รายการอ้างอิง

1. Lee, Eung Suk and Lee, Hyung Seok. A study on torque measurement by excitation coil current in eddy current dynamometer. Proceeding of the Second International Symposium on Instrumentation Science and Technology (2002): 4/669 – 4/675.
2. Gebauer W and Wrichter M. Computer for measuring of torque applied to the car engine dynamometers. Siemens-Z 45 (October1971): 719.
3. Dombrowski, Kurt; Mattheis, Paul and Scharstein, Egbert. DC Machine torque calculator dynamometer system. Siemens Power Engineering 5 (September - October1983): 252-254.
4. T. G. Beckwith and N. Lewis Buck. Mechanical Measurement. Pittsburgh: Addison – Wesley Publishing Company,(n.d).
5. Robert C. Juvinall and Kurt M. Marshek. Fundamental of Machine Component Design. 3rd ed. Michigan: John Wiley & Sons, 2000.
6. ธีรบูลย์ หล่อวิเชียรรุ่ง, นคร ภักดีชาติ และ ชัยวัฒน์ ลิ้มพรจิตรวิไล. ปฏิบัติการไมโครคอนโทรลเลอร์ MCS51 ด้วยโปรแกรมภาษาซี. กรุงเทพมหานคร: อินโนเวตีฟ เอ็กเพอริเมนต์,(ม.ป.ท.)
7. ธีรวัฒน์ ประกอบผล. การพัฒนาไมโครคอนโทรลเลอร์ด้วยภาษาซี. กรุงเทพมหานคร: สมาคมส่งเสริมเทคโนโลยี(ไทย-ญี่ปุ่น),2545.
8. ธีรวัฒน์ ประกอบผล. การประยุกต์ใช้งานไมโครคอนโทรลเลอร์. กรุงเทพมหานคร: สมาคมส่งเสริมเทคโนโลยี(ไทย-ญี่ปุ่น),2540.
9. อุดม จีนประดับ. ไมโครคอนโทรลเลอร์ MCS-51. กรุงเทพมหานคร: สถาบันเทคโนโลยีพระจอมเกล้าพระนครเหนือ,2541.
10. ทีมงาน อีทีที. คู่มือการใช้งาน CP-SPI/RD2 V1 /V2/V3. กรุงเทพมหานคร: อีทีที,2545.
11. ศุภชัย บุศราทิจ. คู่มือภาษาซีสำหรับไมโครคอนโทรลเลอร์ MCS-51 พร้อมโปรแกรมภาษาซี. กรุงเทพมหานคร: อีทีที,2543.

