

CET 241: Day 11-12

DAC (Digital Analog Conversion)

Dr. Noori Kim


Objectives

- Develop a tool for a digital computer to interact with the analog world.
- Study digitization: Quantization, range, precision and resolution.
- Introduce sampling and the Nyquist Theorem.
- Study the basics of sound: electromagnets, speakers, AC vs. DC power, perception of sound.
- Understand how to create sound: loudness, pitch, envelope, and shape
- Use SysTick interrupt to create sounds by programming variable frequencies.

Agenda

- ADC, DAC general concepts
 - Quantization, Sampling
 - Range, Precision, Resolution
 - Nyquist sampling theorem
- Digital Analog Converter (hardware, circuit)
 - Binary weighted, R2R ladder
- DAC application
 - Sound, Speaker, Music
- For practical session
 - SysTick periodic interrupt

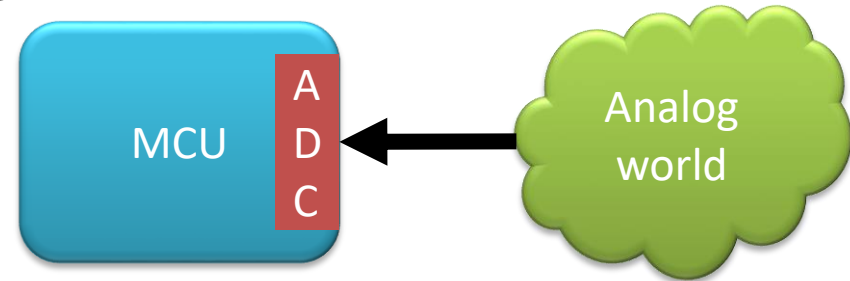
Interaction of world and computers

- Needs a proper **conversion** technique!!!
- The world is inherently Analog (continuous)
 - In both space (x, y, z) and time (t) 
- Computers are inherently Digital (discrete)
 - In both space (x, y, z) and time (t)

- Two possibilities of the conversion:
 1. World to Computers: A to D Conversion
 2. Computers to World: D to A Conversion
- Devices
 1. A to D converters
 2. D to A converters

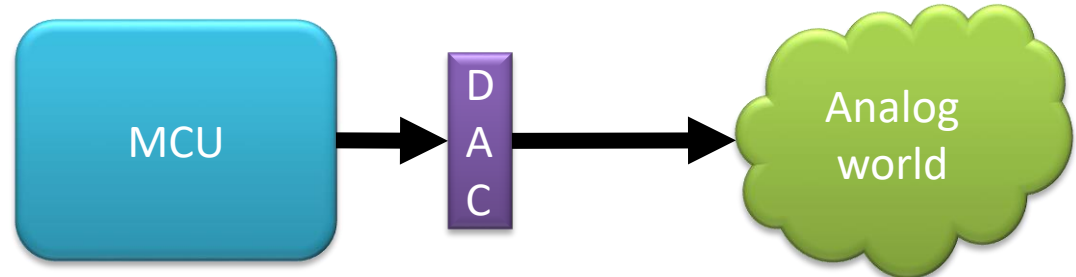
- A to D converters: **Sensors**

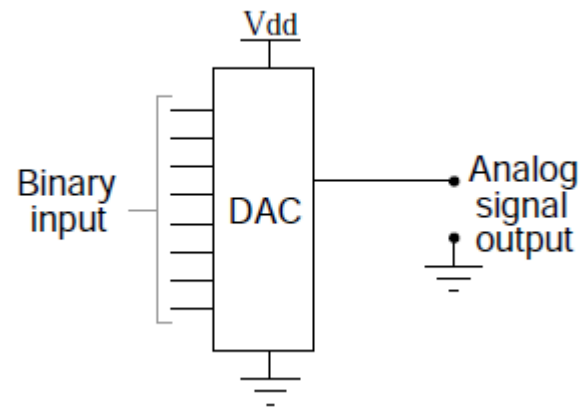
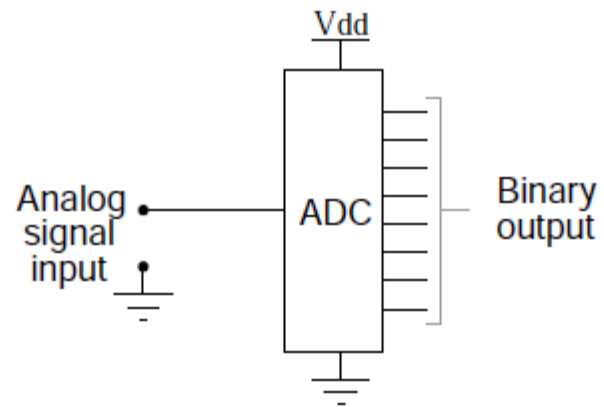
- Temperature
- Sound: microphones
- Optical
- Pressure: sound level meters



- D to A converters: **Actuators**

- Motors
- Sound: speakers
- Brakes ...





We are going to talk about important ADC/DAC concepts including

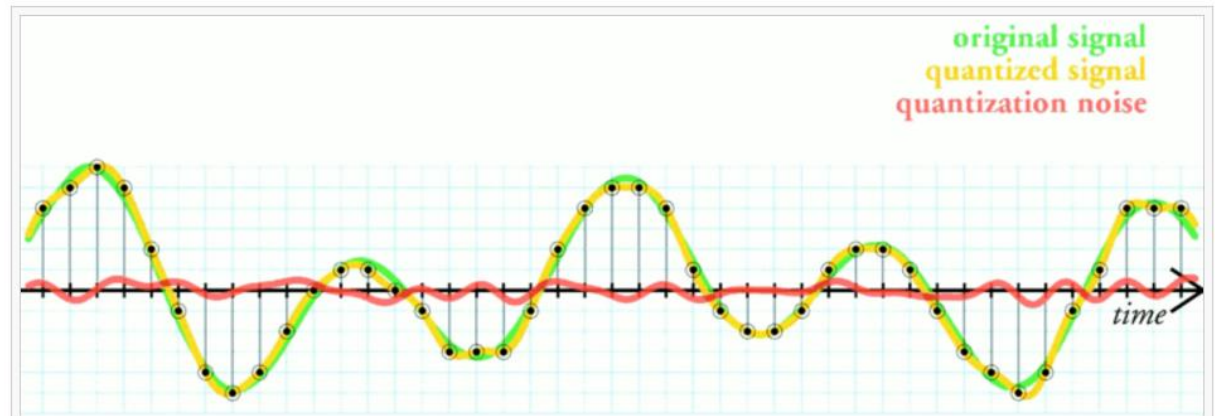
- Quantization (digitization)
- Precision
- Range
- Resolution
- Nyquist Sampling Theorem

Quantization

Quantization (signal processing)

From Wikipedia, the free encyclopedia

Quantization, in mathematics and digital signal processing, is the process of mapping a large set of input values to a (countable) smaller set. Rounding and truncation are typical examples of quantization processes. Quantization is involved to some degree in nearly all digital signal processing, as the

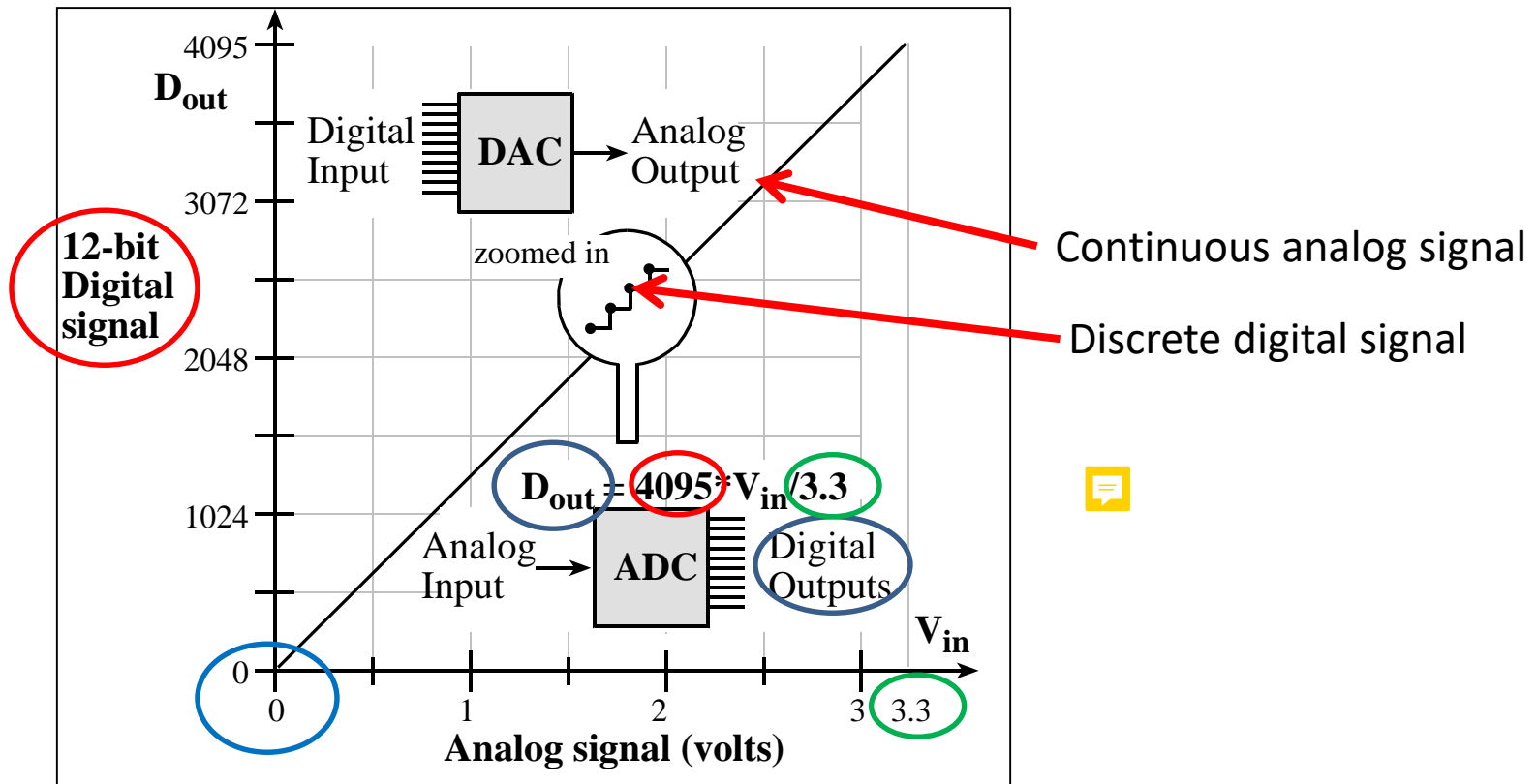


+ considering quantization errors (noise)

An example: a temperature sensor

- Input: analog, continuous, real out-world temperature
- Output: digitized numbers indicating quantized temperature
- Quantization error: real temperature-digital output temperature from the device





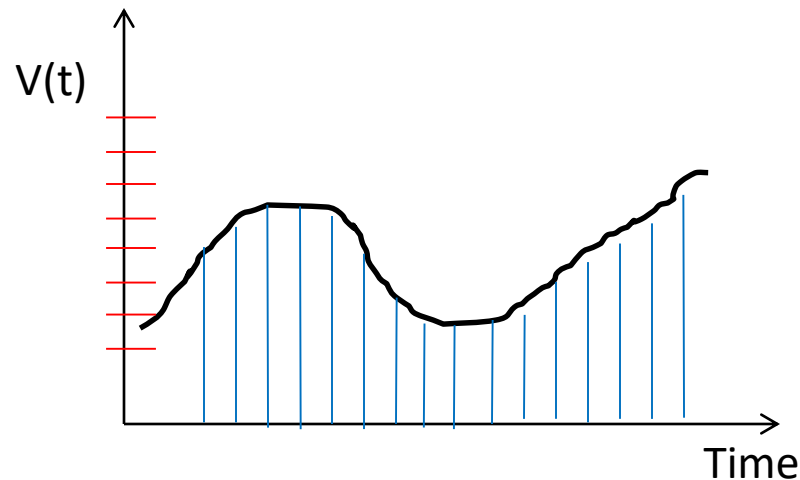
Digitization or quantization

Amplitude of signal (i.e., temp, voltage...)

Time

Digitization

- Purpose: taking an analog signal and digitizing it (= storing it in a computer)
- Two concepts
 1. **Amplitude quantization**: discretizing the signal, separating the amplitude into levels.
 2. **Time quantization**: **sampling**, periodic capture of samples



- **Precision:** number of levels 🗨️
 - i.e., four bits per sample, the precision is 16 (2^4)
- **Range:** min-max analog signal level
 - i.e., voltage
- **Resolution:** the smallest change that the analog signal can be captured
 - i.e., ΔV , but not rigorously defined

– *Resolution(in volt)* = $\frac{\text{Range (in volt)}}{\text{Precision}-1}$

Textbook def.

Case I

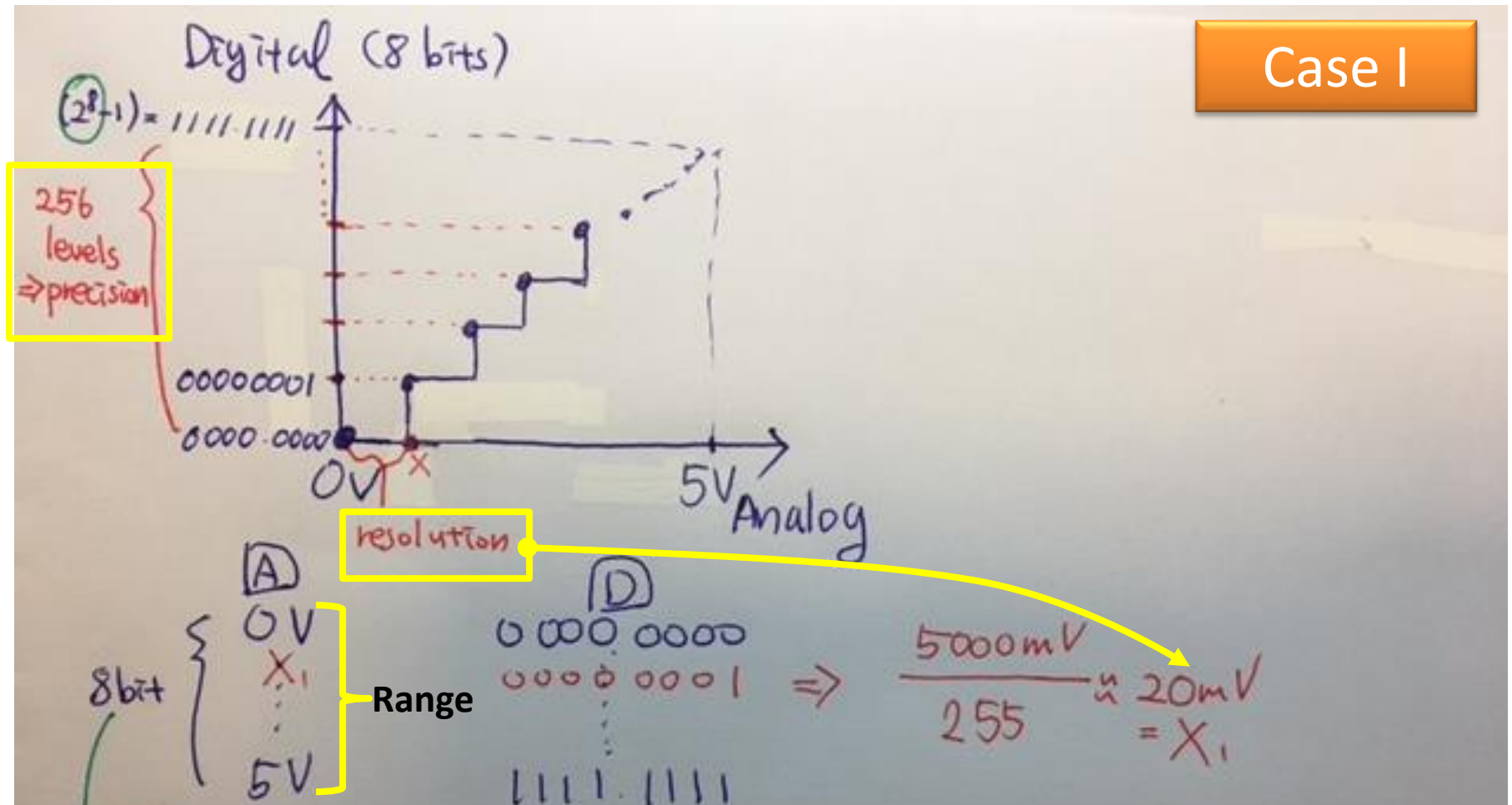
– *Resolution(in volt)* = $\frac{\text{Range (in volt)}}{\text{Precision}}$

Case II

- <http://www.edaboard.com/thread46167.html>

Textbook ex.

