Mini-Project

Morse Code Convertor

จัดทำโดย

นายกรธรรม ศรีธนธรรม

รหัสนักศึกษา 61010022

เสนอ

รศ.ดร.ยุทธพงษ์ รังสรรค์เสรี

รายงานฉบับนี้เป็นส่วนหนึ่งของรายวิชา Microprocessor

ภาคเรียนที่ 2 ปีการศึกษา 2562

ภาควิชาวิศวกรรมโทรคมนาคม คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

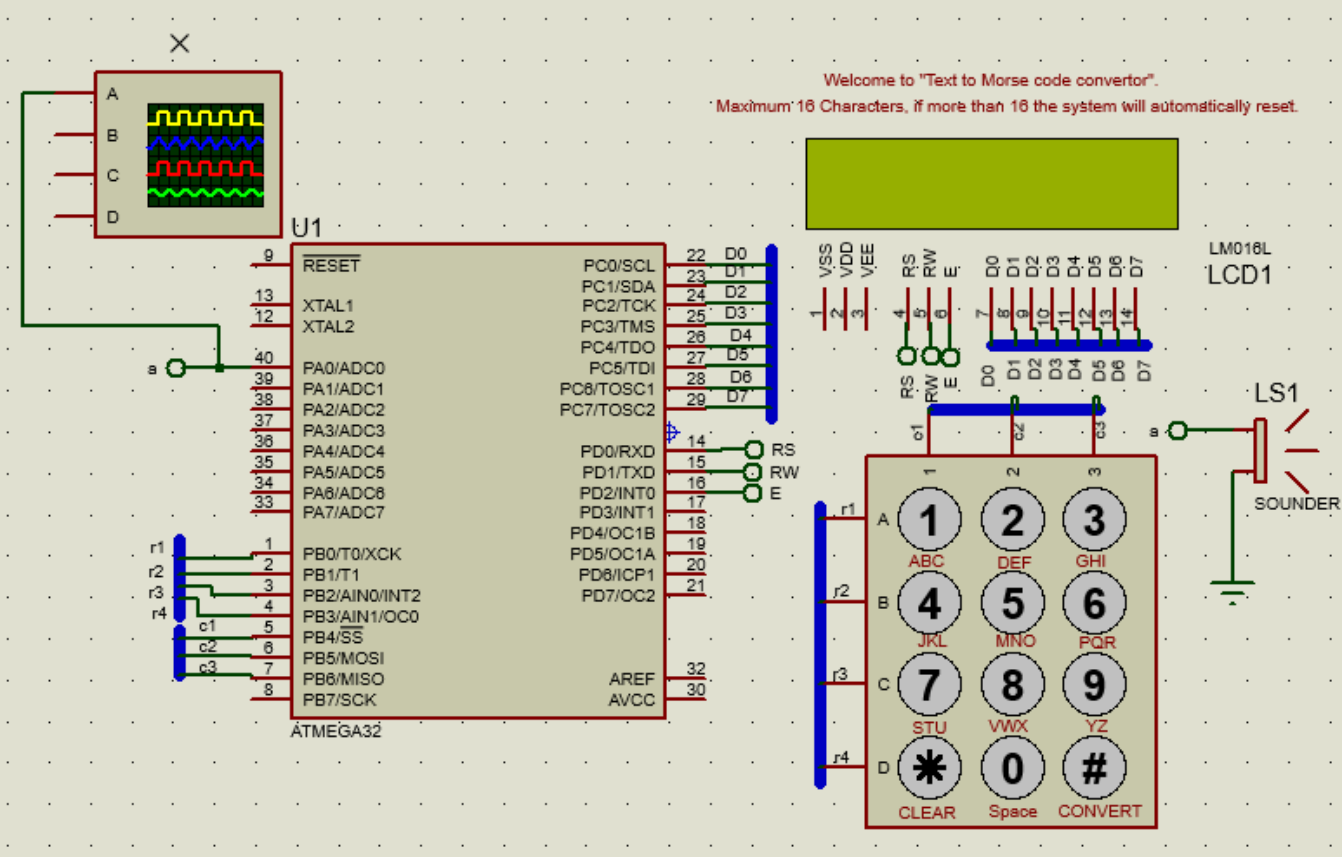
คำนำ

โครงงานนี้เป็นการออกแบบอุปกรณ์ “เครื่องแปลงรหัสมอร์ส (Morse code convertor) “ โดยรหัสมอร์สเป็นวิธีการส่งผ่านสารสนเทศข้อความเป็นชุดสัญญาณเสียงไฟหรือเสียงเคาะ (click) เปิด-ปิด