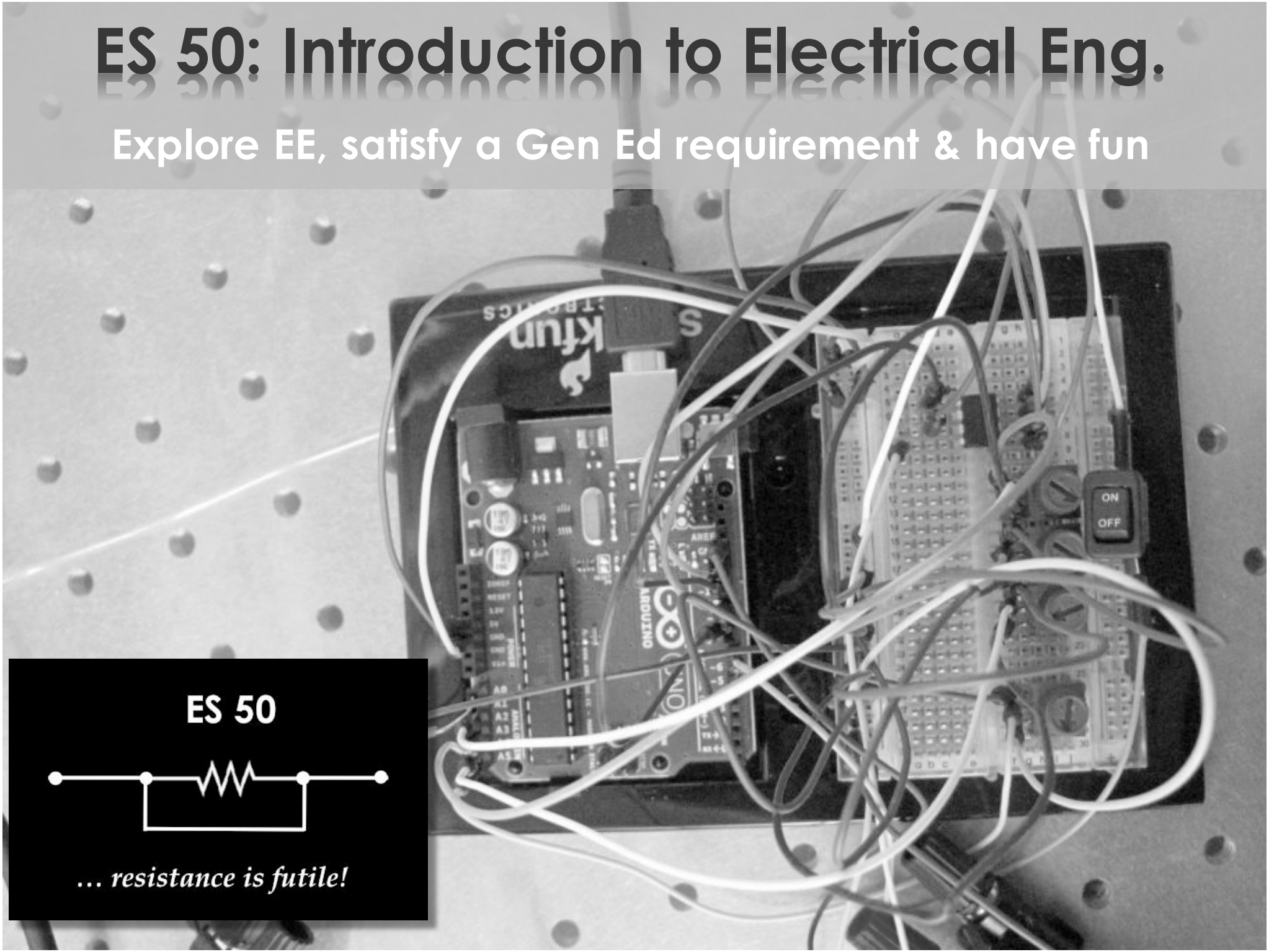
ES 50: Introduction to Electrical Eng.

Explore EE, satisfy a Gen Ed requirement & have fun



ES 50

… resistance is futile!

Lecture 14: Finite State Machines then From Audio to Visual: Images as 2D (DIGITAL) Signals

November 2, 2015

HARVARD

Upcoming Events

In class Project Workshop – Wed Nov 4

- Each group will have 2 min introduce group members & project goals
- Upload slides to CANVAS by Tuesday Nov 3 @ 11:59 PM
- Your entire final group is required to be in class or you will loose points

Parts List

- Due Sun Nov 8 @ 9 PM on CANVAS & by emailing your project TF

HARVARD

Upcoming Events

Sophomore (and Freshman) Advising

- Harvard College Engineering Society – TODAY 4 PM, Pierce 301
- SEAS-wide – Wed Nov 4, 4 PM, MD Lobby

HARVARD

Implementation of FSM using Flip-Flops

Since FSMs have memory, they cannot be represented using combinatory logic only. Sequential logic (flip-flops) are needed to provide memory function. The recipe for synthesis of FSM with D flip-flops is as follows:

1. Determine the number of states, and encode each state with a binary number.
   - For n states we need binary number with k-bits, where $k = \log_2 n$. Remember you need to round up!
   - Each D flip-flop stores 1-bit (digit) so we need total of k flip-flops!
2. Encode each input with binary number. If we have m inputs we will need $e = \log_2 m$ input wires. For example if we have 7 inputs:

   input 0 000
   input 1 001
   ...
   input 7 111

   Note: encoding (typically) reduces the number of input wires required: from m to $\log_2 m$
3. Encode outputs with binary numbers. If we have r outputs we need $p = \log_2 r$ output wires
4. Draw FSM diagram
5. Implement the FSM diagram using combinatory logic & flip-flops

Example: Mod 4 Counter

Synthesize mod 4 synchronous counter with D flip-flops

- The counter counts up on a rising edge of the clock signal
- When counter reaches 3 it goes back to 0, LED is turned on, & counter continues counting.
- Counter has only one input: Reset button R. When $R = 1$, counter goes to 0 & stays there, as long as $R = 1$. (Note, clock signal is not treated as an input in this case.)

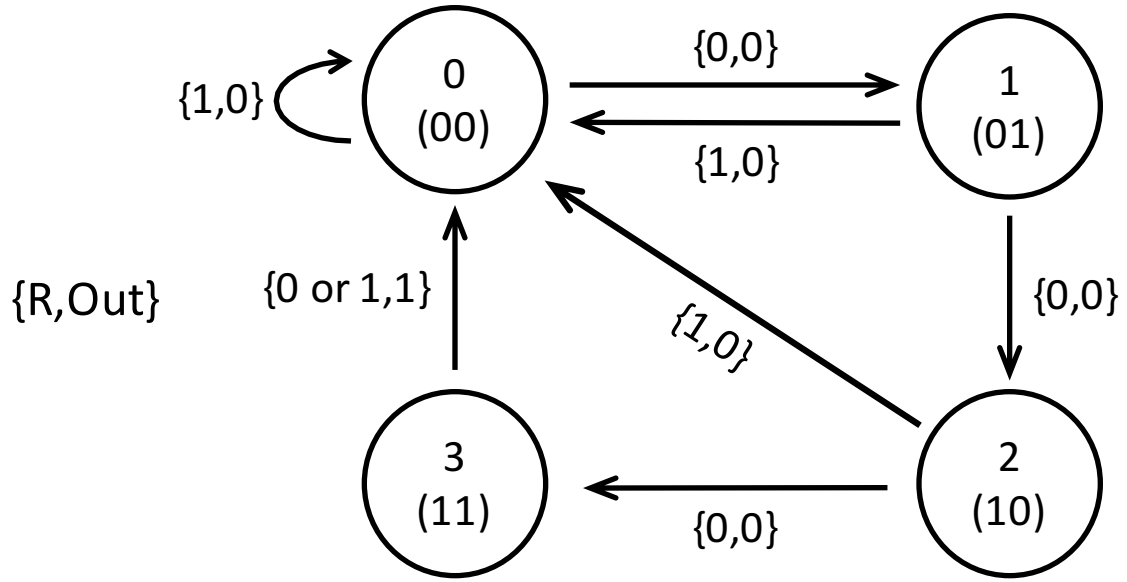
States: {0, 1, 2, 3}

- We need $k = 2$ bits to encode all states so we will need 2 flip-flops
- The states will therefore be: $q_1 q_0 = \{00, 01, 10, 11\}$