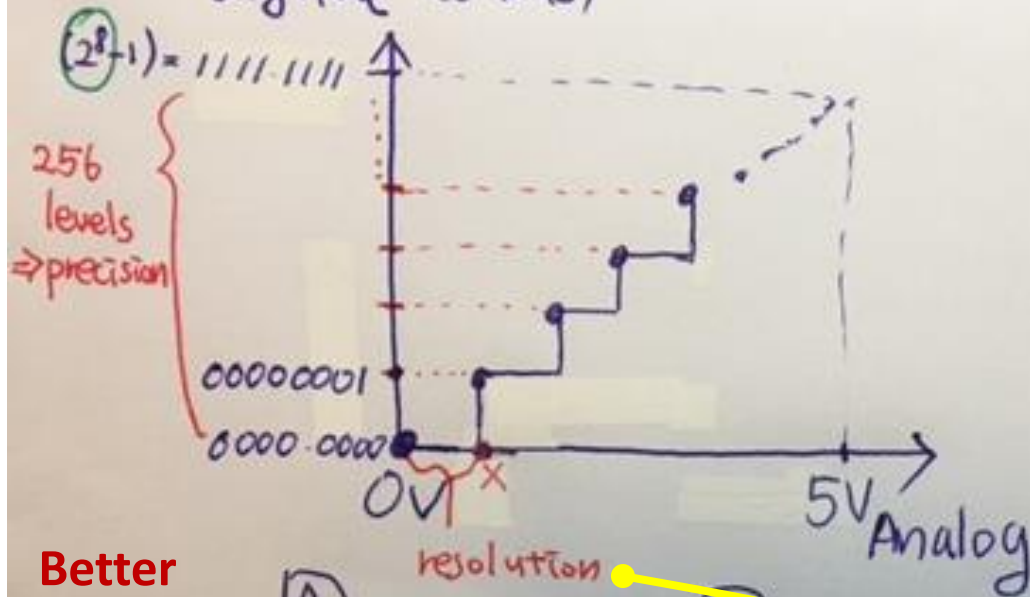
Case I



① Precision : # of digital alternatives , # of levels
i.e., 8 bit DAC \Rightarrow precision is 256 or 8 bit.

② Range : | min - max voltage | of my analog signal
i.e., 5V in the above example case.

Digital (8 bits)



But pricy!!!

Better precision

Better Resolution

(A)

0V
X₁
5V

8bit

more levels

16bit

0V
X₂
5V

(D)

0000.0000
0000.0001
...

$$\Rightarrow \frac{5000\text{mV}}{255} \approx 20\text{mV} = X_1$$

00000000.00000000
0000.0000.0000.0001
...

$$\Rightarrow \frac{5000\text{mV}}{2^{16}-1} = \frac{5000\text{mV}}{65535} \approx 0.07\text{mV} = X_2$$

8 bit DAC case I

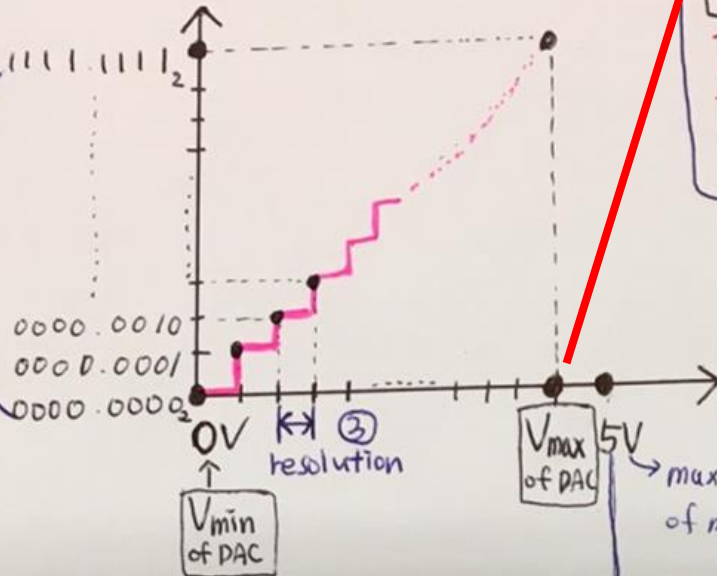
- $Resolution(in\ volt) = \frac{Range\ (in\ volt)}{Precision-1}$
 - $Resolution = \frac{|3.3V - 0V|}{255} = 0.01294$ (Digital_{max}/ V_{dd})
 - $V_{max} = resolution * max\ digital\ value$
 - $3.2994V = 0.01294V * 255$
 - $V_{max} = V_{dd} = 3.3V$
 - The max digital output (1111.1111₂) is mapped to 3.3V

Or another
resolutions

Digital (8 bits)

$(2^8 - 1) = 11111111_2$

① $256(2^8)$ levels
⇒ precision

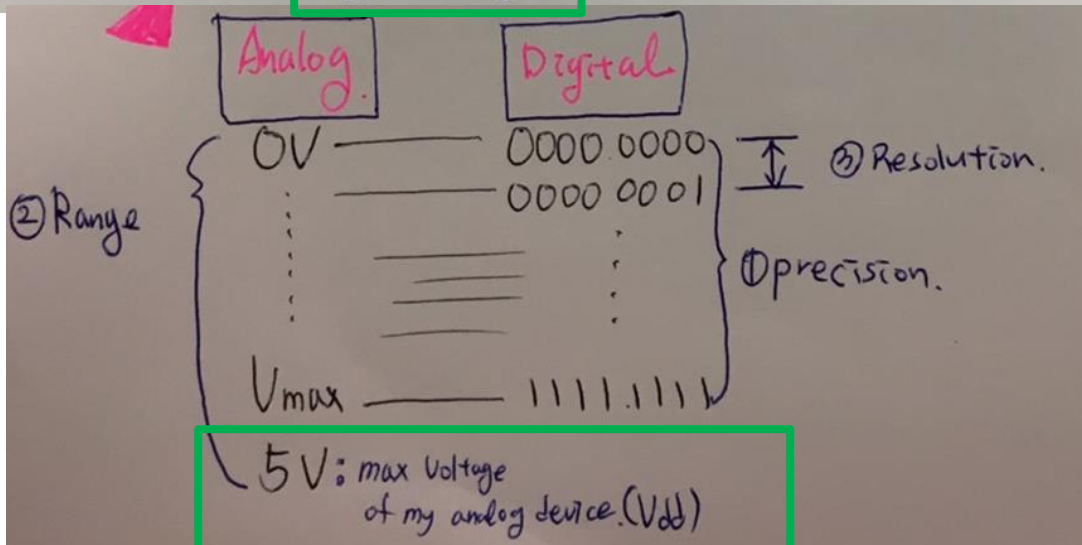


V_{max} of DAC = $\frac{\text{the max of digital value}}{\# \text{ of levels (precision)}} \times V_{dd}$

that one can use for DAC that maps to (1111.1111)

$= \frac{255}{256} \times 5V = ?$

② ⇒ range



8 bit DAC case II

- $Resolution(in\ volt) = \frac{Range\ (in\ volt)}{Precision}$

- $Resolution = \frac{|3.3V - 0V|}{256} = 0.012890625V$

- $V_{max} = resolution * max\ digital\ value$

- $3.287109375V = 0.012890625V * 255$

- $V_{max} = \frac{Digital_{max}}{Precision} \times V_{dd} = \frac{255}{256} \times 3.3V = 3.287V$

- 3.3V (V_{dd}) is not used for DAC output voltage

- The max digital output (1111.1111_2) is mapped to 3.287V

Conclusion for the two resolution formulas

- We have to look at the circuit implementation to calculate the exact resolution that is using for the specific case
- Example 8.2 (Vol2, page 414 or 561)

Example 8.2. Design a 2-bit binary-weighted DAC with a range of 0 to +3.3V using resistors.

Solution: We begin by specifying the desired input/output relationship of the 2-bit DAC. There are two possible solutions depending upon whether we want a resolution of 0.825 V or 1.1 V, as shown as V_1 and V_2 in Table 8.6. Both solutions are presented in Figure 8.30.

N	Q_1	Q_0	V_1 (V)	V_2 (V)
0	0	0	0.000	0.0
1	0	3.3	0.825	1.1
2	3.3	0	1.650	2.2
3	3.3	3.3	2.475	3.3

Table 8.6. Specifications of the 2-bit DAC.

Assume the output high voltage (V_{OH}) of the microcontroller is 3.3 V, and its output low voltage (V_{OL}) is 0. With a binary-weighted DAC, we choose the resistor ratio to be 2/1 so Q_1 bit is twice as significant as the Q_0 bit, as shown in Figure 8.30. Considering the circuit on the right, if both Q_1 and Q_0 are 0, the output V_2 is zero. If Q_1 is 0 and Q_0 is +3.3V, the output V_2 is determined by the resistor divider network



which is 1.1V. If Q_1 is +3.3V and Q_0 is 0, the output V_2 is determined by the network



which is 2.2V. If both Q_1 and Q_0 are +3.3V, the output V_2 is +3.3V. The output impedance of this DAC is approximately 20 k Ω , which means it cannot source or sink much current.

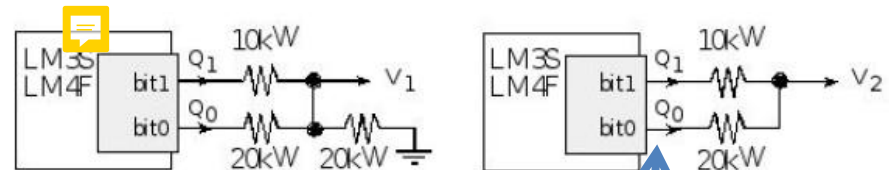


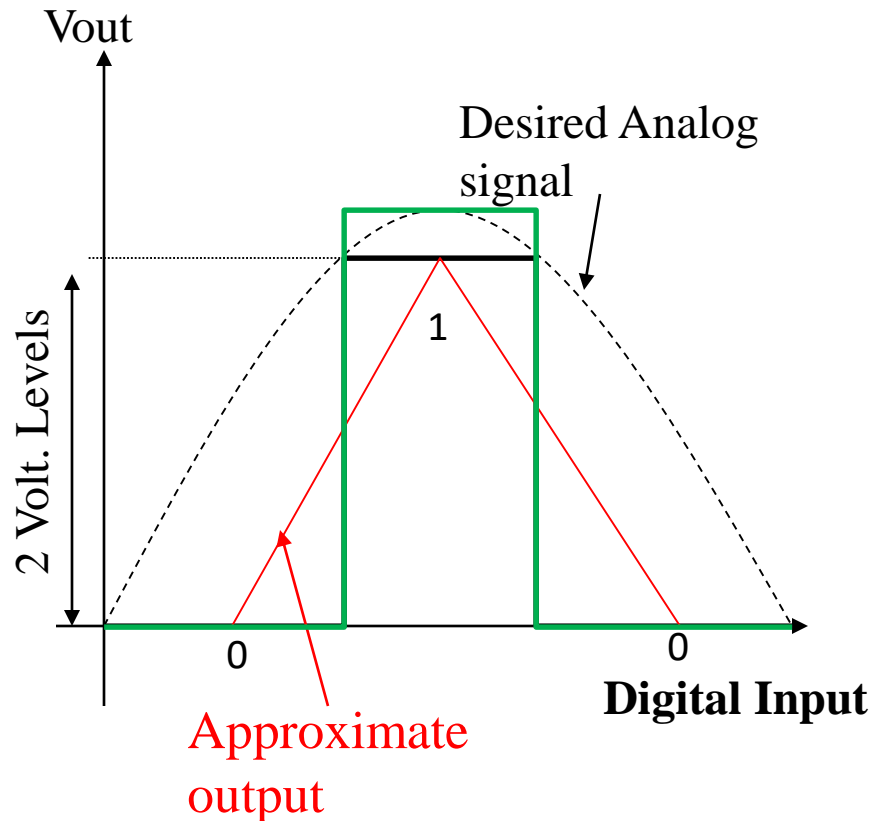
Figure 8.30. Two solutions for a 2-bit DAC.

$$\begin{aligned} \text{Resolution (in volt)} &= \frac{\text{Range (in volt)}}{\text{Precision} - 1} \\ &= \frac{3.3\text{V}}{2^2 - 1} = 1.1\text{V} \end{aligned}$$

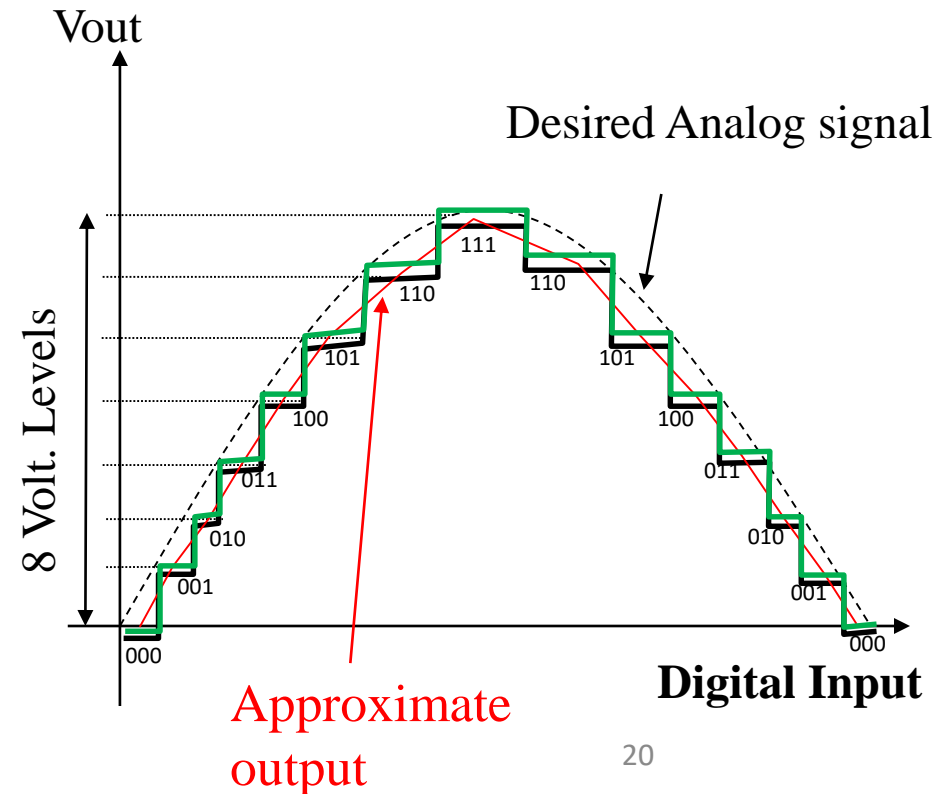
$$\begin{aligned} \text{Resolution (in volt)} &= \frac{\text{Range (in volt)}}{\text{Precision}} \\ &= \frac{3.3\text{V}}{2^2} = 0.825\text{V} \end{aligned}$$

Resolution: the amount of variance in output voltage for every change of the LSB in the digital input.