โดยรหัสมอร์สนั้นยังถูกใช้อย่างแพร่หลายในวงการวิทยุสมัครเล่น โดยเครื่องแปลงรหัสมอร์สนี้จะสามารถแปลงเป็นรหัสมอร์สจากข้อความภาษาอังกฤษที่ผู้ใช้ป้อนเข้ามาได้ ผู้จัดทำหวังว่าโครงงานนี้จะมีประโยชน์ต่อการศึกษาถึงเครื่องแปลงรหัสมอร์สผ่านไมโครคอนโทรลเลอร์ หากมีข้อแนะนำหรือข้อผิดพลาดประการใด ผู้จัดทำขอน้อมรับไว้และขออภัยมา ณ ที่นี้ด้วย

ผู้จัดทำ

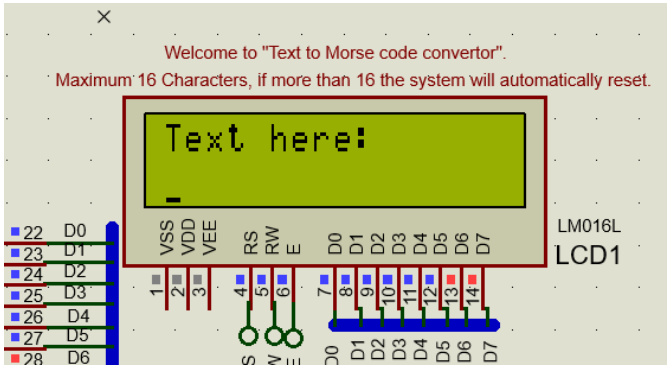
วันที่ 1 พฤษภาคม พ.ศ.2563

1.การทำงานของอุปกรณ์



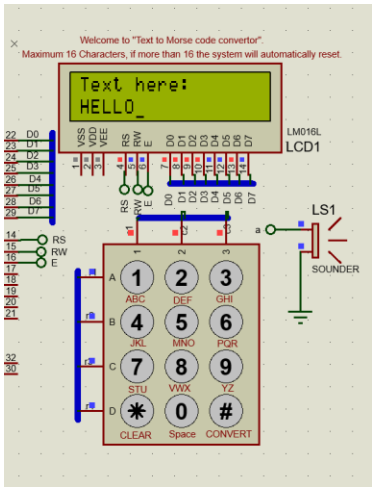
การทำงานของเครื่องแปลงรหัสมอร์สนั้นจะเริ่มจากเมื่อทำการเปิดเครื่องแล้ว