ภาคผนวก

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

ภาคผนวก ก

โปรแกรมคำนวณแรงบิดและกำลังงานประกอบคำอธิบาย

```
#include<reg51.h>          //ประกาศให้คอมไพเลอร์รู้จักรีจิสเตอร์และบิตควบคุมพื้นฐานของไมโครคอนโทรลเลอร์ MCS 51
#include<stdio.h>          // ประกาศเพื่อใช้งานฟังก์ชันที่เกี่ยวกับการสื่อสารพอร์ตอนุกรม
#define pi 3.1415926       // กำหนดให้ pi แทนตัวเลข 3.1415926

sbit e=P3^6;               //กำหนดให้ตัวแปรชื่อ e แทนการเข้าถึงพอร์ต 3 บิต 6(P3.6)ของ AT89S8252
sbit rs=P3^7;              //กำหนดให้ตัวแปรชื่อ rs แทนการเข้าถึงพอร์ต 3 บิต 7(P3.7)ของ AT89S8252
sbit zero=P3^3;            //กำหนดให้ตัวแปรชื่อ zero แทนการเข้าถึงพอร์ต 3 บิต 3(P3.3)ของ AT89S8252
sbit mode=P3^2;            //กำหนดให้ตัวแปรชื่อ mode แทนการเข้าถึงพอร์ต 3 บิต 2(P3.2)ของ AT89S8252
unsigned int tick,show,speed;   //เป็นการประกาศตัวแปร tick,show,speed ให้เป็นตัวแปรแบบ unsigned int โดยตัวแปร tick
                           จะใช้ในโปรแกรมตอบสนองการอินเทอร์รัพท์ ตัวแปร show เป็นตัวแปรที่ใช้ในการแสดงผล ตัว
                           แปร speed สำหรับใช้แสดงผลค่าความเร็วรอบ
sbit gray0=P2^0;           // กำหนดให้ตัวแปรชื่อ gray0 แทนการเข้าถึงพอร์ต 2 บิต 0(P2.0)ของ AT89S8252 ค่าในตัวแปร
                           gray0 จะเป็น 1 หรือ 0 ขึ้นกับ สถานะลอจิกที่บิทวงในสุดของตัวเข้ารหัส gray code
sbit gray1=P2^1;           // กำหนดให้ตัวแปรชื่อ gray1 แทนการเข้าถึงพอร์ต 2 บิต 1(P2.1)ของ AT89S8252 ค่าในตัวแปร
                           gray1 จะเป็น 1 หรือ 0 ขึ้นกับ สถานะลอจิกบิทที่2 ของตัวเข้ารหัส gray code
sbit gray2=P2^2;           // กำหนดให้ตัวแปรชื่อ gray2 แทนการเข้าถึงพอร์ต 2 บิต 2(P2.2)ของ AT89S8252 ค่าในตัวแปร
                           gray0 จะเป็น 1 หรือ 0 ขึ้นกับ สถานะลอจิกบิทที่3 ของตัวเข้ารหัส gray code
sbit gray3=P2^3;           // กำหนดให้ตัวแปรชื่อ gray3 แทนการเข้าถึงพอร์ต 2 บิต 3(P2.3)ของ AT89S8252 ค่าในตัวแปร
                           gray0 จะเป็น 1 หรือ 0 ขึ้นกับ สถานะลอจิกบิทที่4 ของตัวเข้ารหัส gray code
sbit gray4=P2^4;           // กำหนดให้ตัวแปรชื่อ gray4 แทนการเข้าถึงพอร์ต 2 บิต 4(P2.4)ของ AT89S8252 ค่าในตัวแปร
                           gray0 จะเป็น 1 หรือ 0 ขึ้นกับ สถานะลอจิกบิทที่5 ของตัวเข้ารหัส gray code
sbit gray5=P2^5;           // กำหนดให้ตัวแปรชื่อ gray5 แทนการเข้าถึงพอร์ต 2 บิต 5(P2.5)ของ AT89S8252 ค่าในตัวแปร
                           gray0 จะเป็น 1 หรือ 0 ขึ้นกับ สถานะลอจิกบิทที่6 ของตัวเข้ารหัส gray code
sbit gray6=P2^6;           // กำหนดให้ตัวแปรชื่อ gray6 แทนการเข้าถึงพอร์ต 2 บิต 6(P2.6)ของ AT89S8252 ค่าในตัวแปร
                           gray0 จะเป็น 1 หรือ 0 ขึ้นกับ สถานะลอจิกบิทที่7 ของตัวเข้ารหัส gray code
sbit gray7=P2^7;           // กำหนดให้ตัวแปรชื่อ gray7 แทนการเข้าถึงพอร์ต 2 บิต 7(P2.7)ของ AT89S8252 ค่าในตัวแปร
                           gray0 จะเป็น 1 หรือ 0 ขึ้นกับ สถานะลอจิกบิทที่8 ของตัวเข้ารหัส gray code
sbit gray8=P1^7;           // กำหนดให้ตัวแปรชื่อ gray8 แทนการเข้าถึงพอร์ต 1 บิต 7(P1.7)ของ AT89S8252 ค่าในตัวแปร
                           gray0 จะเป็น 1 หรือ 0 ขึ้นกับ สถานะลอจิกบิทที่9 ของตัวเข้ารหัส gray code
sbit gray9=P1^6;           // กำหนดให้ตัวแปรชื่อ gray9 แทนการเข้าถึงพอร์ต 1 บิต 6(P1.6)ของ AT89S8252 ค่าในตัวแปร
                           gray9 จะเป็น 1 หรือ 0 ขึ้นกับสถานะลอจิกที่บิทวงนอกสุดของตัวเข้ารหัส gray code

bit binary0;               //กำหนดให้ตัวแปรชื่อ binary0 เป็นตัวแปรขนาด 1 บิท โดย binary0 จะใช้แทนค่าลอจิก 1 หรือ
                           0 ที่บิทวงในสุดของรหัสฐานสอง
```

```
 bit binary1;              //กำหนดให้ตัวแปรชื่อ binary1 เป็นตัวแปรขนาด 1 บิท โดย binary1 จะใช้แทนค่าลอจิก 1 หรือ
                           0 ในบิทที่ 2 ของรหัสฐานสอง
 bit binary2;              //กำหนดให้ตัวแปรชื่อ binary2 เป็นตัวแปรขนาด 1 บิท โดย binary1 จะใช้แทนค่าลอจิก 1 หรือ
                           0 ในบิทที่ 3 ของรหัสฐานสอง
bit binary3;               //กำหนดให้ตัวแปรชื่อ binary3 เป็นตัวแปรขนาด 1 บิท โดย binary1 จะใช้แทนค่าลอจิก 1 หรือ
                           0 ในบิทที่ 4 ของรหัสฐานสอง

 bit binary4;              //กำหนดให้ตัวแปรชื่อ binary4 เป็นตัวแปรขนาด 1 บิท โดย binary1 จะใช้แทนค่าลอจิก 1 หรือ
                           0 ในบิทที่ 5 ของรหัสฐานสอง