Input: {Reset}

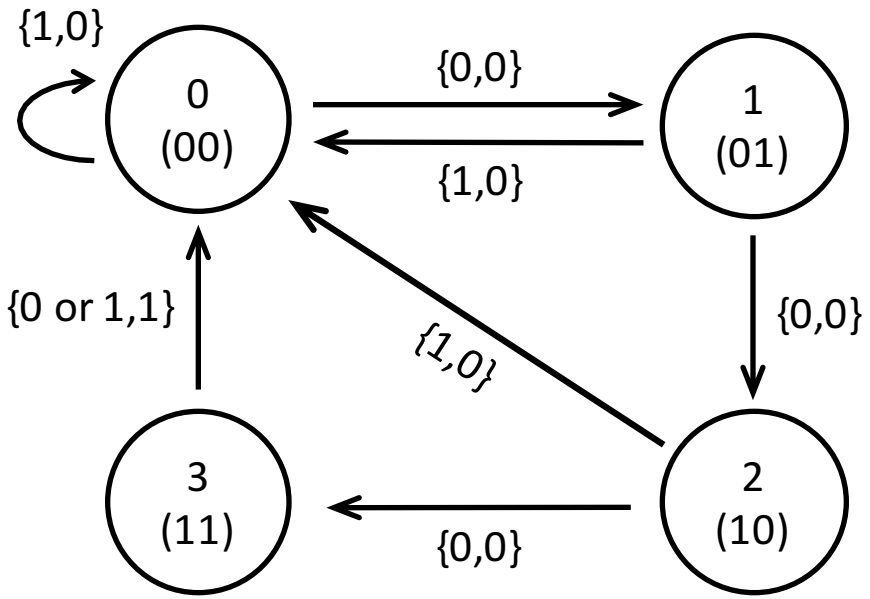
- One input wire

Outputs: {LED OFF, LED ON}



Example: Mod 4 Counter

Now that we have the state diagram, let's make a truth table:

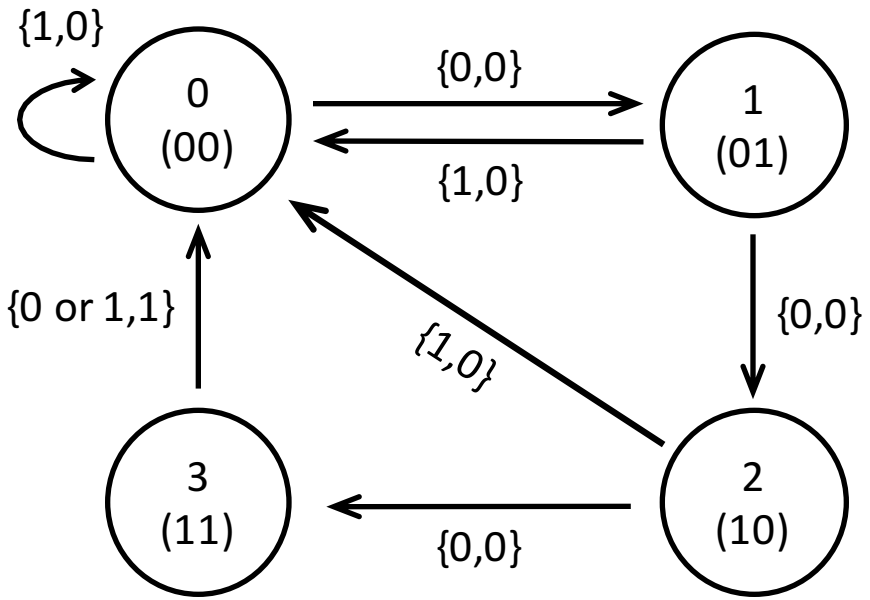


Input	Current State		Next State		Flip-Flops		Output
R	$S_1(t)$	$S_0(t)$	$S_1(t+T)$	$S_0(t+T)$	D_1	D_0	
0	0	0	0	1	0	1	0
0	0	1	1	0	1	0	0
0	1	0	1	1	1	1	0
0	1	1	0	0	0	0	1
1	0	0	0	0	0	0	0
1	0	1	0	0	0	0	0
1	1	0	0	0	0	0	0
1	1	1	0	0	0	0	1

- Each flip-flop memorizes one bit of a current state: $S_1(t)S_0(t)$
- Next state defines where FSM goes next for particular combination of present state and input!
  - Simply put, next state tells us what needs to be written in each flip-flop next!
- How do we make sure that the correct next state is written in?
  - We need to bring appropriate signals to the inputs of flip flops!

Example: Mod 4 Counter

Now that we have the truth table, what are the logic equations:



Input	Current State		Next State		Flip-Flops		Output
R	$S_1(t)$	$S_0(t)$	$S_1(t+T)$	$S_0(t+T)$	D_1	D_0	
0	0	0	0	1	0	1	0
0	0	1	1	0	1	0	0
0	1	0	1	1	1	1	0
0	1	1	0	0	0	0	1
1	0	0	0	0	0	0	0
1	0	1	0	0	0	0	0
1	1	0	0	0	0	0	0
1	1	1	0	0	0	0	1

$S_1(t+T) = R'S_1'S_0 + R'S_1S_0'$

- $S_1(t+T) = R'(S_1'S_0 + S_1S_0')$
- $S_1(t+T) = R'(S_0 \oplus S_1)$

$S_0(t+T) = R'S_1'S_0' + R'S_1S_0'$

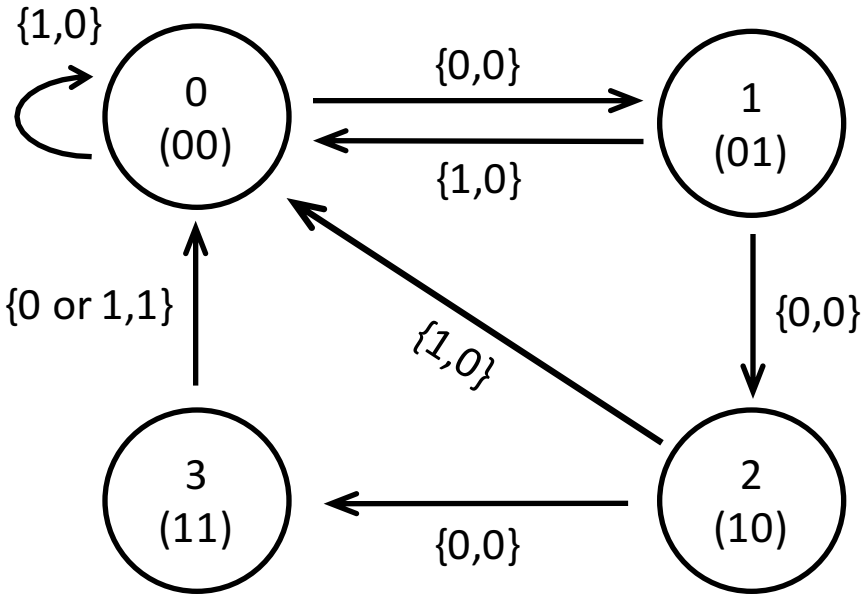
- $S_0(t+T) = R'S_0'(S_1' + S_1)$
- $S_0(t+T) = R'S_0'$