Poor Resolution(1 bit)



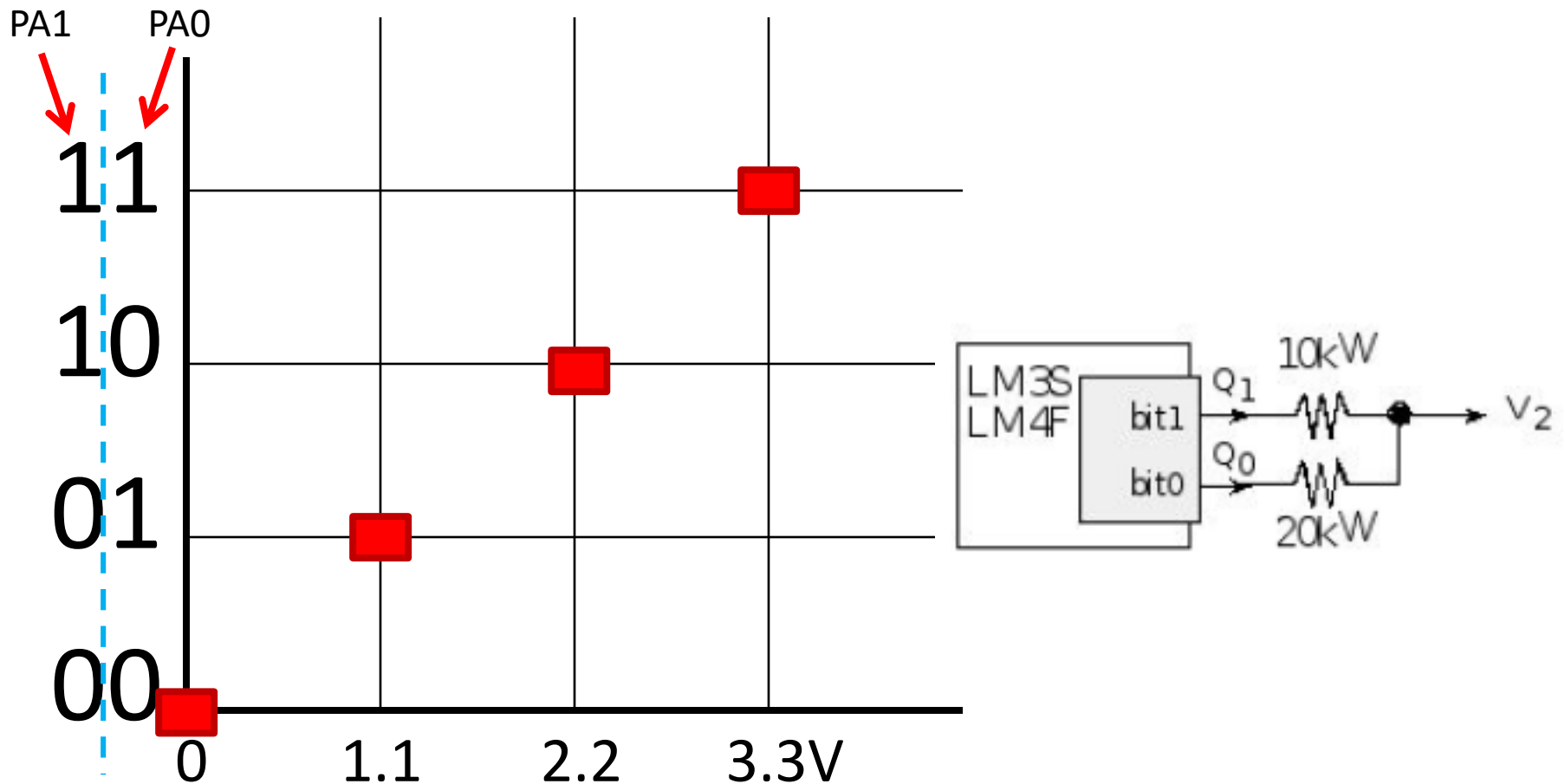
Better Resolution(3 bit)



- We will take case this for our DAC lab (piano)

$$\text{Resolution(in volt)} = \frac{\text{Range (in volt)}}{\text{Precision} - 1}$$

- For your test (Quiz or exam) refer to specific circuits to compute V_{\max} and resolution




Why we call this as DAC?

- Each pin can either output 0 (0V, low) or 1 (3.3V, high) → Digital
- But having parallel digital outputs with extra components (resistors) we can represent various voltage levels → Analog

Trade-offs

	A computing standpoint	An analog standpoint
More levels (i.e. 16 levels, y axis, voltage)	Need more bits per sample (16 levels : 2^4 , 4 bits/sample)	Better precision (16 levels , precision is 16)
More samples (the rate of capture is high, x axis, time)	Need faster processing. (more processing per unit time)	More faithful the digital representation is w.r.t. the analog signal Sampling data more frequently

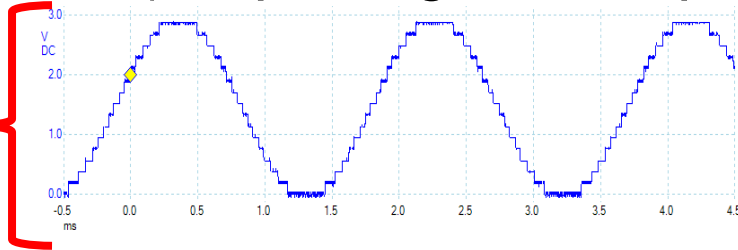


Nyquist theorem: how many samples per second is the best to capture the essence of the analog signal

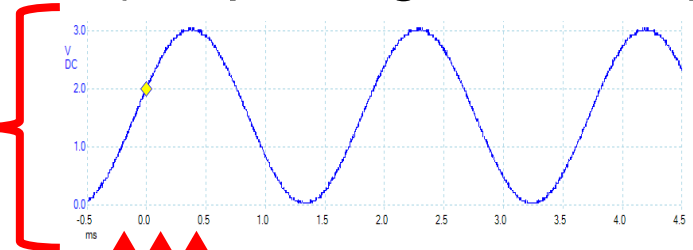
523 Hz sine wave output

12-bit DAC

Case I) capturing 32 samples



Case II) capturing 256 samples



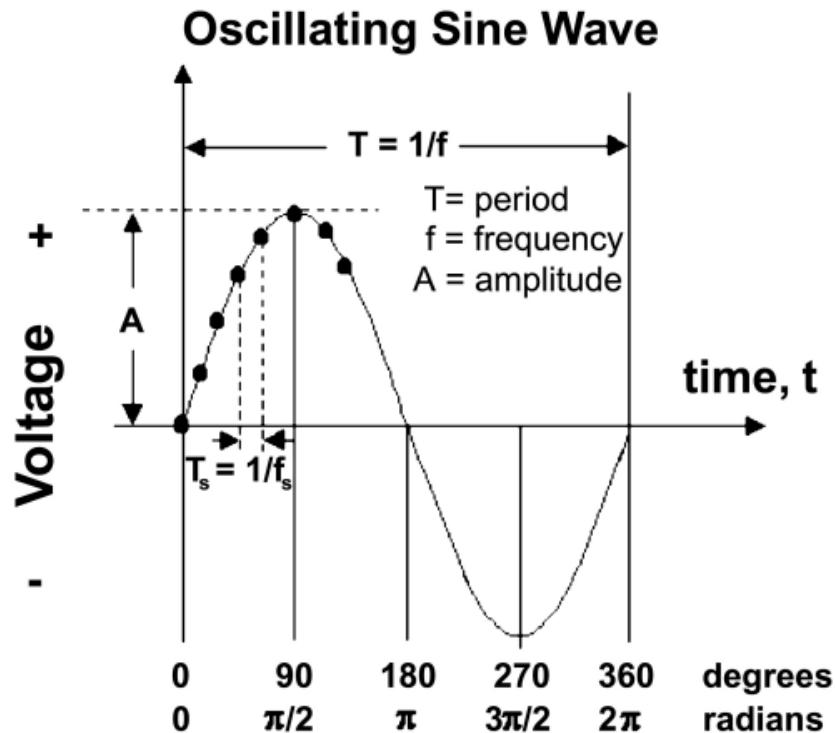
More sampling

- Faster processing required
- More faithful representation

Let's talk about a big theorem

Recall: PLL

- What is phase?
 - The position of a point in time (an instant) on a waveform cycle (wiki)



1. $T = \text{period} = \text{clock period} = \text{cycle}$
2. $F = 1/T$ (frequency)
 - Measured in cycles per second
 - Ex: 1 kHz = 1 000 cycles / s
 - 1 MHz = 1 000 000 cycles / s
3. $A = \text{amplitude}$
4. $T_s = 1/f_s$: Sampling time

Ex) CPU operates at 100 Hz, its "clock cycle" is 0.01 second = 10 ms;

Nyquist Sampling Theorem (NST)

*A band-limited signal of finite energy, which has no frequency components higher than **W Hertz**, may be completely recovered from a knowledge of its samples taken at a rate greater than **$2W$** samples per second.*

The highest frequency, i.e., $W=200\text{Hz}$
(200 samples per second)



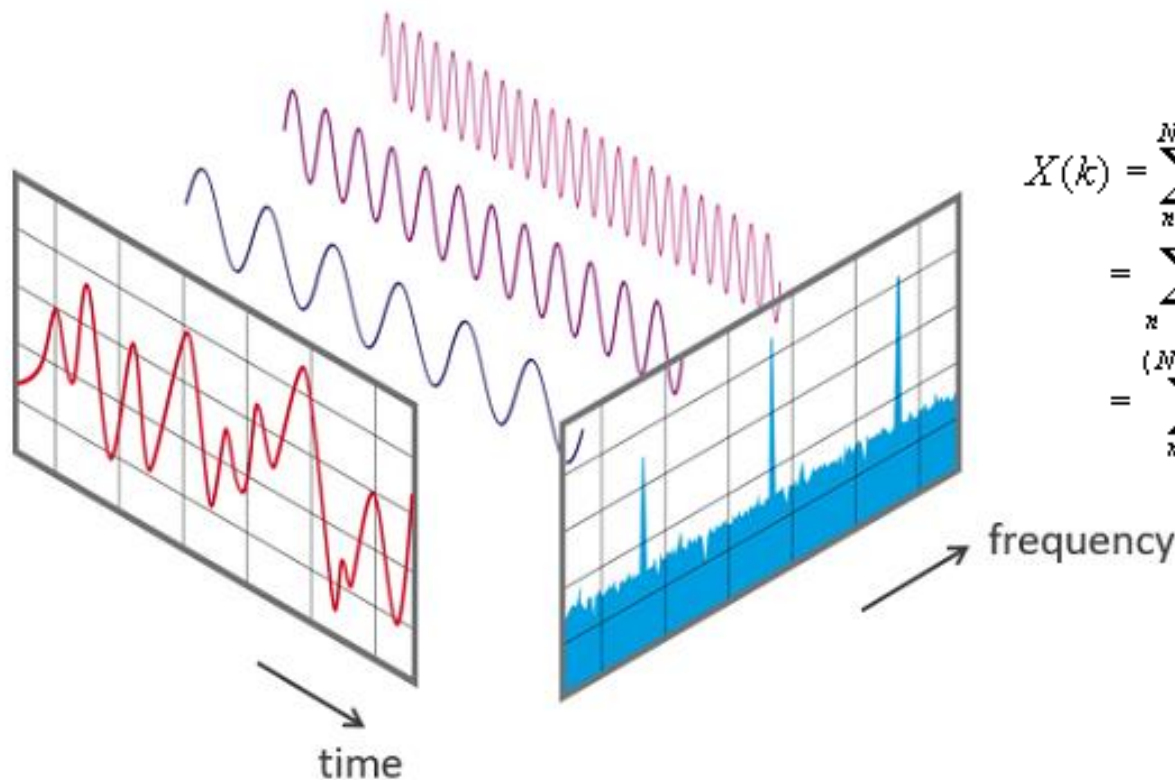
Harry Nyquist
(Also from Bell lab.)

$2W$: 400 samples/sec
400 cycles/sec

General description: NST

- If the signal is **sampled with a frequency of f_s** , then the digital samples only contain frequency components from 0 to $\frac{1}{2} f_s$.
- If the **analog signal** does contain frequency components **larger than $\frac{1}{2} f_s$** , then there will be **an aliasing error** during the sampling process

- Proving the **Nyquist Theorem** mathematically is beyond the scope of this course (FFT)



$$\begin{aligned}
 X(k) &= \sum_{n=0}^{N-1} x(n) W_N^{kn}, \quad k = 0, 1, \dots, N-1 \\
 &= \sum_{n \text{ even}} x(n) W_N^{kn} + \sum_{n \text{ odd}} x(n) W_N^{kn} \\
 &= \sum_{m=0}^{(N/2)-1} x(2m) W_N^{2mk} + \sum_{m=0}^{(N/2)-1} x(2m+1) W_N^{k(2m+1)}
 \end{aligned}$$

Experiments

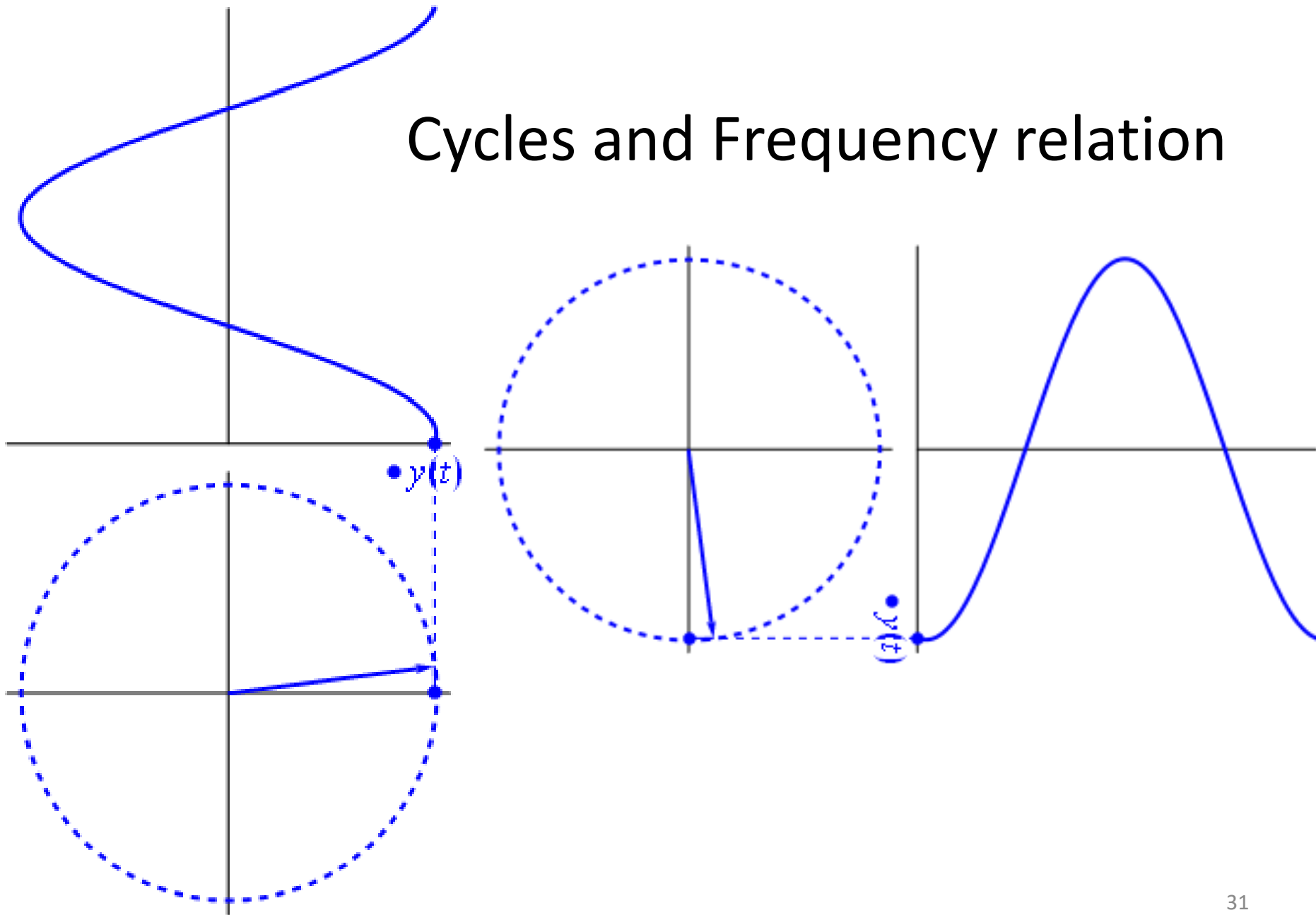


8 secs for one cycle
When I say "start"

- Sample (look at) the needle in every XX seconds
 - After 24 seconds tell me how many cycles does the needle turn around
1. Every 2 sec
 2. Every 4 sec
 3. Every 8 sec
 4. Every 16 sec


- This clocks Freq: $1/8=0.125\text{Hz}$
- Need to sample it at least every 4 secs → NTS $F_s=0.25\text{Hz}$

Cycles and Frequency relation



The Nyquist Theorem says that if a signal is oscillating (or cycling) at frequency of F , in order to capture it faithfully, we must sample at a frequency that is strictly larger than two times of F .

