Amp Summer



$$V_{out} = -\left(\frac{V_A}{2} + V_B\right)$$

B	A	Vout
0	0	0
0	+5V	-2.5V
+5V	0	-5V
+5V	+5V	-7.5V

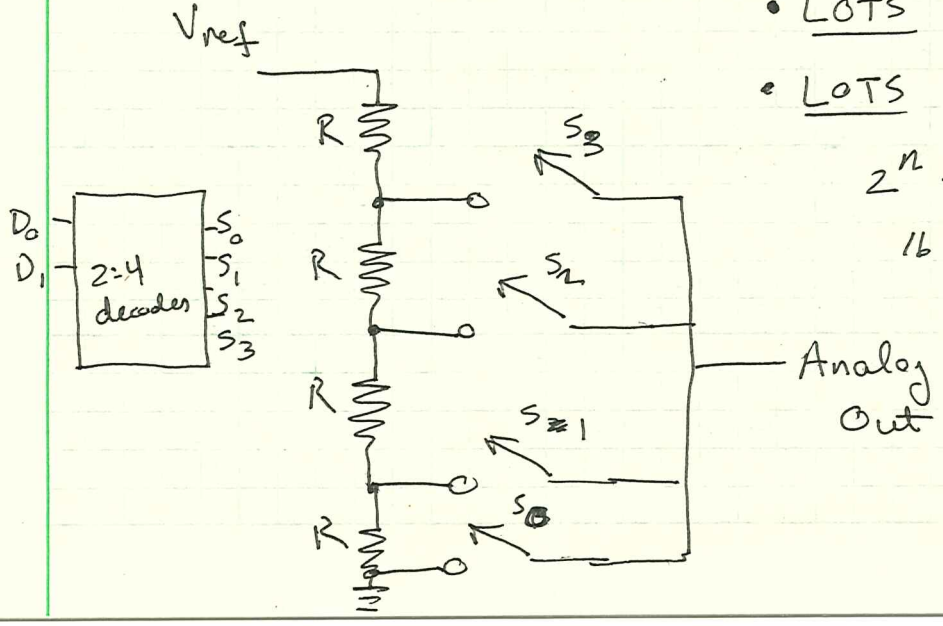
The challenge with this approach is scalability.



← If this resister is too by, I_{bias} starts causing errors

Hard to make precisely

- Method 3: Thermometer DAC



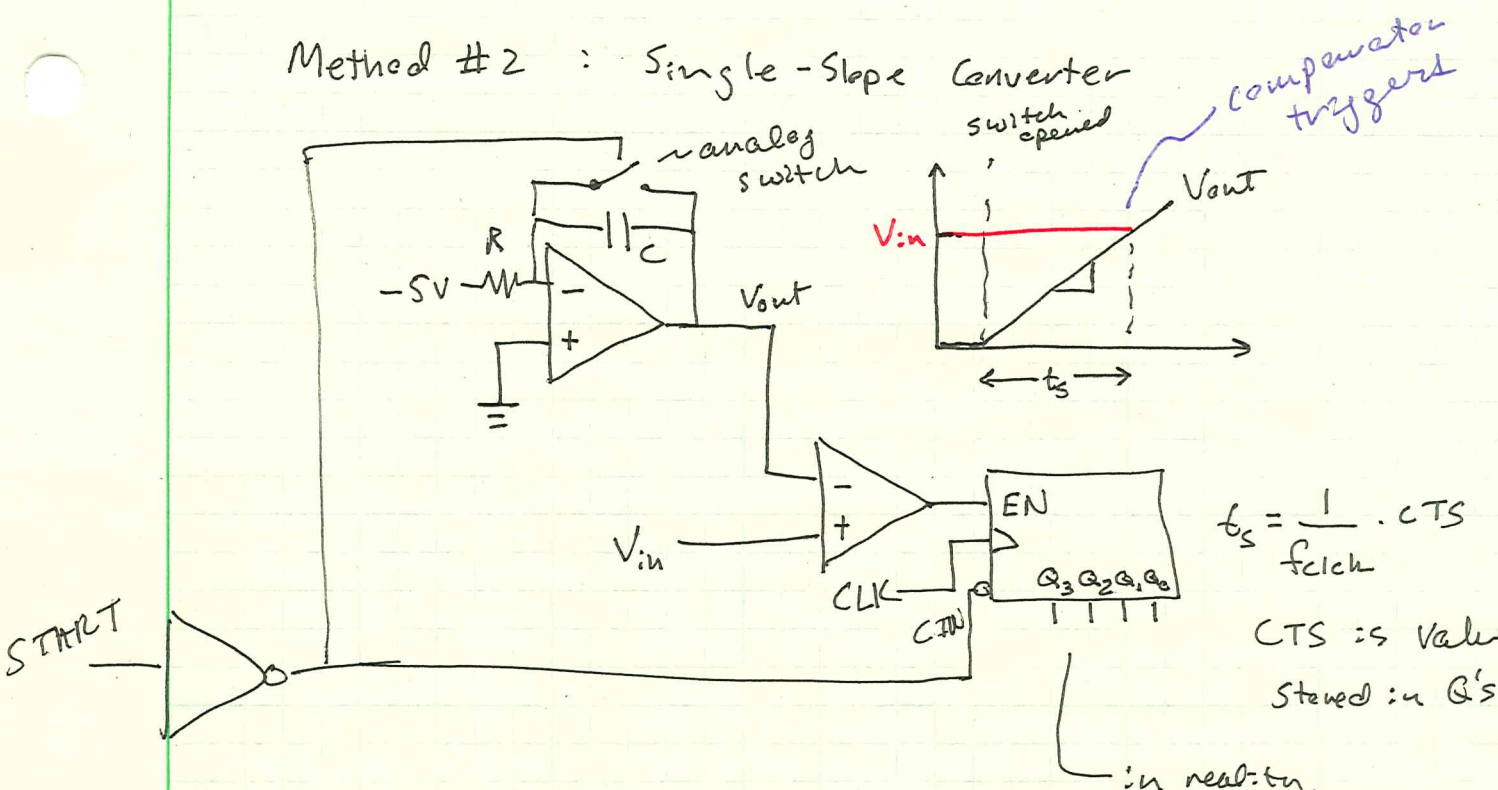
- LOTS of Resisters
- LOTS of switches

2^n # of switches!

16 bits = 65,536 switches

- Fast and high precision but hard to make

Method #2 : Single-Slope Converter



$t_s = \frac{1}{f_{clk}} \cdot CTS$

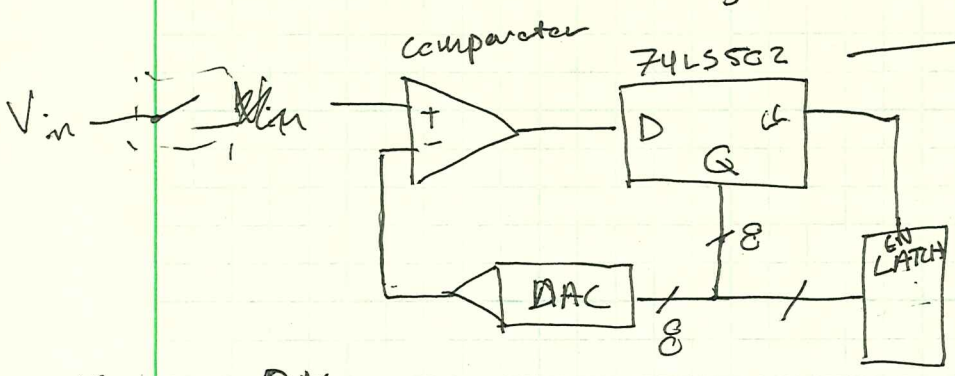
CTS is Value Stored in Q's

in reality, use more bits & higher clock

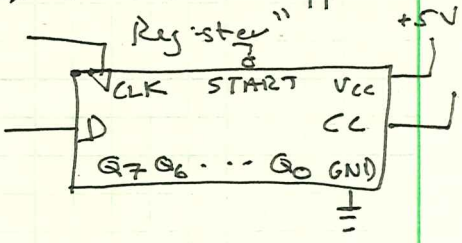
Note: Dual-Slope Converter is better, with a similar approach.

- Eliminates errors in comparator
- Reduces or eliminates some kinds of noise

Method #3 : Binary Search ADC



"Successive Approximation Register"



- Similar to several to Parallel converter

143/255 5

3.33V

160/255 .5

=3.13

1/2 8

+63

191

DAC maps 8 bits to 5V fullscale

Assume Vin = 3V

DAC OUTPUT ≈2.5V

Only 7-steps to Slow

- On first clock tick after we start converting

Q = 0111 1111 ≈ 127₁₀

Q = 1011 1111 ≈ 191₁₀

Q = ~~1001 1111~~ ≈ 159₁₀

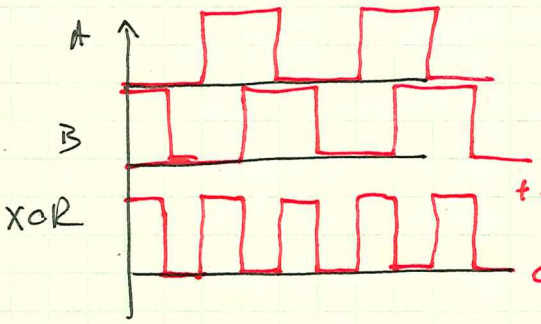
1001 1111

Q = 1000 1111 = 143₁₀

= 1000 0011 ≈

(3) Phased ~~Loop~~ Locked Loop

- Uses Feedback, not on Voltage, but on frequency!
- Used for multiplying frequency & Extracting noisy signals
- Ingredient #1: Phase Detector
  - ↳ Outputs a signal whose average value is proportional to the phase difference
  - ↳ Type I: XOR Gate

