

วิชา Data Communication Laboratory

ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

การทดลองที่ 6 การสร้างกราฟสัญญาณรูปแบบต่างๆ

วัตถุประสงค์

1. ศึกษาการใช้งานโปรแกรม MATLAB เบื้องต้น
2. เข้าใจการทำงานของคำสั่ง MATLAB เบื้องต้น
3. สามารถสร้างสัญญาณด้วยคำสั่ง MATLAB
4. ศึกษาการเขียนโปรแกรมบนบอร์ด Arduino
5. ทดลองการสร้างสัญญาณด้วยการเขียนโปรแกรมสั่งงานจากบอร์ด Arduino

MATLAB เบื้องต้น

MATLAB ย่อมาจาก MATrix LABoratory โดยพัฒนาโดย The Mathworks, Inc เป็นโปรแกรมที่นิยมใช้สำหรับการคำนวณทางวิศวกรรม และวิทยาศาสตร์ เนื่องจากมีฟังก์ชันทางคณิตศาสตร์ให้เลือกใช้หลากหลาย และมีการพัฒนาอยู่เสมอ สามารถใช้ทดสอบ และนำมาพัฒนาอัลกอริทึมต่อได้สะดวกรวดเร็ว ทั้งยังใช้ประมวลผลร่วมกับโปรแกรมอื่นได้ เช่น Fortran, Borland C/C++, Microsoft Visual C++ เป็นต้น ทั้งยังสามารถนำไปใช้งานด้านกราฟิก แสดงผลได้ทั้งภาพสองมิติ ภาพสามมิติ นำภาพมาต่อกันเพื่อแสดงภาพเคลื่อนไหว จุดเด่นที่สำคัญคือ MATLAB เป็นระบบ Interactive คือ ข้อมูลพื้นฐานส่วนใหญ่เป็น Matrix หรือ Array ที่ไม่ต้องกำหนดขนาด หรือมิติเหมือนโปรแกรมภาษาอื่น และมีส่วนที่เป็นโครงสร้างแบบจำลอง (Simulink) ให้ใช้งานสำหรับจำลองระบบต่างๆ

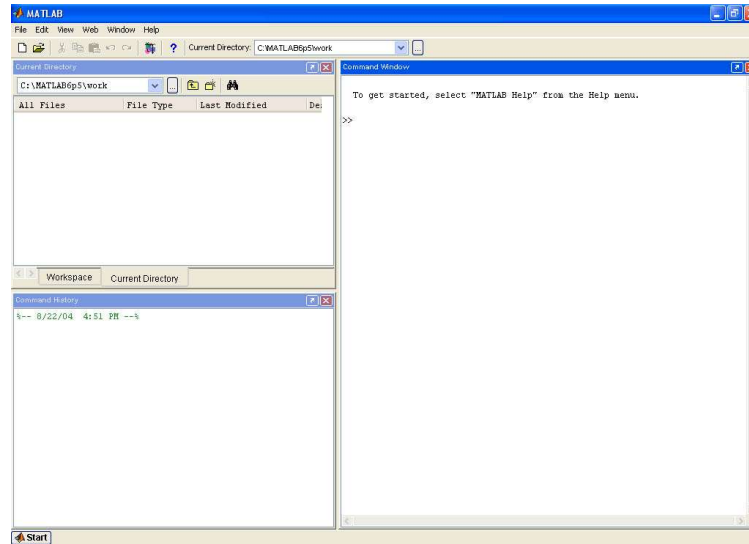
การใช้งานโปรแกรม MATLAB เบื้องต้น

หลังจากเปิดโปรแกรม MATLAB จะมีหน้าต่างซึ่งประกอบด้วยส่วนต่าง ได้แก่ Current Directory, Command History, Command Windows และอาจมีหน้าต่างอื่น ดังรูปที่ 6.1 ในส่วนของการทดลองนี้จะเริ่มกล่าวถึงการใช้งานส่วนที่เป็น Command Windows ซึ่งเป็นส่วนที่สั่งงานหลัก เพื่อการทำการคำนวณหรือเรียกใช้ฟังก์ชัน จะมีลักษณะเป็นส่วนต่อประสานแบบชุดคำสั่ง (Command Line Interface) โดยโปรแกรม MATLAB จะพร้อมใช้งานเมื่อมีเครื่องหมาย prompt ">>" ขึ้นมาสำหรับทำการคำนวณ หรือเรียกใช้คำสั่ง หรือใช้งานฟังก์ชัน

