DIGITAL TRANSMISSION

Analog-to-Digital Conversion: **Pulse Code Modulation (PCM)**
Delta Modulation (DM)

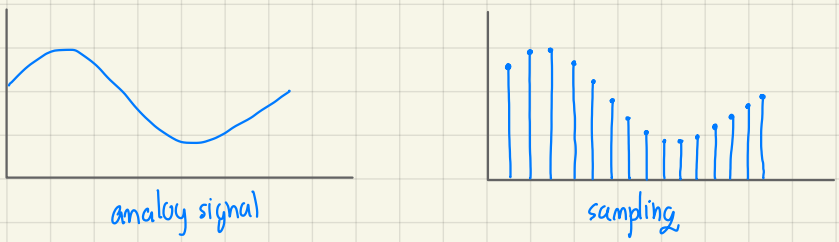
Analog to Digital Conversion (Basic)

↳ Something create analog signal, which are treated as analog data.
  ↳ To transmit analog data over digital signals
    ↳ We need "analog to digital conversion"
    ↳ use "Pulse Code Modulation (PCM)" to convert analog wave into digital signal

PCM

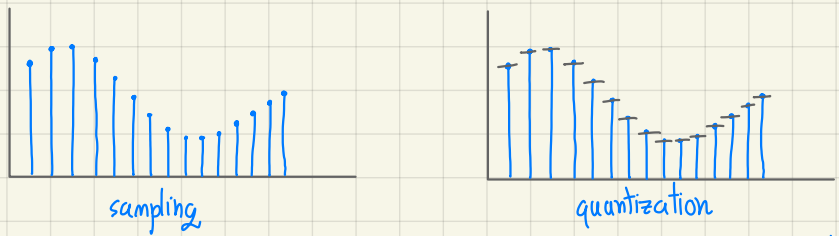
↳ most commonly used method to convert analog data into digital form.
  ↳ step >> sampling >> quantization >> encoding
    ↳ **sampling**

ผลที่ได้จากการทำ PAM
↳ pulse amplitude modulation



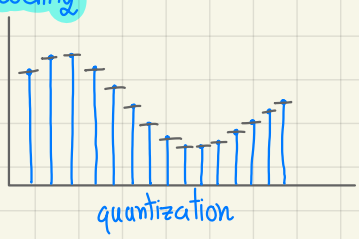
↳ The analog signal is sampled every T interval. Most important factor is sampling rate. (Hz)
  ↳ Sampling rate must be at least two times of high f of the signal

↳ **Quantization**



↳ sampling yield discrete form of continuous analog signal. Every discrete pattern shows the amplitude of the analog signal
↳ The quantization is done between the $max A_n$ and $min A_n$
↳ Quantization is approximation of the instantaneous analog value

↳ **Encoding**
  ↳



each approximate value is converted into binary
→ 10000101

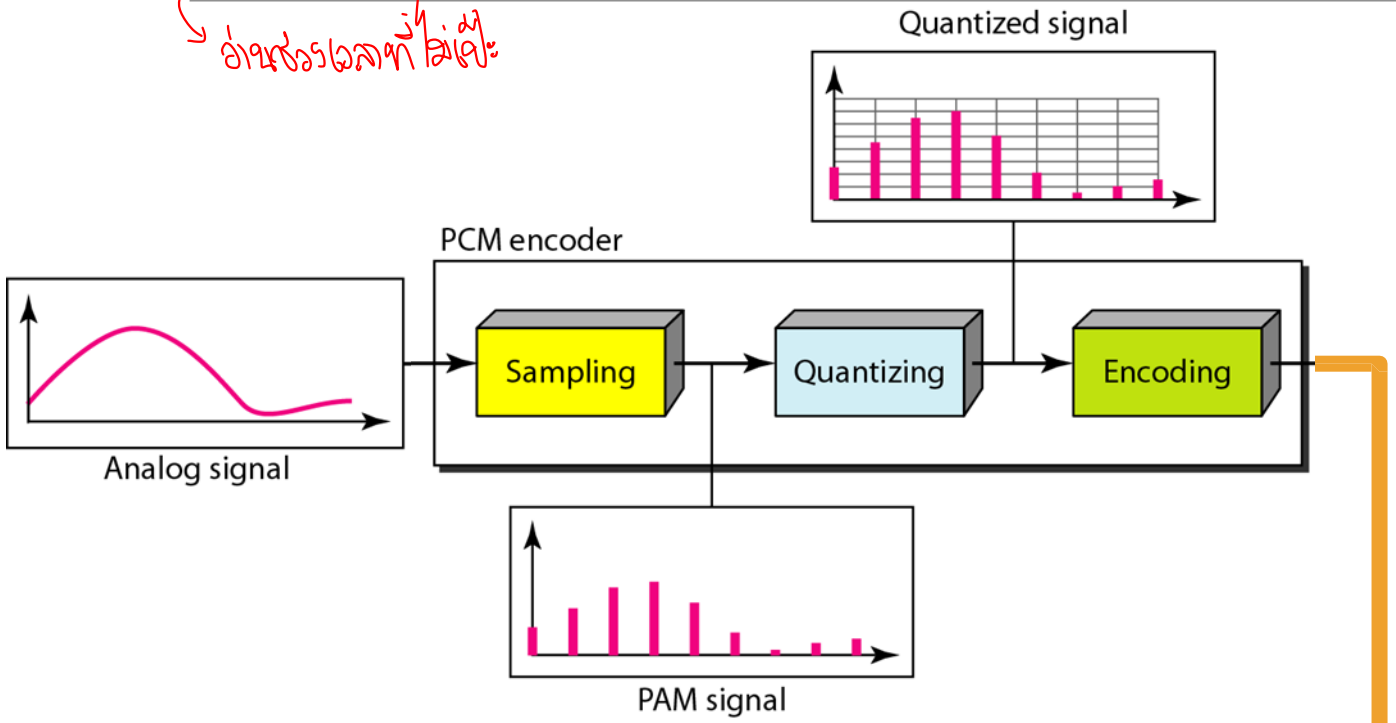
sampling : เทคนิคการสุ่มสัญญาณ

sampling rate : อัตราการสุ่มสัญญาณ (Hz)