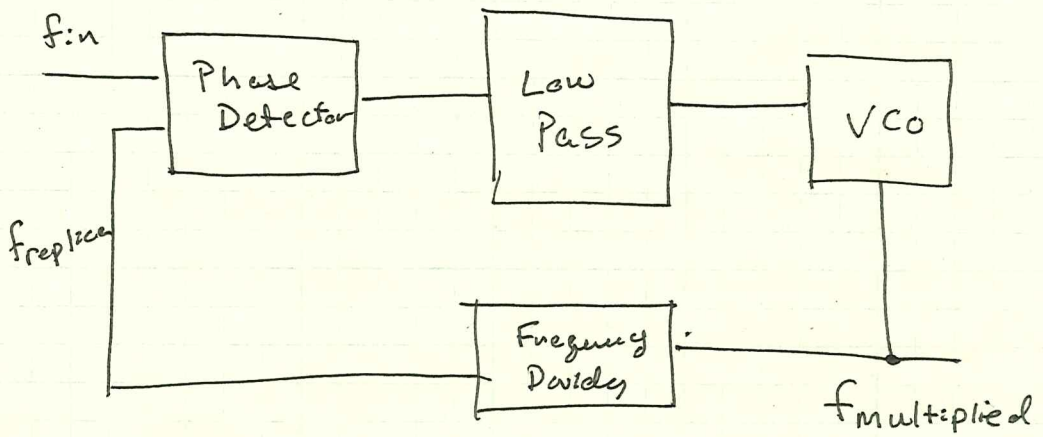


Average is 2.5V => 90° of phase difference

5V => 180°

0V => 0°

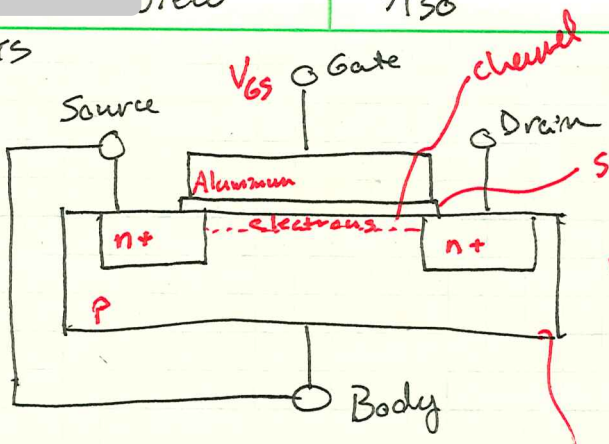
  - ↳ Type II: Edge-Sensitive
- Ingredient #2: Low-Pass filter
  - This extracts the average value from the phase difference signal
- Ingredient #3: Voltage controlled Oscillator
  - Output frequency is a function of input voltage
- Ingredient #4: Frequency Dividers, if desired.



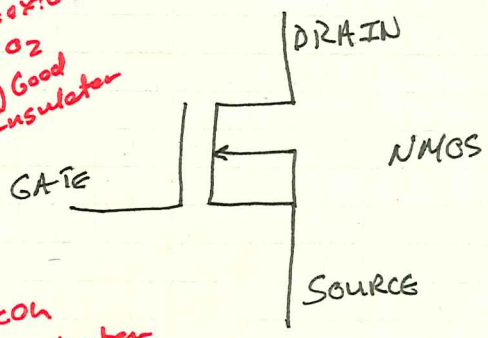
- If we set the circuit up properly, then the circuit "locks" onto the input frequency and freplica matches fin.
- Feedback minimizes phase difference between fin & freplica
- We can get some higher frequency, fmultiplied, as an output

(1) MOSFETS

Know this!



Symbol



- Doping changes electrical properties of semiconductors
  - n+ : heavy doping of excess electrons
  - p : doping of holes
- Apply V_{GS} attracts electrons and forms channel that can readily conduct current (switch is closed state)

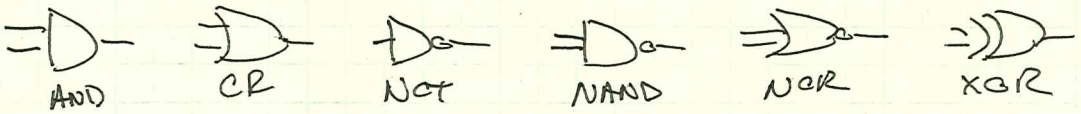
$V_{GS} > V_T$ switch is closed

$V_{GS} < V_T$ switch is open

when $V_{DS} > 0$, current can flow

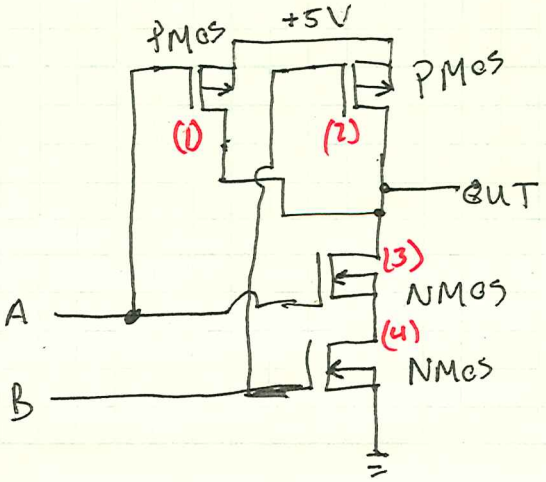
* MOSFETS have very high input impedence! (10^{12}-$10^{14}\ \Omega$)

(2) Logic Gates



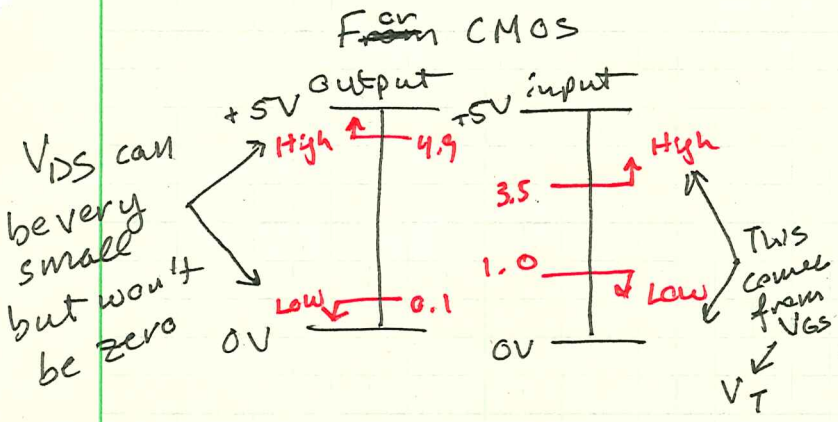
These just come from transistors.

I.e. CMOS NAND:



A	B	Out	Notes
0V	0V	5V	(1) and (2) are on
0V	5V	5V	(1) and (4) are on
5V	0V	5V	(2) turns on
5V	5V	0V	(3) and (4) on

(3) Conversion from Analog World to Digital World: "Logic Levels"



- We're guaranteed that a CMOS OUTPUT High is at least 4.9V and a LOW is no more than 0.1V.
- We're guaranteed that V_{in} from 3.5 to 5V will be treated as a High and V_{in} from 0 to 1.0V will be treated as low

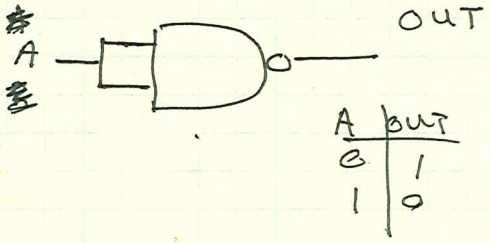
(4) "Universal Gates" NAND and NOR

↳ You can build any other logic gate using only NAND or NOR.

NAND

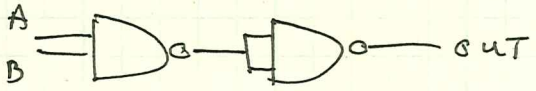
A	B	OUT
0	0	1
0	1	1
1	0	1
1	1	0

- NAND inverter (NOT from NAND)

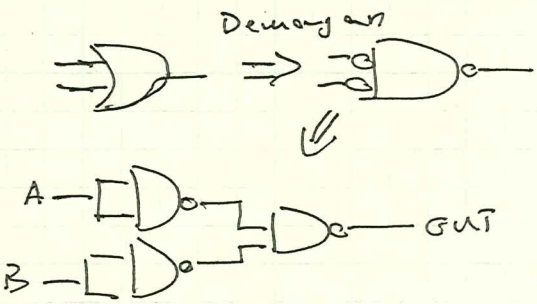


A	OUT
0	1
1	0

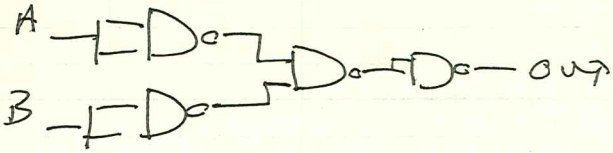
- AND from NANDs



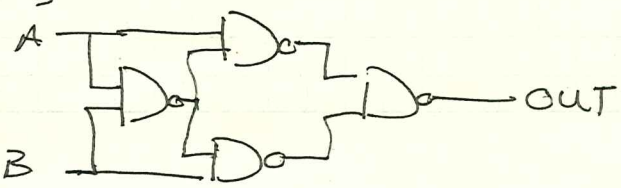
- OR from NANDs



- NOR from NANDs



- XOR from NANDs



(5) Boolean Algebra

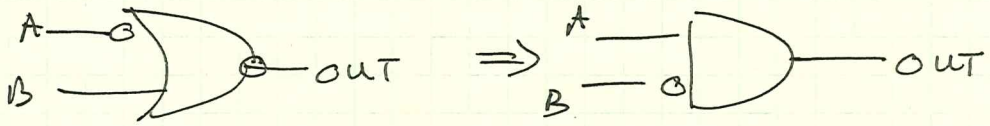
- DeMorgan's Theorem: You can change the gate type from AND to OR or vice versa if you also invert all inputs and outputs.

Ex 1

$\bar{A}\bar{B} = OUT$ $\qquad$ $\overline{A+B} = OUT$

$$\bar{A}\bar{B} = \overline{A+B}$$

Ex 2



$\overline{\bar{A}+B} = OUT$ $\qquad$ $A\bar{B} = OUT$

$$\overline{\bar{A}+B} = A\bar{B}$$

- Misc. Identities

$A + \bar{A} = 1$
$A\bar{A} = 0$
$\bar{\bar{A}} = A$

$A + \bar{A}B = A + B$

- Truth Tables: Giving some input/output relationship in table form

Ex. 1

A	B	OUT
0	0	0
0	1	1
1	0	0
1	1	0

"Out is high ~~if~~ if A is LOW and B is High"

$Out = \bar{A}B$

Ex. 2

A	B	OUT
0	0	1
0	1	1
1	0	0
1	1	1

(1) STATE The logic in words

OUT is High if A is Low and B is Low
or if A is Low and B is High
or if A and B are both High.

(2) Convert to compact logic expression

$OUT = \bar{A}\bar{B} + \bar{A}B + AB$

(3) Simplify

$OUT = \bar{A}(B + \bar{B}) + AB$ (the $(B + \bar{B})$ term = 1)

$= \bar{A} + AB$

$\boxed{OUT = \bar{A} + B}$

(6) Binary and Hex Representations

- Unsigned Binary
- Signed Binary (Two's Compliment)

↳ Treat the MSB as a negative number and add it to the rest of the number

Ex. 0b100 = -4

0b0100 = 4

Be careful with leading zeros!