การคำนวณ

สามารถทำการคำนวณโดยใส่ค่าต่าง และเครื่องหมาย สัญลักษณ์สำหรับการกระทำทางคณิตศาสตร์ เบื้องต้น ได้แก่ + บวก, - ลบ, * คูณ, / หาร, ^ ยกกำลัง ฯลฯ ได้ทันที

```
>> 19+1*29+1
ans =
    49
```



รูปที่ 6.1 แสดงโปรแกรม MATLAB

การใช้งานตัวแปร

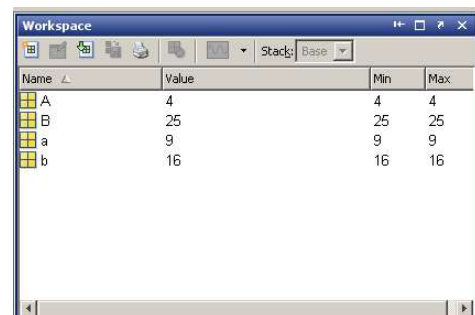
การกำหนดตัวแปรไม่จำเป็นต้องมีการประกาศตัวแปร และกำหนดชนิดของตัวแปร สามารถทำการกำหนดค่า หรือตัวแปรต่างๆ ได้โดยกำหนดให้ตัวแปรมีค่า (และขนาดต่างๆ) ด้วยเครื่องหมาย "=" โดยการกำหนดชื่อตัวแปรจะต้องขึ้นต้นด้วยตัวอักษร และไม่ซ้ำกับคำเฉพาะ (ชื่อฟังก์ชันอื่น) โดยโปรแกรม MATLAB จะตีความหมายของตัวแปรที่เป็นตัวอักษรตัวใหญ่และตัวเล็กจะไม่เหมือนกัน (Case Sensitive) และสามารถเปลี่ยนชนิดตัวแปรโดยกำหนดค่าให้ใหม่ได้ทันที

```
>> ong = 19+1*29+1
ong =
    49
>> ong = 'Jirasak'
ong =
Jirasak
```

คำสั่ง who

who เป็นคำสั่งเพื่อใช้ตรวจสอบว่ามีการใช้งานตัวแปรใดบ้าง (สามารถดูรายละเอียดของตัวแปรได้จากหน้าต่าง Workspace)

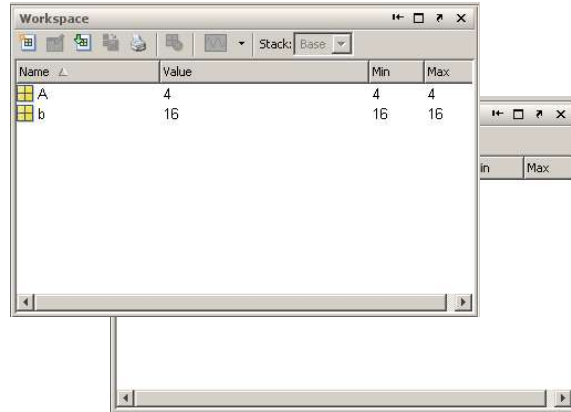
```
>> A = 2^2
A =
    4
>> a = 3^2
a =
    9
>> b = 4^2
b =
   16
>> B = 5^2
B =
   25
>> who
Your variables are:
A B a b
```



คำสั่ง clear

เป็นคำสั่งเพื่อลบตัวแปรออก โดยสั่ง “clear ตามด้วยชื่อตัวแปร” หรือ “clear all” เพื่อลบตัวแปรทั้งหมดที่ใช้อยู่

```
>> who
Your variables are:
A B a b
>> clear a B
>> who
Your variables are:
A b
>> clear all
>> who
>>
```



การใช้เครื่องหมายเซมิโคลอน “;”

การใช้เครื่องหมาย “;” ตามหลังคำสั่งต่างๆ เพื่อกำหนดให้โปรแกรม MATLAB ไม่แสดงผลการทำงาน

```
>> a = 10;
>> A = 19
A =
    19
>> who
Your variables are:
A b
```

ตัวแปร Vectors และ Matrices

การกำหนดตัวแปร Vectors และ Matrices เหมือนการกำหนดตัวแปรธรรมดา แต่จะเพิ่มการกำหนดในลำดับ และแถว โดยใช้เครื่องหมาย คอมา “,” หรือเว้นวรรค (Space Bar) เพื่อกำหนดลำดับในแถว และใช้เครื่องหมายเซมิโคลอน “;” ในการแบ่งระหว่างแถว

การกำหนดแถว

```
>> row1 = [1 2 3 4]
row1 =
     1     2     3     4
>> row2 = [1, 2, 3, 4]
row2 =
     1     2     3     4
```