Technical description: NST

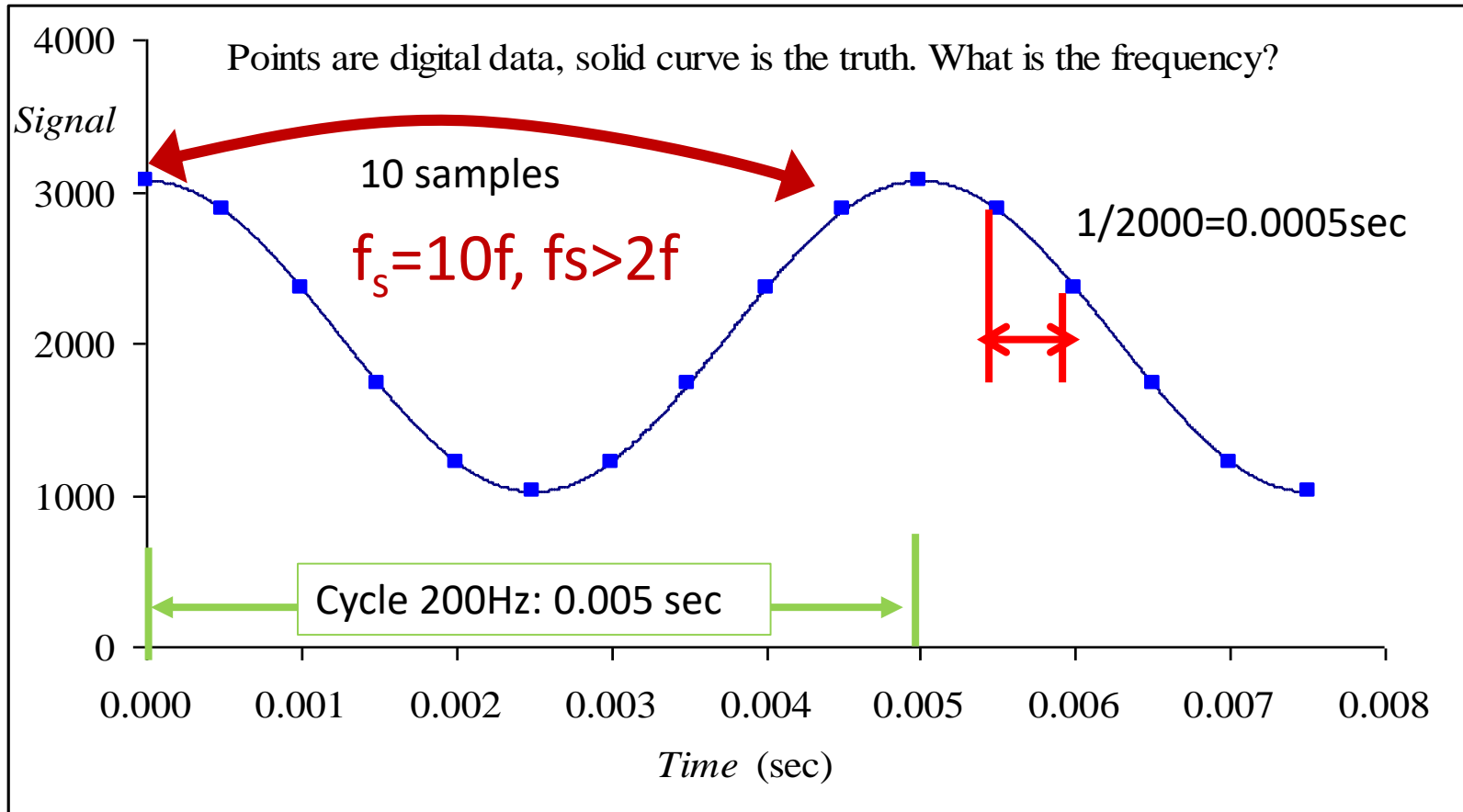
- **A bandlimited analog** signal that has been sampled can be **perfectly reconstructed** from an **infinite** sequence of samples
 - The sampling rate f_s exceeds $2f_{\max}$ samples  per second, where f_{\max} is the highest frequency in the original signal.

- If the analog signal does contain frequency components larger than $(1/2)f_s$, then there will be an **aliasing** error.
 - Aliasing is when the digital signal appears to have a different frequency than the original analog signal.

- If the analog signal does contain frequency components larger than $(1/2)f_s$, then there will be an **aliasing** error.
 - Aliasing is when the digital signal appears to have a different frequency than the original analog signal.