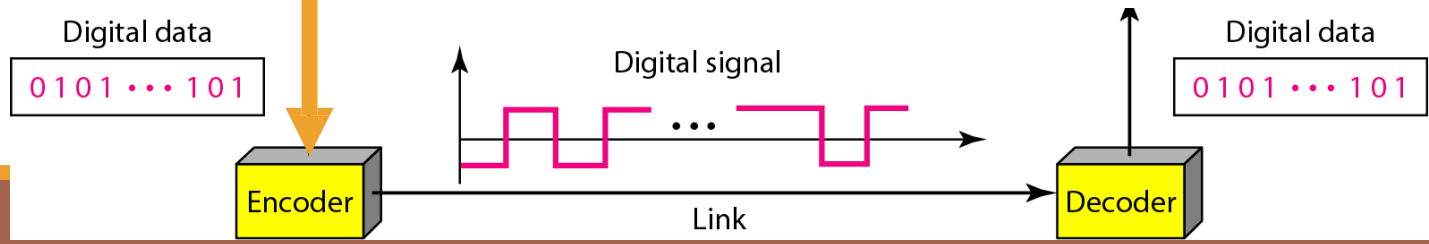
Analog Data on Digital Channel

จุดมุ่งหมายคือ ใช้ hardware ที่ไร้ delay

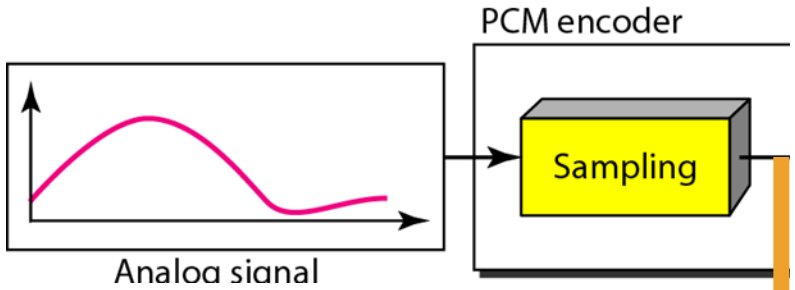
↳ อ่านช่วงเวลาที่ไม่ดี

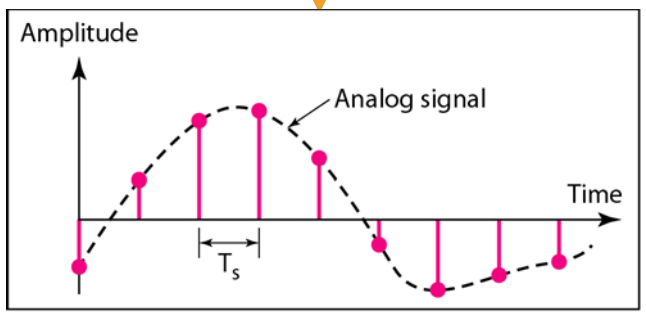


PCM
Pulse Code Modulation

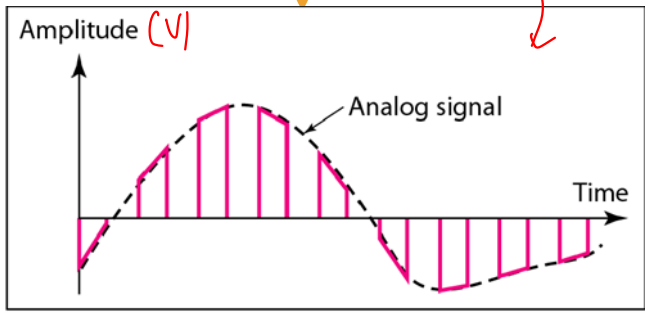


PCM Sampling

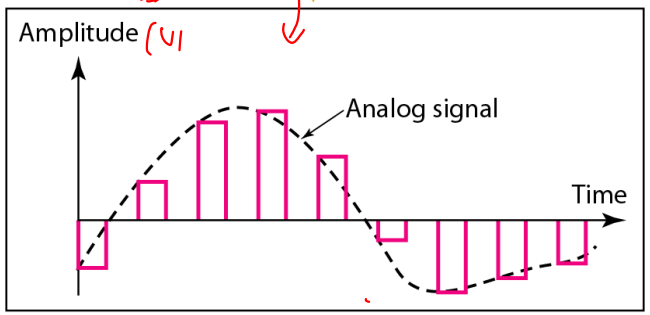




a. Ideal sampling



b. Natural sampling



c. Flat-top sampling

PCM Sampling Patterns

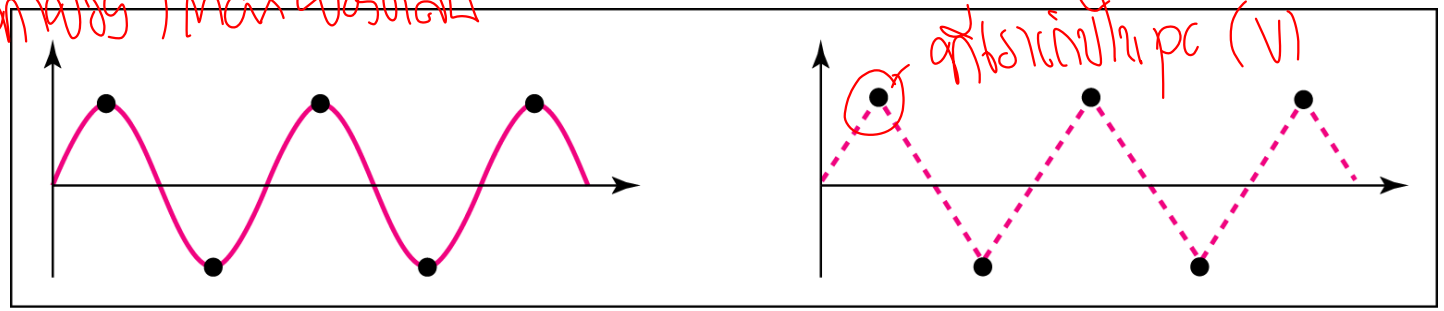
Different Sampling Cases

Sampling @ mininmum Nyquist rate

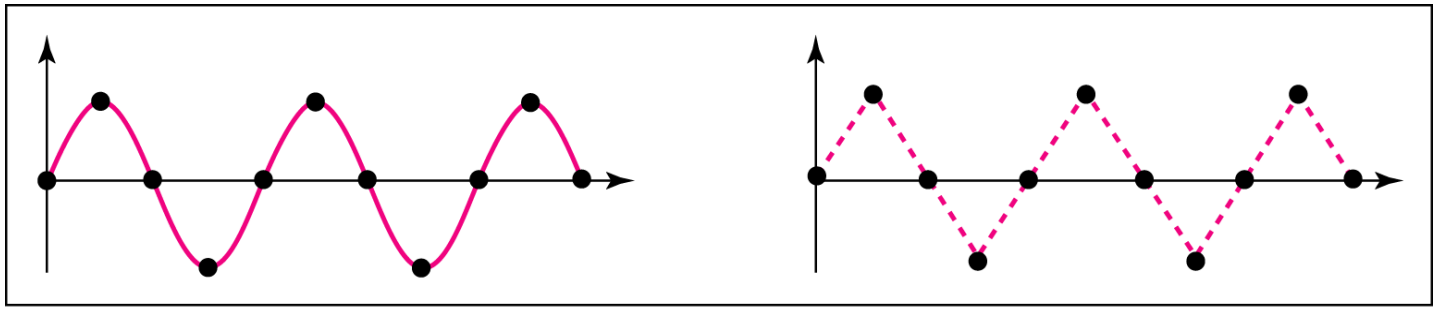
Oversampling rate

Undersampling rate

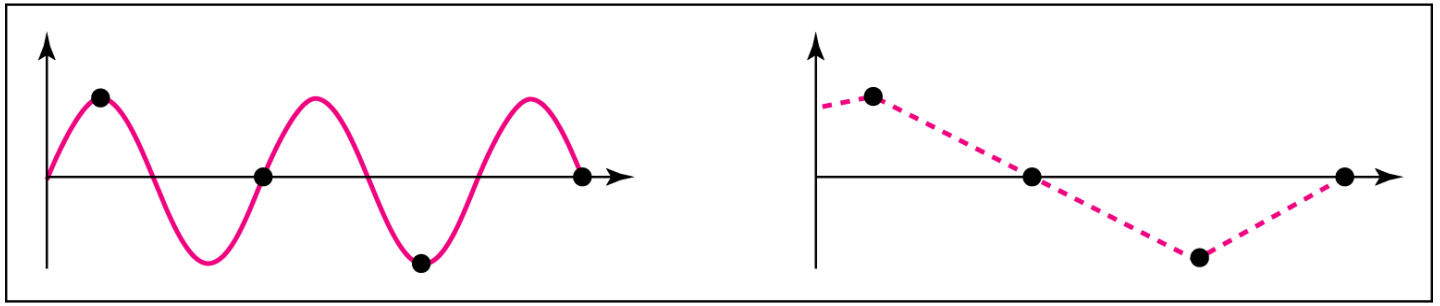
Nyquist Sampling Rate >= 2 x fmax



a. Nyquist rate sampling: $f_s = 2\ f$



b. Oversampling: $f_s = 4\ f$



c. Undersampling: $f_s = f$

Frequency Range Example

Voice

Musical Instrument

แบ่งตามชนิดของเสียง ความถี่จะถูกแบ่งออกดังนี้

เสียงร้อง

- Bass (เสียงต่ำของผู้ชาย) - ความถี่หลัก 70 Hz - 380 Hz / ความถี่รอง (หางเสียง) 380 Hz - 10,000 Hz
- Baritone (เสียงระหว่างต่ำกับสูงของผู้ชาย) - ความถี่หลัก 90 Hz - 400 Hz / ความถี่รอง 400 Hz - 10,000 Hz
- Tenor (เสียงสูงของผู้ชาย) - ความถี่หลัก 130 Hz - 500 Hz / ความถี่รอง 500 Hz - 10,000 Hz
- Alto (เสียงต่ำของผู้หญิง) - ความถี่หลัก 160 Hz - 950 Hz / ความถี่รอง 950 Hz -10,000 Hz
- Soprano (เสียงสูงของผู้หญิง) - ความถี่หลัก 210 Hz - 1,200 Hz / ความถี่รอง 1,200 Hz - 10,000 Hz

เสียงเครื่องดนตรีชนิดดีด,สี,ตี

- Bass viola - ความถี่หลัก 30 Hz - 210 Hz / ความถี่รอง 210 Hz - 15,000 Hz