การกำหนดหลัก

```
>> column = [1; 2; 3]
column =
     1
     2
     3
```

การกำหนดทั้งลำดับและแถว Vs การสลับเปลี่ยนระหว่าง แถว และ หลัก

```
>> x=[11 12 13 14; 21 22 23 24; 31 32 33 34]
x =
    11    12    13    14
    21    22    23    24
    31    32    33    34
```

```
>> y = x'
y =
    11    21    31
    12    22    32
    13    23    33
    14    24    34
```

การกำหนดตัวแปร Vectors และ Matrices ที่เป็นเชิงเส้น และการอ้างอิงค่าที่ตำแหน่งต่างๆ

การกำหนดตัวแปร Vectors และ Matrices ที่เป็นเชิงเส้น โดยใช้เครื่องหมาย โคลอน “:”

การกำหนดตัวแปรที่เพิ่มค่าทีละ 1

โดยสามารถกำหนดโดยใช้เครื่องหมาย “:” ระหว่างค่าเริ่มต้นและค่าสุดท้ายในเครื่องหมาย []

```
>> a = [1:9]
a =
     1     2     3     4     5     6     7     8     9
>> b = [-9:-1]
b =
    -9    -8    -7    -6    -5    -4    -3    -2    -1
```

การกำหนดตัวแปรที่เพิ่มเป็นค่าคงที่

โดยสามารถกำหนดได้ดังนี้ [ค่าเริ่มต้น : ค่าที่เปลี่ยนแปลงไป : ค่าสุดท้าย]

```
>> test1 = [1:2:9]
test1 =
     1     3     5     7     9
>> test2 = [9:-2:-5]
test2 =
     9     7     5     3     1    -1    -3    -5
```

การอ้างอิงค่าที่ตำแหน่งต่างๆ

```
>> Ooo = [0:5:37]
Ooo =
     0     5    10    15    20    25    30    35
>> Ooo(4)
ans =
    15
>> test = [0:11:100;1:11:100]
test =
     0    11    22    33    44    55    66    77    88    99
     1    12    23    34    45    56    67    78    89   100
>> test(2,4)
ans =
    34
>> test(1,[2:5])
ans =
    11    22    33    44
```

การใช้เครื่องหมายสำหรับ Vector และ Matrix

เครื่องหมาย +, -

การบวก หรือ ลบ เมตริกซ์จะใช้เครื่องหมาย “+” หรือ “-” ซึ่งจะกระทำเหมือนกับการบวก (หรือลบ)

เมตริกซ์ธรรมดาโดยจะนำค่าแต่ละตัวมาทำการบวกกัน (หรือลบกัน)

```
>> a = [5:2:15]
a =
     5     7     9    11    13    15
>> b = [5:-1:0]
b =
     5     4     3     2     1     0
>> a+b
ans =
    10    11    12    13    14    15
>> b-a
ans =
     0    -3    -6    -9   -12   -15
```

เครื่องหมาย *, /

การคูณ หรือ หาร เมตริกซ์จะใช้เครื่องหมาย “+” หรือ “-” ซึ่งจะกระทำเหมือนกับการคูณ (หรือหาร) เมตริกซ์ ซึ่งจากตัวอย่างที่ผ่านมาไม่สามารถใช้คำสั่ง $a*b$ ได้เนื่องจากคูณเมตริก จำนวนหลักของตัวตั้งต้องเท่ากับ จำนวนแถวของตัวคูณ จึงต้องใช้คำสั่ง $a*b'$ (เช่นเดียวกันไม่สามารถใช้คำสั่ง a/b ได้เช่นกัน)

```
>> a = [5:2:15]; b = [5:-1:0];
>> a * b'
ans =
    115
>> a / b
ans =
    2.0909
```