$Out = R'S_1S_0 + RS_1S_0$

- $Out = S_1S_0(R' + R)$
- $Out = S_1S_0$

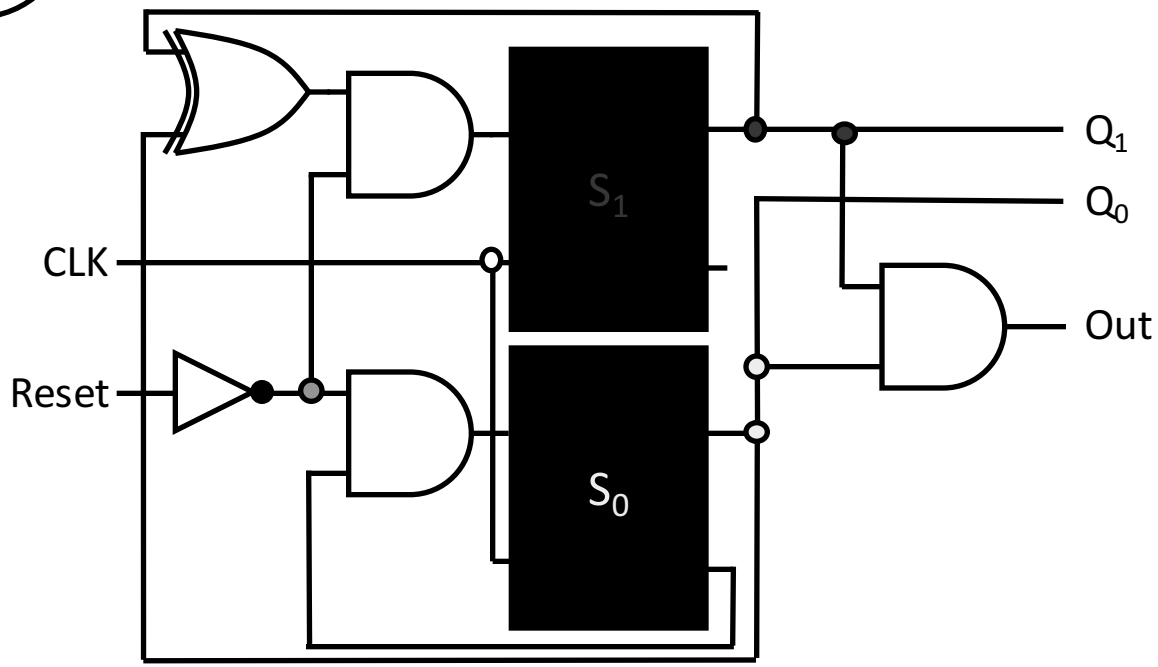
HARVARD

Example: Mod 4 Counter



Input	Current State		Next State		Output
R	$S_1(t)$	$S_0(t)$	$S_1(t+T)$	$S_0(t+T)$	
0	0	0	0	1	0
0	0	1	1	0	0
0	1	0	1	1	0
0	1	1	0	0	1
1	0	0	0	0	0
1	0	1	0	0	0
1	1	0	0	0	0
1	1	1	0	0	1

$$S_1(t+T) = R'(S_0 \oplus S_1)$$
$$S_0(t+T) = R'S_0{}'$$
$$Out = S_1 S_0$$



HARVARD

Another Problem

A. Using synchronous D flip-flops realize 2-bit counter that counts 0, 1, 2, 3, 0, 1, 2… and so on when its input COUNT=1. When COUNT=0, counter remains in 0. Implement the counter using D flip-flops & necessary logic.

B. Modify the outputs of the counter that you synthesized in part (a) by adding necessary logic, in order to produce:
   - The sequence of even numbers between 0 & 7 when signal EVEN=1 (output should be 0, 2, 4, 6, 0, 2, 4, 6, etc)
   - The sequence of odd numbers between 0 & 7 (7 included) when signal EVEN=0 (output should be 1, 3, 5, 7, 1, 3, 5, 7, etc)

C. In both parts you can use any logic gate with as many inputs as you wish. For part (b) you HAVE TO use the counter that you realized in (a) – you cannot synthesize brand new counter. Please use the diagram provided below & complete it with necessary logic. Show all work, and in particular truth tables & logic equations

Another Problem – Part A

Using synchronous D flip-flops realize 2-bit counter that counts 0, 1, 2, 3, 0, 1, 2… and so on when its input COUNT=1. When COUNT=0, counter remains in 0. Implement the counter using D flip-flops & necessary logic.

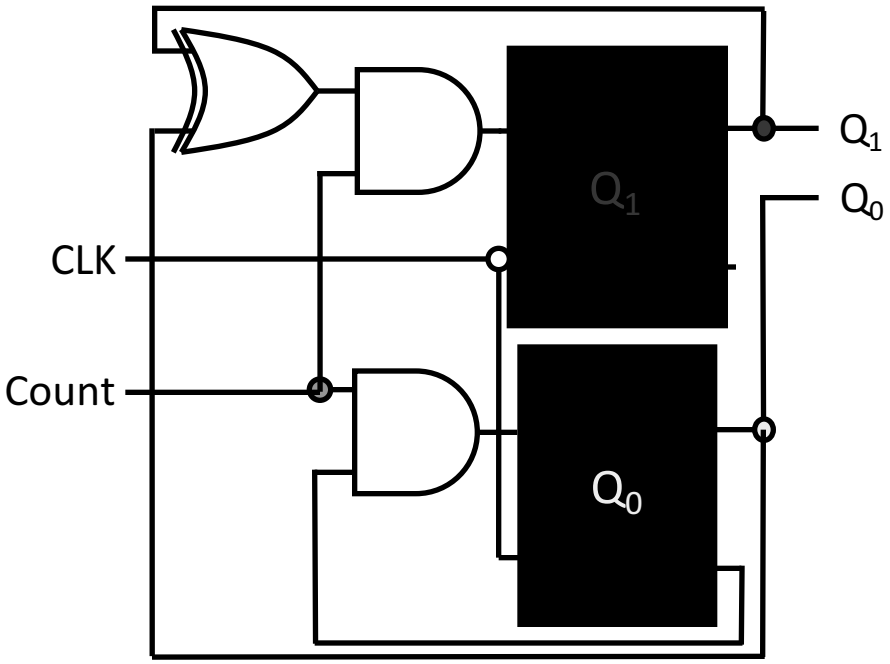
Count	$Q_1(t)$	$Q_0(t)$	$Q_1(t+1)$	$Q_0(t+1)$
0	0	0	0	0
0	0	1	0	0
0	1	0	0	0
0	1	1	0	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	1
1	1	1	0	0

$Q_1(t+1) = \mathrm{C}Q_1'Q_0 + \mathrm{C}Q_1Q_0'$

- $Q_1(t+1) = \mathrm{C}(Q_1'Q_0 + Q_1Q_0')$
- $Q_1(t+1) = \mathrm{C}(Q_0 \oplus Q_1)$

$Q_0(t+1) = \mathrm{C}Q_1'Q_0' + \mathrm{C}Q_1Q_0'$

- $Q_0(t+1) = \mathrm{C}Q_0'(Q_1' + Q_1)$
- $Q_0(t+1) = \mathrm{C}Q_0'$



HARVARD

Another Problem – Part B

Modify the outputs of the counter that you synthesized in part (a) by adding necessary logic, in order to produce:

- The sequence of even numbers between 0 and 7 when signal EVEN=1 (output should be 0, 2, 4, 6, 0, 2, 4, 6, etc)
- The sequence of odd numbers between 0 and 7 (7 included) when signal EVEN=0 (output should be 1, 3, 5, 7, 1, 3, 5, 7, etc)

Even	$Q_1(t+1)$	$Q_0(t+1)$	Z_2	Z_1	Z_0
0	0	0	0	0	1
0	0	1	0	1	1
0	1	0	1	0	1
0	1	1	1	1	1
1	0	0	0	0	0
1	0	1	0	1	0
1	1	0	1	0	0
1	1	1	1	1	0

$$Z_2 = \mathrm{E}'Q_1Q_0' + \mathrm{E}'Q_1Q_0 + \mathrm{E}Q_1Q_0' + \mathrm{E}Q_1Q_0$$