1) จอ LCD จะติดและแสดงคำว่า Text Here: และ Cursor จะขึ้นไปอยู่บรรทัดสอง



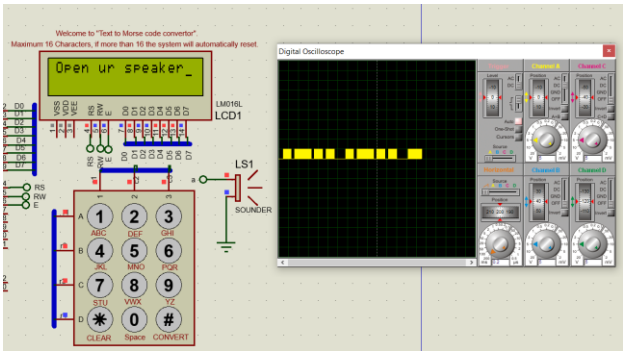
ภาพเมื่อเริ่มโปรแกรม

2) โดยเมื่อ LCD ทำการ initialize เรียบร้อยแล้ว ผู้ใช้จะสามารถป้อนตัวอักษรผ่านแป้นพิมพ์ keypad โดยเป็น keypad แบบโทรศัพท์สมัยก่อนที่สามารถเป็นแป้นพิมพ์ตัวอักษรภาษาอังกฤษได้ผ่านการกดอีกครั้งที่ตัวเดิม



3) เมื่อผู้ใช้ป้อนข้อมูลที่ต้องการที่จะแปลงรหัสมอร์สแล้วให้กดปุ่ม # เพื่อที่จะแปลงเป็นรหัสมอร์ส

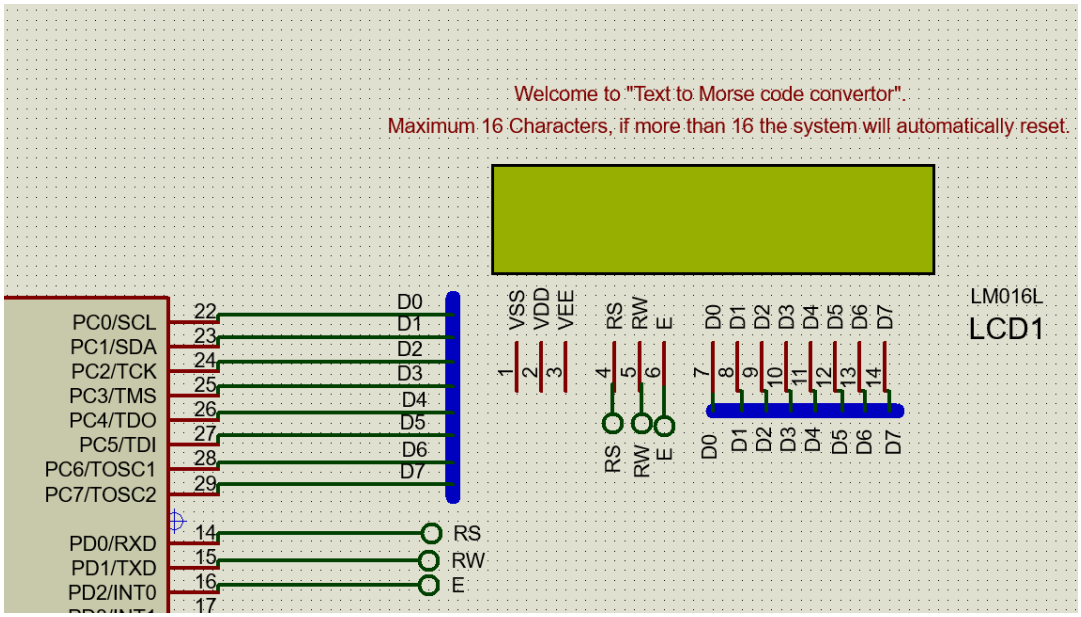
4) เมื่อกด # แล้วรหัสมอร์สจะถูกแปลงและออกมาทางเสียงผ่านอุปกรณ์ SOUNDER และสามารถสังเกตสัญญาณที่ออกมาผ่านออสซิโลสโคป(oscilloscope)