เครื่องหมาย .*, ./

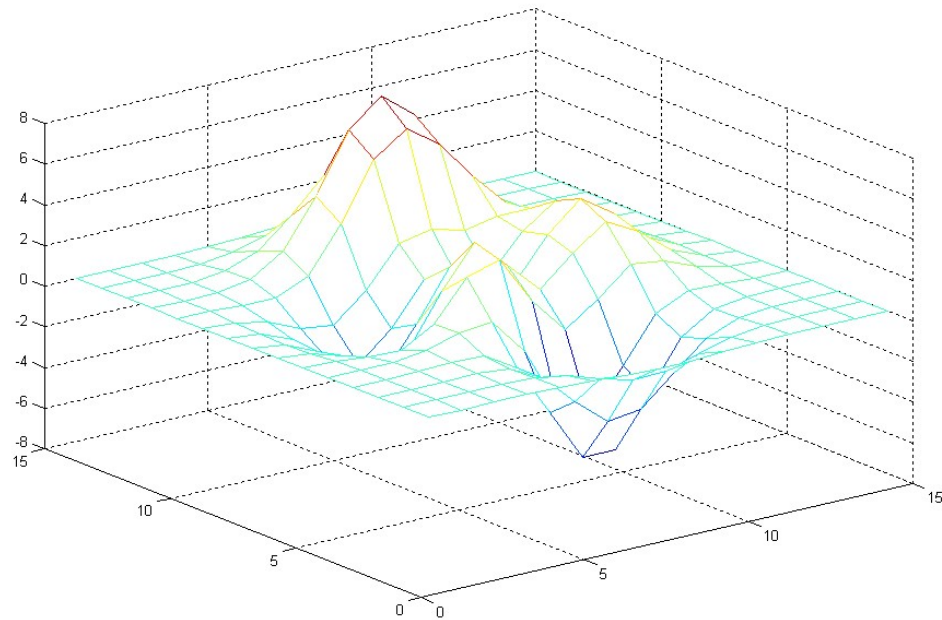
เป็นการสั่งให้เมตริก 2 เมตริกทำการคูณ หรือหารที่ตำแหน่งเดียวกัน (แถว หลัก เดียวกัน คูณหรือการกัน)

```
>> a = [5:2:15]; b = [5:-1:0];
>> a .* b
ans =
    25    28    27    22    13     0
>> b ./ a
ans =
    1.0000    0.5714    0.3333    0.1818    0.0769     0
```

การวาดกราฟ

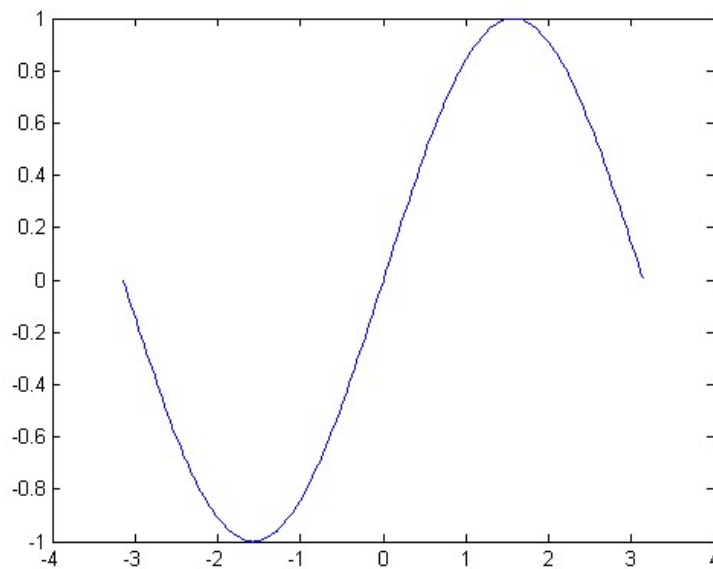
กราฟกล่าวได้ว่าเป็นสิ่งที่สำคัญในการวิเคราะห์ และแสดงผลของโปรแกรม MATLAB โดยสามารถวาดกราฟได้หลายชนิดทั้งสองมิติ และสามมิติ

```
>> mesh(peaks(15))
```



รูปที่ 6.2 แสดงการวาดกราฟจากคำสั่ง mesh(peaks(15))

```
>> x = [-pi : pi/100 : pi];
>> y = sin(x);
>> plot(x,y);
```



รูปที่ 6.3 แสดงการวาดกราฟจากคำสั่ง plot(x,y);

ตารางที่ 6.1 รูปแบบเส้น สัญลักษณ์ และสี สำหรับการวาดกราฟ

สีที่แสดง		สัญลักษณ์ที่ใช้		รูปแบบของเส้น	
สัญลักษณ์ที่ใช้	สี	สัญลักษณ์ที่ใช้	เครื่องหมาย	สัญลักษณ์ที่ใช้	รูปแบบเส้น
R	red	.	:	.	เส้นจุด
G	green	o	วงกลม	-	เส้นทึบ
B	blue	+	บวก	-.	เส้นประและจุด
C	cyan	x	กากบาท	--	เส้นประ
m	magenta	*	ดอกจัน		
Y	yellow	s	สี่เหลี่ยม		
k	black	d	ข้าวหลามตัด		
w	write	v	สามเหลี่ยมล่าง		
		^	สามเหลี่ยมบน		
		>	สามเหลี่ยมขวา		
		<	สามเหลี่ยมซ้าย		