 bit binary5;              //กำหนดให้ตัวแปรชื่อ binary5 เป็นตัวแปรขนาด 1 บิท โดย binary1 จะใช้แทนค่าลอจิก 1 หรือ
                           0 ในบิทที่ 6 ของรหัสฐานสอง
 bit binary6;              //กำหนดให้ตัวแปรชื่อ binary6 เป็นตัวแปรขนาด 1 บิท โดย binary1 จะใช้แทนค่าลอจิก 1 หรือ
                           0 ในบิทที่ 7 ของรหัสฐานสอง
 bit binary7;              //กำหนดให้ตัวแปรชื่อ binary7 เป็นตัวแปรขนาด 1 บิท โดย binary1 จะใช้แทนค่าลอจิก 1 หรือ
                           0 ในบิทที่ 8 ของรหัสฐานสอง
 bit binary8;              //กำหนดให้ตัวแปรชื่อ binary8 เป็นตัวแปรขนาด 1 บิท โดย binary1 จะใช้แทนค่าลอจิก 1 หรือ
                           0 ในบิทที่ 9 ของรหัสฐานสอง
 bit binary9;              //กำหนดให้ตัวแปรชื่อ binary9 เป็นตัวแปรขนาด 1 บิท โดย binary1 จะใช้แทนค่าลอจิก 1 หรือ
                           0 ที่บิทวงนอกสุดของรหัสฐานสอง

void delay(int tik)        //ฟังก์ชันหน่วงเวลา ทำได้โดยใช้คำสั่ง for ซ้อน for โดยการรับค่าจากพารามิเตอร์ tik เข้ามาเพื่อ
{                          เทียบกับตัวแปรในการวนลูป ระยะเวลาในการหน่วงเวลาจะขึ้นอยู่กับค่าตัวแปร tik ที่ส่งผ่านไป
      int i,j;             ให้กับฟังก์ชันและจำนวนรอบของตัวแปร j ในการวนลูป
      for(i=0;i<tik;i++)
      for(j=0;j<250;j++);
}
void lcd_command(unsigned char com) //ฟังก์ชันเพื่อใช้ส่งคำสั่งให้กับโมดูล LCD
{
rs=0;                      //กำหนดให้ข้อมูลที่ขา D0-D7 ของ LCD เป็นข้อมูลคำสั่ง
e=1;                       //เตรียมสร้างสัญญาณพัลส์ขอบขาลงป้อนที่ขา E
P0=com;                    //ส่งข้อมูลคำสั่งออกไปที่ขา D0-D7
delay(5);                  //หน่วงเวลาเพื่อให้โมดูล LCD ประมวลผล
e=0;                       //ป้อนสัญญาณพัลส์เรียบร้อย
delay(5);                  //หน่วงเวลาเพื่อให้โมดูล LCD ประมวลผล
}
void lcd_text(unsigned char text)          //ฟังก์ชันเพื่อใช้ส่งข้อความแสดงผลให้โมดูล LCD
{
rs=1;                      //กำหนดให้ข้อมูลที่ขา D0-D7 ของ LCD เป็นข้อมูลแสดงผล
e=1;                       //เตรียมสร้างสัญญาณพัลส์ขอบขาลงป้อนที่ขา E
```

```
P0=text;                          //ส่งข้อมูลแสดงผลออกไปที่ขา D0-D7
delay(5);                         //หน่วงเวลาเพื่อให้โมดูล LCD ประมวลผล
e=0;                              //ป้อนสัญญาณพัลส์เรียบร้อย
delay(5);                         //หน่วงเวลาเพื่อให้โมดูล LCD ประมวลผล
}

void lcd_init()                   //ฟังก์ชันกำหนดการทำงานเริ่มต้นให้กับโมดูล LCD
{
delay(500);
lcd_command(0x38);                //ส่งค่า 0x38 ผ่านฟังก์ชัน lcd_command เพื่อกำหนดให้โมดูล LCD แสดงผล 2
                                  บรรทัด ความละเอียด 5x7 จุด โดยติดต่อในแบบ 8 บิต
lcd_command(0x0C);                //ส่งค่า 0x0C ผ่านฟังก์ชัน lcd_command เพื่อกำหนดให้เปิดหน้าจอแสดงผล โดยไม่
                                  แสดง เคอร์เซอร์
lcd_command(0x01);                //ส่งค่า 0x01 ผ่านฟังก์ชัน lcd_command เพื่อสั่งให้ลบข้อความบนหน้าจอแสดงใน
}                                 ขณะนั้นให้หมด

float power(float x,float y)      //ฟังก์ชันหาค่าผลลัพธ์ของการยกกำลัง x,y
{
   float i,pow;                   //กำหนดให้ตัวแปร i เป็นตัวแปรในการตรวจสอบเงื่อนไข ตัวแปร pow แทนค่าผลลัพท์จากการ
                                  ยกกำลัง x,y
if(y>0)                           //ตรวจสอบเงื่อนไข y>0 ถ้าใช่ จะคำนวณหาผลลัพธ์ของ x ยกกำลัง y ถ้าไม่ใช่ ผลลัพธ์จากการ
{                                 ยกกำลังเท่ากับ 1
  pow=1;                          //กำหนดค่าเริ่มต้นให้กับตัวแปร pow เท่ากับ 1
   for(i=0;i<y;i++)               //กำหนดให้ทำงานแบบวนรอบด้วยการคูณค่า x เข้ากับตัวแปร pow เป็นจำนวน y ครั้ง
        {
              pow*=x;                     //คูณค่า pow ด้วยค่า x แล้วนำผลลัพธ์ที่ได้เก็บไว้ที่ pow
        }
   }
else{                             //ถ้าตรวจสอบเงื่อนไขแล้วเป็นเท็จ ให้ผลลัพธ์จากการยกกำลังเท่ากับ 1
   pow=1;                         //กำหนดค่าให้ตัวแปร pow เท่ากับ 1
      }
    return(pow);                  //ส่งค่าของ pow ออกไปเป็นค่าผลลัพธ์
}

void main(void)                   //ฟังก์ชันหลักของโปรแกรม
{

 unsigned char x1,x2,x3,x4,count;          //กำหนดให้ตัวแปร x1,x2,x3,x4 เป็นตัวแปรที่ใช้ในการแสดงผลตัวเลข
                                           ,ตัวแปร count แทนค่าความเร็วรอบต่อเวลา 5 วินาที
```