- Cello - ความถี่หลัก 50 Hz - 650 Hz / ความถี่รอง 650 Hz - 15,000 Hz
- Viola - ความถี่หลัก 130 Hz - 1,200 Hz / ความถี่รอง 1,200 Hz -15,000 Hz
- Violin - ความถี่หลัก 180 Hz - 4,000 Hz / ความถี่รอง 4,000 Hz - 15,000 Hz

เสียงเครื่องดนตรีชนิดเป่า

- Bass tuba - ความถี่หลัก 40 Hz -380 Hz / ความถี่รอง 380 Hz - 16,000 Hz
- Bassoon - ความถี่หลัก 45 Hz - 500 Hz / ความถี่รอง 500 Hz - 10,000 Hz
- Bass clarinet - ความถี่หลัก 80 Hz - 500 Hz / ความถี่รอง 500 Hz - 10,000 Hz
- French horn - ความถี่หลัก 120 Hz - 800 Hz / ความถี่รอง 800 Hz - 16,000 Hz
- Trumpet - ความถี่หลัก 140 Hz - 850 Hz / ความถี่รอง 850 Hz - 9,000 Hz
- Clarinet - ความถี่หลัก 160 Hz - 1,600 Hz / ความถี่รอง 1,600 Hz - 16,000 Hz
- Oboe - ความถี่หลัก 240 Hz - 1,600 Hz / ความถี่รอง 1,600 Hz - 16,000 Hz
- Flute - ความถี่หลัก 240 Hz - 2,200 Hz / ความถี่รอง 2,200 Hz - 16,000 Hz
- Piccolo - ความถี่หลัก 500 Hz - 7,000 Hz / ความถี่รอง 7,000 Hz - 16000 Hz

เสียงเครื่องดนตรีชนิดเคาะ

- Piano - ความถี่หลัก 25 Hz - 7,000 Hz

Sampling Rate Example

Regular Human voice

- f_sampling = 8,000 Hz

FM Radio (รองรับเสียงดนตรี 15,000 Hz)

- f_sampling = 32,000 Hz

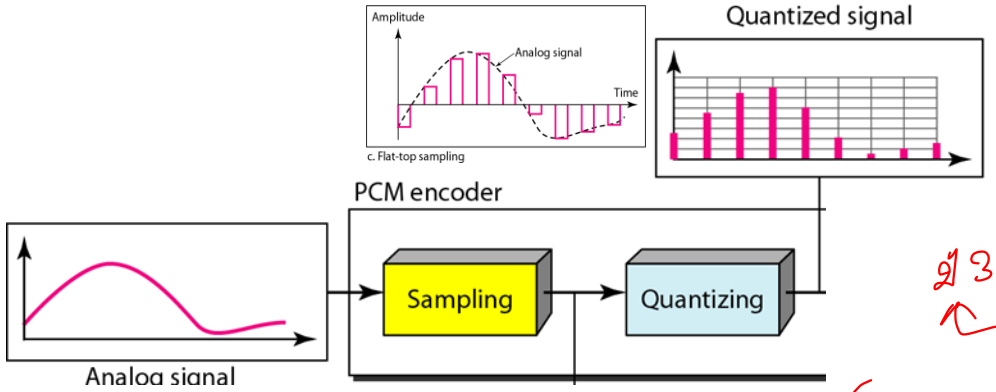
CD Quality (รองรับเสียงดนตรี 20,000 Hz)

- f_sampling = 44,100 Hz

High Quality Sound

- f_sampling = 96,000 Hz

Quantization



#Quantization bit

#Quantization Level

D = delta = level step = (Max-Min)/Qlevel

Qlevel partition:
[-3D, -2D, -D, 0, D, 2D, 3D]

Qlevel codeword:
[-3.5D, -2.5D, -1.5D, -0.5D,0.5D, 1.5D, 2.5D, 3.5D]

#Quantization Level

Qlevel codeword:
[-3.5D, -2.5D, -1.5D, 1.5D, 2.5D, 3.5D]