ตารางที่ 6.2 คำสั่งในรูปแบบการวาดกราฟสองมิติ

คำสั่ง	การทำงาน
plot (x, y)	วาดกราฟของ x, y
plot (x, y, x, z)	วาดกราฟของ x, y และ วาดกราฟของ x, z ในแกน x เดียวกัน
subplot (m,n,p)	แบ่งหน้าต่างรูปภาพเป็นหน้าต่างย่อย m แถว n หลัก และใช้ p เป็นลำดับการวาดกราฟ
figure (n)	สร้างหน้าต่างรูปภาพหมายเลข n
clf	ลบรูปภาพบนหน้าต่าง
close (n)	ลบหน้าต่างรูปภาพหมายเลข n
close all	ลบหน้าต่างรูปภาพทั้งหมด

ตารางที่ 6.3 คำสั่งการแบ่งสเกล

คำสั่ง	การทำงาน
axis(xmain xmax ymain ymax)	กำหนดค่าสูงสุด และค่าต่ำสุดในแกน x และแกน y
axis(xmain xmax ymain ymax zmin zmax)	กำหนดค่าสูงสุด และค่าต่ำสุดในแกน x แกน y และแกน z
X =axis	การหาค่าสูงสุด และค่าต่ำสุด ในแกนต่างๆ
grid on	การเพิ่มเส้น grid ให้แกนปัจจุบัน
grid off	ยกเลิกเส้น grid ให้แกนปัจจุบัน

ตารางที่ 6.4 คำสั่งการแบ่งเซลล์

คำสั่ง	การทำงาน
title ('????')	ตั้งชื่อหัวเรื่องกราฟ
xlabel ('????')	เขียนข้อความที่แกน x
ylabel ('????')	เขียนข้อความที่แกน y
zlabel ('????')	เขียนข้อความที่แกน z
gtext ('????')	เขียนข้อความที่ตำแหน่งเมาส์ชี้
text ('????')	เขียนข้อความที่ตำแหน่ง x y

ตารางที่ 6.5 คำสั่งการวาดกราฟเส้น 3 มิติ

คำสั่ง	การทำงาน
plot3 (x, y, z)	การเขียนกราฟ 3 มิติ
plot3 (x1, y1, z1, S1, x2, y2, z2, S2, x3, y3, z3, S3)	

สามารถศึกษาการวาดกราฟอื่นได้ โดยการใช้ Help ดูคำสั่งต่างๆ เช่น contour, contour3, pcolor (การวาดกราฟเส้นโครงร่าง), mesh, meshc, meshz (การวาดกราฟพื้นผิวร่างแห), surf, surfc (การวาดกราฟพื้นผิว), ฯลฯ

การทดลองที่ 6.1 การใช้คำสั่ง MATLAB บน Command Windows และ บน Editor (M-File)

- ให้นักศึกษาพิมพ์คำสั่งต่อไปนี้บน Command Windows แล้วอธิบายผลที่ได้

```
>> clear all;
>> close all;
>> A = [1:10]
>> B = [1:100]/10
>> C = [1:0.1:10]
>> D = [0.1:0.1:10]
```

- เลือก File => New => Blank M-File หลังจากนั้นให้นักศึกษาพิมพ์คำสั่งต่อไปนี้บน Editor (M-File) แล้วเลือก Debug => Run ทดสอบการทำงาน พร้อมอธิบายผลที่ได้

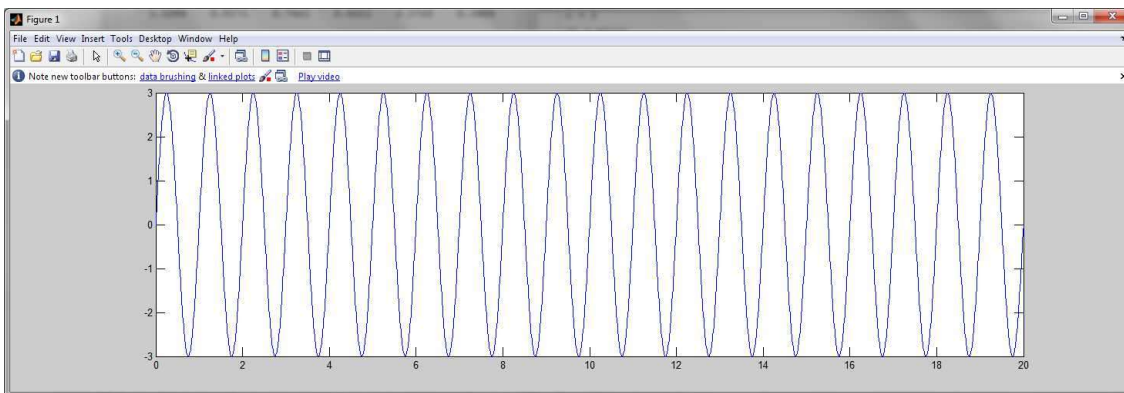
```
clear all;
close all;
x = [0:10]
m = -2
c = 3
y1 = m*x+c
y2 = c.^x
figure (1)
plot (x,y1)
figure (2)
plot (x,y1)
figure (3)
```



```
subplot(2,1,1), plot (x,y1)
subplot(2,1,2), plot (x,y2)
figure (4)
subplot(1,2,1), plot (x,y1)
subplot(1,2,2), plot (x,y2)
```