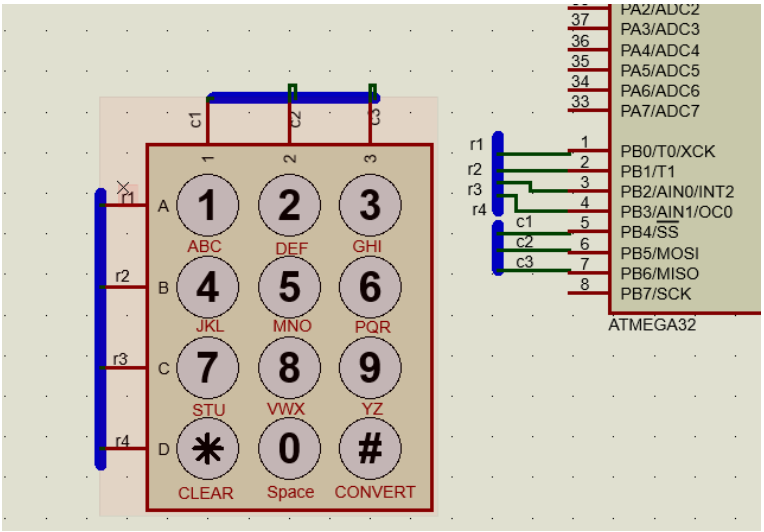
5) เมื่อแปลงรหัสเสร็จแล้วจะกลับไปสู่หน้าเริ่มต้นผู้ใช้สามารถป้อนข้อมูลใหม่เพื่อแปลงเป็นรหัสมอร์สได้

2.วงจรของเครื่องแปรงรหัสมอร์ส

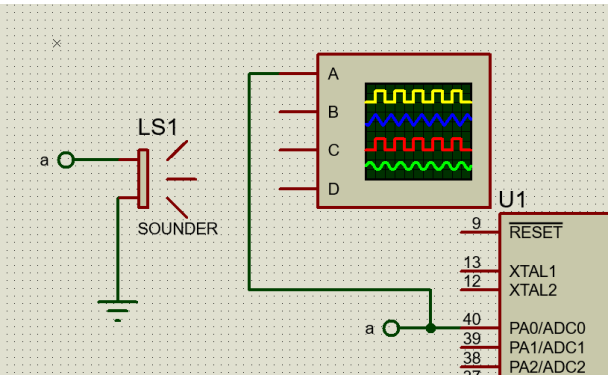
2.1) วงจร LCD : ต่อแบบ 8 bit mode ผ่านพอร์ต C และ D ดังรูปภาพ **โดยใช้อุปกรณ์ชื่อ LM016L**



2.2) วงจร Keypad : โดย row ต่อผ่านพอร์ต B0-B4 และ column ต่อผ่าน B4-B7 โดยkeypad **ใช้อุปกรณ์ที่ชื่อว่า KEYPAD-PHONE**



2.3) วงจร SPEAKER : ต่อผ่านพอร์ต A0 โดยใช้อุปกรณ์ SOUNDER



3.วิธีการออกแบบ

3.1) การออกแบบ LCD : การออกแบบ LCD จะมี function ที่เกี่ยวข้องทั้งหมด 5 functions

3.1.1) lcdCommand เป็นฟังก์ชันไว้ส่ง Command ไปยัง LCD

3.1.2) lcdData เป็นฟังก์ชันไว้ส่งข้อมูลไปยัง LCD

3.1.3) lcd_gotoxy เป็นฟังก์ชันที่จะเลื่อน cursor ไปยังตำแหน่งต่างๆใน LCD

3.1.4) lcd_print เป็นฟังก์ชันที่จะส่งตัวแปร string ไปยังหน้าจอ

3.1.5) lcd_init เป็นฟังก์ชันที่ไว้ initialize ให้หน้าจอ LCD โดยเป็นการใช้ 8-bits mode

โดย D0-D7 ของ LCD จะเชื่อมต่อไป PC0-PC7 ตามลำดับ และ RS, RW, E ไปยัง PD0, PD1, PD2 ตามลำดับ

3.2) การออกแบบ Keypad : โดย Keypad นี้จะอยู่ใน function main เพื่อให้ง่ายต่อการทำงานของอุปกรณ์(เฉพาะในวงจรนี้) โดยการออกแบบนั้นจะต้องมีตัวแปรที่จำเป็นคือ data[16], i, j โดยอุปกรณ์นี้จะรับอินพุตได้ 16 ตัวอักษรและเก็บไว้ในอาเรย์ data จะขออธิบายการออกแบบแบบไล่ลงไปเนื่องจากไม่มีการใช้ interrupt เพราะฉะนั้นการอธิบายแบบ polling