- $Z_2 = Q_1Q_0'(E' + E) + Q_1Q_0(\mathrm{E}' + \mathrm{E})$
- $Z_2 = Q_1Q_0' + Q_1Q_0$
- $Z_2 = Q_1(Q_0' + Q_0)$
- $Z_2 = Q_1$

$$Z_1 = \mathrm{E}'Q_1'Q_0 + \mathrm{E}'Q_1Q_0 + \mathrm{E}Q_1'Q_0 + \mathrm{E}Q_1Q_0$$

- $Z_1 = Q_1'Q_0\ (E' + E) + Q_1Q_0(\mathrm{E}' + \mathrm{E})$
- $Z_1 = Q_1'Q_0 + Q_1Q_0$
- $Z_1 = Q_0(Q_1' + Q_1)$
- $Z_1 = Q_0$

$$Z_0 = \mathrm{E}'Q_1'Q_0' + \mathrm{E}'Q_1'Q_0 + \mathrm{E}'Q_1Q_0' + \mathrm{E}'Q_1Q_0$$

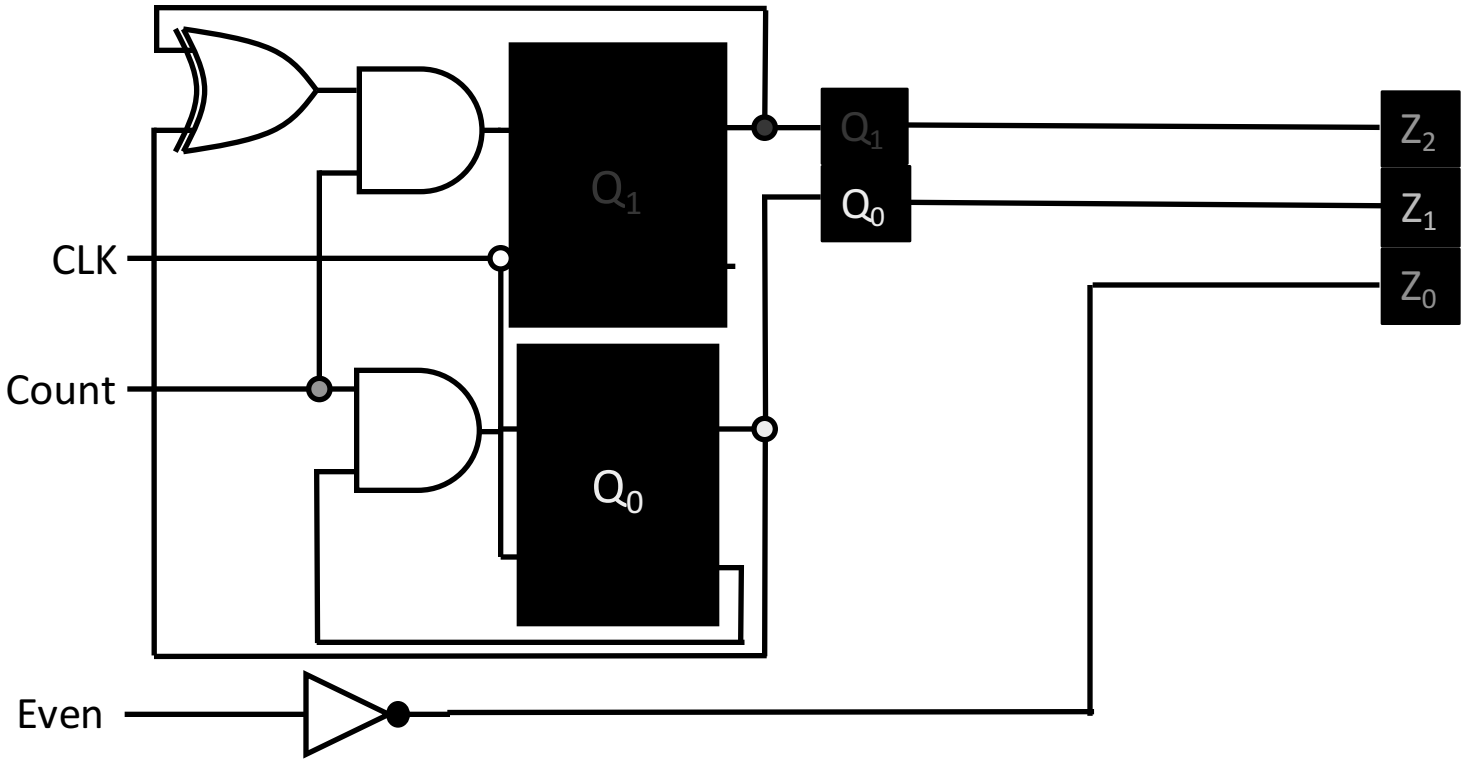
- $Z_0 = \mathrm{E}'(Q_1'Q_0' + Q_1'Q_0 + Q_1Q_0' + Q_1Q_0)$
- $Z_0 = \mathrm{E}'[Q_1'(Q_0' + Q_0) + Q_1(Q_0' + Q_0)]$
- $Z_0 = \mathrm{E}'[Q_1' + Q_1]$
- $Z_0 = E'$

HARVARD

Another Problem – Part C

In both parts you can use any logic gate with as many inputs as you wish. For part (b) you HAVE TO use the counter that you realized in (a) – you cannot synthesize brand new counter. Please use the diagram provided below and complete it with necessary logic. Show all work, and in particular truth tables & logic equations.

$$Z_2 = Q_1$$
$$Z_1 = Q_0$$
$$Z_0 = E'$$

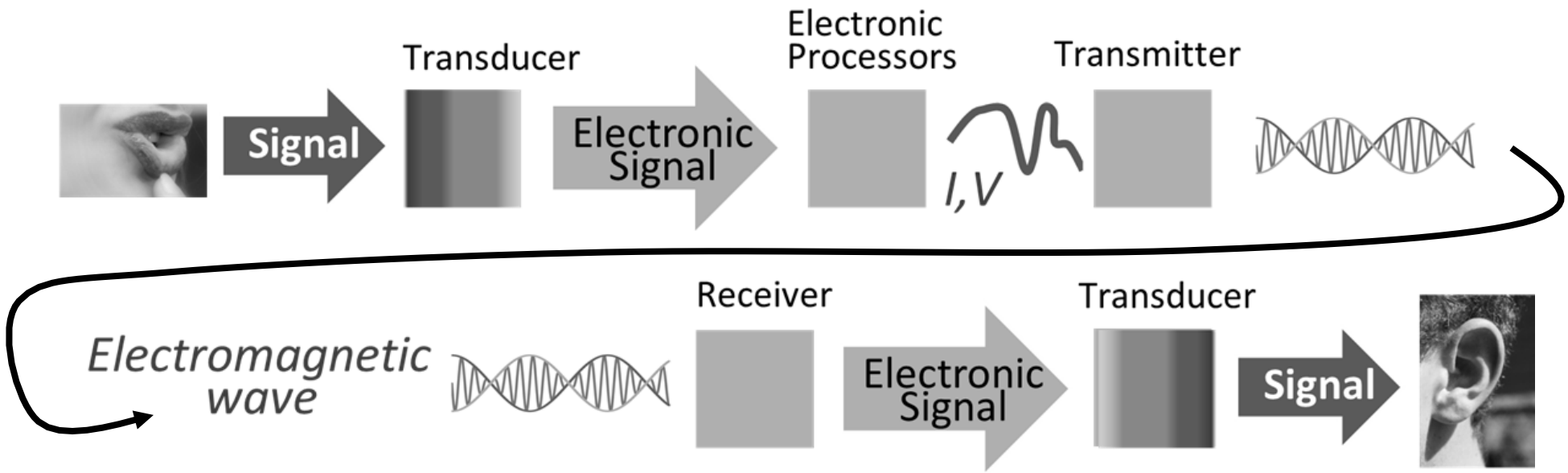


HARVARD

2D Signals Overview

- One dimensional signals: audio
  - Tradeoffs between quality and cost and speed of transmission
- Images: 2D digital signals
  - Same trade-offs, but more so
  - How good is 'good enough' ?
- Sampling & quantization effects on images
- Gray scale images
- Binary images, and processing techniques (in preparation for a future lab)

It's all about information!