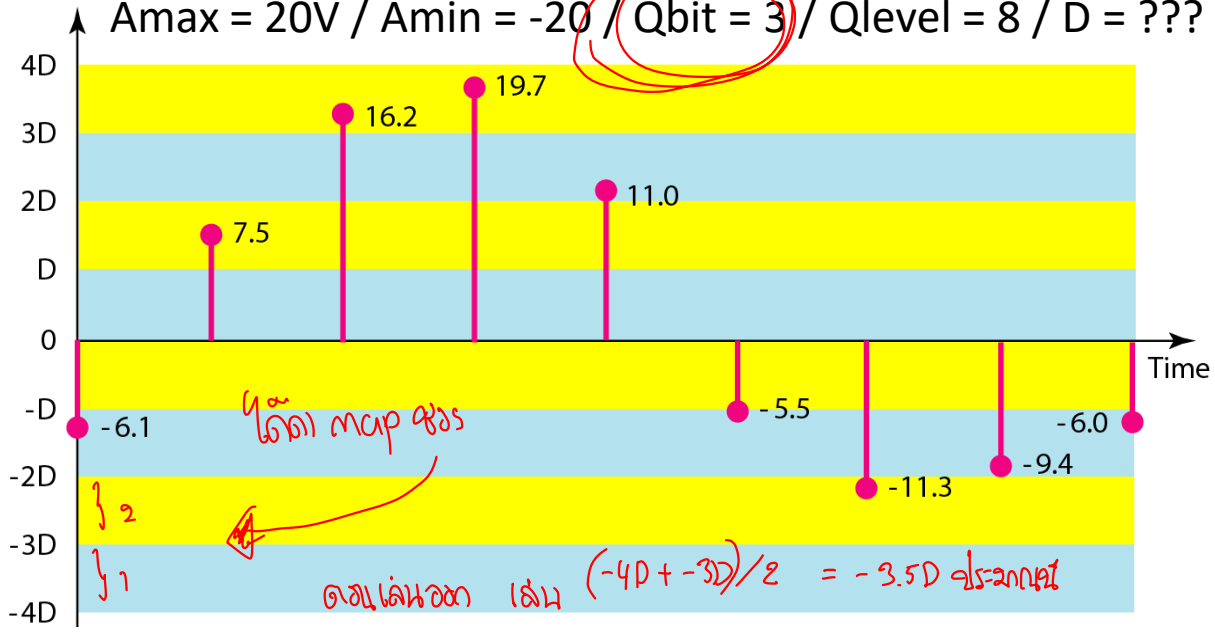
Quantized Encoder (Tx)

#Quantization bit

Quantized Decoder (Rx)

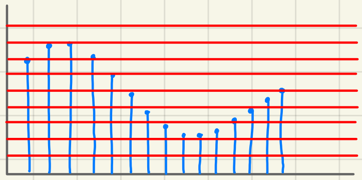
Quantization codes	Normalized amplitude
7	
6	
5	
4	
3	
2	
1	
0	

Amax = 20V / Amin = -20 / Qbit = 3 / Qlevel = 8 / D = ???



Normalized PAM values	-1.22	1.50	3.24	3.94	2.20	-1.10	-2.26	-1.88	-1.20
Normalized quantized values	-1.50	1.50	3.50	3.50	2.50	-1.50	-2.50	-1.50	-1.50
Normalized error	-0.38	0	+0.26	-0.44	+0.30	-0.40	-0.24	+0.38	-0.30
Quantization code	2	5	7	7	6	2	1	2	2
Encoded words	010	101	111	111	110	010	001	010	010

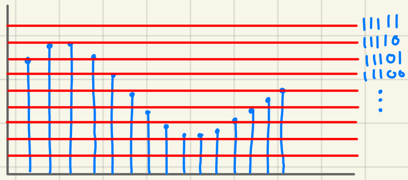
- Quantizing
  - ถ้า sampling คือการขีดเส้นแนวตั้ง quantize คือการขีดเส้นแนวนอน



sampling

- มารู้จักกับ bit depth กัน
  - จำนวน bit ในการ sample 1 ครั้ง
    - เช่น 8 bit, 16 bit, 24 bit
  - สามารถนำไปแบ่งเป็น level ต่างๆ ของการ sampling ได้
    - 3 บิท → $2^3 = 8$ level
      8 บิท → $2^8 = 256$ level

- เส้นแนวนอนต่างๆ ก็คือ level



sampling

- ขั้นตอนที่เราจะทำคือกำหนด Qbit (bit depth) → ยิ่ง Qbit มากเสียงยิ่งไม่เหลี่ยม
  - Qbit = 3 → Qlevel = 8
- ต่อไปทำการหาความกว้างของแต่ละ level
  - $D = \frac{V_{max} - V_{min}}{Q_{level}}$ ตัวอย่าง $\frac{20-(-20)}{8} = 5\ V$
- Q partition = $_{-40}[-3D, -2D, -D, 0, D, 2D, 3D]_{40}$ เราไม่เอาหัวท้าย

  = $_{min=-20}[-15, -10, -5, 0, 5, 10, 15]_{max=20}$ # ค่า voltage ที่ถูกแบ่ง
- Q codeword
  - การเอา Q partition มาจับคู่กัน ⌒ แล้วหาร 2

    Q codeword = [-3.5D, -2.5D, -1.5D, -0.5D, 0.5D, 1.5D, 2.5D, 3.5D]

    = [-17.5, -12.5, -7.5, -2.5, 2.5, 7.5, 12.5, 17.5]
- อยากรู้ว่าค่าที่เรา sampling มาอยู่ช่วงไหน เอาไปหาร D
  - $\frac{6.1}{5} \Rightarrow 1.22$ → ตัวนี้ normalize PAM
    - ลองเทียบกับค่า Q codeword ขณะยังไม่คูณ D 1.22 (เราเลือกค่าแบบปัดขึ้น)
      - 1.5 ค่านี้คือ normalize quantize value

↳ ลองหาผลต่าง 1.22 − 1.5 = −0.38 ค่านี้คือ normalize error

↳ ยิ่ง level น้อยค่านี้แนวโน้มที่มากขึ้น

↳ เทียบค่า codeword ต่อ 1.5 คือ index ที่เท่าไหร่ ?

↳ ตอบ: 5 ไง

↳ เราก็ทำการ encode สัญญาณด้วย 101 #

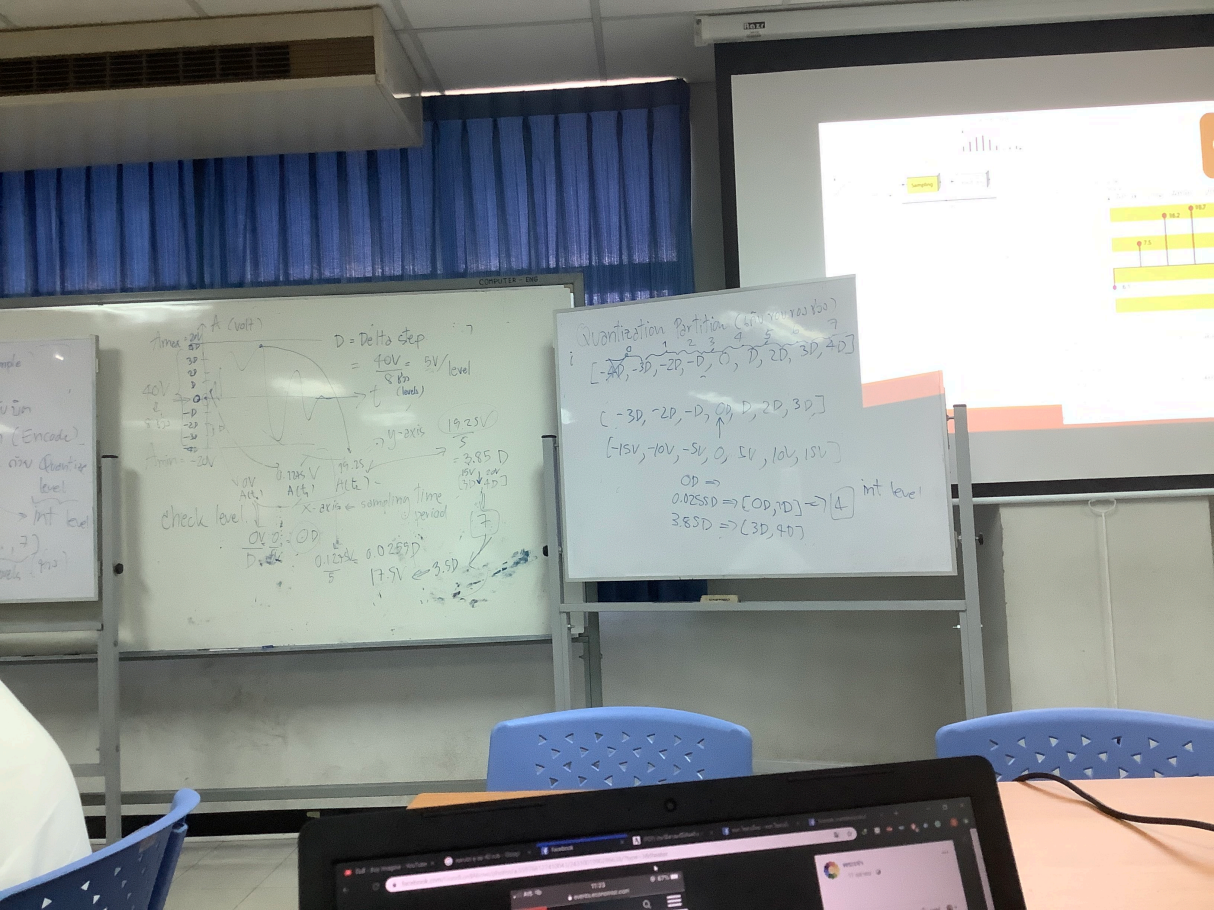
จากตัวอย่างก่อน ถ้าเราสุ่มเก็บมาได้ค่าเป็น −9.4 จงหา normalized PAM, normalized quantize value, normalize error, ข้อมูลที่ encode