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

```
float angle,decimal,buffer ,rad,torque,kumlung,r,k,x; //กำหนดให้ตัวแปร angle แทนค่ามุมหมุนของรอก, ตัวแปร
                                                      decimal ใช้เก็บค่าตำแหน่งของรอกในรูปเลขฐานสิบ, ตัวแปร
                                                      buffer เก็บค่าตำแหน่งเริ่มต้นของรอกในรูปเลขฐานสิบ,ตัวแปร rad แทน
                                                      ค่าความเร็วรอบในหน่วย rad/s ,torque แทนค่าแรงบิด , ตัวแปร
                                                      kumlung แทนค่ากำลังงาน ,ตัวแปร r คือ รัศมีของรอก,ตัวแปร k คือ ค่า
                                                      ความแข็งตึงสปริง,ตัวแปร x คือ ระยะยืดสปริงที่สัมพันธ์กับมุมหมุนรอก
  count=0;                                            //กำหนดค่าเริ่มต้นให้กับตัวแปร count เท่ากับ 0
  angle=0;                                            //กำหนดค่าเริ่มต้นให้กับตัวแปร angle เท่ากับ 0
  decimal=0;                                          //กำหนดค่าเริ่มต้นให้กับตัวแปร decimal เท่ากับ 0
  buffer=0;                                           //กำหนดค่าเริ่มต้นให้กับตัวแปร buffer เท่ากับ 0
  show=0;                                             //กำหนดค่าเริ่มต้นให้กับตัวแปร show เท่ากับ 0
   rad=0;                                             //กำหนดค่าเริ่มต้นให้กับตัวแปร rad เท่ากับ 0
   torque=0;                                          //กำหนดค่าเริ่มต้นให้กับตัวแปร torque เท่ากับ 0
   kumlung=0;                                         //กำหนดค่าเริ่มต้นให้กับตัวแปร kumlung เท่ากับ 0

  TMOD=0x61;              //กำหนดค่า 0x61 ให้กับรีจิสเตอร์ TMOD เพื่อตั้งค่าไทเมอร์ 0 ทำงานเป็นตัวจับเวลาในโหมด
                          และตั้งค่าไทเมอร์ 1 ทำงานเป็นตัวนับในโหมด 2
  TH0=0xF9;               //ตั้งค่าการนับเริ่มต้นให้กับรีจิสเตอร์ TH0 เท่ากับ 0xF9
  TL0=0xFF;               //ตั้งค่าการนับเริ่มต้นให้กับรีจิสเตอร์ TL0 เท่ากับ 0xFF
  TL1=0x00;               //ตั้งค่าการนับเริ่มต้นให้กับรีจิสเตอร์ TL1 เท่ากับ 0x00
   TR0=1;                 //ให้ไทเมอร์0 เริ่มต้นทำงาน
   TR1=1;                 //ให้ไทเมอร์1 เริ่มต้นทำงาน
    IE=0x82;              //กำหนดค่า 0x82 ให้กับรีจิสเตอร์ IE เพื่อให้อินเทอร์รัพท์จากไทเมอร์ 0 ได้

  lcd_init();             //เรียกใช้ฟังก์ชัน lcd_init เพื่อเริ่มต้นการทำงานของโมดูล LCD

while(1)
{

        binary0=gray0;    //เริ่มการแปลงจากรหัสเกรย์ไปเป็นรหัสฐานสองโดยการแทนค่า gray0 ลงใน binaary0
        if(gray1==1)              //ถ้า gray1 เท่ากับ 1 ค่าใน binary1 จะตรงข้ามกับค่า binary0
            binary1= ~binary0;
        else                      //ถ้า gray1 ไม่เท่ากับ 1 ค่าใน bianry1 จะเหมือนกับค่าใน binary0
            binary1=binary0;
        if(gray2==1)              //ถ้า gray2 เท่ากับ 1 ค่าใน binary2 จะตรงข้ามกับค่า binary1
            binary2= ~binary1;
        else                      //ถ้า gray2 ไม่เท่ากับ 1 ค่าใน bianry2 จะเหมือนกับค่าใน binary1
            binary2=binary1;
        if(gray3==1)              //ถ้า gray3 เท่ากับ 1 ค่าใน binary3 จะตรงข้ามกับค่า binary2
```

```
    binary3= ~binary2;
else                        //ถ้า gray3 ไม่เท่ากับ 1 ค่าใน bianry3 จะเหมือนกับค่าใน binary2
    binary3=binary2;
if(gray4==1)                //ถ้า gray4 เท่ากับ 1 ค่าใน binary4 จะตรงข้ามกับค่า binary3
    binary4= ~binary3;
else                        //ถ้า gray4 ไม่เท่ากับ 1 ค่าใน bianry4 จะเหมือนกับค่าใน binary3
    binary4=binary3;


if(gray5==1)                //ถ้า gray5 เท่ากับ 1 ค่าใน binary5 จะตรงข้ามกับค่า binary4
    binary5= ~binary4;
else                        //ถ้า gray5 ไม่เท่ากับ 1 ค่าใน bianry5 จะเหมือนกับค่าใน binary4
    binary5=binary4;
if(gray6==1)                //ถ้า gray6 เท่ากับ 1 ค่าใน binary6 จะตรงข้ามกับค่า binary5
    binary6= ~binary5;
else                        //ถ้า gray6 ไม่เท่ากับ 1 ค่าใน bianry6 จะเหมือนกับค่าใน binary5
    binary6=binary5;
if(gray7==1)                //ถ้า gray7 เท่ากับ 1 ค่าใน binary7 จะตรงข้ามกับค่า binary6
    binary7= ~binary6;
else                        //ถ้า gray7 ไม่เท่ากับ 1 ค่าใน bianry7 จะเหมือนกับค่าใน binary6
    binary7=binary6;
if(gray8==1)                //ถ้า gray8 เท่ากับ 1 ค่าใน binary8 จะตรงข้ามกับค่า binary7
    binary8= ~binary7;
else                        //ถ้า gray8 ไม่เท่ากับ 1 ค่าใน bianry8 จะเหมือนกับค่าใน binary7
    binary8=binary7;
if(gray9==1)                //ถ้า gray9 เท่ากับ 1 ค่าใน binary9 จะตรงข้ามกับค่า binary8
    binary9= ~binary8;
else                        //ถ้า gray9 ไม่เท่ากับ 1 ค่าใน bianry9 จะเหมือนกับค่าใน binary8
    binary9=binary8;

decimal=0;                  //กำหนดค่าเริ่มต้นให้ decimal เท่ากับ 0

if(binary9==1)              //ทำการแปลงจากรหัสฐานสองเป็นเลขฐานสิบ โดยเริ่มเช็คที่ bianry9 ถ้ามีค่าเท่ากับ
decimal+=power(2,0);        1 จะเพิ่มค่า 2 ยกกำลัง 0 ให้กับตัวแปร decimal
else                        //ถ้า binary9 ไม่เท่ากับ 1 จะไม่มีการเพิ่มค่าให้กับตัวแปร decimal
decimal+=0;

if(binary8==1)              //ถ้า binary8 เท่ากับ 1 จะเพิ่มค่า 2 ยกกำลัง 1 ให้กับ decimal
decimal+=power(2,1);
```

```
else                        //ถ้า binary8 ไม่เท่ากับ 1  ค่าใน decimal ยังคงเท่าเดิม
decimal+=0;

if(binary7==1)              //ถ้า binary7 เท่ากับ 1 จะเพิ่มค่า 2 ยกกำลัง 2 ให้กับ decimal
decimal+=power(2,2);
else                        //ถ้า binary7 ไม่เท่ากับ 1  ค่าใน decimal ยังคงเท่าเดิม
decimal+=0;

if(binary6==1)              //ถ้า binary6 เท่ากับ 1 จะเพิ่มค่า 2 ยกกำลัง 3 ให้กับ decimal
decimal+=power(2,3);
else                        //ถ้า binary6 ไม่เท่ากับ 1  ค่าใน decimal ยังคงเท่าเดิม
decimal+=0;

if(binary5==1)              //ถ้า binary5 เท่ากับ 1 จะเพิ่มค่า 2 ยกกำลัง 4 ให้กับ decimal
decimal+=power(2,4);
else                        //ถ้า binary5 ไม่เท่ากับ 1  ค่าใน decimal ยังคงเท่าเดิม
decimal+=0;

if(binary4==1)              //ถ้า binary4  เท่ากับ 1 จะเพิ่มค่า 2 ยกกำลัง 5 ให้กับ decimal
decimal+=power(2,5);
else                        //ถ้า binary4ไม่เท่ากับ 1  ค่าใน decimal ยังคงเท่าเดิม
decimal+=0;

if(binary3==1)              //ถ้า binary3  เท่ากับ 1 จะเพิ่มค่า 2 ยกกำลัง 6 ให้กับ decimal
decimal+=power(2,6);
else                        //ถ้า binary3 ไม่เท่ากับ 1  ค่าใน decimal ยังคงเท่าเดิม
decimal+=0;

if(binary2==1)              //ถ้า binary2  เท่ากับ 1 จะเพิ่มค่า 2 ยกกำลัง 7 ให้กับ decimal
decimal+=power(2,7);
else                        //ถ้า binary2 ไม่เท่ากับ 1  ค่าใน decimal ยังคงเท่าเดิม
decimal+=0;

if(binary1==1)              //ถ้า binary1  เท่ากับ 1 จะเพิ่มค่า 2 ยกกำลัง 8 ให้กับ decimal
decimal+=power(2,8);
else                        //ถ้า binary1 ไม่เท่ากับ 1  ค่าใน decimal ยังคงเท่าเดิม
decimal+=0;

if(binary0==1)              //ถ้า binary0  เท่ากับ 1 จะเพิ่มค่า 2 ยกกำลัง 9 ให้กับ decimal
```