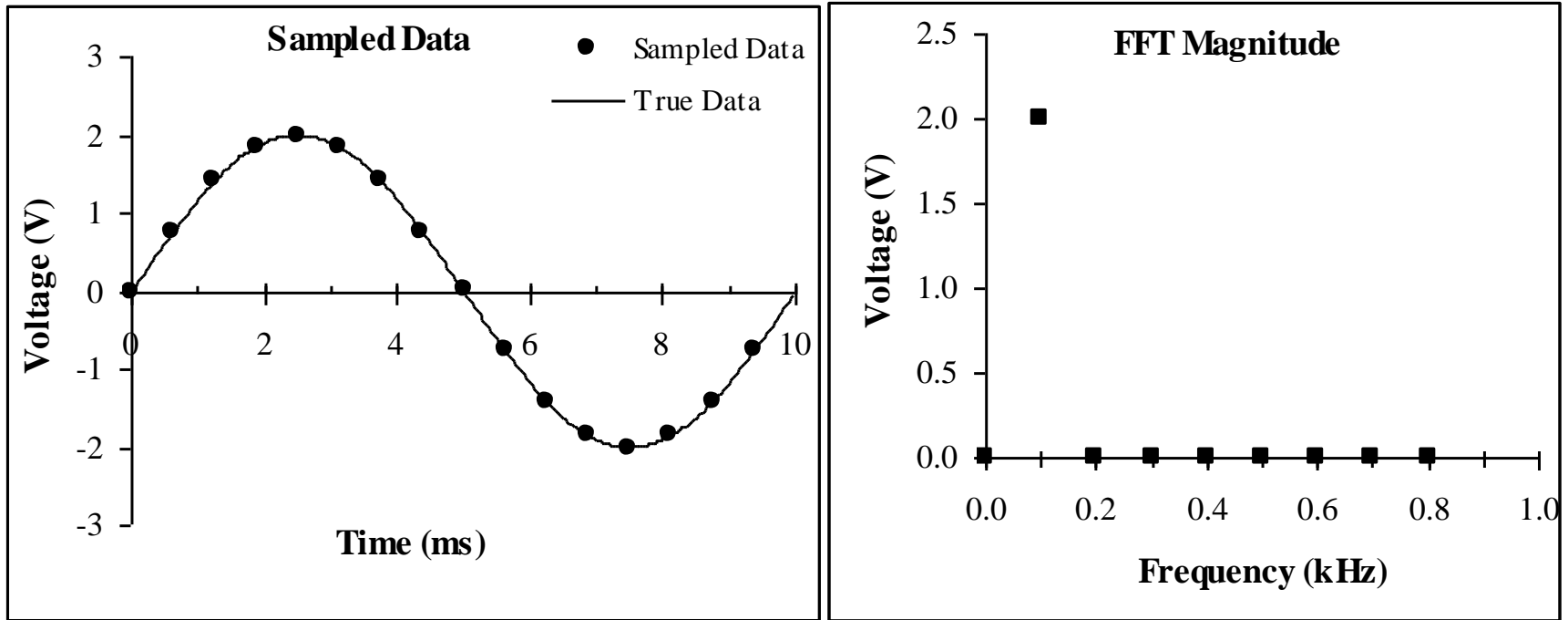
f_s

- 200Hz signal sampled at 2000Hz



<http://www.ece.utexas.edu/~valvano/Volume1/Nyquist.xls>

- f 100Hz signal sampled at f_s 1600Hz

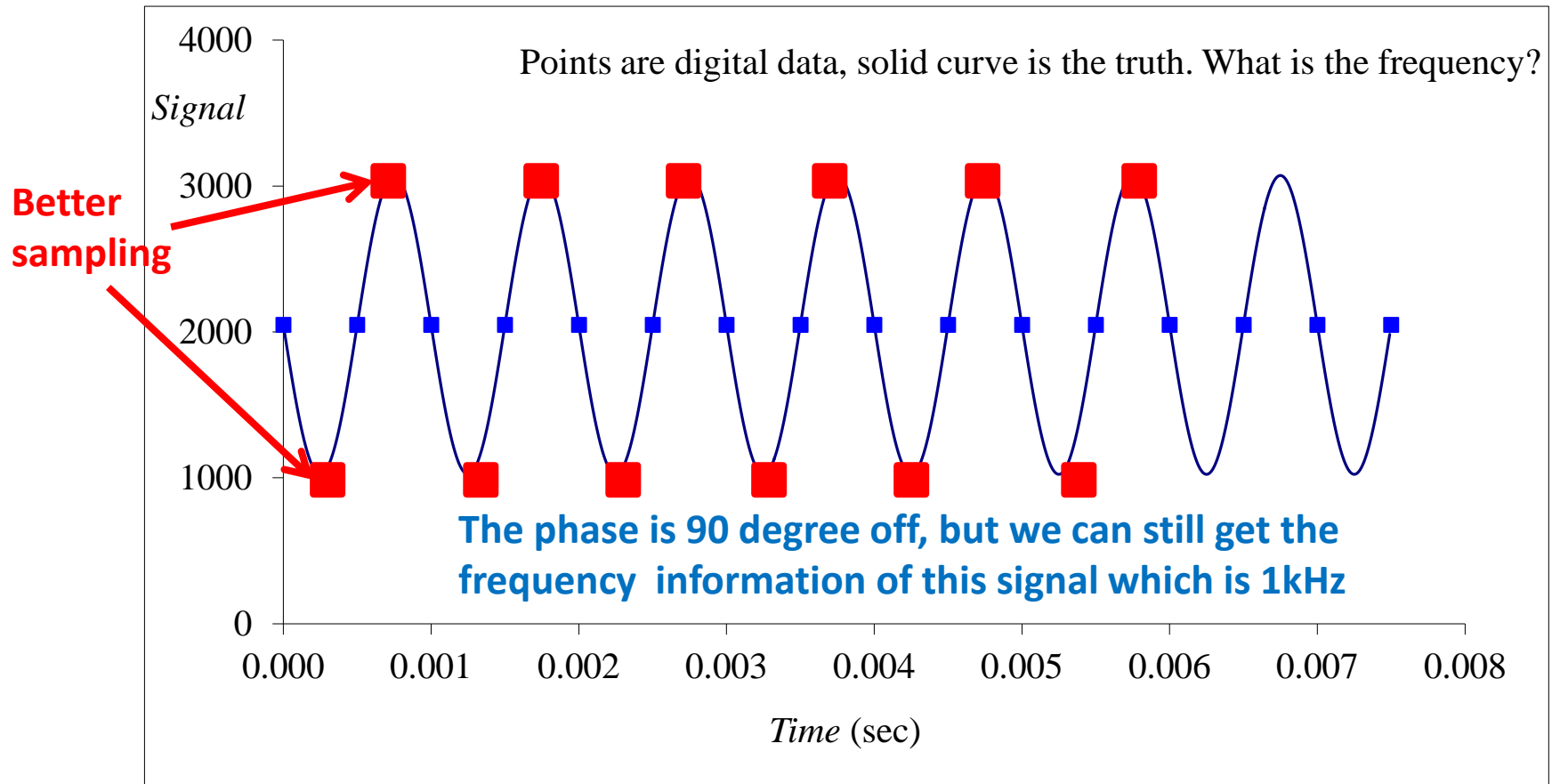


<http://www.ece.utexas.edu/~valvano/EE345L/Labs/Fall2011/FFT16.xls>

$$f = (1/2)f_s$$

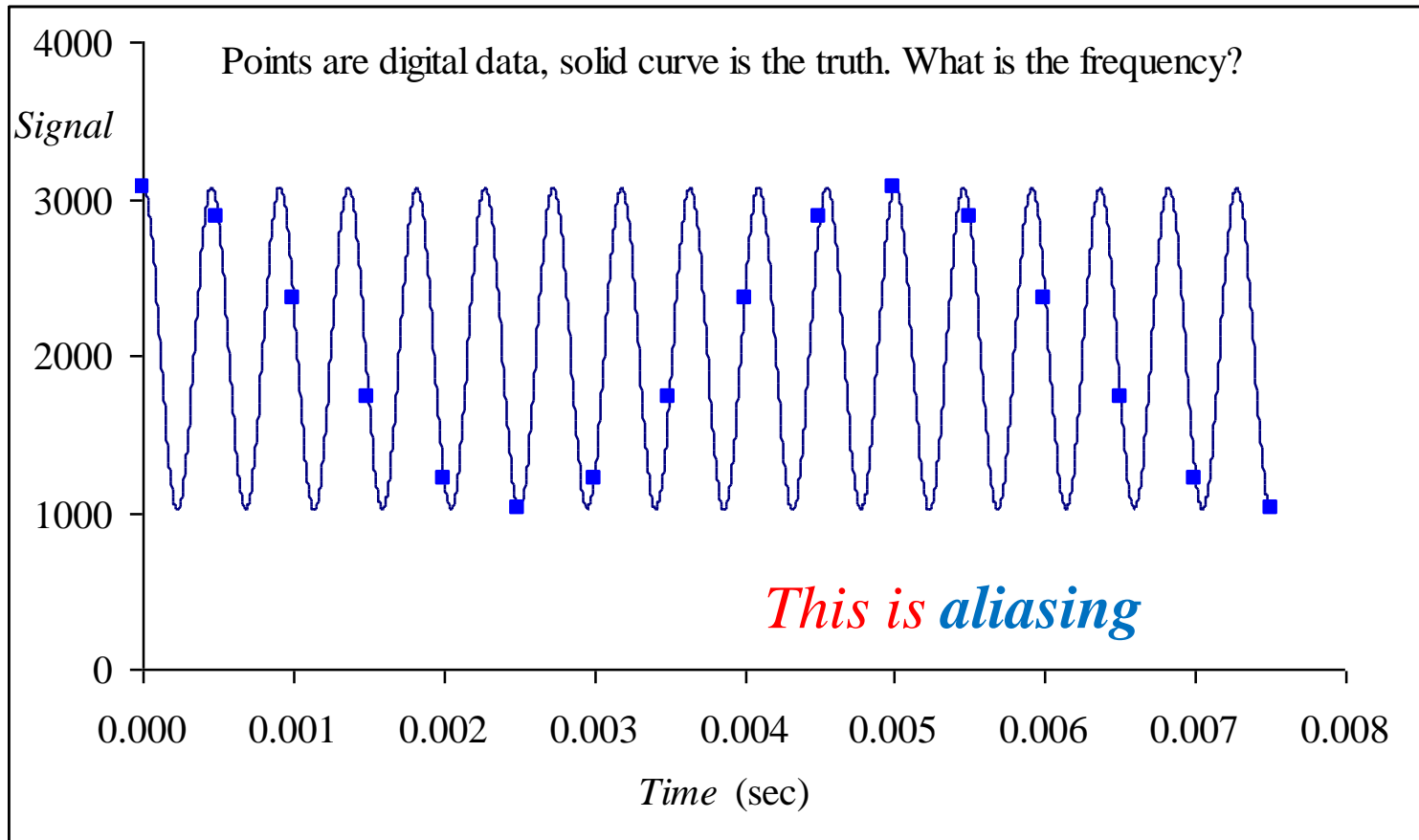
f_s : minimum f_s to reconstruct f

- 1000Hz signal sampled at 2000Hz



<http://www.ece.utexas.edu/~valvano/Volume1/Nyquist.xls>

- 2200Hz signal sampled at 2000Hz

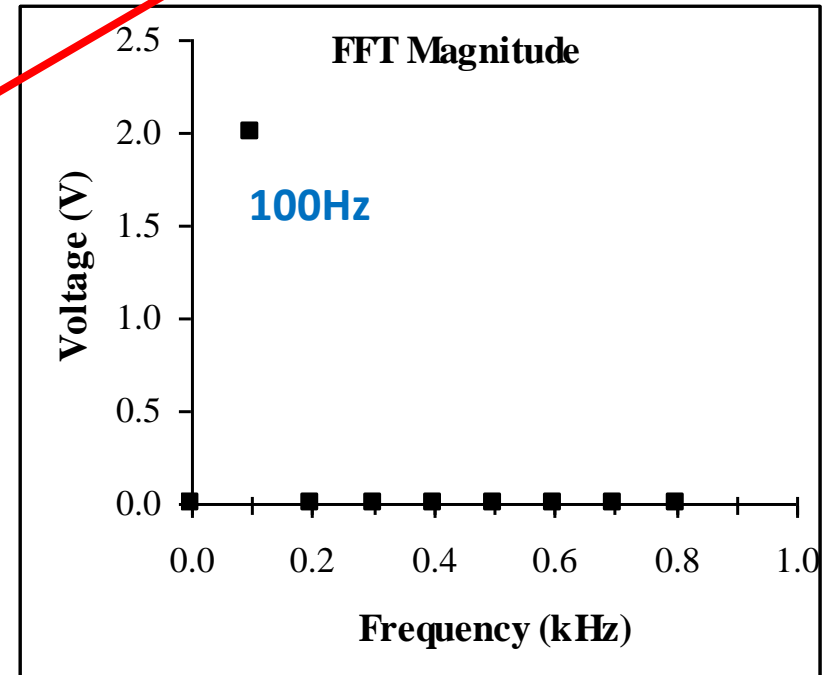
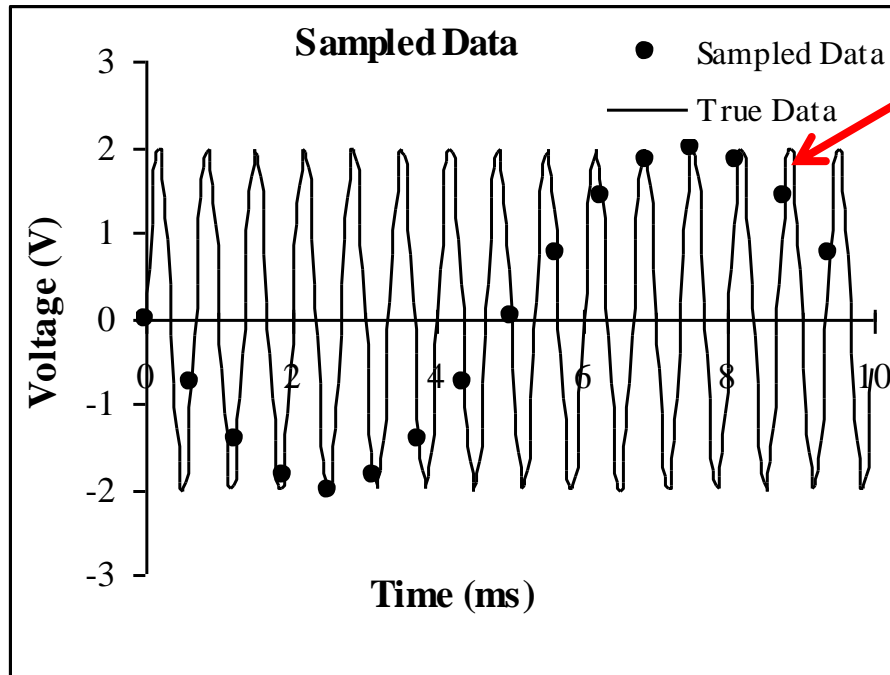


<http://www.ece.utexas.edu/~valvano/Volume1/Nyquist.xls>

Freq of aliased result?

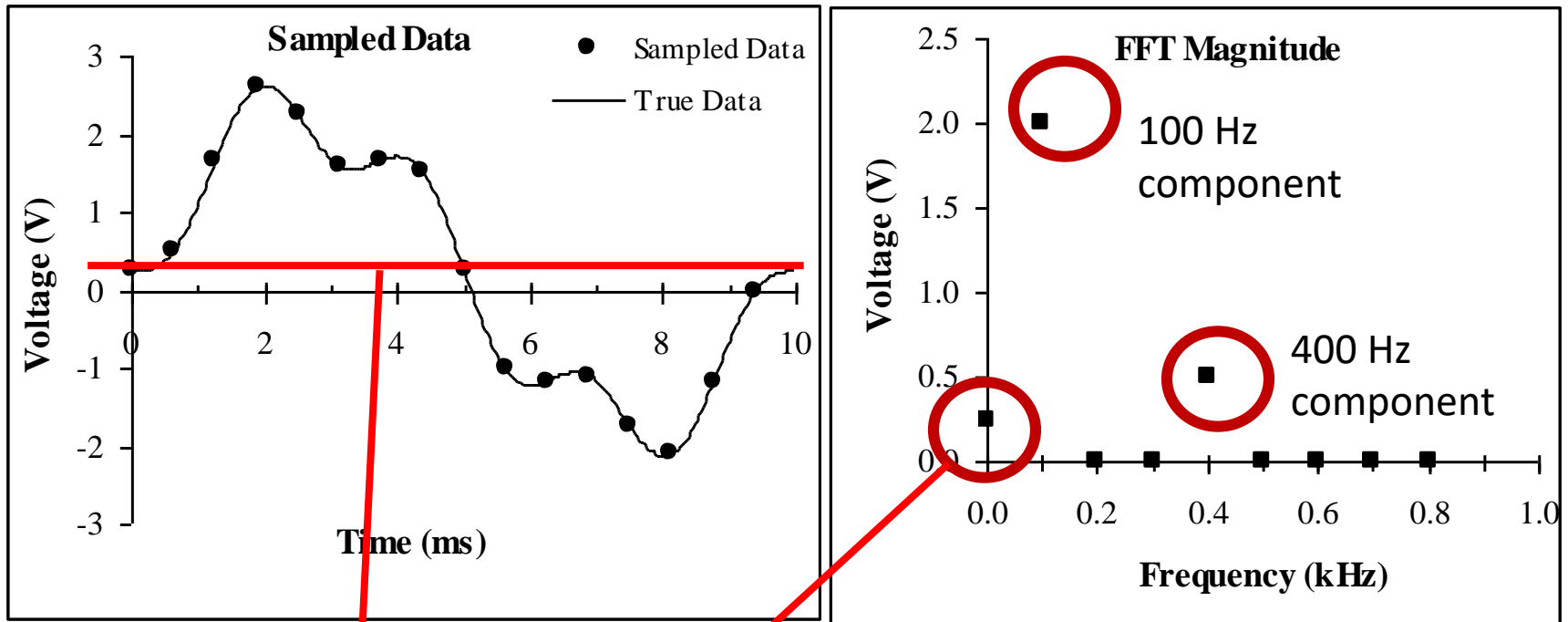
- 1500Hz signal sampled at 1600Hz

This is aliasing



<http://www.ece.utexas.edu/~valvano/EE345L/Labs/Fall2011/FFT16.xls>

- A signal with DC, 100Hz and 400Hz sampled at 1600Hz = 0.625ms interval



<http://www.ece.utexas.edu/~valvano/EE345L/Labs/Fall2011/FFT16.xls>

DC (zero freq component)

0 Hz+ 100Hz+400Hz; obey superposition

Theories for both
ADC/DAC



DAC Hardware
implementation

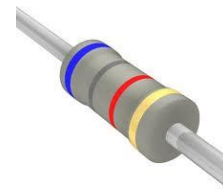
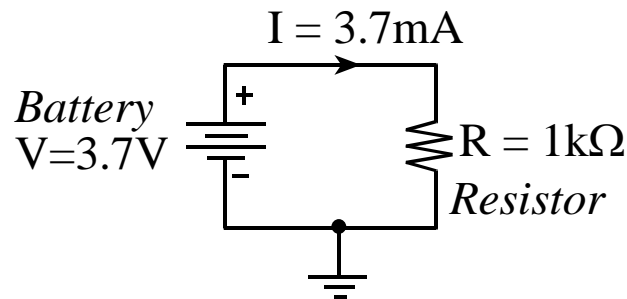
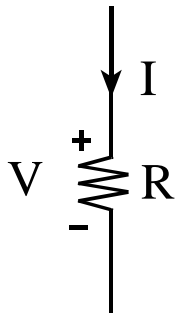
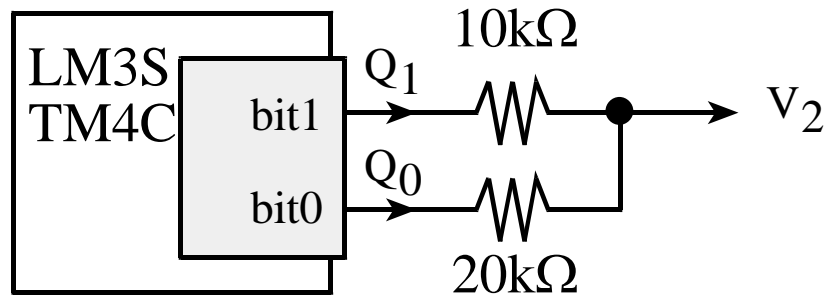
Types of DAC

- Binary Weighted DAC
 - One resistor for each bit of output
- R-2R Ladder DAC
 - Binary weighted cascading ladder
 - Improved precision by selecting resistors of equal value
- Thermometer coded DAC
 - Resistor for each possible output value controlled by a switch (resistor string)
- Segmented DAC
 - Thermometer for most significant bits and binary weighted for least significant bits
- Hybrid DAC
 - Any combination of the techniques

Binary Weighted DAC

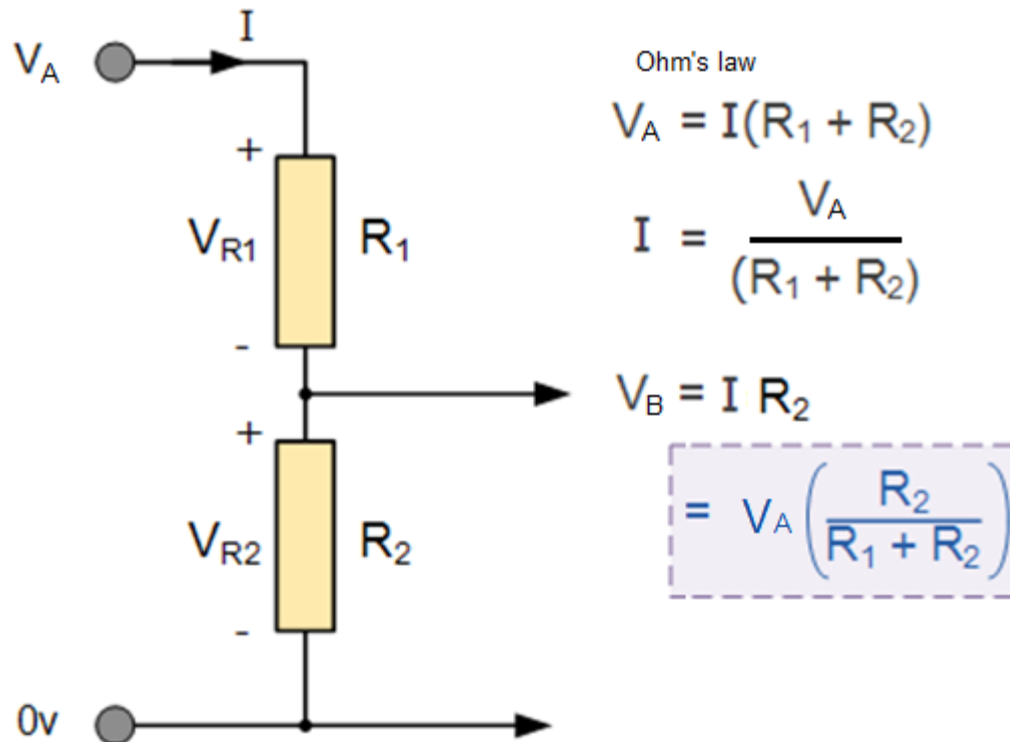
Binary Weighted DAC

- Binary Weighted DAC
 - One resistor for each bit of output
 - Resistor values in powers of 2

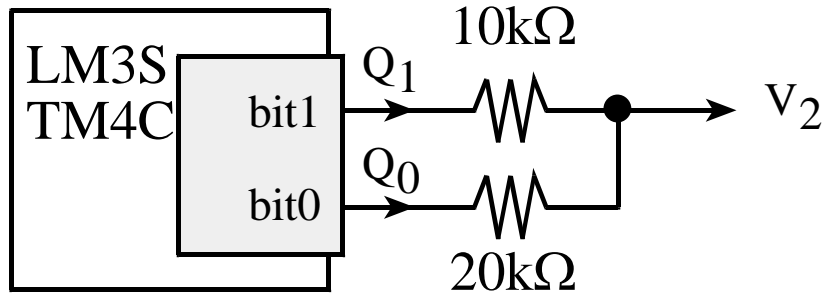


Voltage divider rule

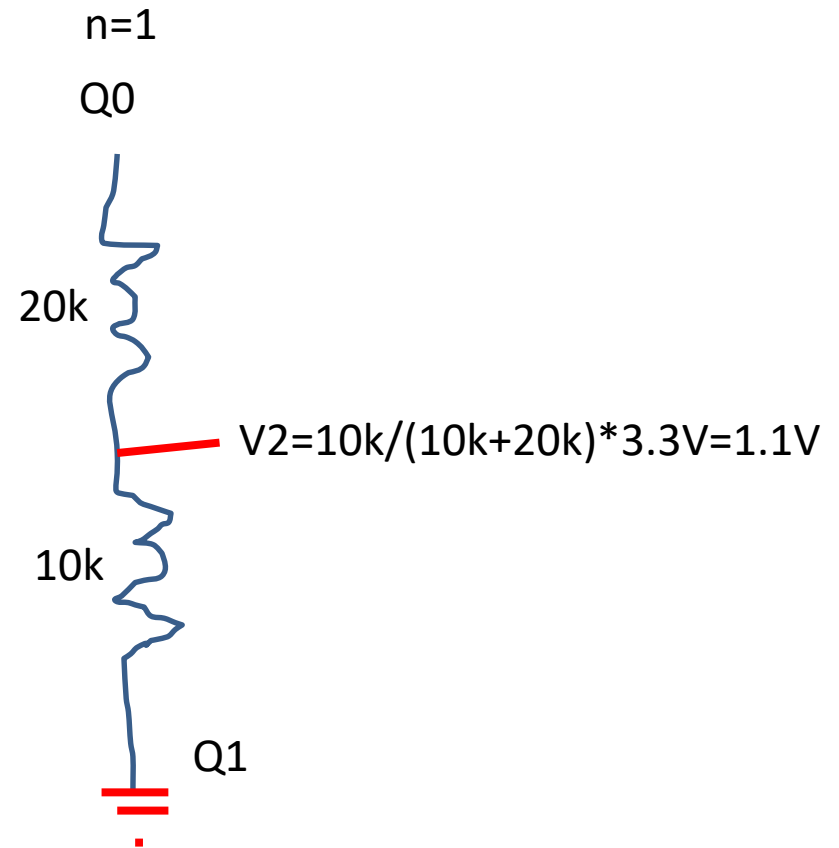
- The **voltage divider rule** in node methods (KCL, KVL)



1 = 3.3 V
0 = 0V



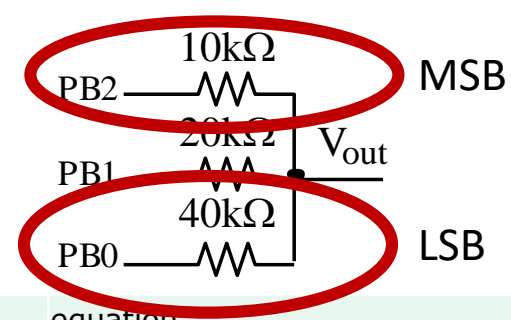
n	Q1	Q0	V2
0	0	0	0
1	0	1	1.1
2	1	0	?
3	1	1	3.3



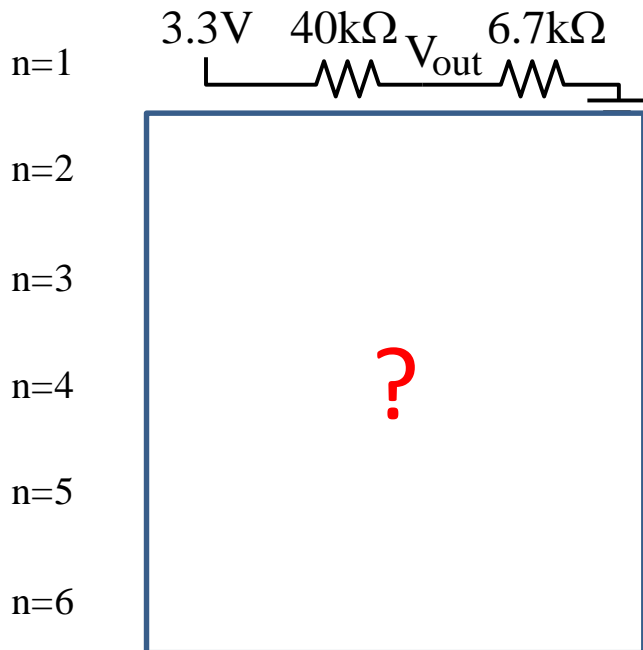
- Based on the simplest 2 bit case exercise, you try this later at home
- 3 bit Binary weighted DAC analysis