normalized PAM $= \frac{V}{D} = \frac{-9.4}{5} = -1.88$

normalized quantized = ปัดขึ้น = −1.5

normalized error = จริง − ปลอม $= -1.88 - (-1.5) = -0.38$

index ที่ nq อยู่ = 2 ข้อมูลที่ encode = 010


Quantization Partition
Check level
D = Delta Step
int level

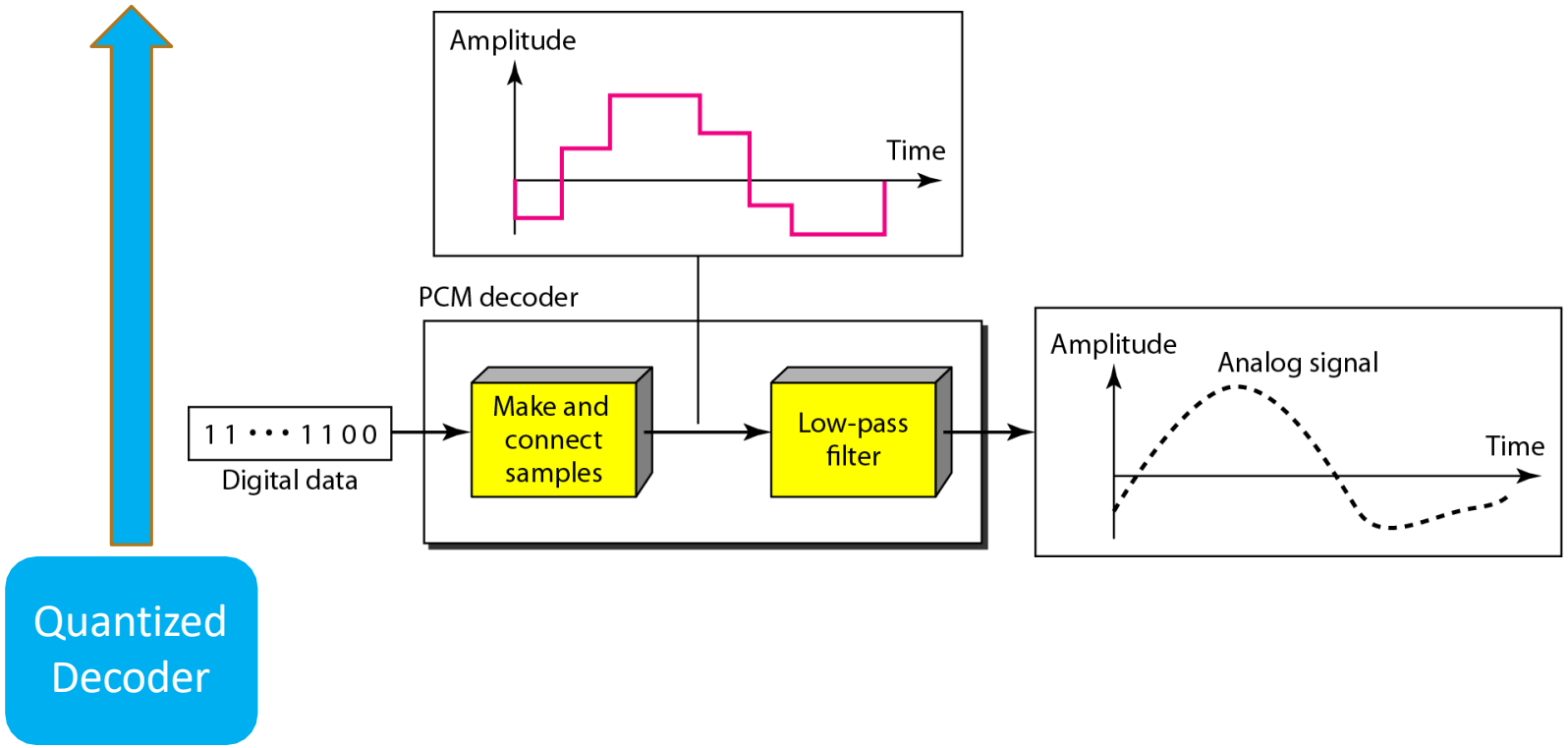
PCM Decoder

Quantization Decoder

Low Pass Filtering

#Quantization Level

Qlevel bound:
[-3.5D, -2.5D, -1.5D, 1.5D, 2.5D, 3.5D]



Quantization Encoder

Sampled Quantizer Encode	Quantization partition	[-.5 0 .5]
	Quantization codebook	[-.75 -.25 .25 .75]
	Input signal vector length	1
	Sample time (*เปลี่ยนตาม Period ทุกครั้ง*)	.01
Integer to Bit Converter	Number of bit per integer	2