3.2.1) ลูปนอกสุดคือ for(i=0;i<16;i++) โดยลูปนี้จะเป็นลูปที่คอยรับข้อความที่ผู้ใช้ป้อนเข้าสู่อาเรย์ตัวที่ i ของ data โดยถ้าเกิน 16 ตัวแล้วจะหลุดลูป

3.2.2) ลูปถัดไปคือ while loop while(j!=1) โดยคำสั่งนี้มีไว้เพื่อให้รันรับค่าจาก keypad ไปเรื่อยๆจนกว่าค่า j จะเป็น 1 (Default = 0) โดยค่า j จะเป็น 1 เมื่อรหัสมอร์สส่งออกไปครบแล้ว

3.2.3) ต่อไปจะเป็นการเช็คค่าการกดปุ่มของ Keypad ต่อการที่จะกดปุ่มจะมีการเขียนโปรแกรมมารองรับก่อนเข้ารับค่า

```
do
{
    KEY_PRT &= 0xF0;      /* mask PORT for column read only */
    asm("NOP");
    colloc = (KEY_PIN & 0xF0); /* read status of column */
}while(colloc != 0xF0);
```

1

```
do
{
    do
    {
        _delay_ms(20);              /* 20ms key debounce time */
        colloc = (KEY_PIN & 0xF0); /* read status of column */
    }while(colloc == 0xF0);         /* check for any key press */

    _delay_ms (40);             /* 20 ms key debounce time */
    colloc = (KEY_PIN & 0xF0);
}while(colloc == 0xF0);
```

2

<u>ส่วนที่ 1</u> จะเป็นการเช็คค่าว่าไม่มีการกดปุ่มใดๆ <u>ส่วนที่ 2</u> จะเป็นการเช็คว่ามีการกดปุ่มแล้วก่อนที่จะเข้ารับค่าจริง

3.2.4) จะเข้าเช็คค่าว่ามีการกดปุ่มไหนโดยใช้วิธีการ ไล่เช็คทีละ row โดย และเก็บค่าของพอร์ต B เก็บไว้ใน colloc

```
KEY_PRT = 0xFE;              /* check for pressed key in 1st row */
colloc = KEY_PIN;
```

เมื่อเข้าเช็คที่ row นั้นและเก็บค่าไว้ที่ colloc แล้วจะทำการไล่เช็คแต่ละ column โดยถ้า colloc ที่เก็บค่ามีค่าตรงกับ column และ row นั้นจะเข้าสู่เงื่อนไข โดยมีการเก็บค่าไว้ที่ data

```
if(colloc == 0xEE) //r1c1
    {
            lcd_print("A");
            data[i]='A';
```

3.2.5) เมื่อส่งค่าไปที่ LCD แล้วจะยังไม่หลุดลูปจะทำการเช็คว่ามีการกดค่าที่ row,column นั้นอีกไหม(ให้เหมือนโทรศัพท์รุ่นปุ่มกด) โดยจะมีการใช้ timer0 (Normal mode, no prescaler) เพื่อ เช็คว่าในระหว่างที่ timer ทำงานมีการกดปุ่มไหม โดยขณะที่ TIFR ยังไม่มีค่าเป็น 1 ก็จะมีการเช็คค่าว่ามีการกดปุ่มนั้นอีกครั้งไหม ภายในเวลา 3.5 Second ถ้ามีการกดปุ่มก็จะลบค่าที่ส่งไปยัง LCD และเปลี่ยค่าใหม่ด้วยตัวอักษรตัวถัดไป (เวลาของ timer0 นั้นมีค่าน้อยมากจนไม่นำมาคิด) แต่ถ้าภายใน 3.5 second ไม่มีการกดก็จะหลุดลูปและค่า i จะเพิ่มขึ้นเพื่อรับค่าต่อไป เช่นเดียวกันกับตัวที่ 3 ก็จะมีการใช้ timer0 เพื่อเช็คว่ามีการกดปุ่มครั้งที่ 3 ไหม

```
TCNT0 = 0x20;
TCCR0 = 0x01;
while((TIFR&0x1)==0)
{
   colloc = 0;
   delay_ms(3500);
   colloc = KEY_PIN;
   if(colloc == 0xEE)
   {
      lcdCommand(0x10)
      lcd_print("B");
      data[i]='B';

   }
}
TCCR0 = 0;
TIFR = 0x1;
```