R2	10	kΩ
R1	20	kΩ
R0	40	kΩ

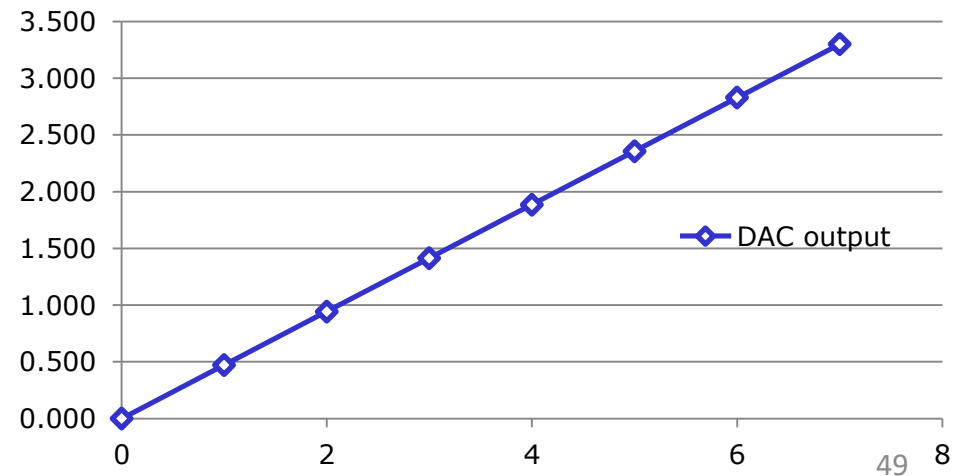
3 bit DAC



n	PB2	PB1	PB0	kohm	equation	Vout (V)
0	0	0	0			0.000
1	0	0	3.3	R2 R1	$3.3 \cdot (R1 R2) / (R0 + R1 R2)$?
2	0	3.3	0	R2 R0	$3.3 \cdot (R2 R0) / (R1 + R2 R0)$	
3	0	3.3	3.3	R1 R0	$3.3 \cdot R2 / (R2 + R1 R0)$	
4	3.3	0	0	R1 R0	$3.3 \cdot (R1 R0) / (R2 + R1 R0)$	
5	3.3	0	3.3	R2 R0	$3.3 \cdot R1 / (R1 + R2 R0)$	
6	3.3	3.3	0	R2 R1	$3.3 \cdot R0 / (R0 + R2 R1)$	
7	3.3	3.3	3.3			3.300

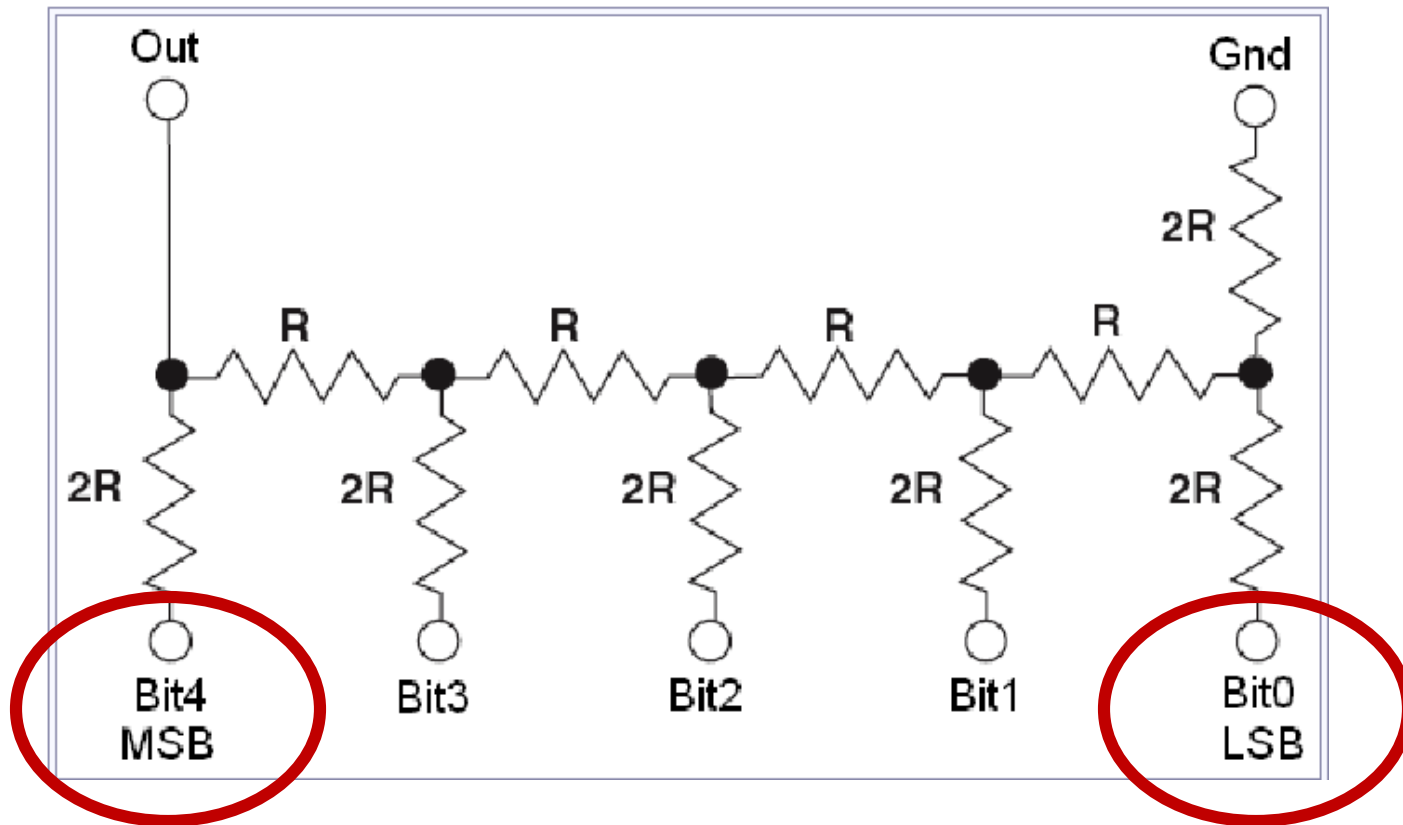


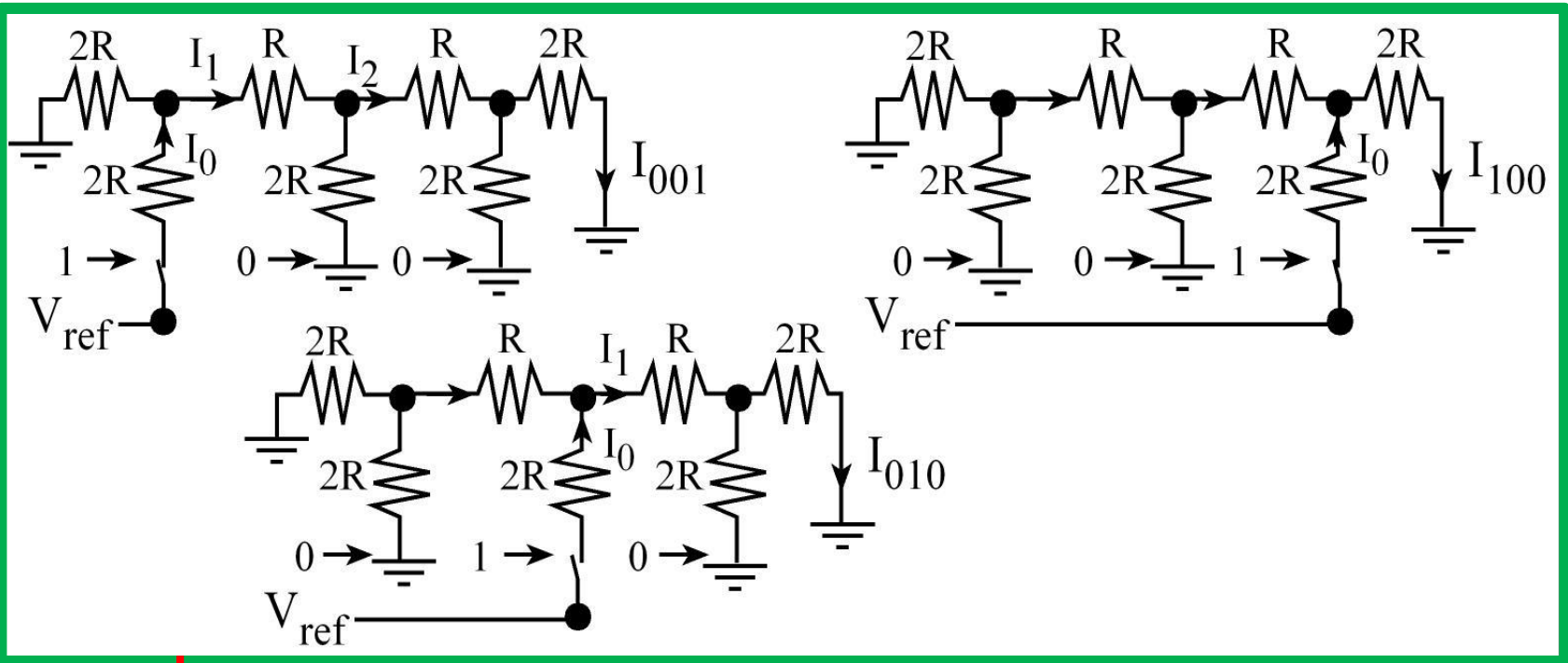
DAC output



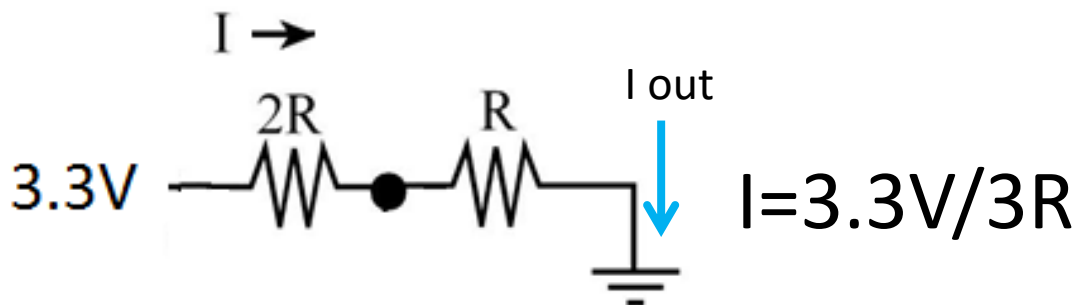
R-2R Ladder DAC

- R-2R Ladder DAC
 - Binary weighted cascading ladder
 - Improved precision owing to ability to select resistors of equal value