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

```
decimal+=power(2,9);
else                          //ถ้า binary0 ไม่เท่ากับ 1 ค่าใน decimal ยังคงเท่าเดิม
decimal+=0;


if(zero==0)                   //ตรวจสอบเงื่อนไข zero เท่ากับ 0 หรือไม่ ถ้าใช่ แสดงว่ามีการกดสวิทช์
{                             เซตศูนย์ โปรแกรมก็จะแทนค่าในตัวแปร decimal ลงในตัวแปร buffer
buffer=decimal;               ดั้งนั้น buffer จะเก็บค่าที่บอกถึงตำแหน่งของจานเข้ารหัสในขณะที่มี
}                             การกดสวิทช์ไว้


if(decimal>buffer)                  //ตรวจสอบเงื่อนไข decimal มากกว่า buffer หรือไม่ ถ้าใช่ มุม
  angle=((decimal-buffer)*0.351);   หมุนของรอกเท่ากับ decimal ลบด้วย buffer แล้วนำมาคูณด้วย
else                                0.351 ถ้าไม่ใช่ มุมหมุนของรอกเท่ากับ buffer ลบด้วย decimal
   angle=((buffer-decimal)*0.351);  แล้วนำมาคูณด้วย 0.351

 if(tick>=895)                //ตรวจสอบเงื่อนไข tick >= 895 หรือไม่ ถ้าใช่ แสดงว่าเกิดการอินเทอร์รัพท์
{                             ครบ 895 ครั้ง หรือจับเวลามาครบ 1 วินาทีแล้ว
    TR1=0;                    //ปิดไทเมอร์ 1 ให้หยุดการนับสัญญาณพัลส์
     tick=0;                  //ตั้งค่าให้กับตัวแปรตรวจสอบเงื่อนไข tick เท่ากับ 0
     count=TL1;               //แทนค่าใน TL1 ลงในตัวแปร count
     TL1=0x00;                //รีเซตค่า TL1 เท่ากับ 0x00
      TR1=1;                  //เปิดไทเมอร์1 เพื่อเริ่มต้นการนับใหม่
}

r=0.115;                      //กำหนดให้ตัวแปร r (รัศมีรอก)เท่ากับ 0.115 เมตร
k=4859 ;                      //กำหนดให้ตัวแปร k(ค่าคงที่สปริง) เท่ากับ 4859 N/mm
rad=count*0.25*pi;            //คำนวณหาความเร็วรอบเชิงมุมจากจำนวนพัลส์ที่นับได้
x=angle*pi/180*r*1000;        //คำนวณหาระยะยืดสปริงจากมุมหมุนของรอก
torque=((k*x/1000)+113.796)*r;//คำนวณหาค่าแรงบิดจากตัวแปร x,r,k
kumlung=rad*torque;           //คำนวณหาค่ากำลังจากตัวแปร rad,torque
speed=count*7.5;              //คำนวณความเร็วรอบในหน่วย รอบ/นาที โดยนำ count คูณกับ 7.5
                              แล้วนำค่าเก็บไว้ใน speed

if(mode==1)                   //ตรวจสอบเงื่อนไข mode เท่ากับ 1 หรือไม่ ถ้าใช่ แสดงว่าได้กดสวิทช์เพื่อ
{                             เลือกให้แสดงผล ค่าความเร็วรอบ และมุมหมุนของรอก ถ้าไม่ใช่ แสดงว่า
                              เลือกให้แสดงผล แรงบิดและกำลังงาน
x1=(speed/1000)|0x30;         //เอาตัวเลขหลักพันในตัวแปร speed เก็บไว้ที่ x1
```

```
x2=((speed%1000)/100)|0x30;          //เอาตัวเลขหลักร้อยในตัวแปร speed เก็บไว้ที่ x2
x3=(((speed%1000)%100)/10)|0x30;     //เอาตัวเลขหลักสิบในตัวแปร speed เก็บไว้ที่ x3
x4=(((speed%1000)%100)%10)|0x30;     //เอาตัวเลขหลักหน่วยในตัวแปร speed เก็บไว้ที่ x4

lcd_command(0x81);                   //กำหนดให้ LCD เริ่มแสดงผลที่ตำแหน่งที่2 บรรทัดบน
lcd_text('S');                       //ส่งตัวอักษร 'S' ไปแสดงที่ตำแหน่งที่2 บรรทัดบน
lcd_text('p');                       //ส่งตัวอักษร 'p' ไปแสดงที่ตำแหน่งที่3 บรรทัดบน
lcd_text('e');                       //ส่งตัวอักษร 'e' ไปแสดงที่ตำแหน่งที่4 บรรทัดบน
lcd_text('e');                       //ส่งตัวอักษร 'e' ไปแสดงที่ตำแหน่งที่5 บรรทัดบน
lcd_text('d');                       //ส่งตัวอักษร 'd' ไปแสดงที่ตำแหน่งที่6 บรรทัดบน
lcd_text(' ');                       //ส่งตัวอักษร ' ' ไปแสดงที่ตำแหน่งที่7 บรรทัดบน
lcd_text(' ');                       //ส่งตัวอักษร ' ' ไปแสดงที่ตำแหน่งที่8 บรรทัดบน
lcd_text(x1);                        //ส่งค่าตัวเลขใน x1 ไปแสดงที่ตำแหน่งที่9 บรรทัดบน
lcd_text(x2);                        //ส่งค่าตัวเลขใน x2 ไปแสดงที่ตำแหน่งที่10 บรรทัดบน
lcd_text(x3);                        //ส่งค่าตัวเลขใน x3 ไปแสดงที่ตำแหน่งที่11 บรรทัดบน
lcd_text(x4);                        //ส่งค่าตัวเลขใน x4 ไปแสดงที่ตำแหน่งที่12 บรรทัดบน
lcd_text(' ');                       //ส่งตัวอักษร ' ' ไปแสดงที่ตำแหน่งที่13 บรรทัดบน
lcd_text('R');                       //ส่งตัวอักษร 'R ' ไปแสดงที่ตำแหน่งที่14 บรรทัดบน
lcd_text('P');                       //ส่งตัวอักษร ' P' ไปแสดงที่ตำแหน่งที่15 บรรทัดบน
lcd_text('M');                       //ส่งตัวอักษร 'M ' ไปแสดงที่ตำแหน่งที่16 บรรทัดบน

show=angle*10;                       //นำค่าใน angle คูณกับ 10 แล้วเก็บในตัวแปร show เพื่อให้สามารถ
แสดง