1

```
TCNT0 = 0x20;
TCCR0 = 0x01;
while((TIFR&0x1)==0)
{
         colloc = 0;
         delay_ms(5000);
         colloc = KEY_PIN;
         if(colloc == 0xEE)
         {
            lcdCommand(0x10);
            lcd_print("C");
            data[i]='C';
         }
}
TCCR0 = 0;
TIFR = 0x1;
break;
```

2

3.2.5) โดยตัวKEYPAD จะมีปุ่ม * ที่เป็นการ CLEAR ค่าออกไป 1 ค่า (หรือก็คือลบค่าก่อนหน้านั่นเอง) การทำงานนั้นจะเริ่มจากการ shift cursor ไปทางซ้าย 1 ค่าและปริ้นค่า" "(blank) ออกไปเพื่อลบตัวอักษรออกจาก LCD 1 ตัวก่อนหน้า cursor และที่ data นั้นจะลบค่าข้อมูลตัวที่ i-1 (เพราะปัจจุบันลูปอยู่ที่ i) และจะลบค่า i-=2 เพราะว่าเมื่อ break แล้วค่า i จะถูกเพิ่มไป 1 เพราะงั้นต้องลบไป 2 ค่าเพื่อกลับไปยังค่าก่อนหน้า และ shift cursor ไปทางซ้ายเพื่อรอรับค่า

```
if(colloc == 0xE7)
{
    lcdCommand(0x10);
    lcd_print(" ");
    data[i-1]=0;
    i-=2;
    lcdCommand(0x10);
   break;
}
```

3.2.6) เมื่อผู้ใช้อินพุตข้อมูลเสร็จเรียบร้อยแล้ว พร้อมที่จะแปลงรหัสมอร์สก็จะทำการกดปุ่ม # โดยมีการทำงานในขณะที่ j ไม่เท่ากับ 1 เท่านั้น (เพราะว่าเมื่อแปลงเสร็จค่า j จะเป็น 1 และหลุดลูปออกไป) เมื่อกดปุ่ม # แล้วจะทำการเคลียร์หน้าจอ LCD และปริ้น Open ur speaker ไปยังหน้าจอ LCD และเข้าสู่ section ของการแปลงรหัสมอร์สซึ่งจะขออธิบายส่วนถัดไป

```
while(j!=1)
{
lcdCommand(0x01);
   delay_ms(1000);
lcd_print("Open ur speaker");
```

3.3) การออกแบบการแปลงรหัสมอร์ส : หลังจากที่ผู้ใช้กดปุ่ม # แล้วจะเข้าสู่ช่วงการแปลงรหัสมอร์ส โดยจะมี function ภายนอกที่เกี่ยวข้อง 2 อัน คือ s(), l() โดยทั้งสองเป็นตัวแทนของการจำลอง Square wave ความถี่สูงและส่งค่าออกไปที่ A0 และออกไปยัง SOUNDER

```
void l()
{
   int a,b,c = 0;

  while(a<150)
  {
  for(a=0;a<150;a++)
  {
   for(b=0;b<1000;b++)
   {
      PORTA = 0x01;
   }
   for(c=0;c<1000;c++)
   {
      PORTA = 0x00;
   }
 }

 }
}
```

```
void s()
{
   int a,b,c = 0;

  while(a<75)
  {
  for(a=0;a<75;a++)
  {
   for(b=0;b<1000;b++)
   {
      PORTA = 0x01;
   }
   for(c=0;c<1000;c++)
   {
      PORTA = 0x00;
   }
   }
   }
}
```