ต้องกำ bit depth ของทั้ง 2 และ codebook ต้องเท่ากัน ตอน encode , decode

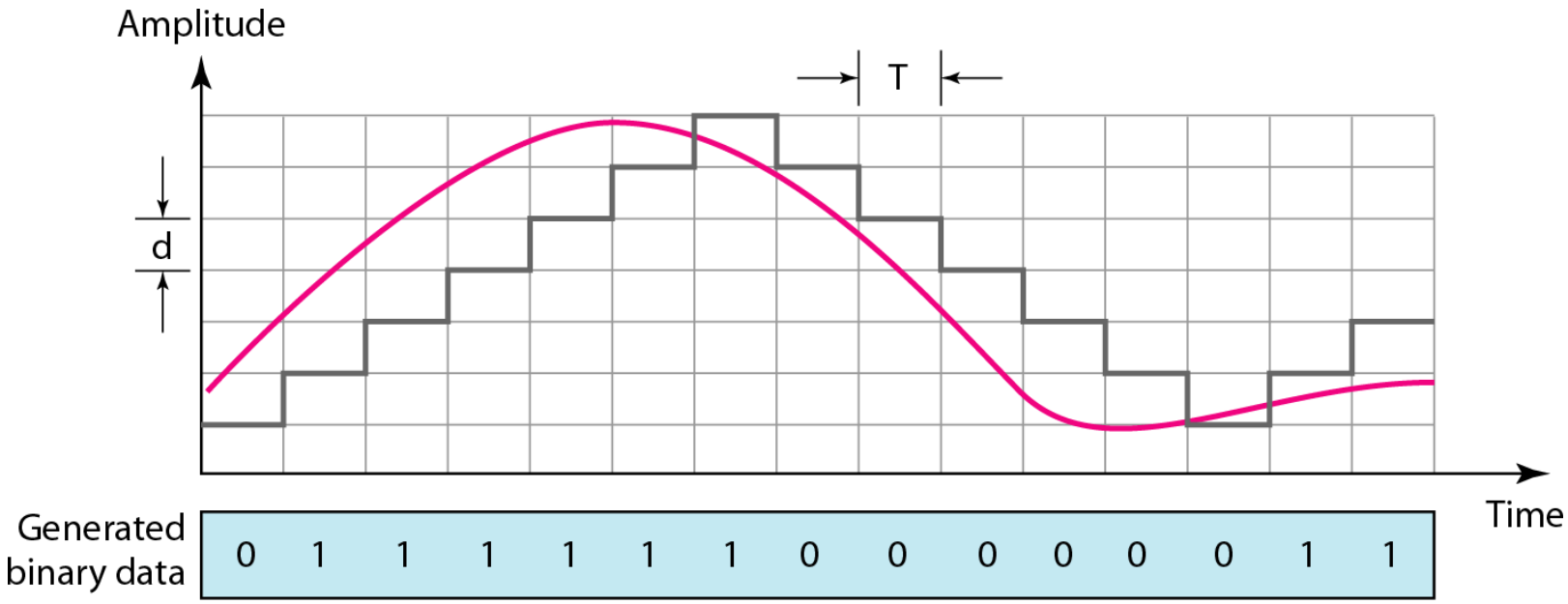
Quantization Decoder

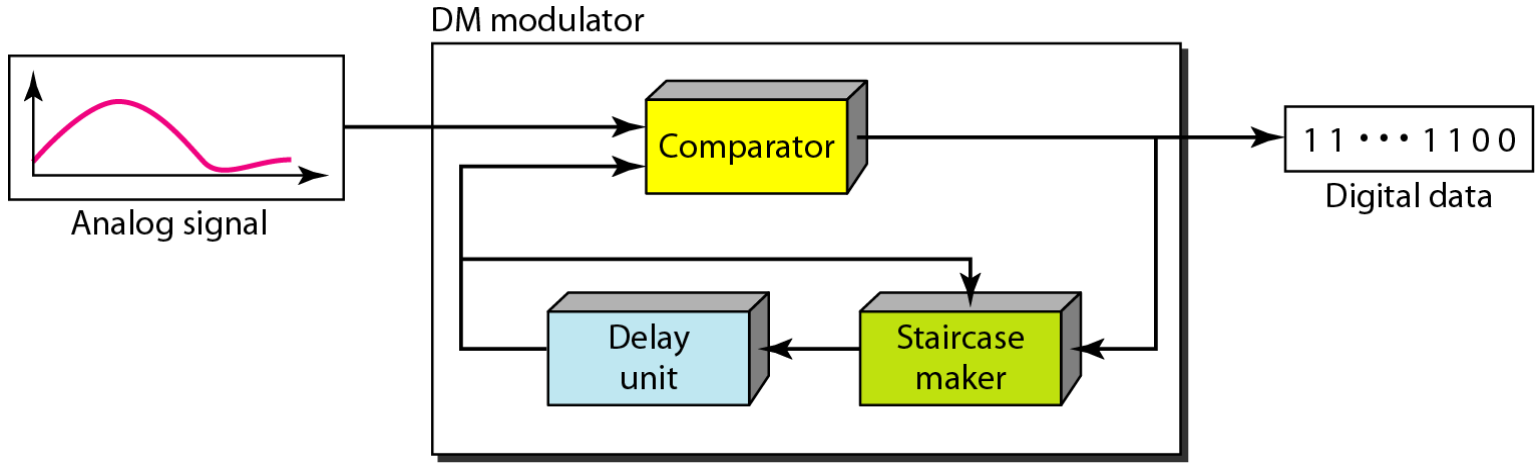
Bit to Integer Converter	Number of bit per integer	2
Quantizer Decode	Quantization codebook	[-.75 -.25 .25 .75]