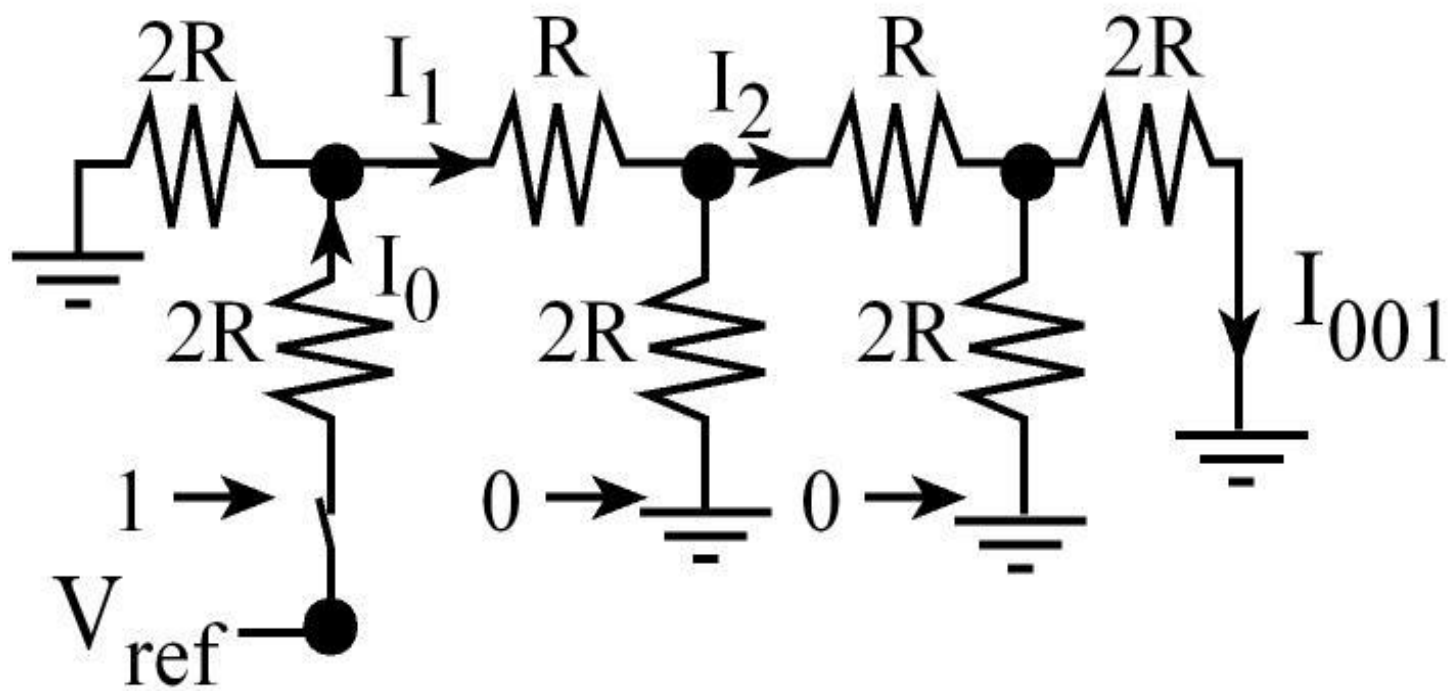


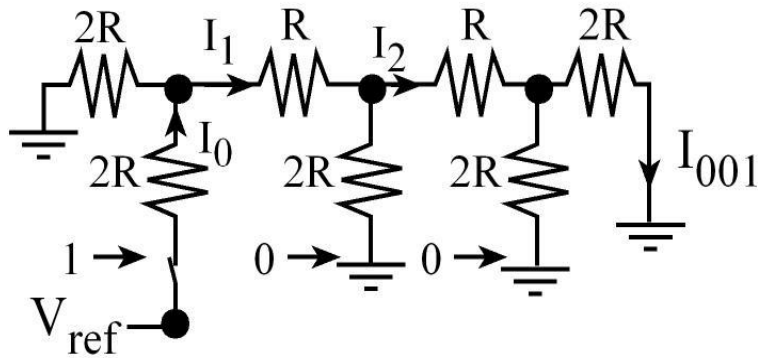
Final reduced form



If $R = 11k$ then
 $I = 0.1mA$

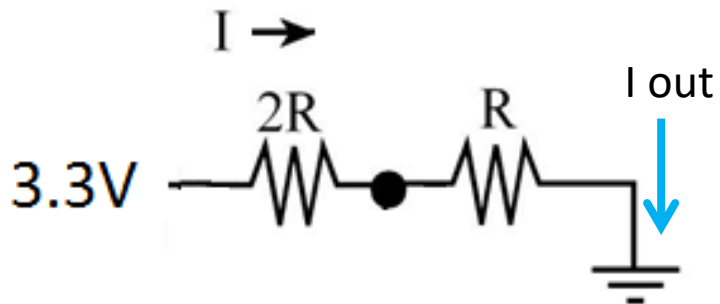
Compute I_{out}





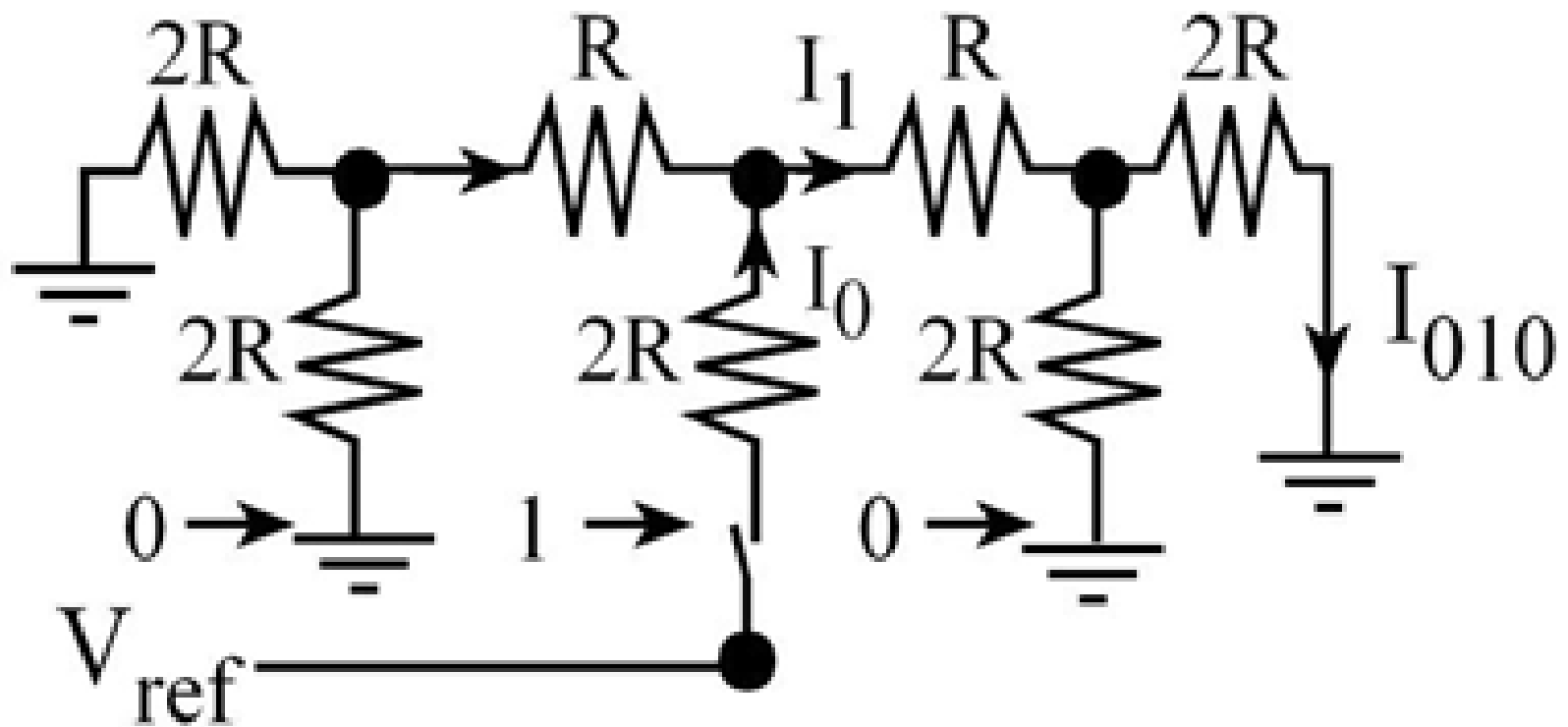
$V_{ref}=3.3V$

Final reduced form

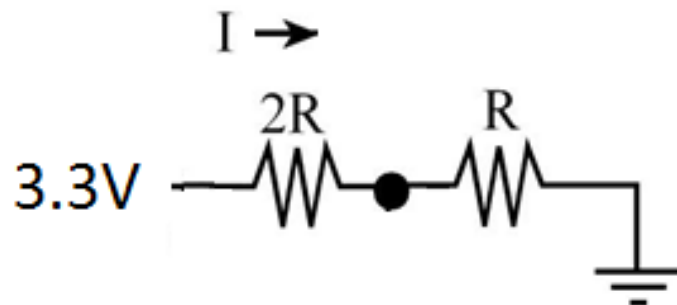


$$I = 3.3V / (2R + R)$$

If $R=11k$ then
 $I = 0.1mA$



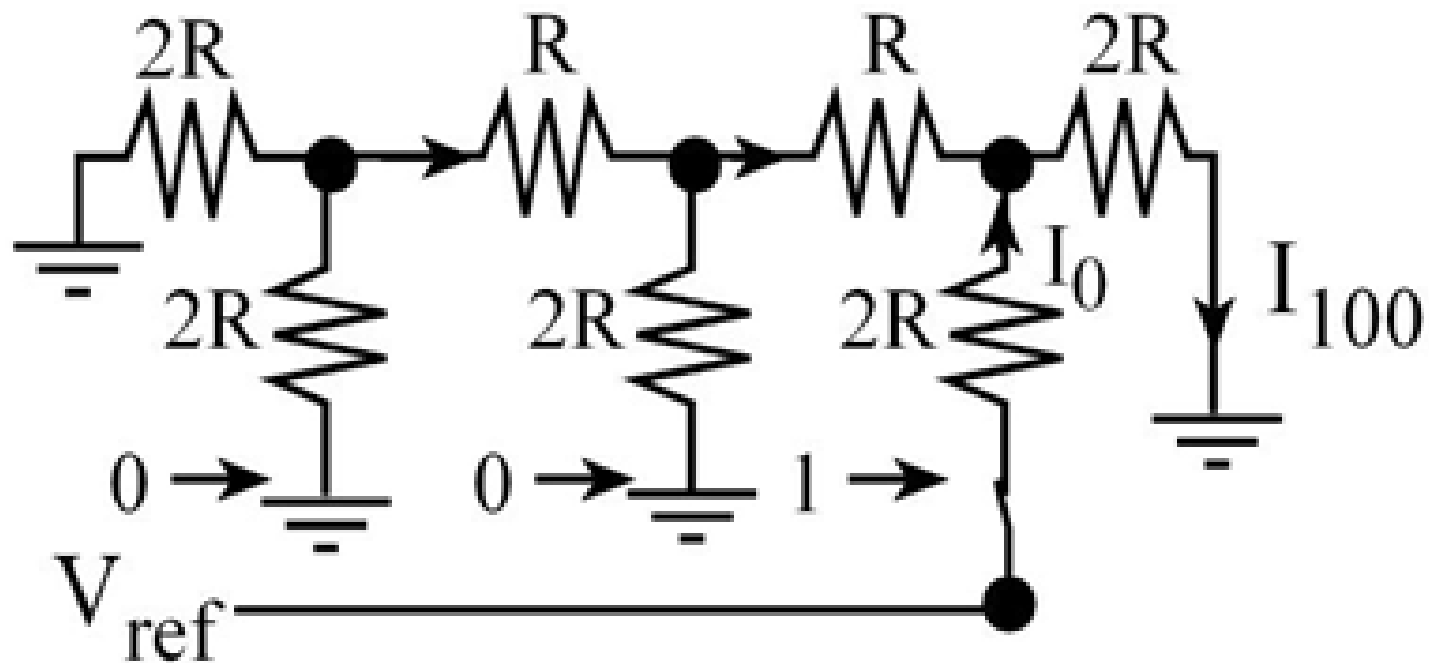
Final reduced form



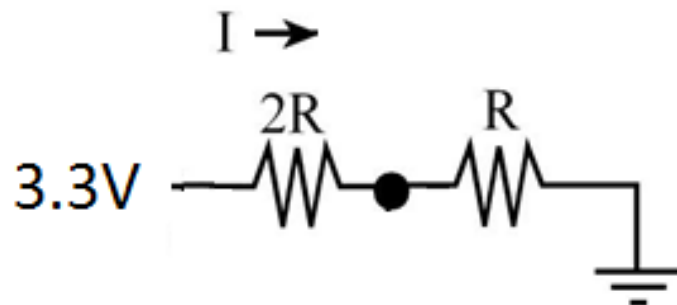
$$V_{ref} = 3.3V$$

$$I = 3.3V / (2R + R)$$

If $R = 11k$ then
 $I = 0.1mA$



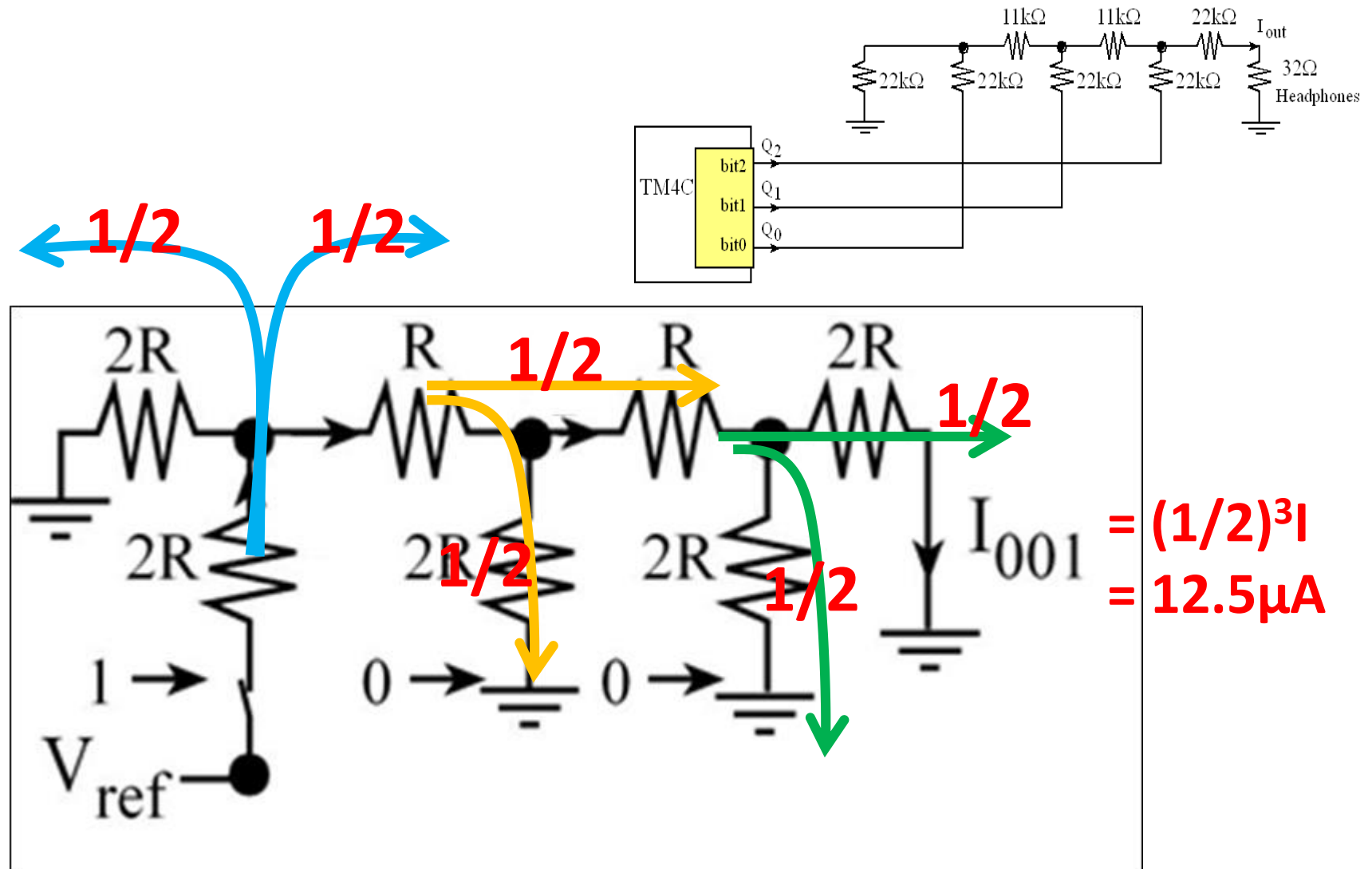
Final reduced form



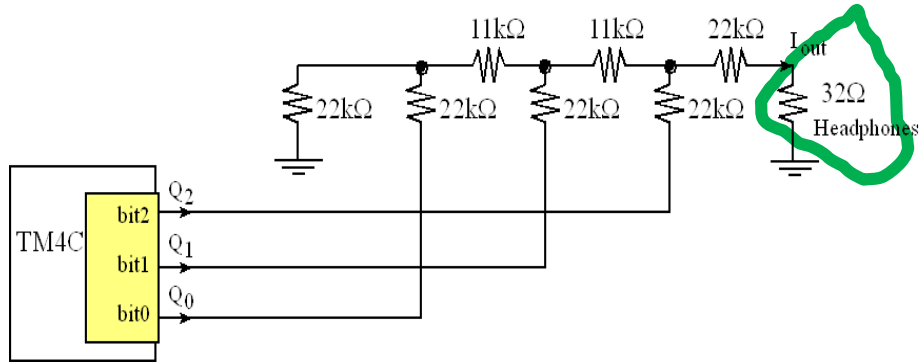
$$V_{ref} = 3.3V$$

$$I = 3.3V / (2R + R)$$

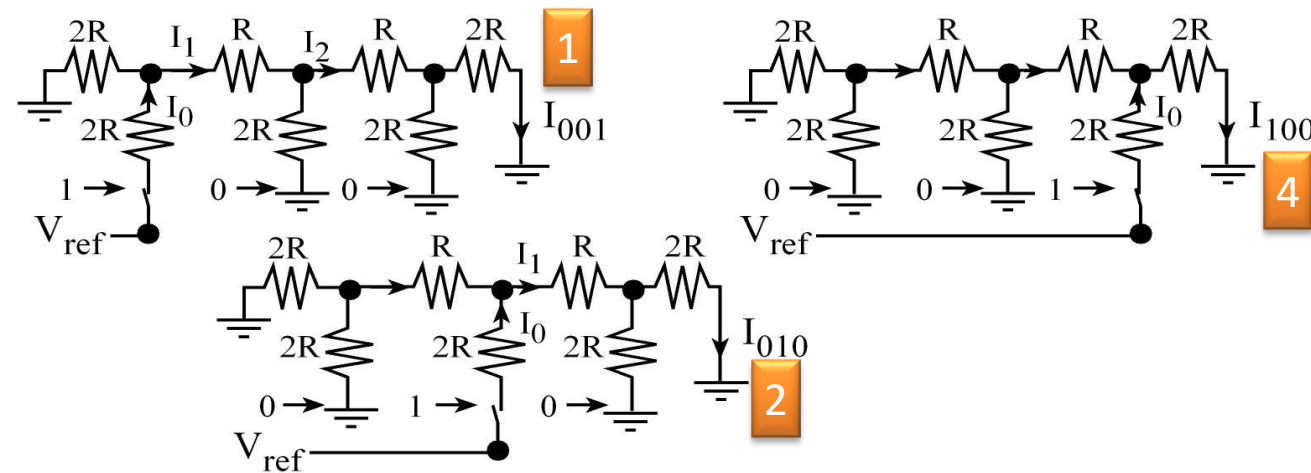
If $R = 11k$ then
 $I = 0.1mA$



3 bit case



N	Q_2	Q_1	Q_0	$I_{out} (\mu A)$
0	0	0	0	0.0
1	0	0	3.3V	12.5
2	0	3.3V	0	?
3	0	3.3V	3.3V	
4	3.3V	0	0	?
5	3.3V	0	3.3V	
6	3.3V	3.3V	0	
7	3.3V	3.3V	3.3V	?