โดย l นั้นแทนค่า แดช (-) จะมีค่าที่ยาว และ s จะแทนค่า ดอต(.) จะมีค่าที่สั้น

หลังจากกด # แล้วจะเข้าสู่ section ของการแปลงรหัส โดยมีตัวแปร ch1, ch2 และมีลูปที่จะมาควบคุม โดย ch0 จะมีค่าตั้งแต่ 0 จนถึงขนาดของ data

```
int ch1=0;
int ch2=0;
for(ch1=0;ch1<sizeof data;ch1++)
{
```

และจะเข้าสู่การเช็คว่า data[ch1] นั้นตรงกับตัวอักษรอะไร ตัวอย่างเช่น data[0] ตรงกับค่า ‘A’ ก็จะเข้าสู่ **else if(data[ch1] == ‘A’) //assume ch1 ==0 and data[0] == A**

โดย ‘A’ นั้นจะมีรหัสมอร์สคือ .- (dot-dash) โดยจะมีดีเลย์ระหว่างรหัสภายใน 0.5 sec และจะดีเลย์ 1.75 sec ก่อนที่จะเข้าไปสู่รหัสภายนอก (ch1 == 1) เพื่อให้ผู้ฟังสามารถแยกออกว่าจบตัวอักษรแรกแล้ว

```
if(data[ch1] == 0)
{
   delay_ms(1);
}
else if(data[ch1] == ' ')
{
   delay_ms(300);
}
else if(data[ch1] == 'A')
{
   s();
   delay_ms(500);
   l();
   delay_ms(1750);
}
```

เมื่อจบการแปลงข้อมูลทั้งหมดของ data แล้วค่า i จะเท่ากับ 0 และค่า j จะเท่ากับ 1 แล้วจะหลุดลูป

```
        }
    }
                --
                  i=0;j=1;


                  }

                  break;
                  }
            }
```

เมื่อหลุดลูปแล้วจะทำการเคลีย LCD ทั้งหมดและจะทำการลบค่า data ทั้งหมดและ set ค่า j ให้เท่ากับ 0 และวนกลับไปที่ main loop และมีการ initialize LCD และจะวนกลับสู่โปรแกรมเดิม

```
lcdCommand(0x01);
for(j=0;j<16;j++)
{
   data[j]=0;
}
j=0;
```

4.ผลการทดสอบ

เครื่องแปลงรหัสมอร์สมีการแปลงวงจรได้และตรงตามรหัสมอร์ส โดยมีเสียงสัญญาณรหัสมอร์สผ่านอุปกรณ์ SOUNDER และสามารถวัดสัญญาณได้ผ่าน oscilloscope

5.วิจารณ์ผลการทดลองและสรุป

5.1) วิจารณ์ผลการทดลอง

จากการทดลองจะพบว่าในส่วนของ LCD นั้นจะไม่มีปัญหาอันเนื่องมาจากการรันผ่านโปรแกรม แต่ในส่วนของ KEYPAD นั้นจะยังมีปัญหาอันเนื่องมาจากการจำลองผ่านคอมพิวเตอร์ เช่น เมื่อกดปุ่มครั้งที่ 2 แล้วจะยังไม่เปลี่ยนตัวอักษรให้เท่าที่ควร

5.2) สรุปการทดลอง

การทดลองจะพบว่าเมื่อทำการแปลงรหัสมอร์สแล้ว รหัสมอร์สจะออกมาในรูปของเสียง(ผ่านอุปกรณ์ SOUNDER) และสัญญาณไฟฟ้า(Square-wave) ที่สามารถวัดได้ (พอร์ต A0)

โดยจะมีการใช้ LCD ที่ไว้แสดงข้อความที่ผู้ใช้ป้อนผ่าน KEYPAD โดย KEYPAD จะสามารถป้อนเป็นอักษรภาษาอังกฤษได้