                                     ผลในหลักทศนิยมอีก 1 ตำแหน่ง
x1=(show/1000)|0x30;                 //เอาตัวเลขหลักพันในตัวแปร show เก็บไว้ที่ x1
x2=((show%1000)/100)|0x30;           //เอาตัวเลขหลักร้อยในตัวแปร show เก็บไว้ที่ x2
x3=(((show%1000)%100)/10)|0x30;      //เอาตัวเลขหลักสิบในตัวแปร show เก็บไว้ที่ x3
x4=(((show%1000)%100)%10)|0x30;      //เอาตัวเลขหลักหน่วยในตัวแปร show เก็บไว้ที่ x4

lcd_command(0xC1);                   //กำหนดให้ LCD เริ่มแสดงผลที่ตำแหน่งที่2 บรรทัดล่าง
lcd_text('A');                       //ส่งตัวอักษร 'A' ไปแสดงที่ตำแหน่งที่2 บรรทัดล่าง
lcd_text('n');                       //ส่งตัวอักษร 'n' ไปแสดงที่ตำแหน่งที่3 บรรทัดล่าง
lcd_text('g');                       //ส่งตัวอักษร 'g' ไปแสดงที่ตำแหน่งที่4 บรรทัดล่าง
lcd_text('l');                       //ส่งตัวอักษร 'l' ไปแสดงที่ตำแหน่งที่5 บรรทัดล่าง
lcd_text('e');                       //ส่งตัวอักษร 'e' ไปแสดงที่ตำแหน่งที่6 บรรทัดล่าง
lcd_text(' ');                       //ส่งตัวอักษร ' ' ไปแสดงที่ตำแหน่งที่7 บรรทัดล่าง
lcd_text(x1);                        //ส่งค่าตัวเลขใน x1 ไปแสดงที่ตำแหน่งที่8 บรรทัดล่าง
lcd_text(x2);                        //ส่งค่าตัวเลขใน x2 ไปแสดงที่ตำแหน่งที่9 บรรทัดล่าง
lcd_text(x3);                        //ส่งค่าตัวเลขใน x3 ไปแสดงที่ตำแหน่งที่10 บรรทัดล่าง
```

```
lcd_text('.');                    //ส่งตัวอักษร '. ' ไปแสดงที่ตำแหน่งที่11 บรรทัดล่าง
lcd_text(x4);                     //ส่งค่าตัวเลขใน x4 ไปแสดงที่ตำแหน่งที่12 บรรทัดล่าง
lcd_text(' ');                    //ส่งตัวอักษร ' ' ไปแสดงที่ตำแหน่งที่13 บรรทัดล่าง
lcd_text('D');                    //ส่งตัวอักษร ' D' ไปแสดงที่ตำแหน่งที่14 บรรทัดล่าง
lcd_text('E');                    //ส่งตัวอักษร ' E' ไปแสดงที่ตำแหน่งที่15 บรรทัดล่าง
lcd_text('G');                    //ส่งตัวอักษร ' G' ไปแสดงที่ตำแหน่งที่16 บรรทัดล่าง

}
else                              ถ้าตรวจสอบเงื่อนไขแล้วเป็นเท็จ ให้แสดงผลแรงบิดและกำลังงาน
{
show=torque*100;                  //นำค่าใน torque คูณกับ 100 แล้วเก็บในตัวแปร show เพื่อให้สามารถ
                                  แสดงผลในหลักทศนิยมเพิ่มอีก 2 ตำแหน่ง
x1=(show/1000)|0x30;              //เอาตัวเลขหลักพันในตัวแปร show เก็บไว้ที่ x1
x2=((show%1000)/100)|0x30;        //เอาตัวเลขหลักร้อยในตัวแปร show เก็บไว้ที่ x2
x3=(((show%1000)%100)/10)|0x30;   //เอาตัวเลขหลักสิบในตัวแปร show เก็บไว้ที่ x3
x4=(((show%1000)%100)%10)|0x30;   //เอาตัวเลขหลักหน่วยในตัวแปร show เก็บไว้ที่ x4

lcd_command(0x80);                //กำหนดให้ LCD เริ่มแสดงผลที่ตำแหน่งแรก บรรทัดบน
lcd_text('T');                    //ส่งตัวอักษร 'T' ไปแสดงที่ตำแหน่งแรก บรรทัดบน
lcd_text('o');                    //ส่งตัวอักษร 'o' ไปแสดงที่ตำแหน่งที่2 บรรทัดบน
lcd_text('r');                    //ส่งตัวอักษร 'r' ไปแสดงที่ตำแหน่งที่3 บรรทัดบน
lcd_text('q');                    //ส่งตัวอักษร 'q' ไปแสดงที่ตำแหน่งที่4 บรรทัดบน
lcd_text('u');                    //ส่งตัวอักษร 'u' ไปแสดงที่ตำแหน่งที่5 บรรทัดบน
lcd_text('e');                    //ส่งตัวอักษร 'e' ไปแสดงที่ตำแหน่งที่6 บรรทัดบน
lcd_text(' ');                    //ส่งตัวอักษร ' ' ไปแสดงที่ตำแหน่งที่7 บรรทัดบน
lcd_text(x1);                     //ส่งค่าตัวเลขใน x1 ไปแสดงที่ตำแหน่งที่8 บรรทัดบน
lcd_text(x2);                     //ส่งค่าตัวเลขใน x2 ไปแสดงที่ตำแหน่งที่9 บรรทัดบน
lcd_text('.');                    //ส่งตัวอักษร '.' ไปแสดงที่ตำแหน่งที่10 บรรทัดบน
lcd_text(x3);                     //ส่งค่าตัวเลขใน x3 ไปแสดงที่ตำแหน่งที่11 บรรทัดบน
lcd_text(x4);                     //ส่งค่าตัวเลขใน x4 ไปแสดงที่ตำแหน่งที่12 บรรทัดบน
lcd_text(' ');                    //ส่งตัวอักษร ' ' ไปแสดงที่ตำแหน่งที่13 บรรทัดบน
lcd_text('N');                    //ส่งตัวอักษร 'N' ไปแสดงที่ตำแหน่งที่14 บรรทัดบน
lcd_text('.');                    //ส่งตัวอักษร '.' ไปแสดงที่ตำแหน่งที่15 บรรทัดบน
lcd_text('M');                    //ส่งตัวอักษร 'M' ไปแสดงที่ตำแหน่งที่16 บรรทัดบน


show=kumlung;                     //แทนค่าในตัวแปร kumlung ลงในตัวแปร show
x1=(show/1000)|0x30;              //เอาตัวเลขหลักพันในตัวแปร show เก็บไว้ที่ x1
x2=((show%1000)/100)|0x30;        //เอาตัวเลขหลักร้อยในตัวแปร show เก็บไว้ที่ x2
```