Modulation
Transmission

Sampling
Quantization

HIGH QUALITY
- Faithful replication of signal
- No distortion
- No aliasing
- High Signal-to Noise

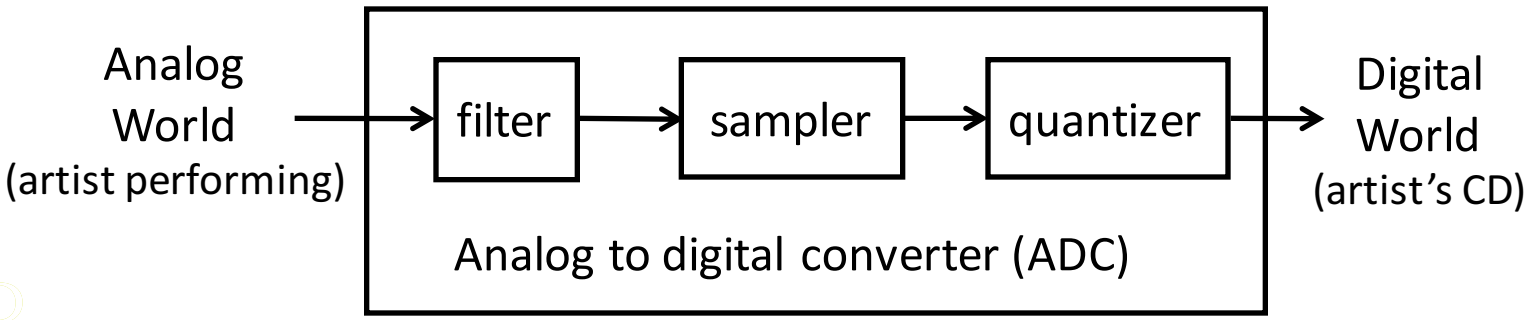
LOW COST
HIGH SPEED TRANSMISSION

Compression
How much information is 'just enough?

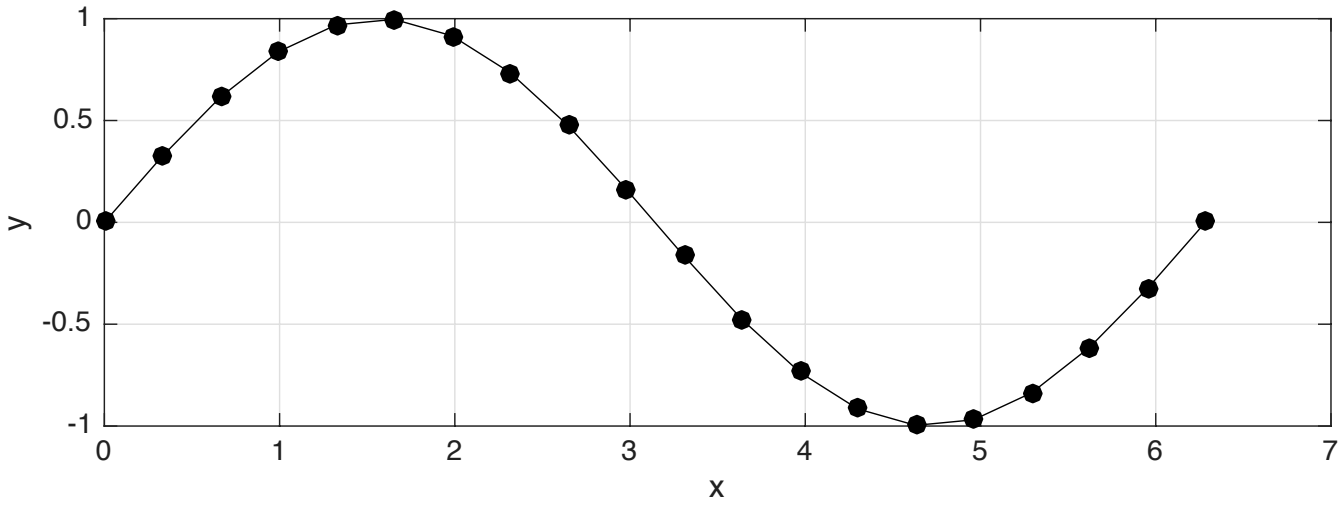
HARVARD

Summary: Analog to Digital Converter Design

1. Low pass filter, with cut-off frequency f_{max}
   - Needed to make sure that signal is *bandwidth limited*
2. Sampling frequency needs to be large enough to obey Nyquist Criterion
   - $f_s > 2 \cdot f_{max}$
   - CD quality: $f_{sample} = 44.1\ kHz$
3. Quantizer needs to have enough bits to avoid quantization noise
   - CD quality: 16-bit quantizer ➔ $SNR = 90\ dB$
4. Quantizer needs to be fast & have enough dynamic range to be able to capture large and small amplitudes



Our audio analog signals were ‘1D’ (amplitude as a function of time)



A MATLAB PROGRAM
to calculate & plot a time-varying SINE wave

```
clear all
close all

x = [0:19]/19*2*pi;
signal=sin(x);
subplot(211);
plot(x,signal,'ko-','MarkerFaceColor','k');
grid
xlabel('x');
ylabel('y');
```

A ‘string’ of 20 numbers, different values of the signal at various times

```
signal = [ 0, 0.325, 0.614, 0.837, 0.969, 0.997, 0.916,
0.736, 0.476, 0.165, -0.165, -0.476, -0.736, -0.916, -
0.997, -0.969, -0.837, -0.614, -0.325, 0];
```

Like a ‘vector’ with 20 elements

HARVARD

Images are ‘2D’: with a spatial extent

There are many ways and levels of sophistication in representing 2D images….
Let’s begin with a ‘grey scale’

Image of Venus, reflected microwaves

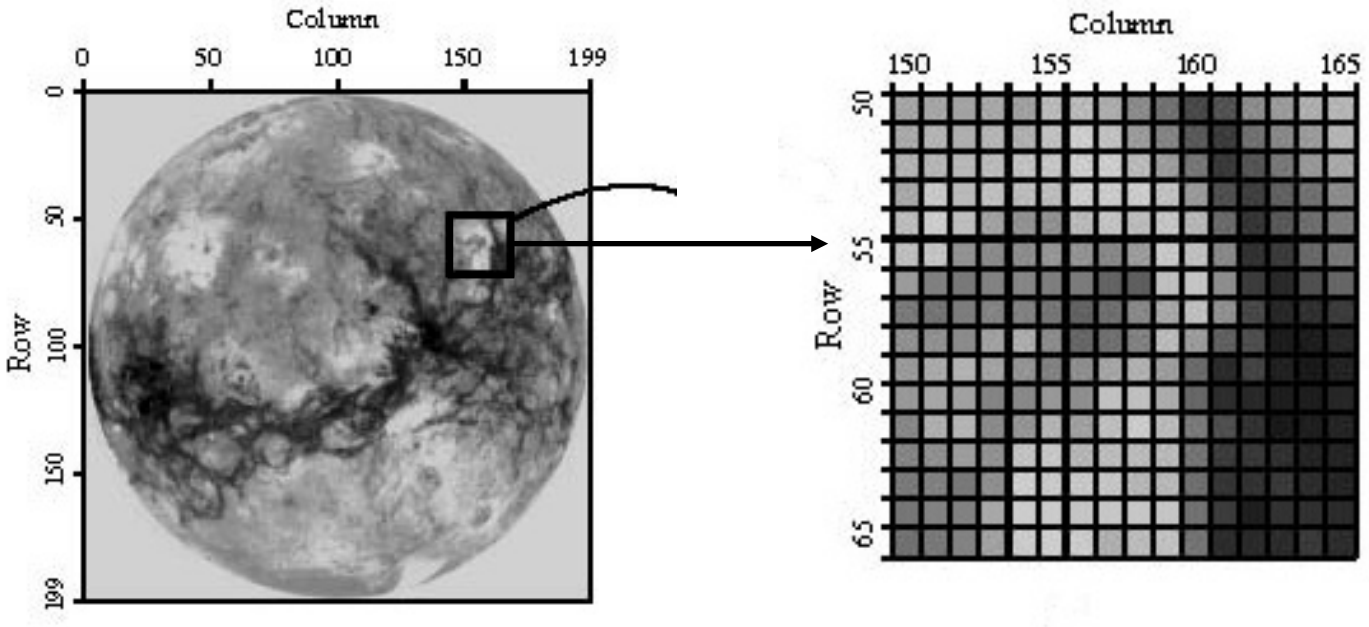
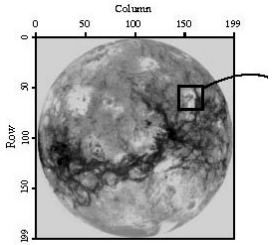