- Truncation! This is not rounding. It is keeping certain bits and throwing away others

Ex. 0xA97

Truncate to the 8 least significant bits:

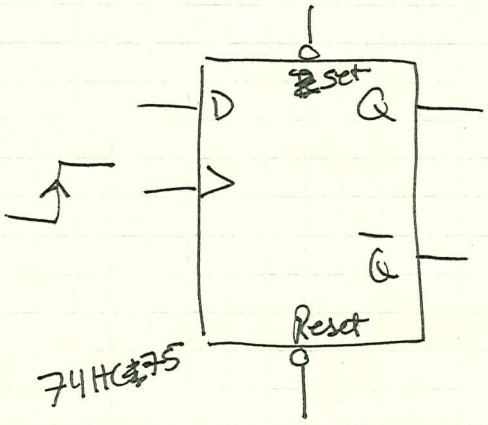
We get: 0x97

Truncate the ~~top~~ 4 most significant bits:

We get: 0x97

Truncate = "Remove" or "Eliminate"

(7) D Flip Flop (MID: Most Important Device)

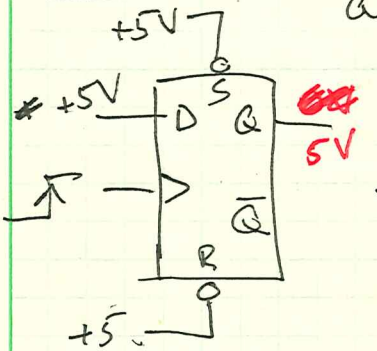


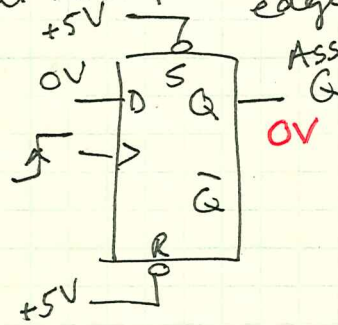
How it works

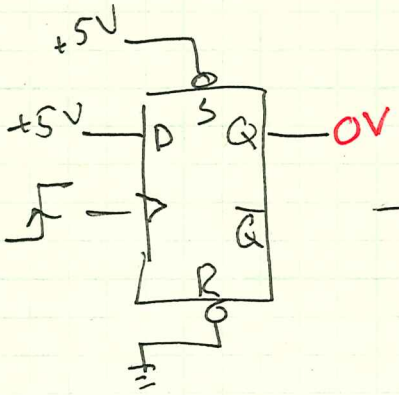
- At a rising clock edge, data gets transfered from D to Q.
- $\overline{Q}$ is always inverse of Q.
- Set and Reset are async function (does not depend on the clock)
  - ↳ Set drives $Q = 1$, $\overline{Q} = 0$
  - Reset drives $Q = 0$, $\overline{Q} = 1$

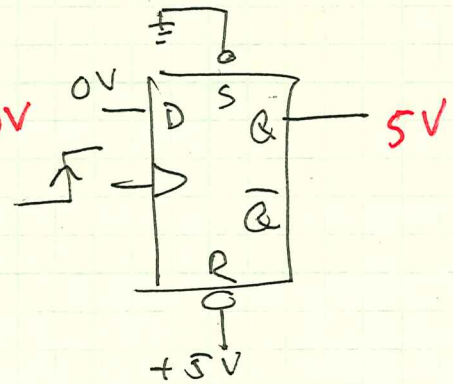
Examples:

What happens at next pos. clock edge? Assume Q's start at 0.



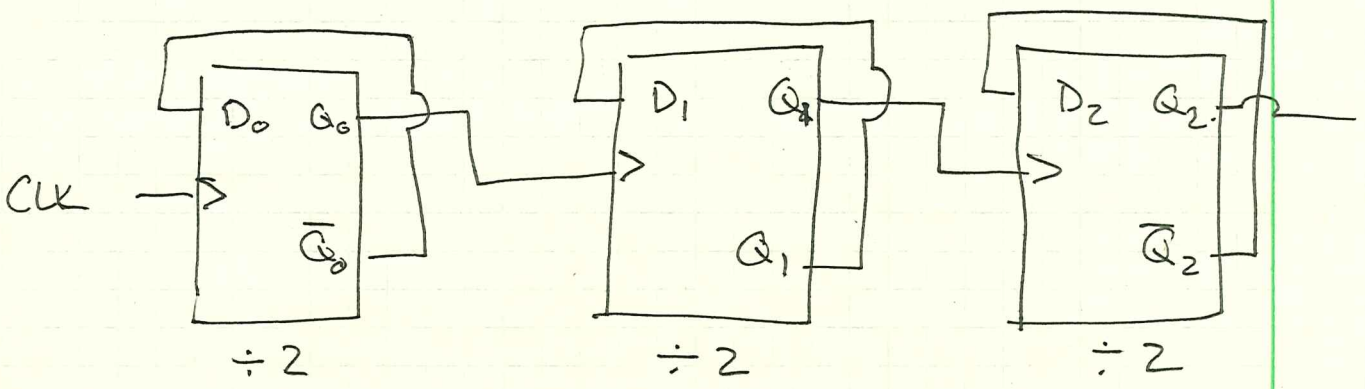






HC4040 – 12 bit ripple counter

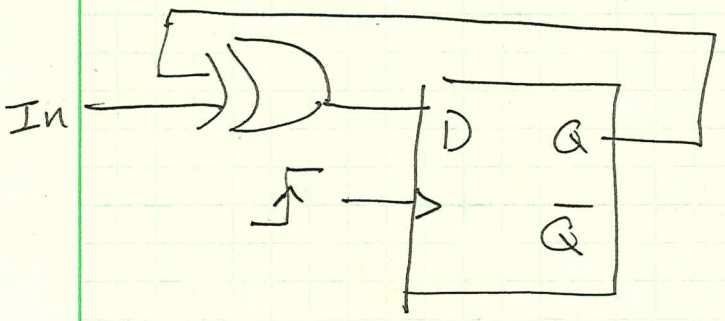
(8) Ripple Counter:



- Together is a ÷8
- Equivalently think of as counting to 8.

~~(9) Synchronous Counter~~

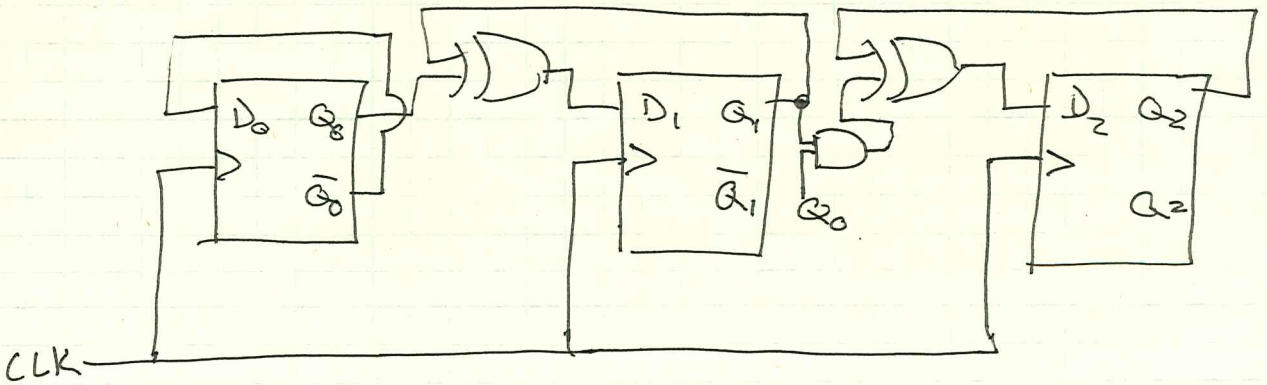
(9) Toggle Flop



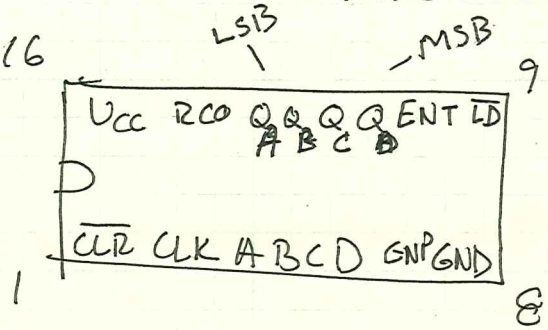
IN	Q	D	
0	0	0	hold
0	1	1	
1	0	1	toggle
1	1	0	

(10) Synchronous Counter

"Synchronize your watches"



Review 74~~LS~~HC163 functions:



$\overline{CLR}$: Sync Clear

$\overline{LD}$: Sync LOAD

ENT, ENP: Carry In (enables counting)

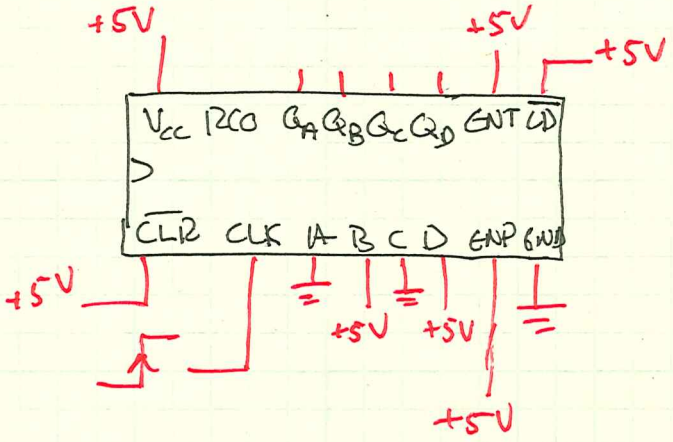
RCO: Carry Out (indicates that the counter is full)

A, B, C, D: Parallel load inputs "Data inputs"

Q_A, Q_B, Q_C, Q_D: Output bits

CLK: Clock

Example: Assume all Q's start out 0



(a) What are the values of Q's after the next clock tick?

$Q_A = 1$ $Q_B = 0$ $Q_C = 0$ $Q_D = 0$

(b) What if ENT/ENP = 0V?

$Q_A = 0$ $Q_B = 0$ $Q_C = 0$ $Q_D = 0$

(c) What if $\overline{LD}$ = 0V?

$Q_A = 0$ $Q_B = 1$ $Q_C = 0$ $Q_D = 1$

(d) What if $\overline{CLR}$ = 0V?