A Slide hided

	Binary Weighted	R-2R
Pros	Easily understood	Only 2 resistor values Easier implementation Easier to manufacture Faster response time
Cons	Limited to ~ 8 bits Large # of resistors Susceptible to noise Expensive Greater Error	More confusing analysis

Practical session

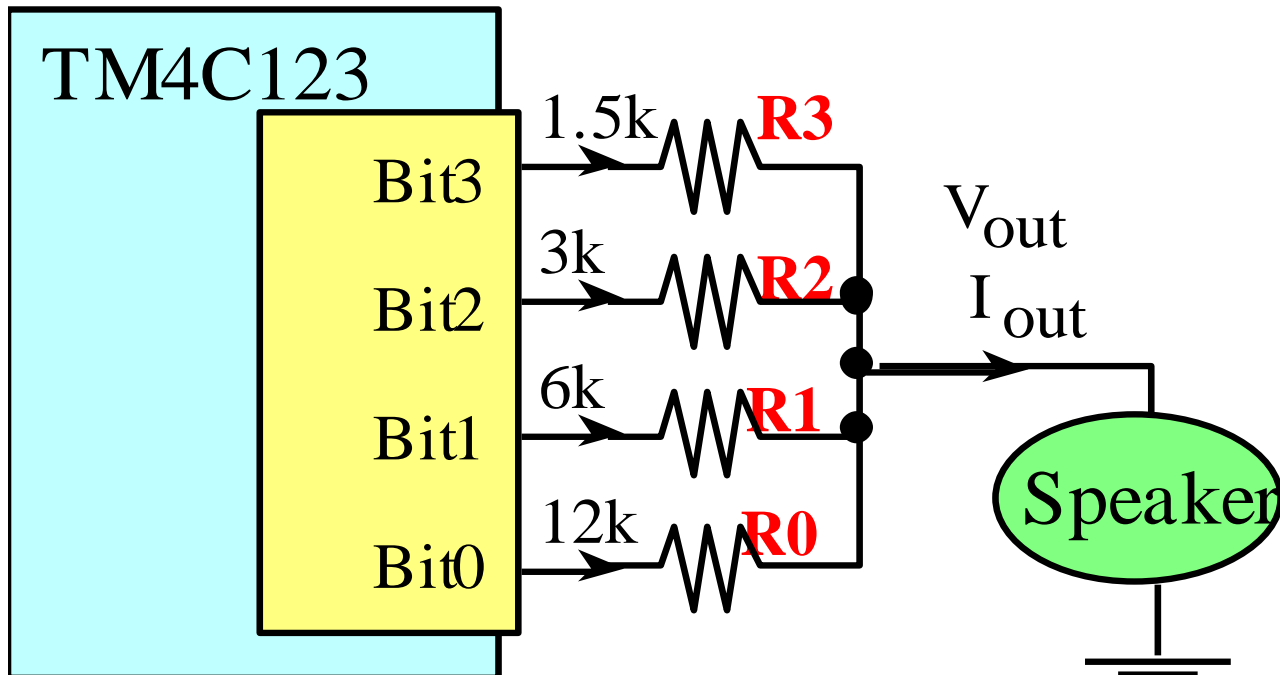
- Having this knowledge, you should be able to generate a sine wave

`SineWave[16] = {4,5,6,7,7,7,6,5,4,3,2,1,1,1,2,3};`  

- We will implement 4 bit binary weighed DAC with 4-key piano
 - Do not use port E as your input piano keys
 - Do not use Port B as your 4-bit DAC
 - If you use port E and port B, the maximum score you can get is 80/100.

Resistor Network for 4-bit DAC

- DAC hardware
 - Employ least significant four bits of a GPIO port
 - Arrange resistor network in 1, 2, 4, 8 sequence
 - Each port bit can assume digital levels of 0 and 3.3 V
 - Ports are *current limited* – max 8 mA



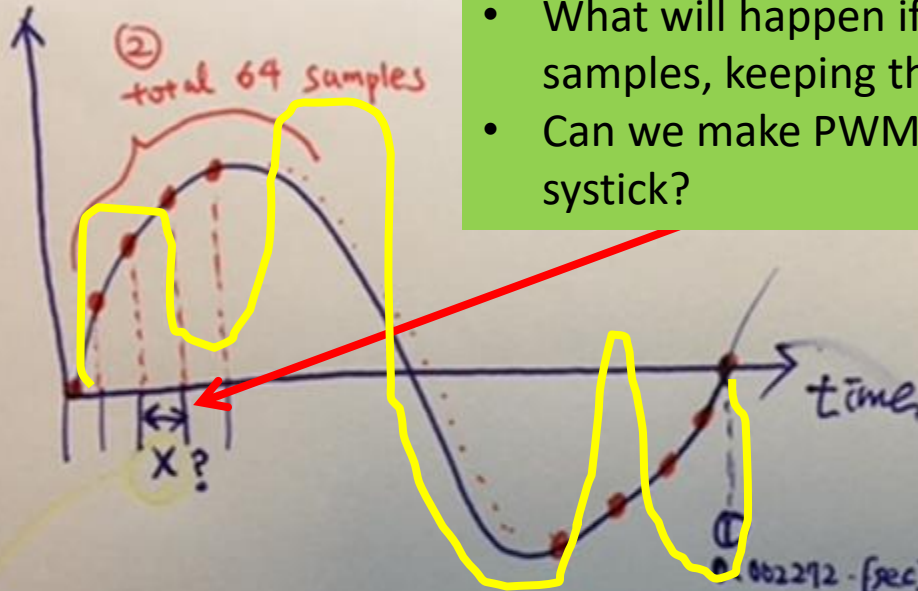
OK, I can make a sine wave using parallel digital outputs, but

How do we control frequencies?

➔ We will use SysTick interrupt!

A simple concept is described next page, more will come next week.

[Tone generation.]



- What will happen I change my wave shape, keeping the period?
- What will happen if I reduce the # of samples, keeping the period and precision?
- Can we make PWM signal by manipulating systick?

/ 352 μ s



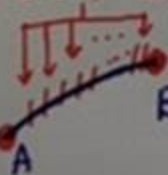
③ My bus clock speed :

$$80 \text{ MHz} = 80,000,000 \text{ cycle /sec}$$

$$\Rightarrow 12.5 \text{ nano Sec.} = 12.5 \times 10^{-9}$$

④ X ; Sampling Interval

Bus cycles



$$\frac{80 \text{ MHz}}{440 \text{ Hz}} = \frac{2.272 \times 10^{-3}}{12.5 \times 10^{-9}} \approx 2841 \text{ Counts}$$

$\Leftrightarrow 2841 \text{ bus cycles from A to B}$

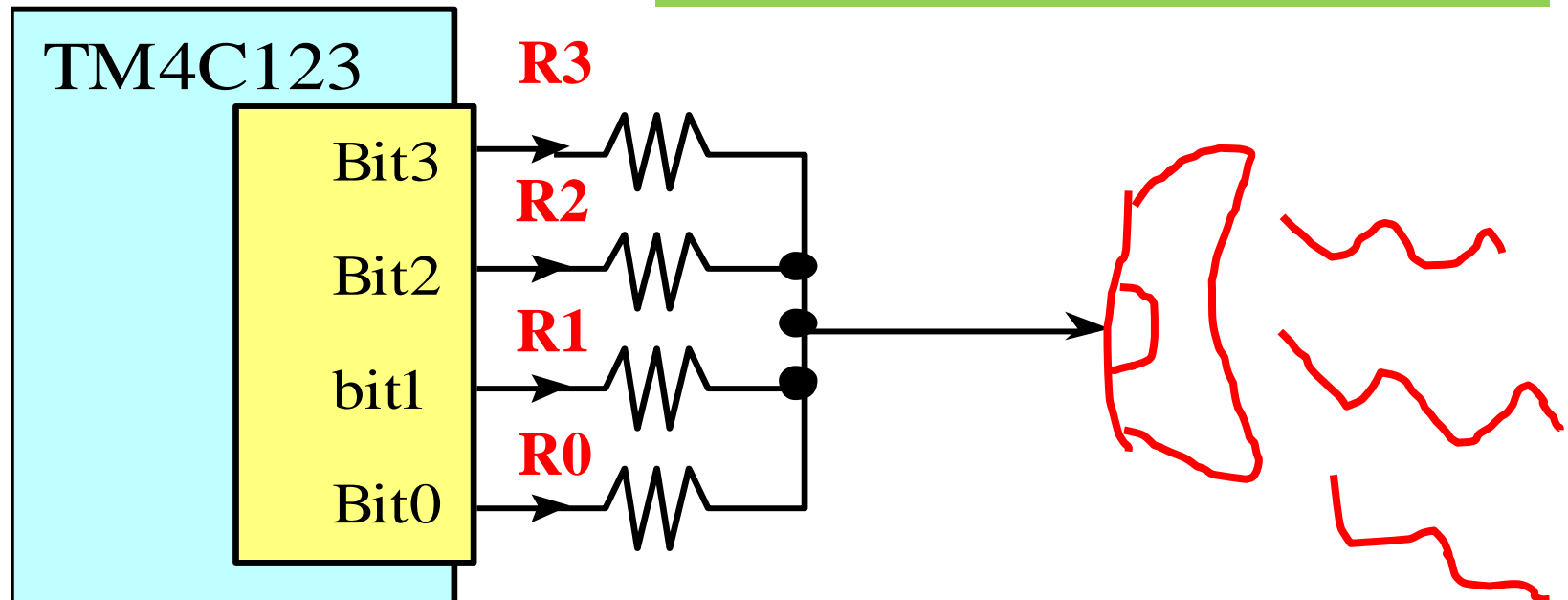
$$\rightarrow \frac{1}{2841} = 352 \mu\text{sec}$$

about 352 μ sec from A to B

Systick says: Please interrupt me every 2841 cycles
Bus clock: OK, I will

Interesting questions!

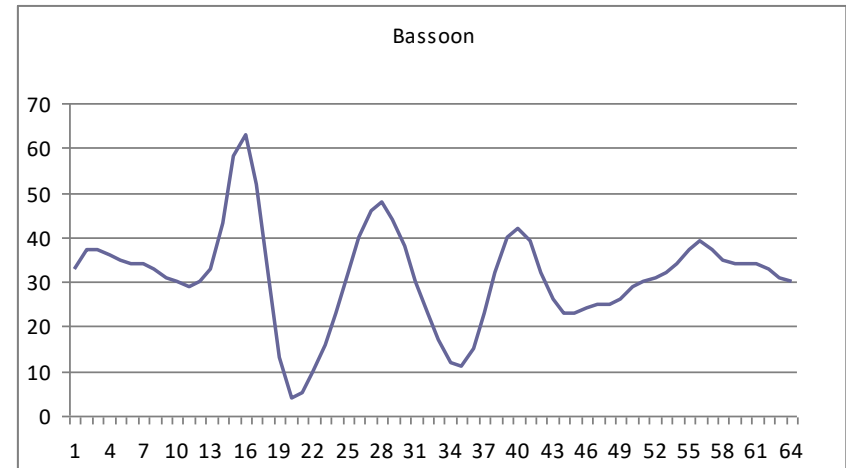
- What will happen if I remove one of the resistors?
- Why we do not set AMSEL register for this lab?
- Where is the place for converting digital to analog?
- Is the Bit0 always LSB?
- Why do we increase resistances (R0-R3) in twice manner?



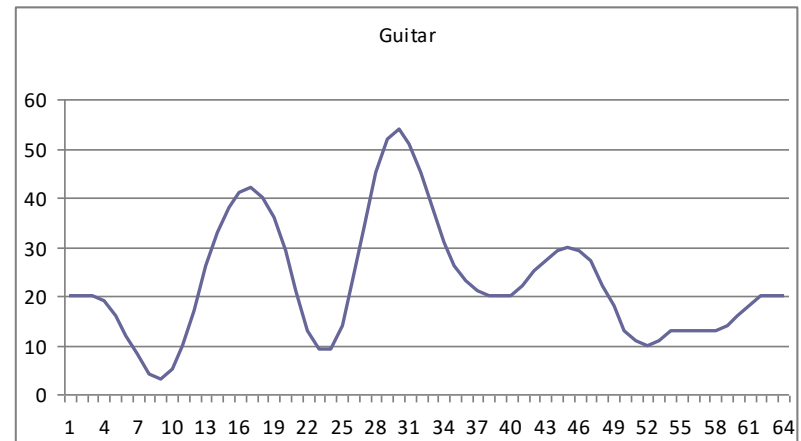
what would happen if you keep
the period, change the shape of
your wave?

Other Instruments (diff. shape)

```
// 6-bit 64-element bassoon wave
const uint8_t Bassoon[64] = {
    33,37,37,36,35,34,34,33,31,30,29,
    30,33,43,58,63,52,31,13,4,5,10,16,
    23,32,40,46,48,44,38,30,23,17,12,11,
    15,23,32,40,42,39,32,26,23,23,24,25,
    25,26,29,30,31,32,34,37,39,37,35,34,
    34,34,33,31,30};
```



```
// 6-bit 64-element guitar wave
const uint8_t Guitar[64] = {
    20,20,20,19,16,12,8,4,3,5,10,17,
    26,33,38,41,42,40,36,29,21,13,9,
    9,14,23,34,45,52,54,51,45,38,31,
    26,23,21,20,20,20,22,25,27,29,
    30,29,27,22,18,13,11,10,11,13,13,
    13,13,13,14,16,18,20,20,20};
```



Musical Notes

Note	f (Hz)	T (milli s)
C	523	1.91
B	494	2.02
B ^b	466	2.15
A	440	2.27
A ^b	415	2.41
G	392	2.55
G ^b	370	2.70
F	349	2.87
E	330	3.03
E ^b	311	3.22
D	294	3.40
D ^b	277	3.61
C	262	3.82

i.e.)

Two octave up: $\text{Freq} * (2)^2$

One octave up:
double up freq

One octave down:
half of freq down

i.e.)

Two octave down: $\text{Freq} * (1/2)^2$

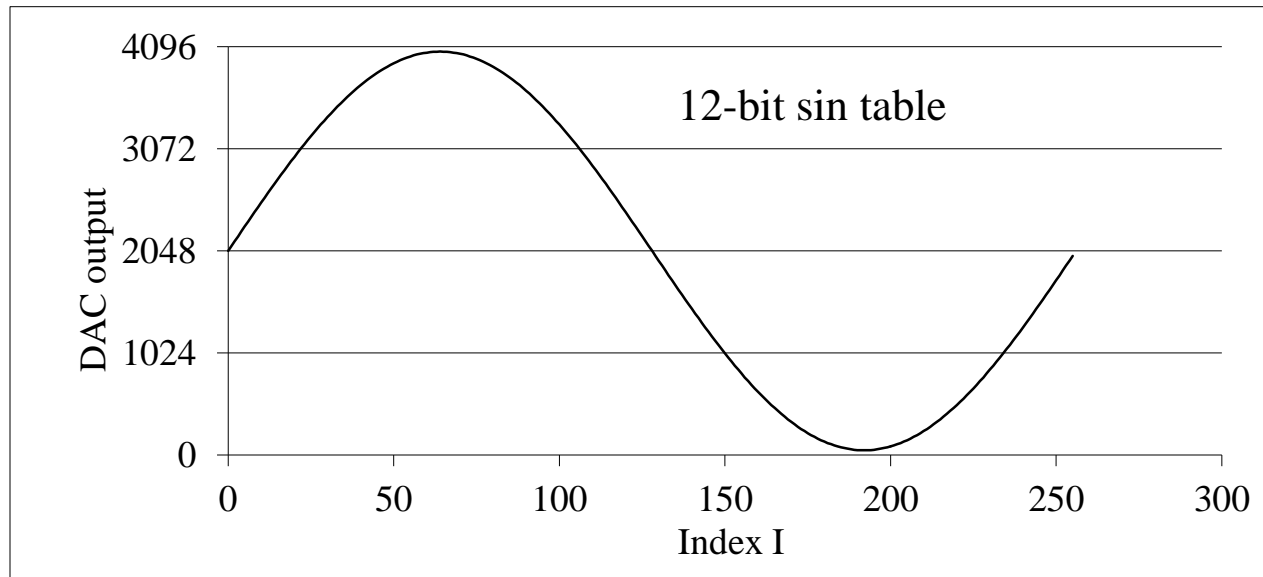
Tone Generation

- Picturizing the followings and computing sampling intervals
 - 440 Hz tone
 - 64 samples per a sinusoid period
 - 80 MHz clock (12.5 ns interval)
 - $(80\text{Mcounts/s}/440\text{Hz})/64 \text{ points} = 2841 \text{ counts} = 355.1 \mu\text{s/point}$
- If 256 points/period ?
 - 89 $\mu\text{s/point}$ (you try)

Tone Generation

```
// 11-bit outputs 64-element sine wave
const uint16_t Wave[64] = {
    1024,1122,1219,1314,1407,1495,1580,1658,1731,1797,1855,
    1906,1948,1981,2005,2019,2024,2019,2005,1981,1948,1906,
    1855,1797,1731,1658,1580,1495,1407,1314,1219,1122,1024,
    926,829,734,641,553,468,390,317,251,193,142,
    100,67,43,29,24,29,43,67,100,142,193,
    251,317,390,468,553,641,734,829,926
};
```

- Why 11 bits?
- What is the max value of 11 bit binary?



Tone generated from 256 discrete 12-bit outputs

Reading for two weeks

Vol.1	Vol.2
Ch.8 (8.1)	Ch.6 (6.4)
Ch.10 (10.1, 10.2, 10.3, 10.4)	Ch.8 (8.4, 8.5) Ch.10 (10.2)