3. ให้นักศึกษาทดลองสร้างสัญญาณ Sine Wave ที่ความถี่ 1 Hz ขนาด 3 Volt ตามค่าเวลา $t = [0:20]$ แล้วแสดงผลกราฟสัญญาณด้วยคำสั่ง plot บน Editor (M-File)

4. ให้นักศึกษาพิมพ์คำสั่งบน Command Windows (แก้คำสั่งในข้อ 3.) เพื่อให้ได้ผลดังรูปที่ 6.5



รูปที่ 6.4 สัญญาณ $3\sin(2\pi t)$

5. เชิญอาจารย์ตรวจการทดลอง

.....
ลายเซ็นอาจารย์ผู้ตรวจการทดลอง

การทดลองที่ 6.2 การสร้าง Pulse Wave บน Arduino

1. ให้นักศึกษาเขียนโปรแกรมบนบอร์ด Arduino UNO R3 ดังนี้

1.1. ส่วนหัวโปรแกรมให้กำหนดแปรดังนี้

- 1) zeta เป็น Array ชนิด float เก็บค่ามุม 4 มุม ได้แก่ (0° , 90° , 180° , 270°)
- 2) S เป็น Array ชนิด float สำหรับเก็บค่าสัญญาณ Sine ของมุมต่างๆ
- 3) pwmDuty เป็น Array ชนิด uint16_t สำหรับเก็บค่า Duty Cycle ของ สัญญาณ PWM

1.2. ส่วนฟังก์ชัน setup()

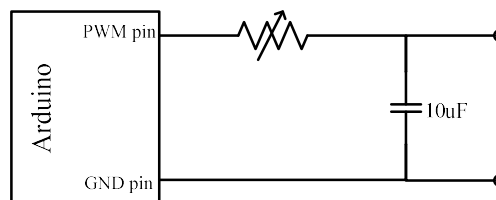
- 1) ให้ตั้งค่าขา Digital ที่ต้องการสร้างสัญญาณ Pulse Width Modulation (PWM) โดยสามารถเลือก pin_number (3, 5-6, 9-11) ด้วยคำสั่งรูปดังนี้

```
Serial.begin(115200);
pinMode(pin number, OUTPUT);
```
- 2) ทำการคำนวณค่า S[i] และ pwmDuty[i] เพื่อนำไปใช้ต่อในฟังก์ชัน loop() โดยที่
 - o ให้ S[i] เก็บค่าจากการคำนวณค่า Sine ของมุมต่างๆ ได้แก่ 0° , 90° , 180° และ 270°
 - o ให้ pwmDuty[i] เก็บค่า Duty Cycle ของ สัญญาณ PWM Pulse ($0 \leq \text{Duty Cycle} \leq 255$) ตามค่า Sine ที่ได้จาก S[i]
 - o Debug ค่า S[i] และ pwmDuty[i] แต่ละลำดับด้วยคำสั่ง Serial.println; บันทึกค่าที่สัมพันธ์กับ zeta ต่างๆ

1.3. ส่วนฟังก์ชัน loop() ให้เขียนโปรแกรมเพื่อวนแสดงค่า PWM Pulse จากข้อ 1.2 ที่ละชุดด้วยคำสั่ง

```
for(int i=0; i<4; i++){
  analogWrite(pin number, pwmDuty[i]);
  delayMicroseconds(4000);
}
```

2. ต่อวงจร Low Pass Filter เพื่อกรองสัญญาณความถี่ต่ำ โดยด้วยตัวต้านทานปรับค่าได้ 10K Ω (ต้องบดกรีสายไฟที่ขาตัวต้านทานปรับค่า) และ ตัวเก็บประจุขนาด 10 μF ดังรูปที่ 6.6 แล้วใช้ Oscilloscope Ch1 วัดสัญญาณตกคร่อม C กับ Ground และ Ch2 วัดสัญญาณตกคร่อม PWM กับ Ground ปรับค่าตัวต้านทานปรับค่าเพื่อให้สัญญาณที่ Ch1 ใกล้เคียง Sine Wave มากที่สุด