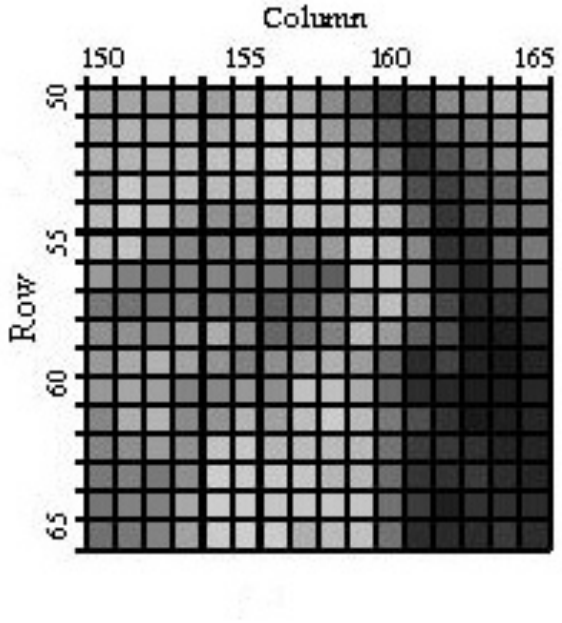
Image is segmented into pixles

- *Pixel = picture element*
- Each pixel represents a *sample* of the original
- The value of each pixel is assigned a number within a grey scale
- 0 = black, 255 = white, one BYTE (8 bits) of information/pixel

2D Grey Scale Image

Image of Venus, reflected microwaves





Each pixel has an assigned number value in the grey scale

Column

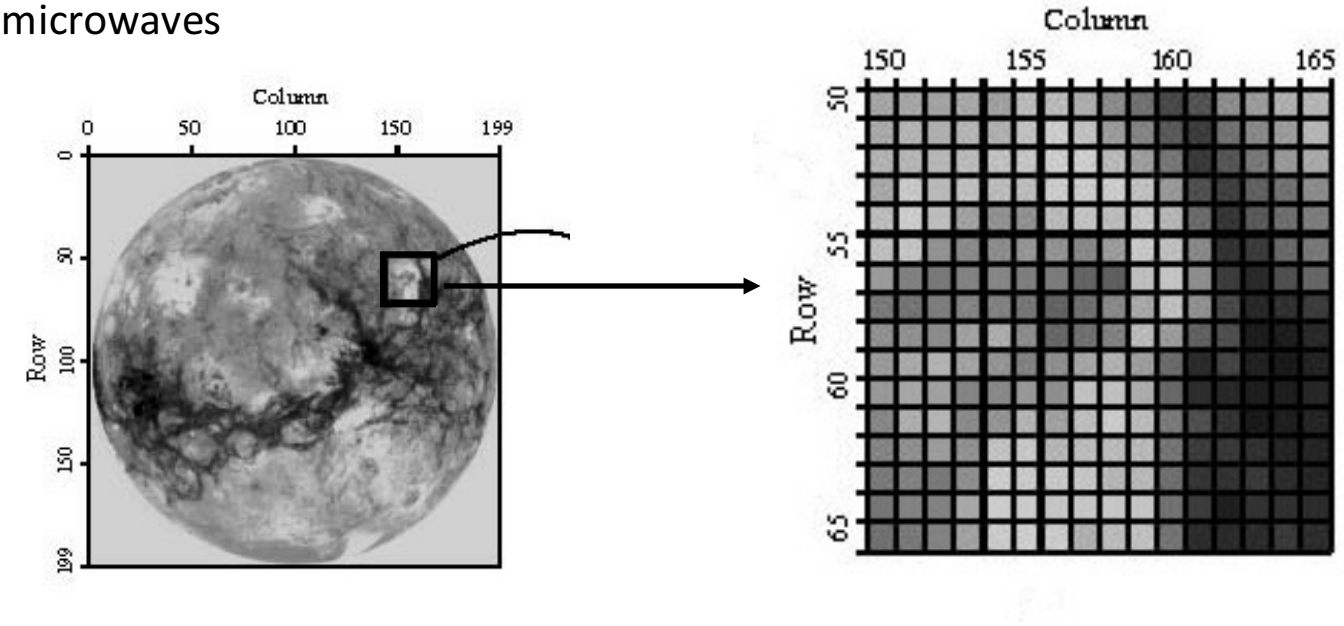
Row	150	151	152	153	154	155	156	157	158	159	160	161	162	163	164	165
50	183	183	181	184	177	200	200	189	159	135	94	105	160	174	191	196
51	184	195	190	195	191	205	216	206	174	153	112	80	134	157	174	196
52	194	196	198	201	206	209	215	216	199	175	140	77	106	142	170	186
53	184	212	200	204	201	202	214	214	214	205	173	102	84	120	134	159
54	202	215	203	179	165	165	199	207	202	208	197	129	73	112	131	146
55	203	208	166	159	160	168	166	157	174	211	204	158	69	79	127	143
56	174	149	143	151	156	148	146	123	118	203	208	162	81	58	101	125
57	143	137	147	153	150	140	121	133	157	184	203	164	94	56	66	80
58	164	165	159	179	188	159	126	134	150	199	174	119	100	41	41	58
59	173	187	193	181	167	151	162	182	192	175	129	60	88	47	37	50
60	172	184	179	153	138	172	163	207	205	188	127	63	56	43	42	55
61	156	191	196	159	167	195	178	203	214	201	143	101	69	38	44	52
62	154	163	175	165	207	211	197	201	201	199	138	79	76	67	51	53
63	144	150	143	162	215	212	211	209	197	198	133	71	69	77	63	53
64	140	151	150	185	215	214	210	210	211	209	135	80	45	69	66	60
65	135	143	151	179	213	216	214	191	201	203	138	61	59	61	77	63

ENTRIES IN ROWS AND COLUMNS: A MATRIX!

MATLAB = MATrix LABoratory: a program you will use in lab to carry out IMAGE PROCESSING

Images are ‘2D’ : with a spatial extent

Image of Venus,
reflected microwaves



More Data, More Challenges

- One second of digital audio information = 8 *kilobytes*
- One second of television = eight *Megabytes*