$Q_A = 0$ $Q_B = 0$ $Q_C = 0$ $Q_D = 0$

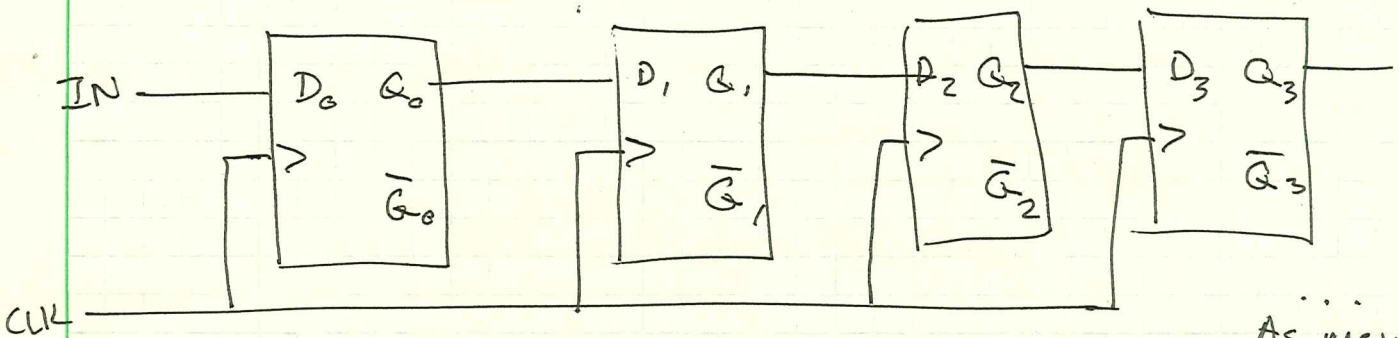
(e) Now assume Q's are all 1. What is the value of RCO? And what are the values of Q's ~~are~~ after the next clock edge?

RCO = 1 $Q_A = 0$ $Q_B = 0$ $Q_C = 0$ $Q_D = 0$

* Internals of 163. Load dominant?

Counter overflows and starts again!

(11) Shift Register



Assume all Q's start LOW and IN is High.

(a) What are the Q values after the next clock tick?

$Q_0 = 1$ $Q_1 = 0$ $Q_2 = 0$ $Q_3 = 0$

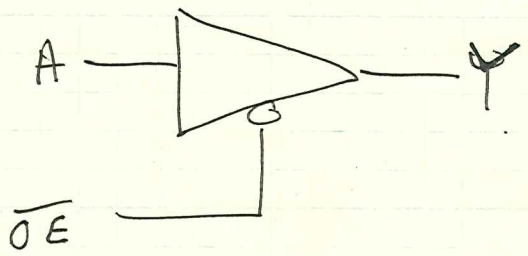
(b) Next clock tick?

$Q_0 = 1$ $Q_1 = 1$ $Q_2 = 0$ $Q_3 = 0$

(c) How many clock ticks does it take to fully shift in data to our four bit register?

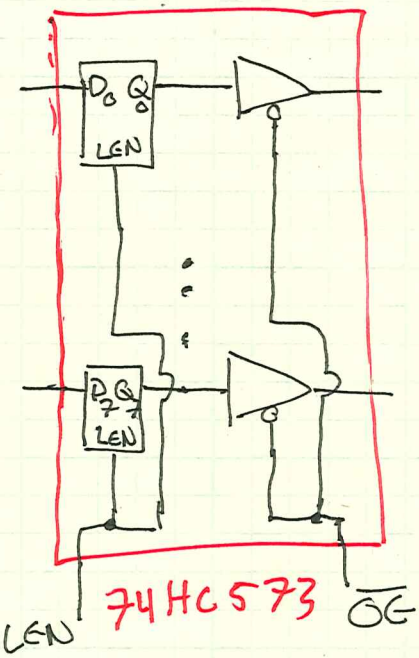
4 clock ticks for the 4-bit register

(12) Three State Buffer



A	$\overline{OE}$	Y
1	0	1
0	0	0
1	1	Hi-Z
0	1	Hi-Z

(13) Transparent D Latch

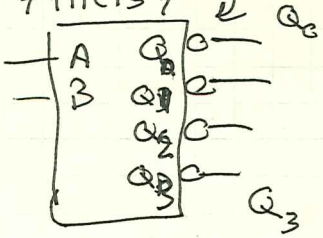


Latch Enable

Output Enable

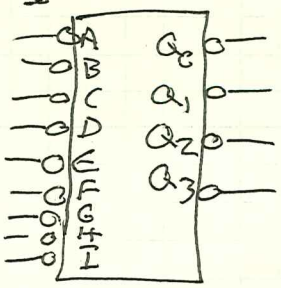
(14) Decoders & Encoders

74HC139 ↓



2:4

74HC147 ↓



10:4

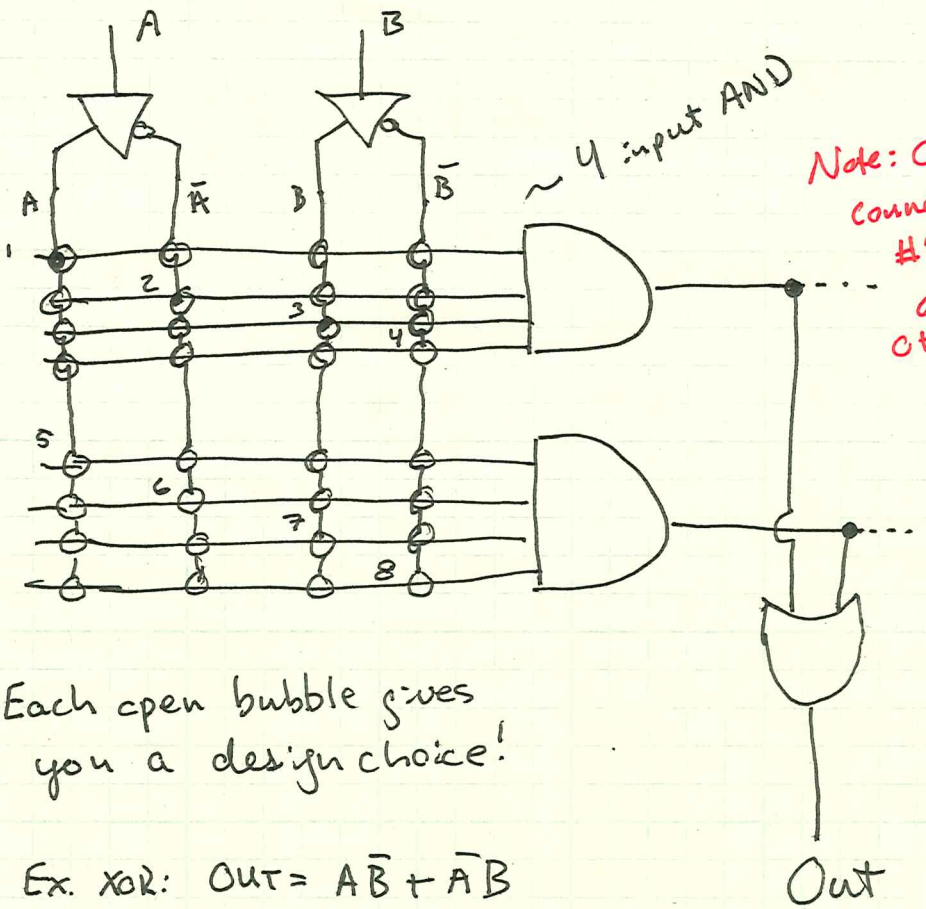
- Decoders assert a single output, based on the input code
- Encoders expect a single input, and set an output code based on which one is asserted.

(15) RAM: Decoding, three states, Latching ... just logic gates ... just transistors!

July 31, 2019 1

Programmable Logic:

(1) Simplest view: 2 inputs and 1 output



"Combinatorial Programmable Array Logic"

Note: Only connections #'d (1) through (8) are available. Other bubble should not be present.

Bad to float inputs to AND Gates?

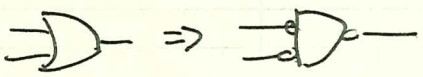
Maybe put connected to GND or V_{DD} in programming?

- Each open bubble gives you a design choice!
- Ex. XOR: $OUT = A\bar{B} + \bar{A}B$

  connect (1) & (-1) then (6) & (8)
- Ex. $OUT = A \cdot B$

  connect (3) & (4) then (5) & (2)
- Ex. $OUT = A + B$

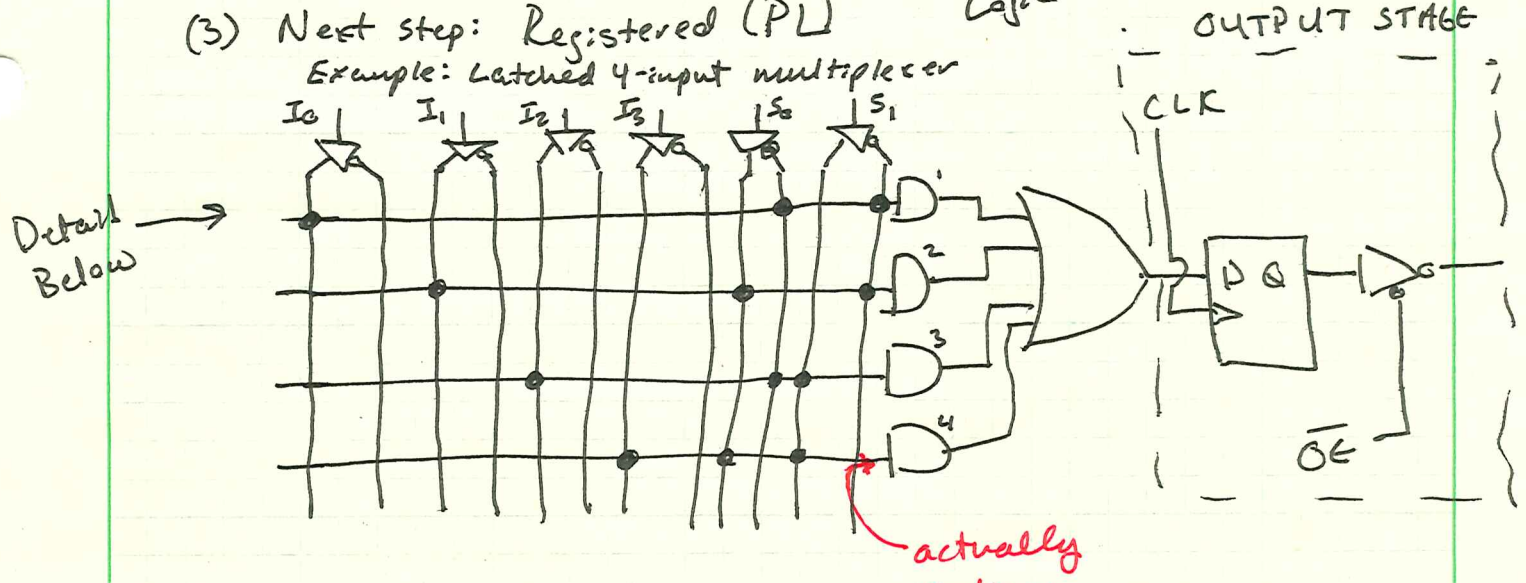
  Leave as exercise for the reader... connect (1) and (7)