รูปที่ 6.5 วงจร Low Pass Filter

3. ให้นักศึกษาวาดรูปกราฟของสัญญาณและเปรียบเทียบความแตกต่างของสัญญาณที่ได้จากข้อ 1 (Ch2) และ ข้อ 2 (Ch1)

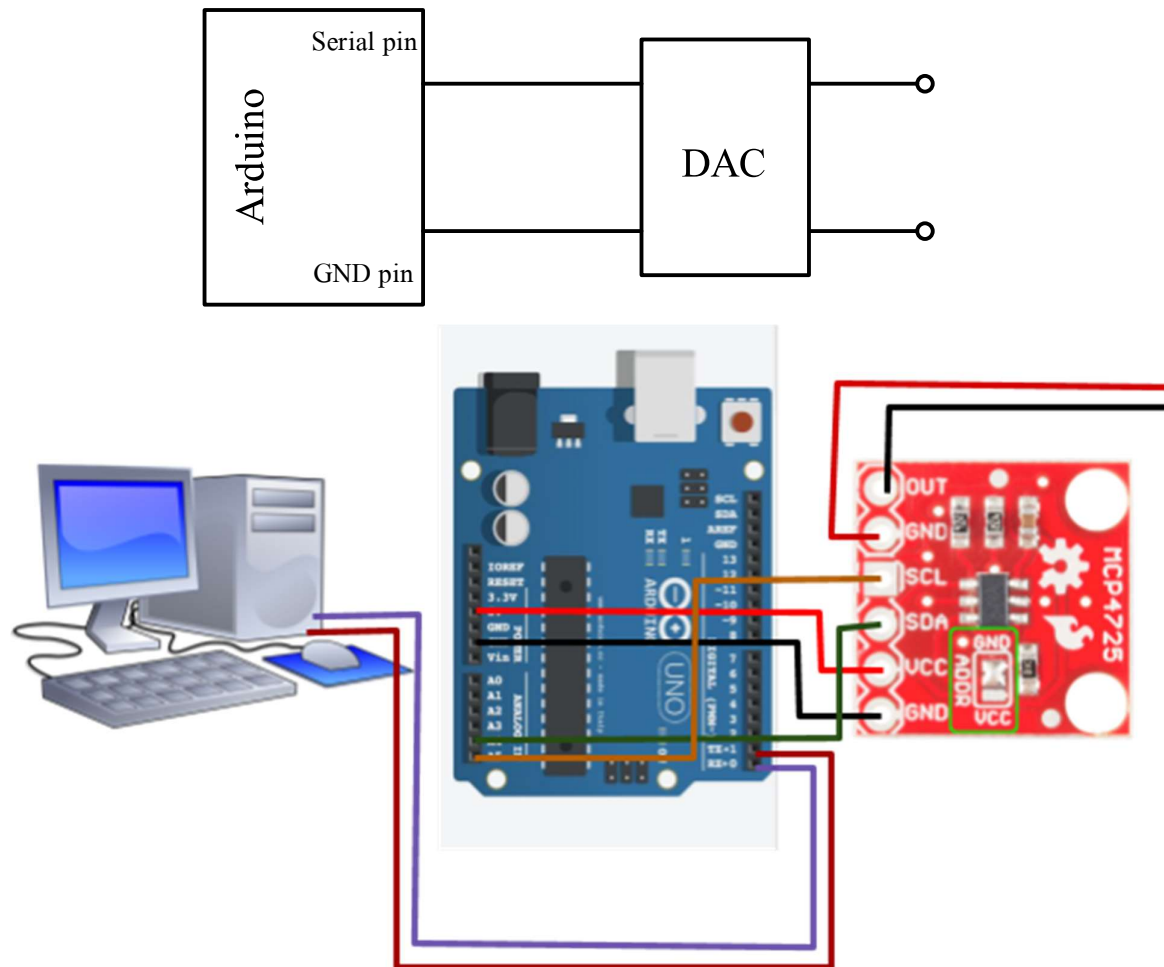
4. ทำการทดลองเพิ่มเติม โดยปรับเปลี่ยนโปรแกรมให้เพิ่มจำนวนมุมที่คำนวณค่าองศา Sine เป็นจำนวน 8 และ 16 มุม และคำนวณค่าของ PWM ที่สัมพันธ์กัน
5. ให้นักศึกษาวาดรูปกราฟของสัญญาณ และเปรียบเทียบความแตกต่างของสัญญาณที่ตกคร่อม C เมื่อจำนวนมุมเป็น 4, 8 และ 16 มุม