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

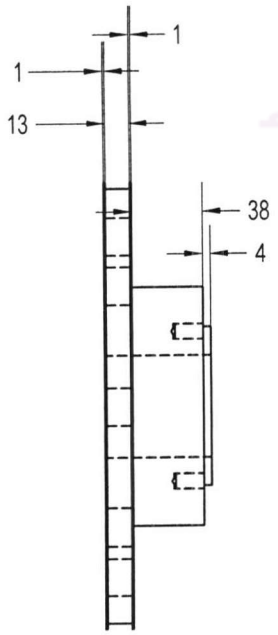
```
        x3=(((show%1000)%100)/10)|0x30;     //เอาตัวเลขหลักสิบในตัวแปร show เก็บไว้ที่ x3
        x4=(((show%1000)%100)%10)|0x30;     //เอาตัวเลขหลักหน่วยในตัวแปร show เก็บไว้ที่ x4

        lcd_command(0xC1);                  //กำหนดให้ LCD เริ่มแสดงผลที่ตำแหน่งที่2 บรรทัดล่าง
        lcd_text('P');                      //ส่งตัวอักษร 'P' ไปแสดงที่ตำแหน่งที่2 บรรทัดล่าง
        lcd_text('o');                      //ส่งตัวอักษร 'o' ไปแสดงที่ตำแหน่งที่3 บรรทัดล่าง
        lcd_text('w');                      //ส่งตัวอักษร 'w' ไปแสดงที่ตำแหน่งที่4 บรรทัดล่าง
        lcd_text('e');                      //ส่งตัวอักษร 'e' ไปแสดงที่ตำแหน่งที่5 บรรทัดล่าง
        lcd_text('r');                      //ส่งตัวอักษร 'r' ไปแสดงที่ตำแหน่งที่6 บรรทัดล่าง
        lcd_text(' ');                      //ส่งตัวอักษร ' ' ไปแสดงที่ตำแหน่งที่7 บรรทัดล่าง
        lcd_text(x1);                       //ส่งค่าตัวเลขใน x1 ไปแสดงที่ตำแหน่งที่8 บรรทัดล่าง
        lcd_text(x2);                       //ส่งค่าตัวเลขใน x2 ไปแสดงที่ตำแหน่งที่9 บรรทัดล่าง
        lcd_text(x3);                       //ส่งค่าตัวเลขใน x3 ไปแสดงที่ตำแหน่งที่10 บรรทัดล่าง
        lcd_text(x4);                       //ส่งค่าตัวเลขใน x4 ไปแสดงที่ตำแหน่งที่11 บรรทัดล่าง
        lcd_text(' ');                      //ส่งตัวอักษร ' ' ไปแสดงที่ตำแหน่งที่12 บรรทัดล่าง
        lcd_text('W');                      //ส่งตัวอักษร ' W' ไปแสดงที่ตำแหน่งที่13 บรรทัดล่าง
        lcd_text('A');                      //ส่งตัวอักษร 'A' ไปแสดงที่ตำแหน่งที่14 บรรทัดล่าง
        lcd_text('T');                      //ส่งตัวอักษร 'T' ไปแสดงที่ตำแหน่งที่15 บรรทัดล่าง
        lcd_text('T');                      //ส่งตัวอักษร 'T' ไปแสดงที่ตำแหน่งที่16 บรรทัดล่าง
        }


 }
}

void timer0_service() interrupt 1           //โปรแกรมตอบสนองการอินเทอร์รัปต์ของไทเมอร์ 1
{
   tick=tick+1;                     //เพิ่มค่าการนับรอบโอเวอร์โฟลว อีก 1 โดยเพิ่มค่า tick ขึ้นหนึ่งค่า
   TH0=0xF9;                        //รีโหลดค่าเริ่มต้นให้ตัวจับเวลา เพื่อให้ช่วงเวลาการนับเป็น 1 มิลลิวินาที
   TL0=0xFF;
}
```