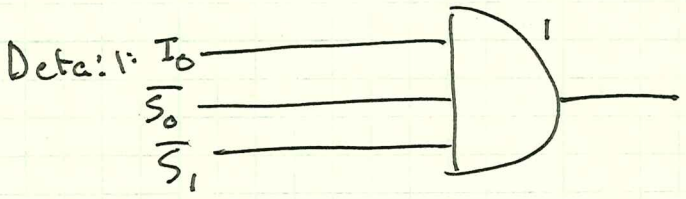
(2) Next step in complexity:
- Add 3-state buffer to the output
- Feed back the output to the logic array

(3) Next step: Registered (PL) — Programmable Logic

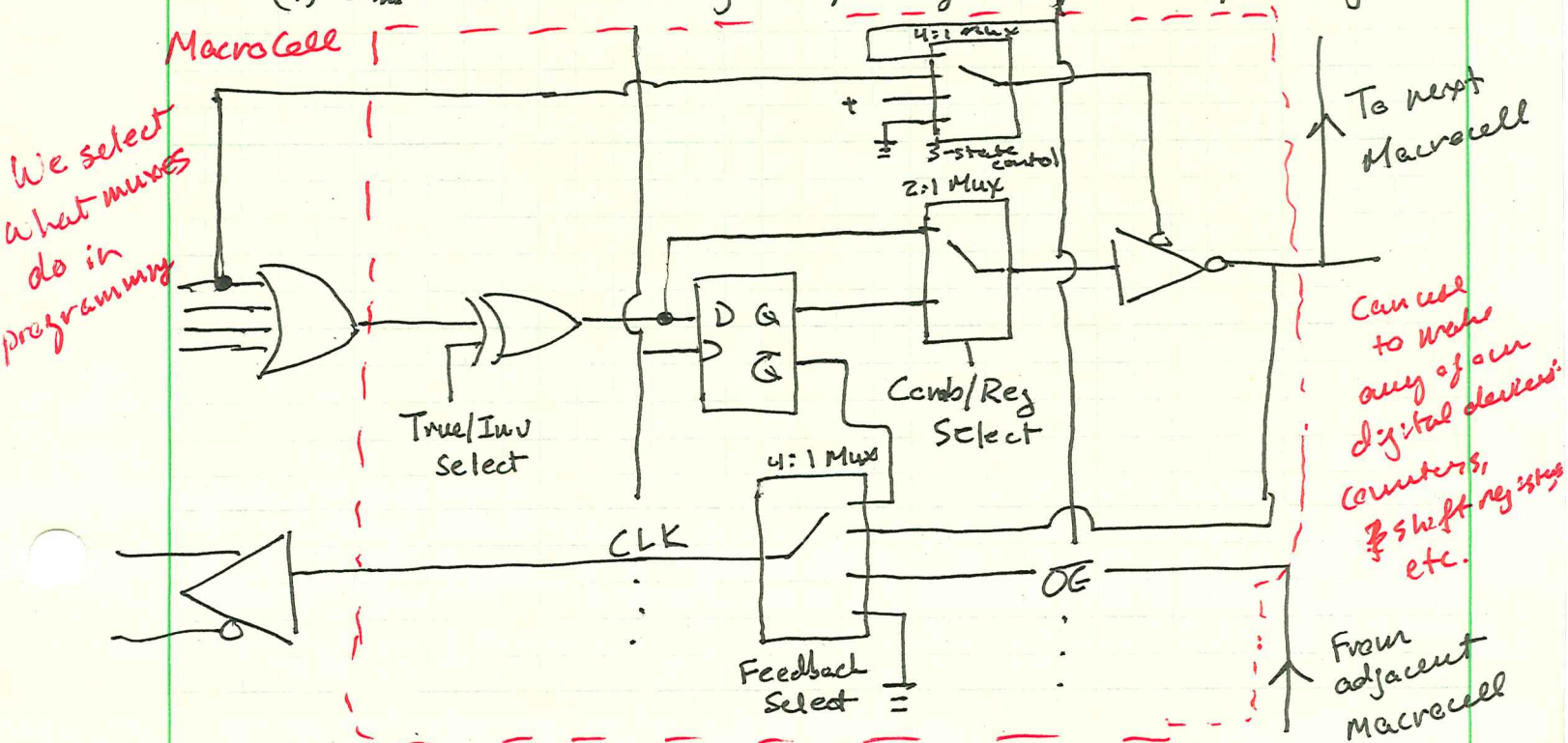
Example: Latched 4-input multiplexer



- When we draw a "•" it means we connect an input to the respective AND Gate



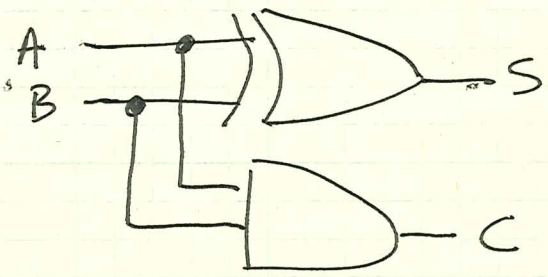
(4) One more level of complexity: Improve output stage



- To program, use a "Hardware Description Language"
  i.e. Verilog, VHDL

(2) Arithmetic Logic Unit (ALU)

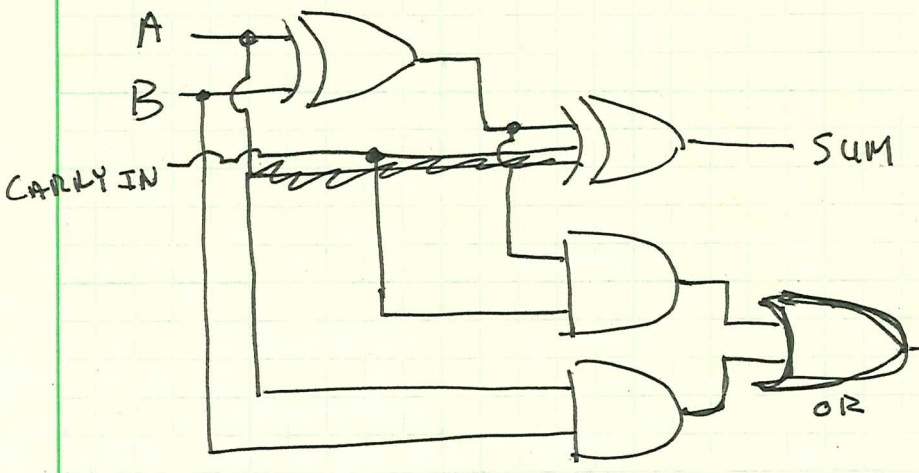
Version 1: Half Adder



In		Out	
A	B	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

reverse? To make cleared?

Version 2: Full Adder



A	B	C_{in}	S	C_{out}
0	0	0	0	0
0	1	0	1	0
1	0	0	1	0
1	1	0	0	1
0	0	1	1	0
0	1	1	0	1
1	0	1	0	1
1	1	1	1	1

"$S = A + B + C_{in}$"

In math form

Binary:

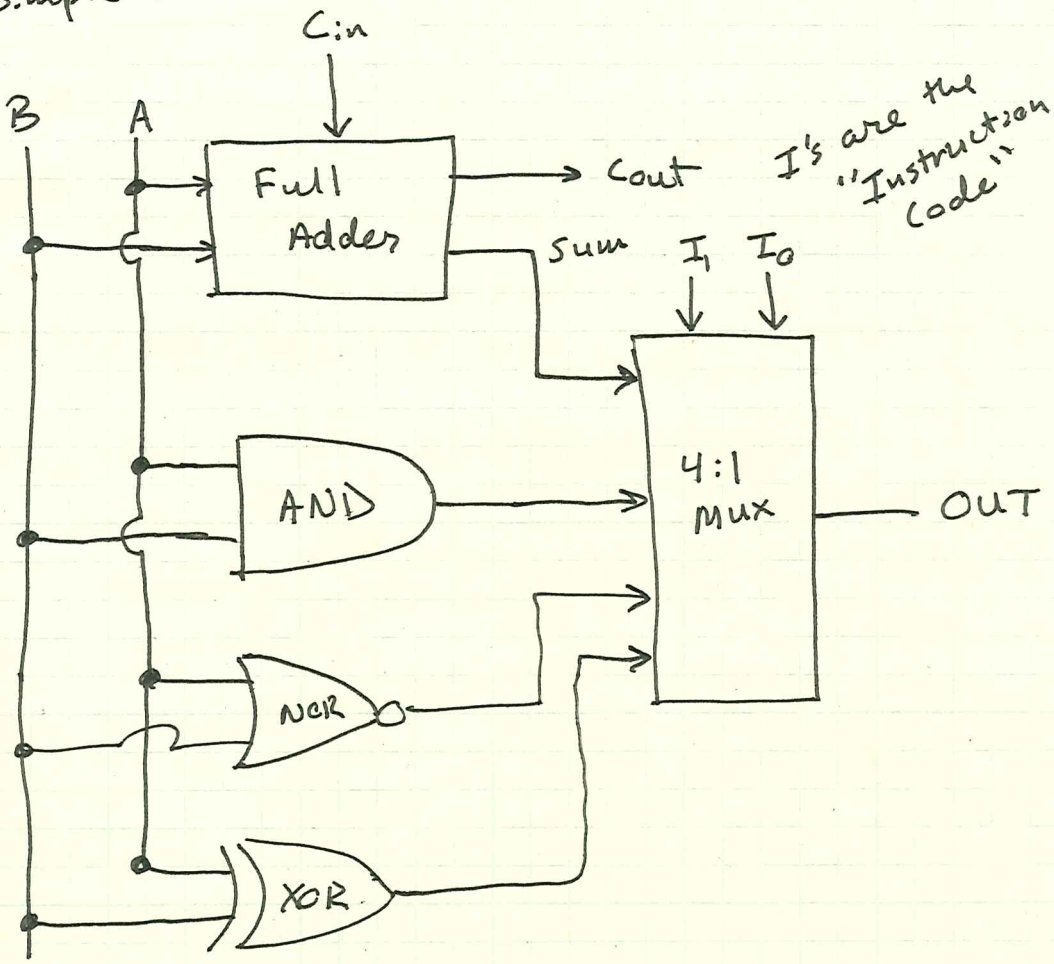
```
     1 <- CARRY OUT
A:   1
B:   1
-----
0 <- SUM
```

```
     1 <- CARRY OUT
A:   1
B:   1
Cin: 1
-----
1 <- sum
```

=> Next step would be to add 2-bit values

A = A1 A0
B = B1 B0

Simple ALU



- Add More operations to get a more functional computer!

For Our Computer, A and B are each 8 bits!

Instruction Code is also 8 bits!