6. เชิญอาจารย์ตรวจการทดลอง

.....
ลายเซ็นอาจารย์ผู้ตรวจการทดลอง

การทดลองที่ 6.3 การสร้าง Sine Wave บน Arduino

1. Download Library Adafruit เพื่อใช้ส่งสัญญาณติดต่อกับ วงจร DAC (MCP4725) จาก
<https://learn.adafruit.com/mcp4725-12-bit-dac-tutorial/using-with-arduino>
https://github.com/adafruit/Adafruit_MCP4725
<https://cdn-learn.adafruit.com/downloads/pdf/mcp4725-12-bit-dac-tutorial.pdf>



รูปที่ 6.6 การต่อ Arduino กับ วงจร DAC (MCP4725)

2. ให้นักศึกษาเขียนโปรแกรมบนบอร์ด Arduino UNO R3 ดังนี้

2.1. ส่วนหัวโปรแกรม

- 1) ให้เขียนโปรแกรมหาดังนี้

```
#include <Wire.h>
#include <Adafruit_MCP4725.h>
Adafruit_MCP4725 dac;
int delay0;

#define defaultFreq 1700 // dac speed
#define freq0 500 // sine wave frequency
```

- 2) พร้อมกำหนดตัวแปรอื่นที่จำเป็น

2.2. ส่วนฟังก์ชัน setup()

- 1) ให้ตั้งค่า Serial Communication และการเชื่อมต่อวงจรแปลงสัญญาณ Digital-to-Analog (DAC: MCP4725 12-Bit Digital to Analog (DAC) Converter I2C Interface Module) ดังนี้

```
void setup(void)
{
    Serial.begin(115200); // set buadrate serial
    dac.begin(0x62);      // set to default For MCP4725A1
    //dac.begin(0x64);    // set For MCP4725A2
    //dac.begin(0x60);    // set For MCP4725A0
    delay0 = (1000000/freq0 - 1000000/defaultFreq)/4;
                        // delay for period of sine
    Serial.print("delay0 is ");
    Serial.println(delay0);
}
```

- 2) ทำการคำนวณค่า $S[i]$ และ $S_DAC[i]$ เพื่อนำไปใช้ต่อในฟังก์ชัน loop() โดยที่
 - o ให้ $S[i]$ เก็บค่าจากการคำนวณค่า Sine ของมุมต่างๆ ได้แก่ 0° , 90° , 180° และ 270°
 - o ให้ $S_DAC[i]$ เก็บค่า Sine ที่ได้ ในรูปแบบของ Digital ขนาด 12 บิต เพื่อส่งให้วงจร DAC ($0 \leq DAC \leq 4095$)
 - o **Debug ค่า $S[i]$ และ $S_DAC[i]$** แต่ละลำดับด้วยคำสั่ง Serial.println;

2.3. ส่วนฟังก์ชัน loop() ให้เขียนโปรแกรมเพื่อส่งสัญญาณ Sine จากมุมต่างๆ ไปยังวงจร DAC ด้วยคำสั่ง

```
for(int i=0 ; i<4 ; i++){
    dac.setVoltage(S_DAC[i], false);
    delayMicroseconds(delay0); // สำหรับ sine freq0
}
```

3. ทำการทดลองเพิ่มเติม โดยปรับโปรแกรมในข้อ 2 ให้เพิ่มจำนวนมุมที่คำนวณค่าองศา Sine เป็นจำนวน 8, 16 และ 32 มุม ที่ความถี่ $freq0 = 500$ Hz เท่าเดิม
4. ให้นักศึกษาวาดรูปกราฟของสัญญาณ และอธิบายความแตกต่างของสัญญาณที่ได้ เมื่อจำนวนมุมเป็น 4 และ 16 มุม ที่ความถี่ $freq0 = 500$ Hz

5. ทำการทดลองเพิ่มเติม โดยเขียนโปรแกรมสร้างสัญญาณ Sine จำนวน 32 มุม ที่ความถี่ `freq0` เป็น 500, 1,000 และ 5,000 Hz ตามลำดับ
6. ให้นักศึกษาวาดรูปกราฟของสัญญาณ และอธิบายความแตกต่างของสัญญาณที่ได้จากข้อ 5 เมื่อส่งค่า Sine ที่ความถี่ 500 และ 5,000 Hz

7. สรุป และวิเคราะห์ผลการทดลอง

8. เชิญอาจารย์ตรวจการทดลอง

.....

ลายเซ็นอาจารย์ผู้ตรวจการทดลอง