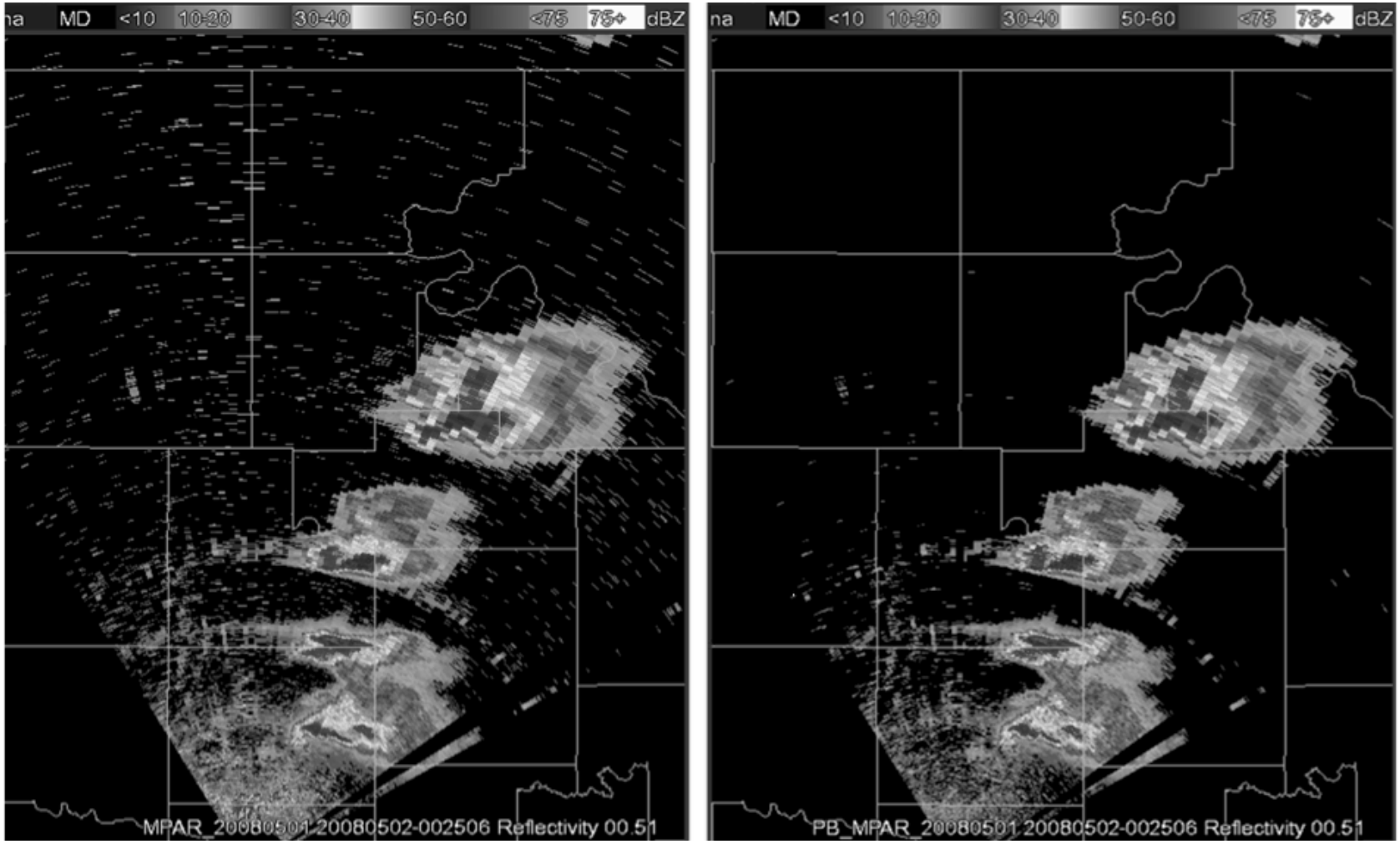
Is 8 bits of quantization too little?

A brightness step of 1/256 (0.39%) LESS THAN THE EYE CAN PERCEIVE!

HARVARD

But really, what if we miss something important?

It’s all about information!



Radar images with (right) and without (left) signal processing to remove background noise

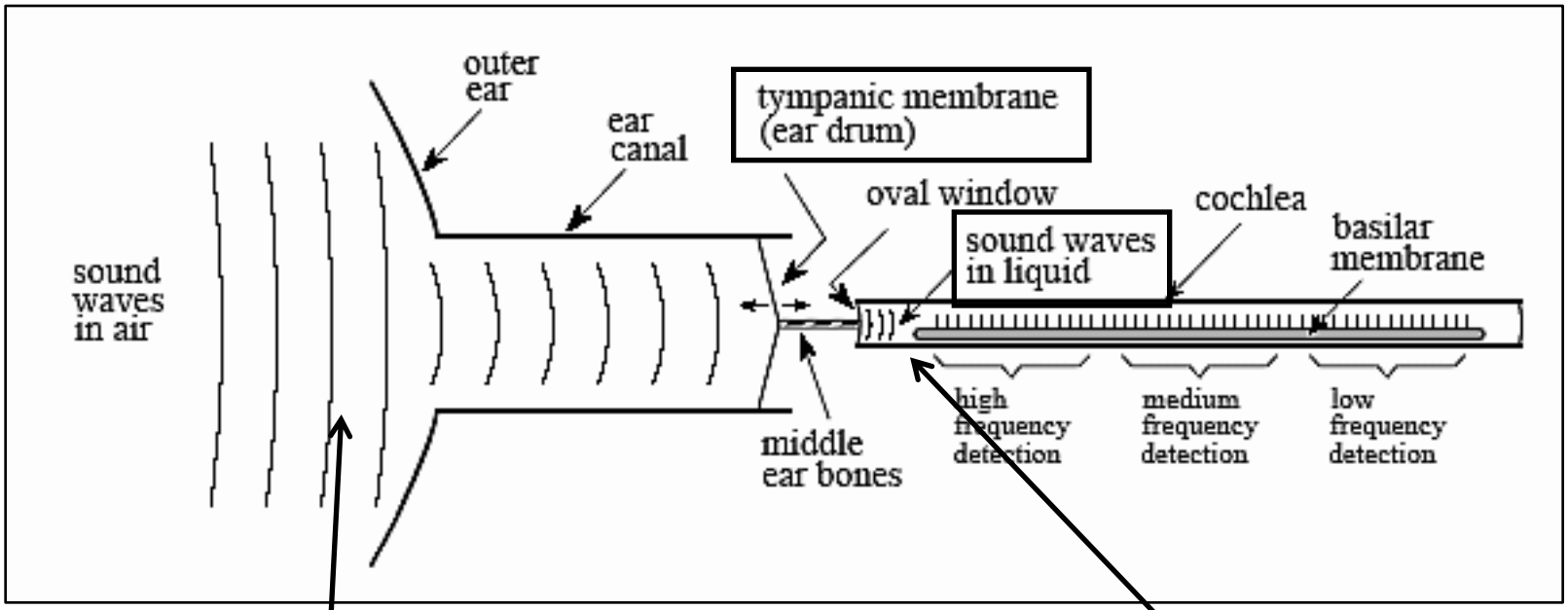
HARVARD

http://www.nssl.noaa.gov/projects/mparsup/dsp.php

The important issues of signals 'good enough'; let's go back to audio & the real detector

Electrical engineering ideas implicit in the 'ear'

- 'Impedance matching': e.g. high resistance to low resistance



Sound waves in air: low mechanical resistance (impedance)