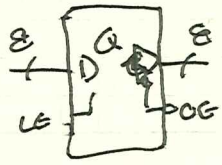
256 possible instructions ($I_7 \ldots I_0$)

Expected things such as: ADD, SUBTRACT, MULTIPLY, DIVIDE, COMPLIMENT, AND, OR, XOR, INCREMENT, DECREMENT

NOT "invert"

Note: Other instruction codes are reserved for manipulating internal registers, I/O, or external memory.

Register:



Think of as: i.e. HC573

- For OUR COMPUTER, we store the instructions in External RAM

- Each Machine Cycle, which is typically 4 clock cycles for our μC, the μC executes the instruction at the current address, and then goes to the next address.

  * Typically the μC just advances to the very next location in memory.

    I.e. 0x0000 ⟶ 0x0001

    Address at current machine cycle — Address at next machine cycle

  * But, some instructions allow us to jump to a different location in memory

    ↳ Allows Looping!

    Typically 2 or more instruction

    For example:

    - 1ST ONE says: next instruction will contain address to jump to.
    - 2ND & 3RD is: Address to jump to

A couple notes on today's Lab

GLUE LOGIC: Interfaces Different ports of the ~~xxx~~ computer

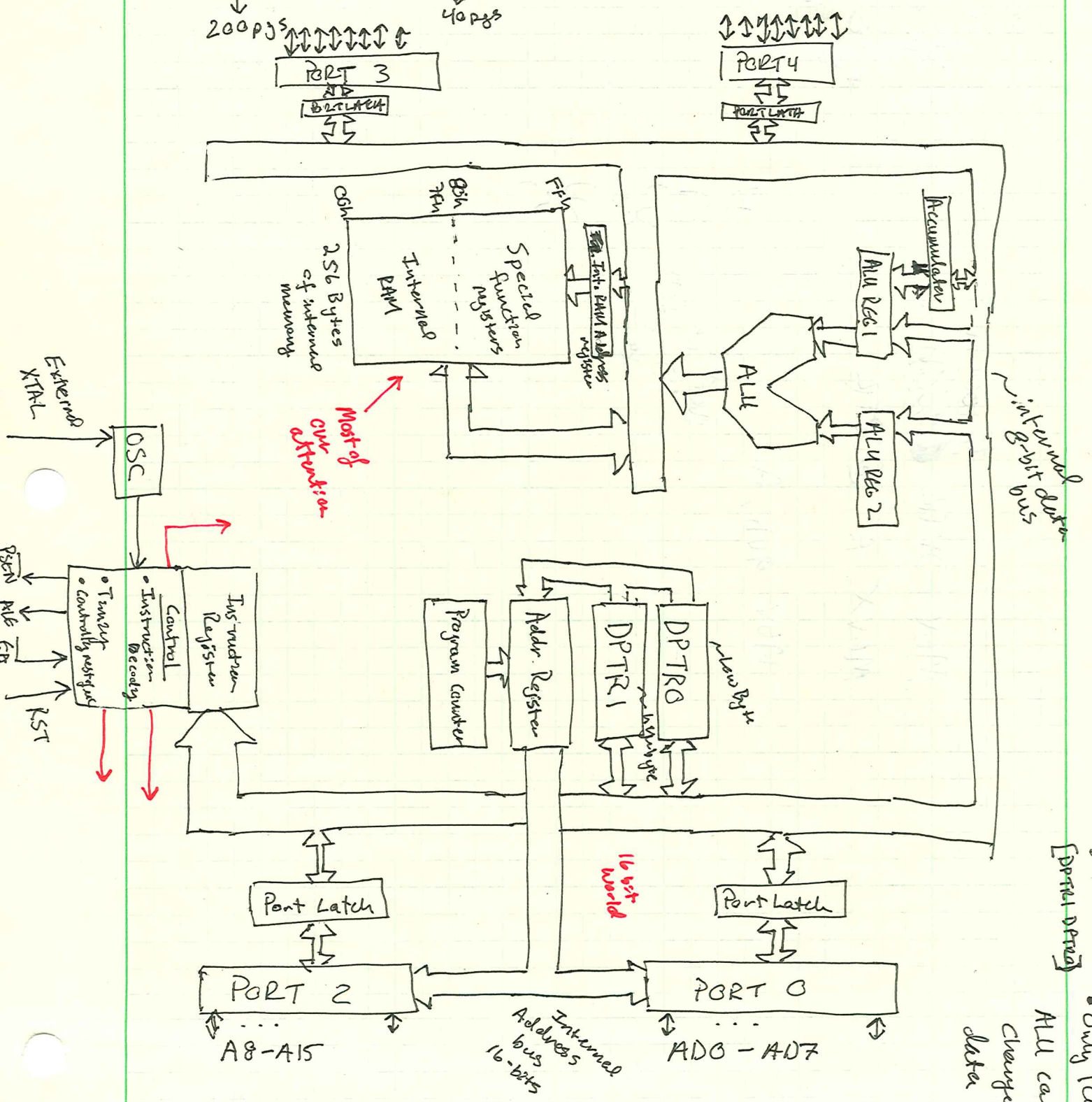
STEP LOGIC: Lets us advance a single Machine Cycle at a time. So we can see what's happening.

$\overline{BR}$ is a switch on the keypad that chooses whether you or the uC has control over the Address and Data lines.

Asserting $\overline{BR}$ should bring you back to where you were at end of memory lab.

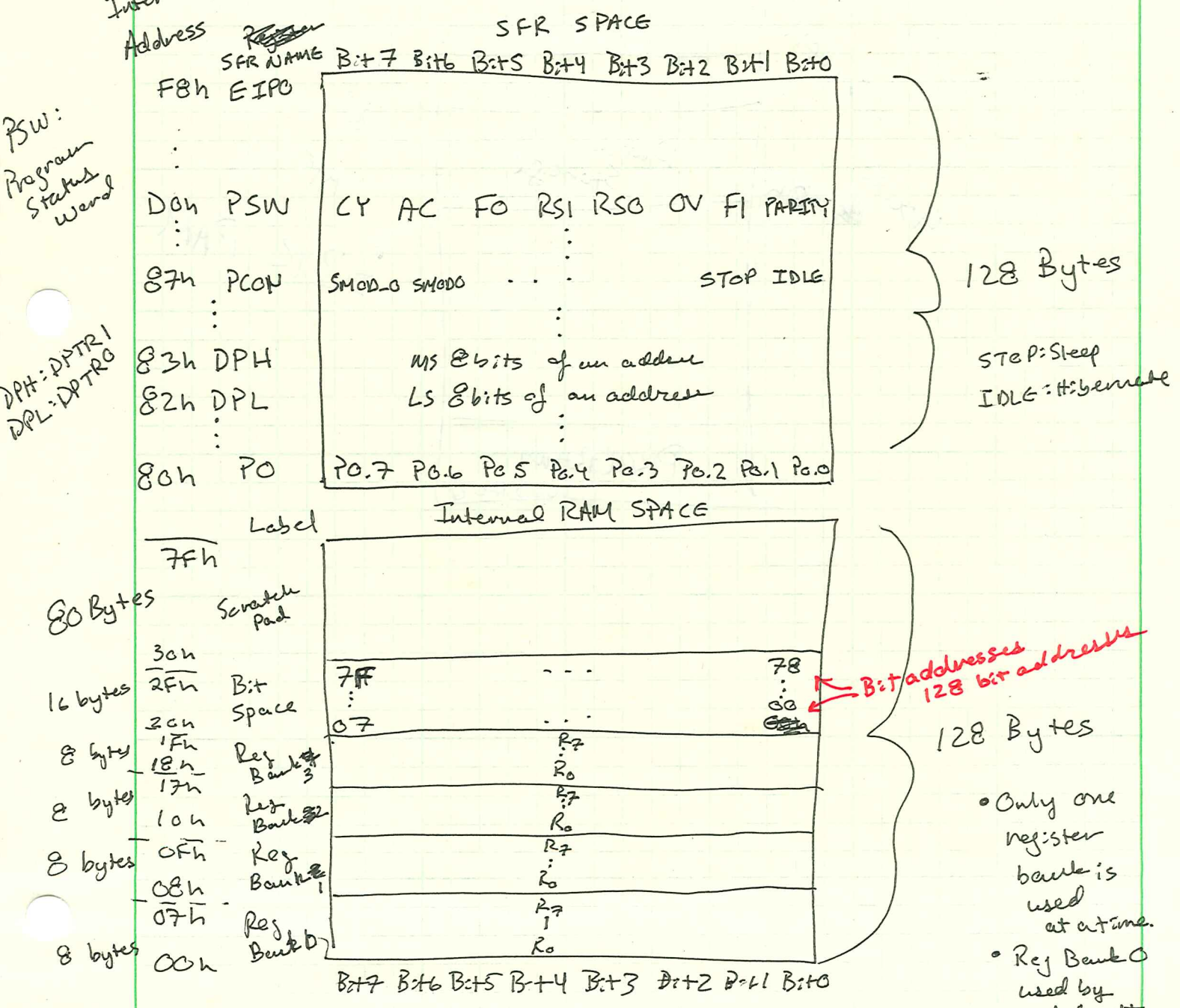
At the end of class today you should have a working uC running a simple program.

- In our computer, we use an 8051 variant, the Dallas DS89C430
- User Guide & Datasheet for the Dallas Board
  - 200 pgs
  - 40 pgs



- DPTR = [DPTR1 DPTR0]
- Only true ALU can change/modify data

- Special Function Registers (SFRs)
  - ↳ All peripherals and operations that are not explicit instructions in the µC are controlled via SFRs.
  - ↳ They control individual functions or can report a function's state.
  - ↳ Each is 1 byte.



(3) Instruction Set

- Each instruction has a code, often called an "opcode"
  - ↳ 256 codes for the 8051

A: ACCUMULATOR "ACC"

OPcode		Description	Mnemonic	OPERANDS
0x00	:	No Operation	NOP	
0x02	:	Long Jump	LJMP	addr16
0x04	:	Increment the Accumulator	INC	A
0xE8	:	Move Memory from R0 into A	MOV	A, R0
0xFF		Move Memory from A into R7	MOV	R7, A

Machine Code ↔ Equivalent statements but Assembly is easier to read. ↔ Living at "Assembly Language"

(4) Assembly Language

Typical Instruction: `MOV DPTR, #8000h`

- MOV: opcode mnemonic
- DPTR: Destination
- #: means "immediate"
- 8000h: Source

* This loads 8000h into DPTR. DPTR points to external data and I/O

In Machine Code: `908000`

- 90: opcode
- 8000: #const16

External Addressing: `MOVX A, @DPTR`

- MOVX: opcode
- A: Destination
- @DPTR: source
- @: means "indirect"

- Go find the data living at the address stored by DPTR and dump it into the ACCUMULATOR

MOVX @DPTR, A → Output the data stored in the Accumulator to the address specified by DPTR.