ตัวอย่างของ sampling rate และ bit depth และกิจกรรมที่ใช้ในการเรียนการสอน

Activity# 9.1

Delta Modulation Encoder

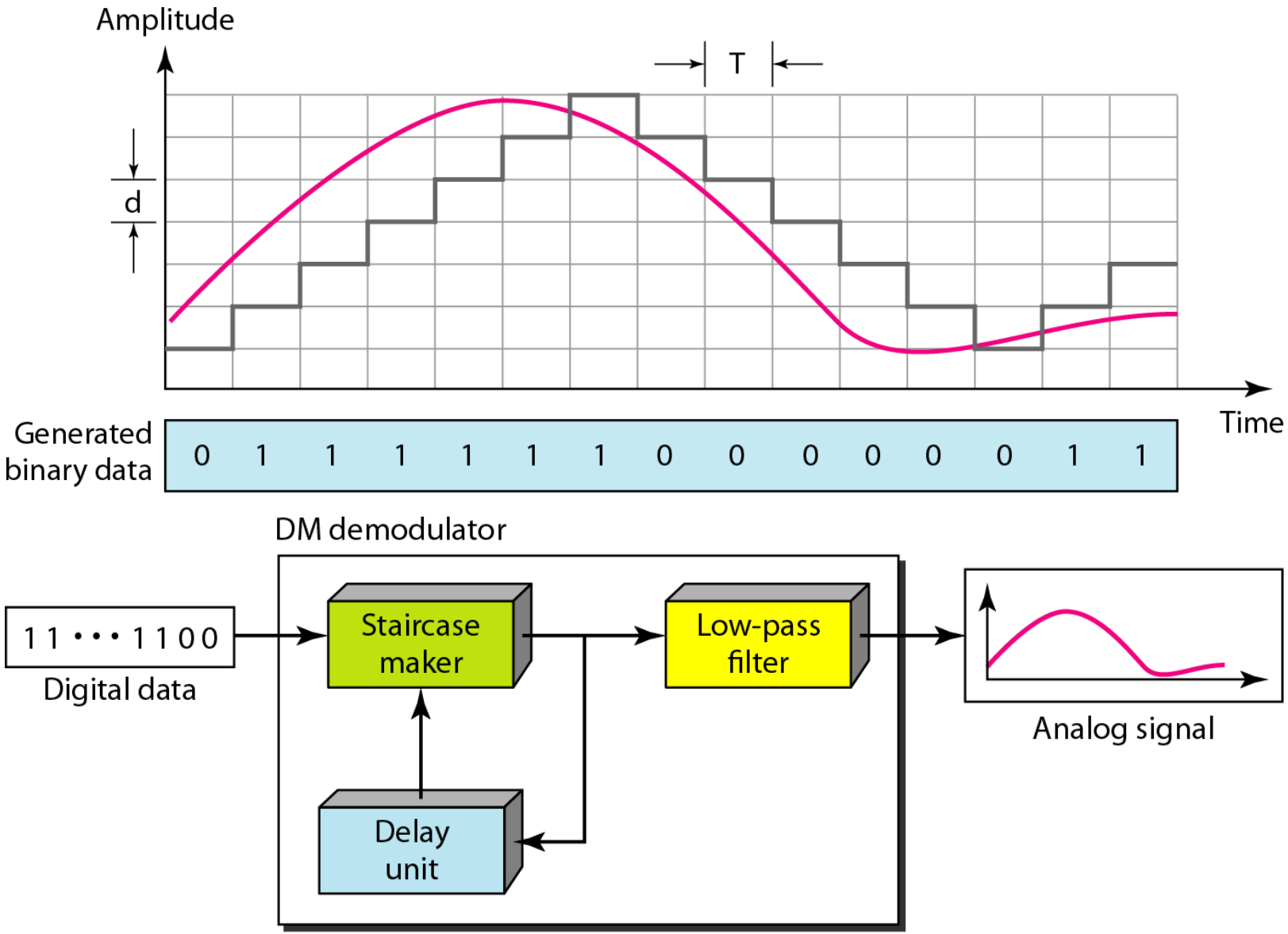




Delta Modulation Decoder

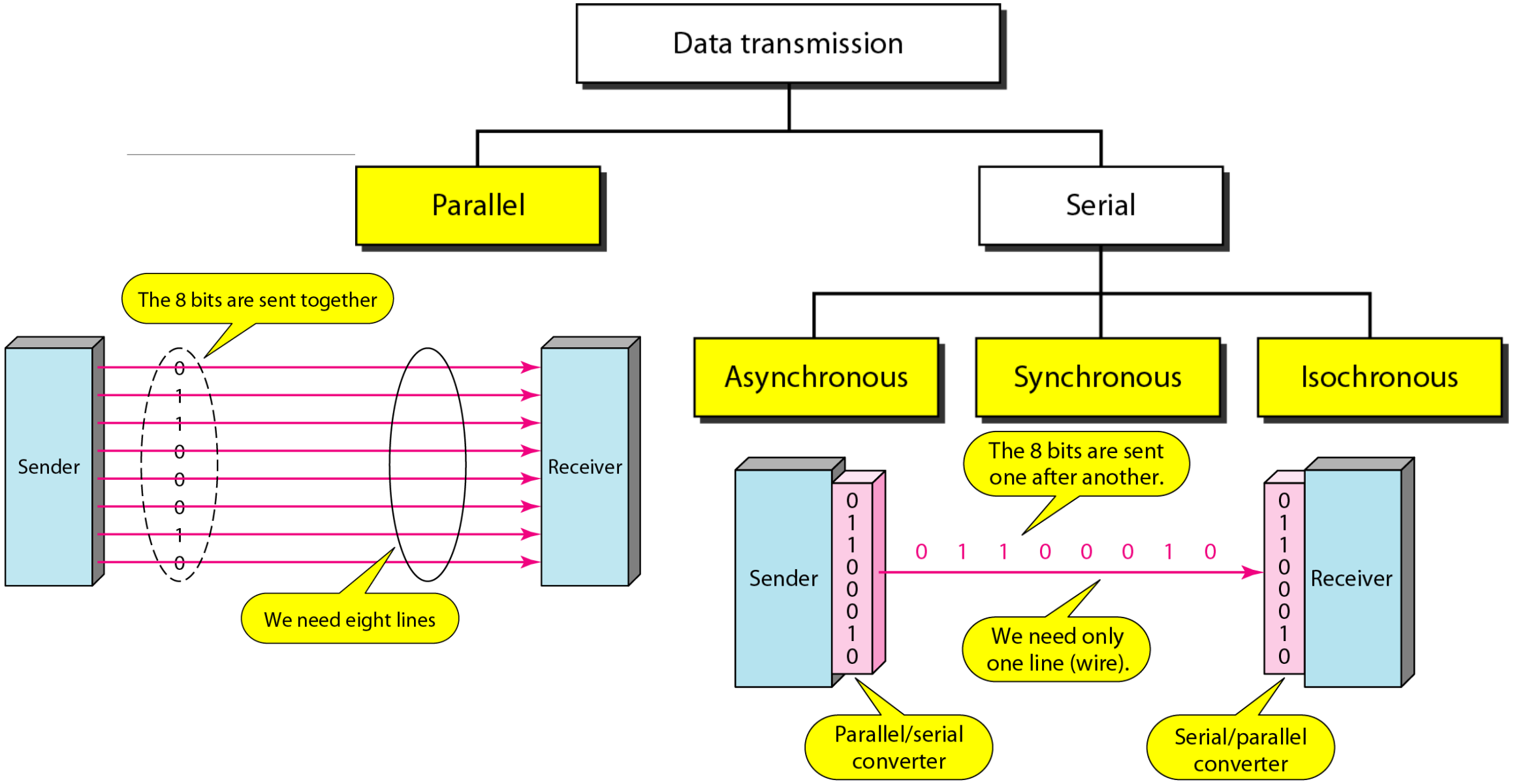
Quantization Decoder

Low Pass Filtering

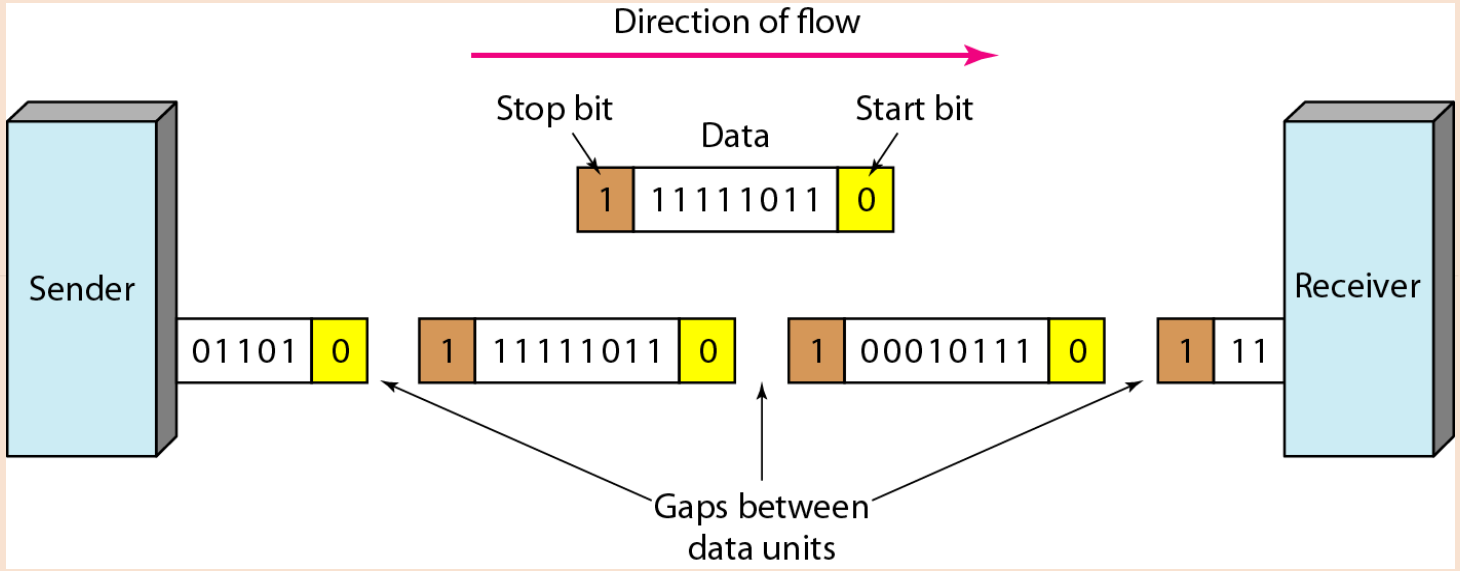


DIGITAL TRANSMISSION MODE

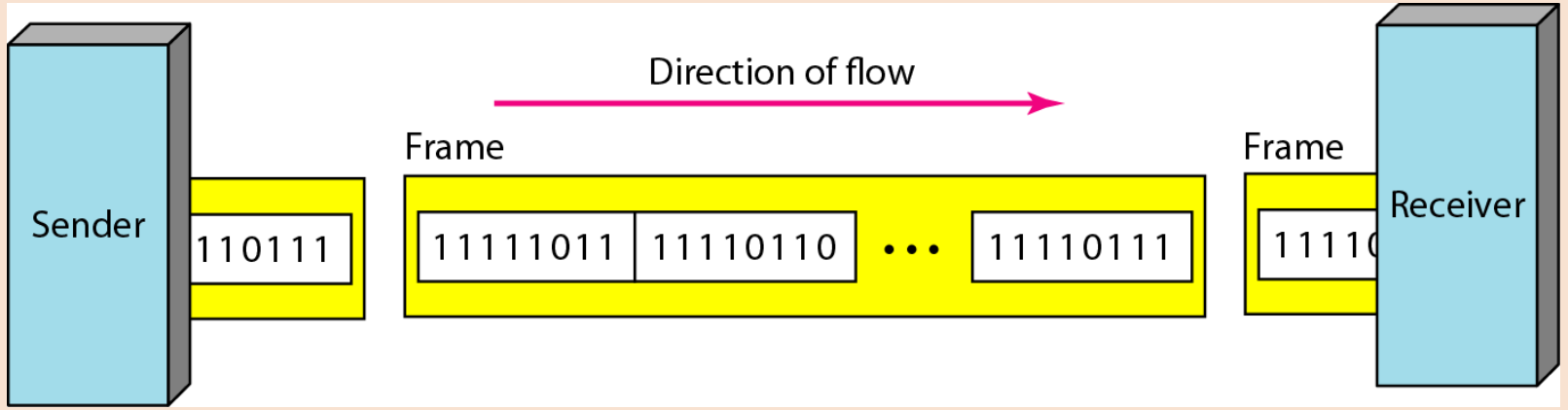
Parallel vs Serial


Data transmission
Parallel
Serial
Asynchronous
Synchronous
Isochronous
The 8 bits are sent together
Sender
0
1
1
0
0
0
1
0
Receiver
We need eight lines
The 8 bits are sent one after another.
Sender
0 1 1 0 0 0 1 0
Receiver
We need only one line (wire).
Parallel/serial converter
Serial/parallel converter

Asynchronous Transmission



Synchronous Transmission



Activity# 9.2

นำ NOTEBOOK ลง MATLAB มา ด้วย

Playing with PCM audio files

1) Read Audio files (download from Google Classroom)

use:

```
[y,Fs] = audioread(filename)
```

```
[y,Fs] = audioread(filename,samples)
```

```
[start,finish].
```

2) What are the sampling rate (Fs) and quantization bits (nBits: Bits per sample) ?

use:

```
info = audioinfo(filename)
```

3) Play with different Fs and nBits

use:

```
player = audioplayer(Y,Fs,nBits)
```

y(:, 2:3)

Fs = [8000 22050 44100 96000]

nBits = [8 16 24]

Reading PCM audio files (Options)

`[y,Fs] = audioread(filename)` reads data from the file named `filename`, and returns sampled data, `y`, and a sample rate for that data, `Fs`.

`[y,Fs] = audioread(filename,samples)` reads the selected range of audio samples in the file, where `samples` is a vector of the form `[start,finish]`.

File Format	BitsPerSample	Data Type of y	Data Range of y
WAVE (`.wav`)	8	`uint8`	0 ≤ y ≤ 255
	16	`int16`	-32768 ≤ y ≤ +32767
	24	`int32`	-2^31 ≤ y ≤ 2^31–1
	32	`int32`	-2^31 ≤ y ≤ 2^31–1