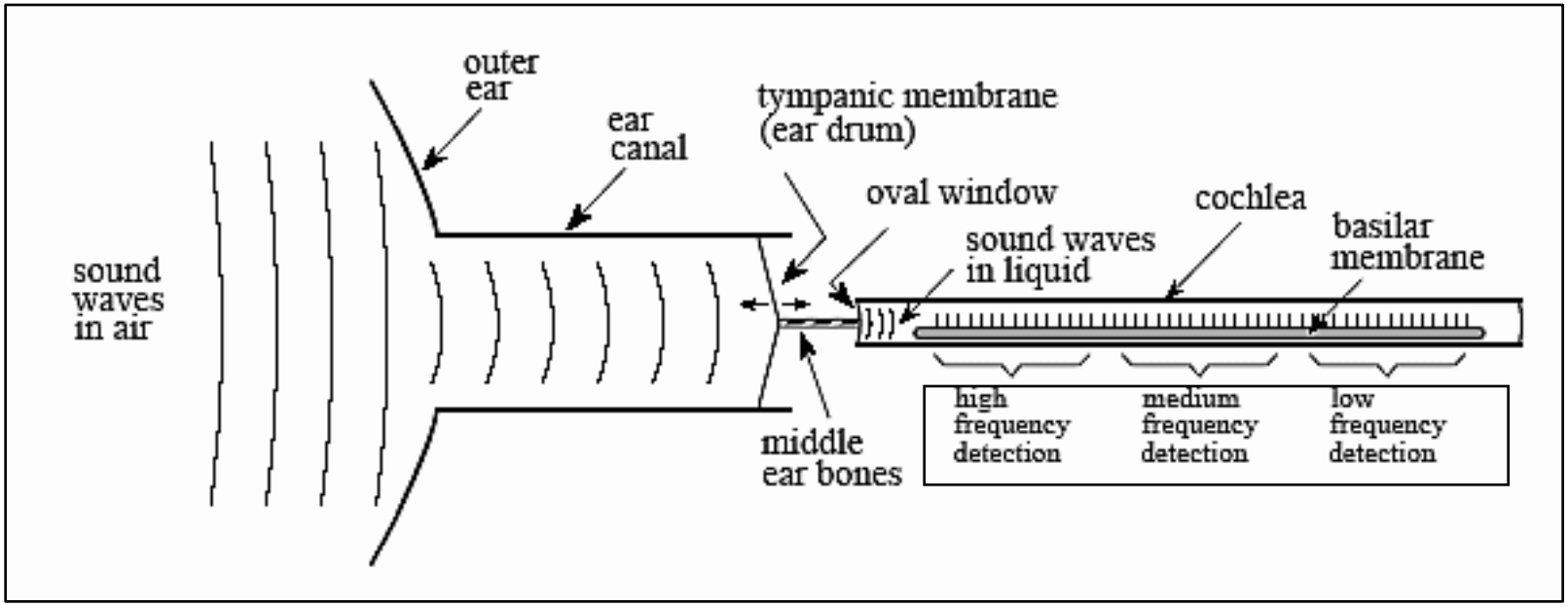
Sound waves in liquid: high mechanical resistance (impedance)

MIDDLE EAR : impedance matching network

The electrical engineering of the ear

Electrical engineering ideas implicit in the 'ear'

- 'Impedance matching': e.g. high resistance to low resistance
- Frequency spectrum analyzer



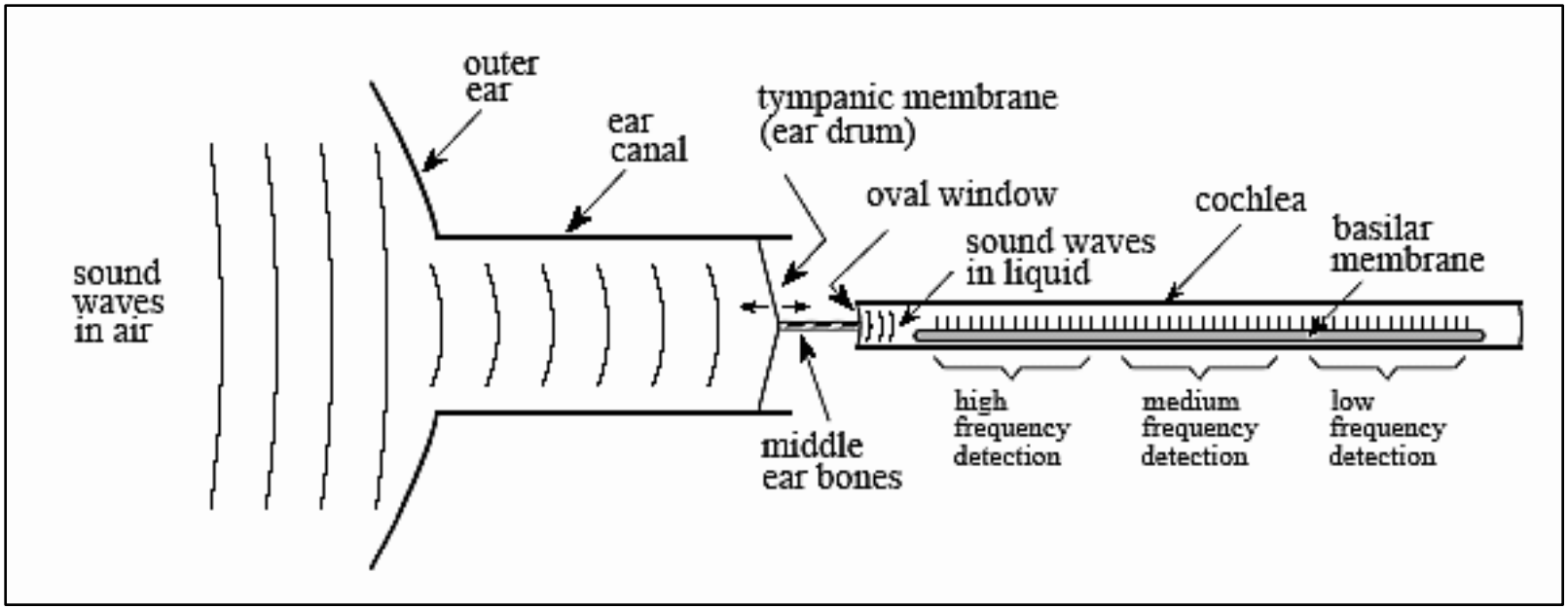
Basilar membrane: stiffest near oval window, responsive to highest frequencies
more flexible membrane, responsive to lower frequency
supports ~ 12,000 nerve cells

Place principle: frequency response a function of 'place'

The electrical engineering of the ear

Electrical engineering ideas implicit in the 'ear'

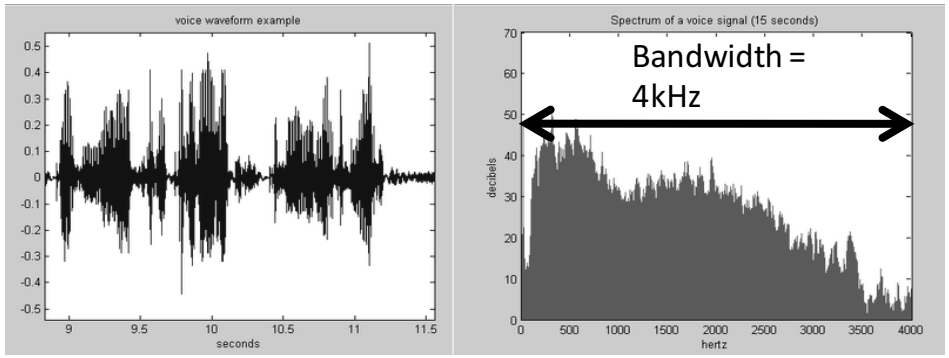
- 'Impedance matching': e.g. high resistance to low resistance
- Frequency spectrum analyzer
- Achieving higher frequency response: how neurons 'encode' information



A single neuron produces *an action potential* for each vibration cycle….but
A single neuron can only produce action potentials < 500 Hz

Volley Principle: several neurons take turns firing
e.g. 10 neurons each fire 300 times/sec → 3 kHz

The important issues of signals 'good enough'; let's go back to audio



Waveform of speech

Response to AMPLITUDE of Audio Signal, in decibel (dB) Sound Power Level, a logarithmic scale

Watts/cm^2	Decibels SPL	Example sound
10^{-2}	140 dB	Pain
10^{-3}	130 dB	
10^{-4}	120 dB	Discomfort
10^{-5}	110 dB	Jack hammers and rock concerts
10^{-6}	100 dB	
10^{-7}	90 dB	OSHA limit for industrial noise
10^{-8}	80 dB	
10^{-9}	70 dB	
10^{-10}	60 dB	Normal conversation
10^{-11}	50 dB	
10^{-12}	40 dB	Weakest audible at 100 hertz
10^{-13}	30 dB	
10^{-14}	20 dB	Weakest audible at 10kHz
10^{-15}	10 dB	
10^{-16}	0 dB	Weakest audible at 3 kHz
10^{-17}	-10 dB	
10^{-18}	-20 dB	

Softer ↓ Louder ↑

For us…..

- Difference between *loudest* and *faintest* sounds: 120 dB
- Can distinguish between 1 dB (12% change)
- But greatest sensitivity between 1-4 KHz
- So we DON'T necessarily need to 'quantize' amplitude equally into 12 one-dB intervals: use less bits, (8?) more judiciously

HARVARD