(1) Review The Tiny Test Program (end of 20L)

	Program		Assembly
	LOC	OBJ	
(1)	00	800E	SJMP, 0E
	10	00	NOP
	11	80FD	SJMP, FD

~rel address for jump

AJMP "Absolute Jump"

Address
0000 80
0001 0E

(2)

SJMP: Jumps from Address where it would have found its instruction if there wasn't a jump.

(1) We want to goto $10h = 16_{10}$

Would have gone to $02h = 2_{10}$

$14_{10} \Rightarrow 1110_2 = 0Eh$

(2) We want to goto $10h = 16_{10}$

Would have gone to $13h = 19_{10}$

-3_{10}

(i) Write out the positive version of the # 0b00000011

(ii) Flip all the bits : 0b11111100

(iii) Add one : 0b11111101

F D

FDh

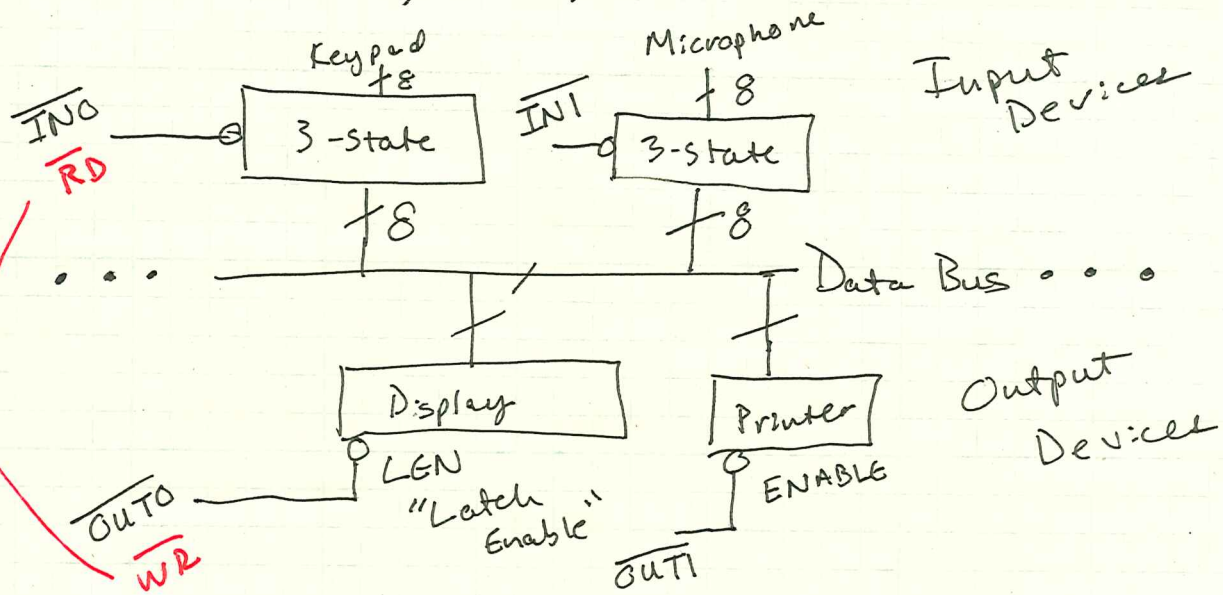
(2) I/O Decoding - We want to add perpherals

Our Computer:

- All peripherals attach to the data bus
- To avoid conflict, we need a way to enable and disable individual devices.
- Control lines: (outputs from uC)

  $\overline{WR}^*$, $\overline{RD}^*$, $\overline{PSEN}^*$, A15-A0

mouse
printer
keyboard
microphone
camera
Display
Touch input device
flash drive



Only use one input and one output

How do we generate $\overline{IN0}$, $\overline{IN1}$, etc.?

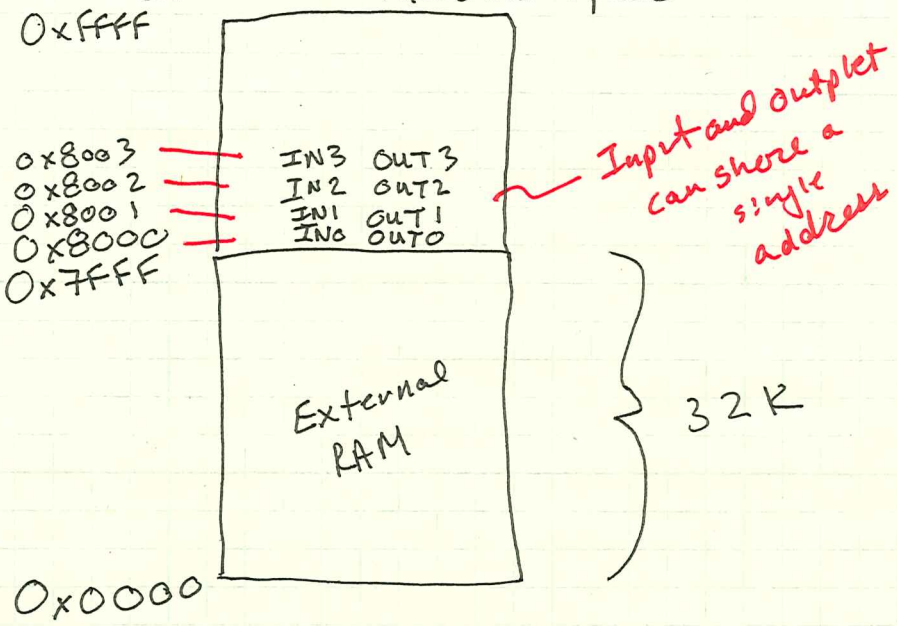
$\overline{WR}^*$ means Processor Write
$\overline{RD}^*$ means Processor Read
} uC is self-centered

Use the address lines! Treat I/O just like you would memory!

For our computer, we have 16 address lines.

↳ 65,536 addresses!

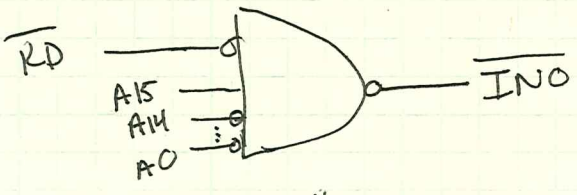
External Address Space



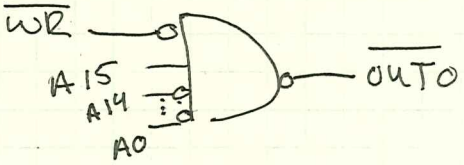
Input and output can share a single address

The general idea:

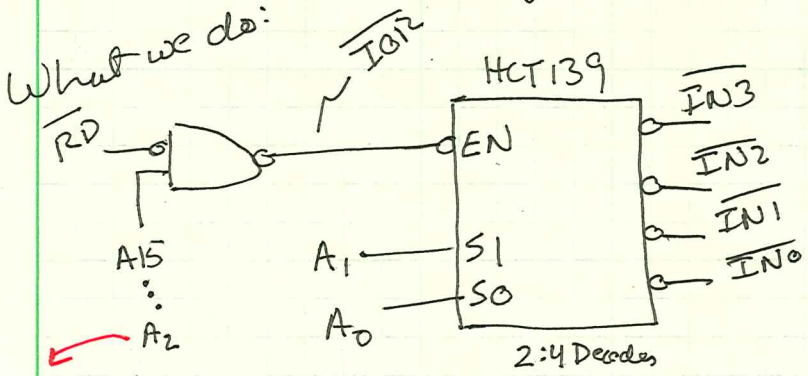
Ex. $\overline{IN0}$

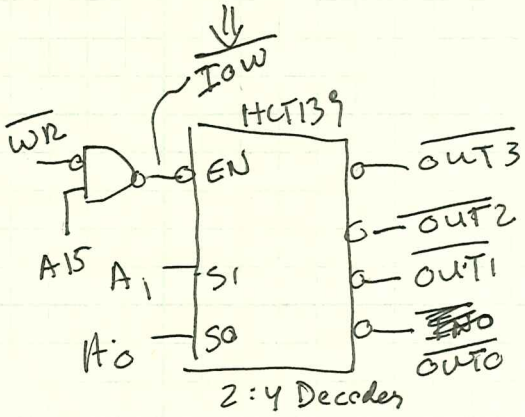


Ex. $\overline{OUT0}$



What we do:





This moves I/O to very top of address space

In our computer:

0x8000 => IN0 and OUT0

0x8FF0 => IN0 and OUT0

0xFFF0 => IN0 and OUT0

Only looking at A15, A1, and A0.

Tiny Program To interact with Peripherals

Address	Machine Code	Assembly Code
0x 0028	908000	MOV DPTR, #8000h
0x 002B	E0	MOVX A, @DPTR
0x 002C	F0	MOVX @DPTR, A
0x 002D	80FC	SJMP.

This line just saves the address generated

RD

inputs from IN0

generates RD or WR

outputs to OUT0

generates WR

Want to go to 0x2B $= x11_{10}$

Would go to ~~0x30~~ 0x2F $= x15_{10}$

✓ $= -4_{10} \Leftrightarrow$ 0xFC

$-3_{10} \Leftrightarrow$ 0xFD

(3) Conditional Branching in Assembly

↳ FOR loops, while loops, IF STATEMENTS

Example: JNB, JB : Jump if Bit set/not set "if statement"

DJNZ : Decrement Jump Not Zero "for loop"

CJNE : Compare JUMP NOT EQUAL "while loop"

Note: "8051 Instruction Set" is an important resource

(4) Subroutines : A program within a program

ACALL : Absolute Call

LCALL : Long Call

(5) When we use subroutines, we often employ the stack.

Stack: Temporary Storage Area located in internal RAM.