	32	`single`	-1.0 ≤ y ≤ +1.0
	64	`double`	-1.0 ≤ y ≤ +1.0
WAVE (`.wav`) (u-law)	8	`int16`	-32124 ≤ y ≤ +32124
WAVE (`.wav`) (A-law)	8	`int16`	-32256 ≤ y ≤ +32256
FLAC (`.flac`)	8	`uint8`	0 ≤ y ≤ 255
	16	`int16`	-32768 ≤ y ≤ +32767
	24	`int32`	-2^31 ≤ y ≤ 2^31–1
MP3 (`.mp3`), MPEG-4 AAC (`.m4a`, `.mp4`), OGG (`.ogg`), and certain compressed WAVE files	N/A	`single`	-1.0 ≤ y ≤ +1.0

Playing PCM audio files (Options)

`player = audioplayer(Y,Fs)` creates an `audioplayer` object for signal `Y`, using sample rate `Fs`. The function returns a handle to the `audioplayer` object, `player`.

`player = audioplayer(Y,Fs,nBits)` uses `nBits` bits per sample for signal `Y`.

`player = audioplayer(Y,Fs,nBits,ID)` uses the audio device identified by `ID` for output.

`Y`	Audio signal represented by a vector or two-dimensional array containing `single`, `double`, `int8`, `uint8`, or `int16` values. The value range of the input sample depends on the data type. The following table lists these ranges. **Data Type** / **Sample Value Range** `int8` / -128 to 127 `uint8` / 0 to 255 `int16` / -32768 to 32767 `single` / -1 to 1 `double` / -1 to 1
`Fs`	Sampling rate in Hz. Valid values depend on both the sample rates permitted by MATLAB® and the specific audio hardware on your system. MATLAB has a hard restriction of 1000 Hz <= Fs <= 384000 Hz, although further hardware-dependent restrictions apply. Typical values supported by most sound cards are 8000, 11025, 22050, 44100, 48000, and 96000 Hz.
`nBits`	Bits per sample. Specify only when signal `Y` is represented by floating-point values. Valid values depend on the audio hardware installed: `8`, `16`, or `24`. **Default:** `16`
`ID`	Device identifier. To obtain the ID of a device, use the `audiodevinfo` function. **Default:** `-1` (default device)