↳ Location is controllable by you via the Stack Pointer SFR

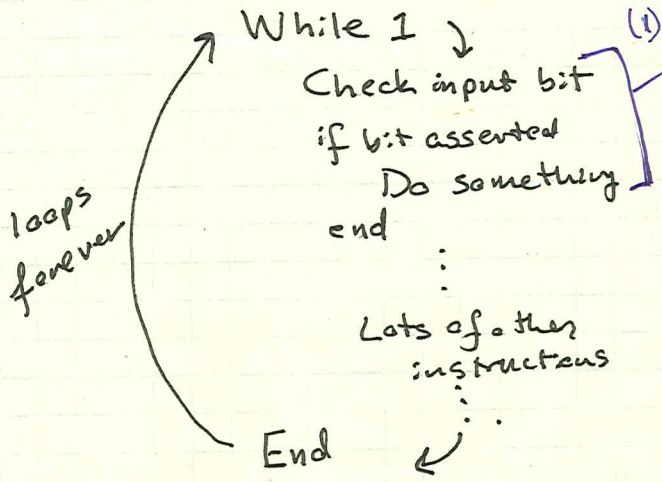
Lab Today:
21.L.1.4, 21.L.1.5, 22.L.1.1, 22.L.1.2

Lab 23 Micro IV | Interrupts

Interrupt: A hardware or software event that jumps us to a special kind of subroutine

Why? Say you are monitoring an input pin and want to respond very quickly (~ μs), but say the event only occurs every few seconds

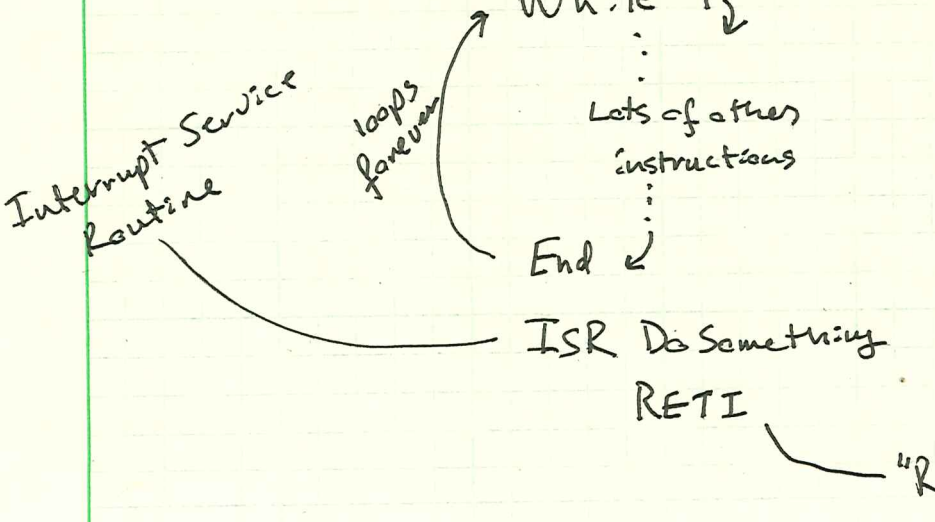
YOUR PROGRAM w/o interrupts:



(1) Very inefficient We waste a lot of time checking this every few μs when we know it is going to only come every few sec

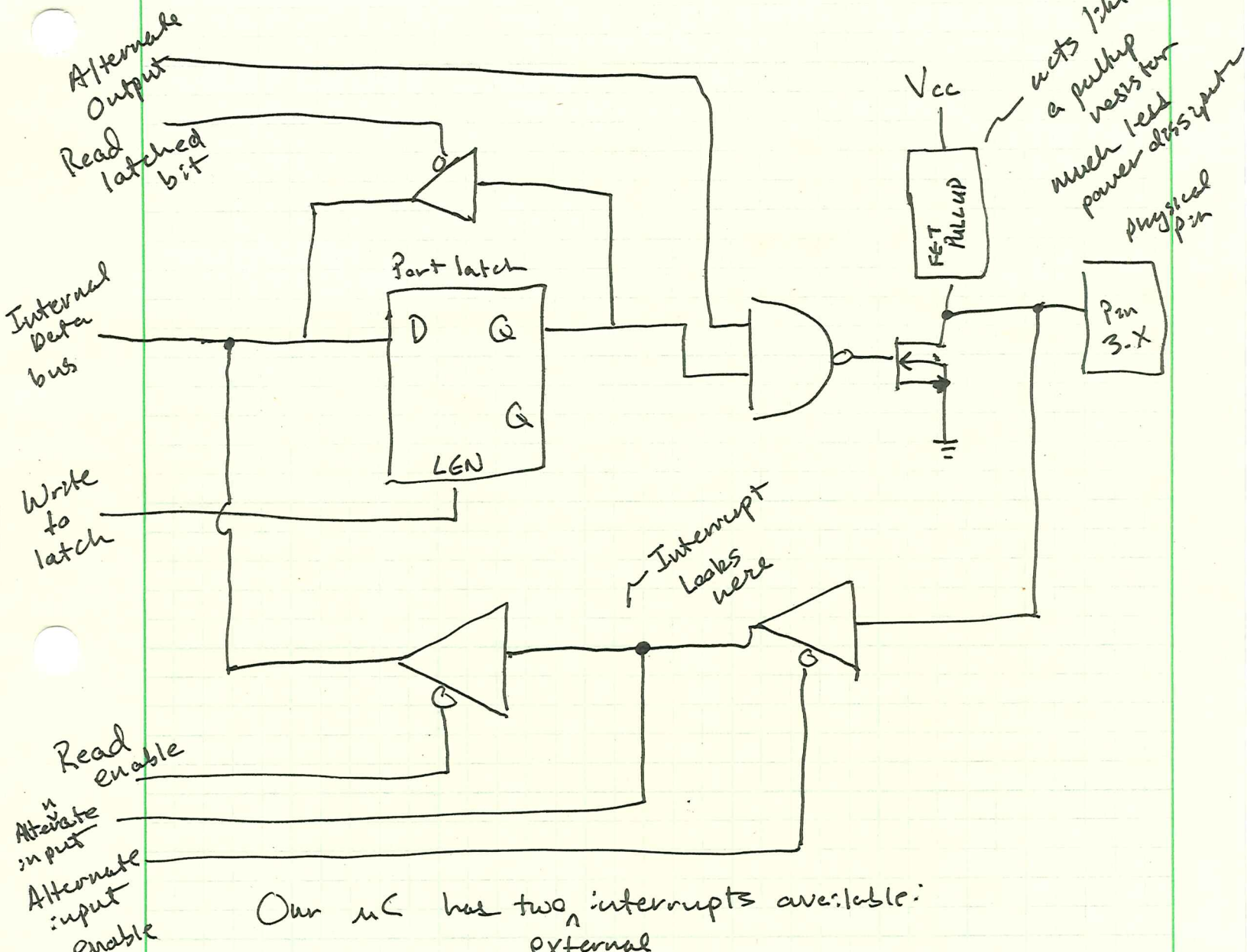
(2) We don't guarantee fast response - depends on where we are in the program loop

YOUR Program with Interrupts



- New, main program is free to focus on its jobs
- Interrupt jumps in as needed.

In Hardware: Look at Port3 output stage



Our uC has two external interrupts available:

INT0 : Port3 Pin 2
INT1 : Port3 Pin 3

To Configure these Pins as Interrupts, use Interrupt Enable (IE)

Special Function Register

A8h

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
IE:	EA	...	..	...	...	EX1	...	EX0

EA — global interrupt enable
EX1 — INT1 Enable
EX0 — INT0 ENABLE

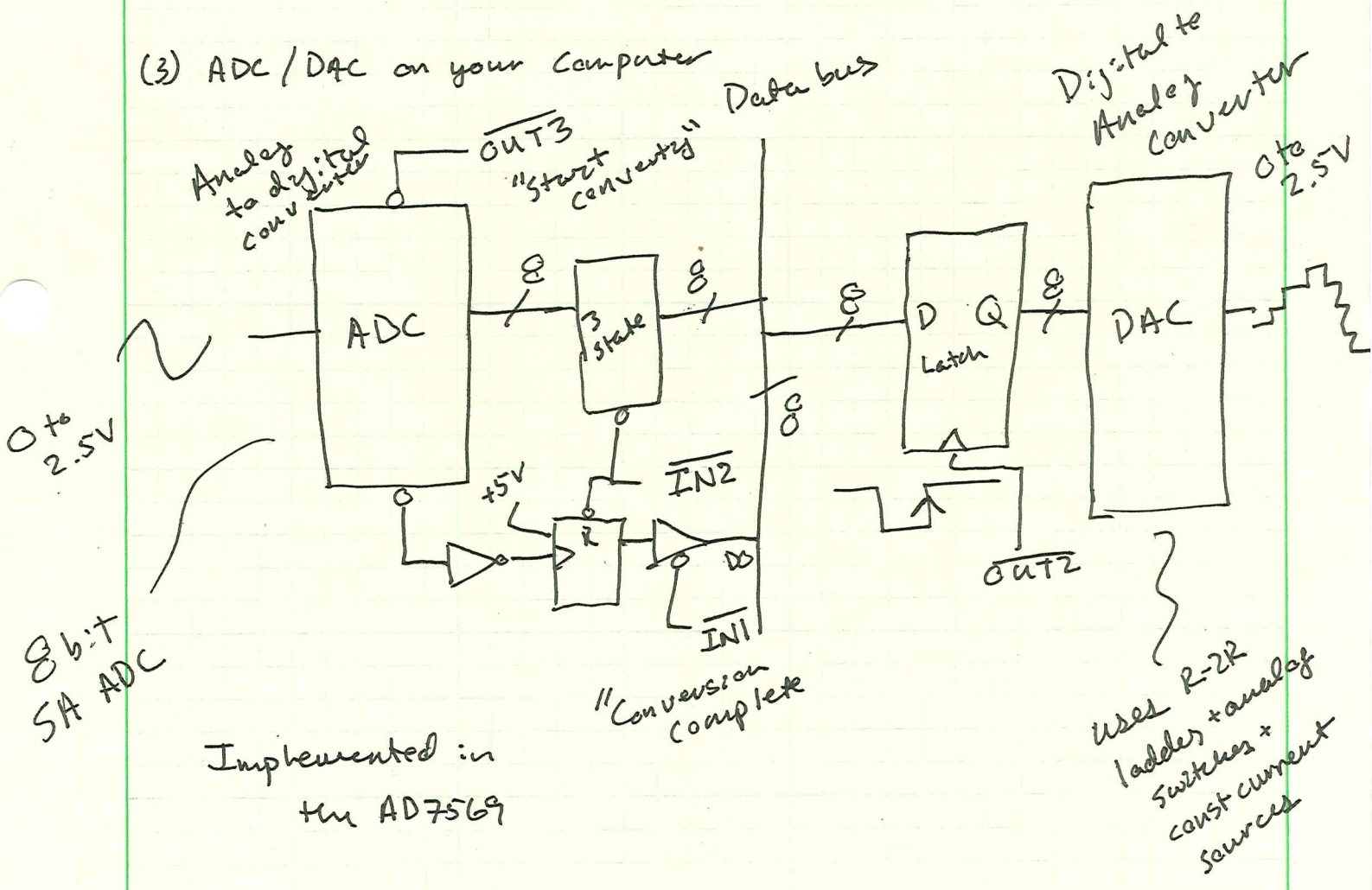
ISRs Must be at known locations in program memory.

"Interrupt Vectors" — ISR Locations.

ISR for INT0 default location is 0x0003h in program memory
ISR for INT1 default location is 0x0013h ↗

Note, INT0 has higher default "priority" but you can choose!

(3) ADC / DAC on your Computer



Implemented in the AD7569