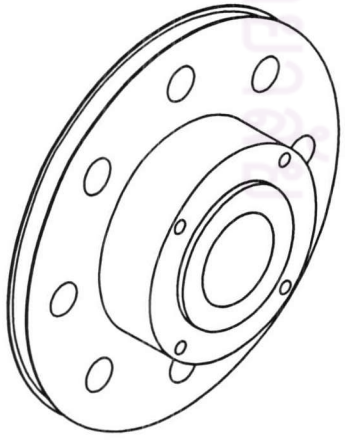
ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

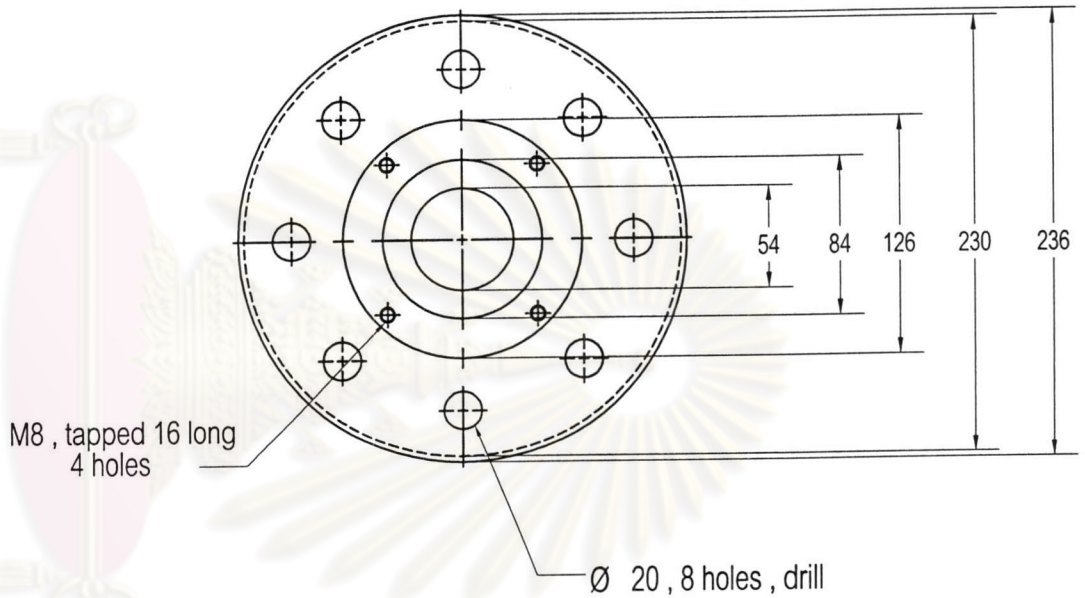
ภาคผนวก ข

แบบรายละเอียดรอกและสปริง

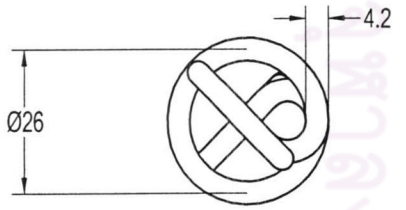
ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

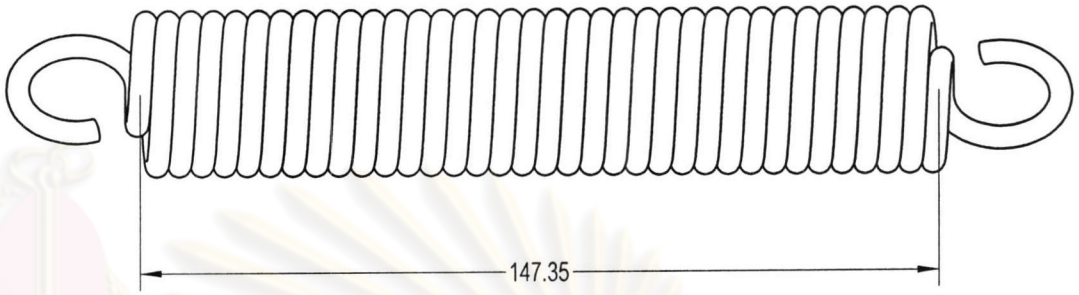

1
1
13
38
4

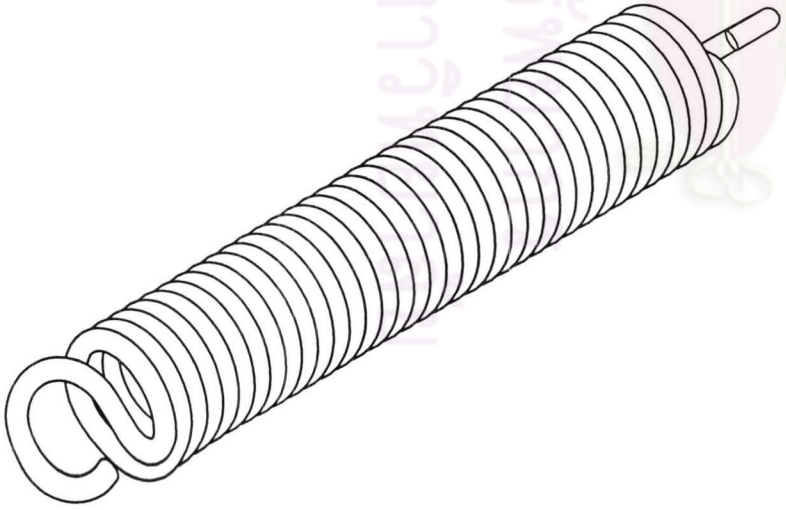



54
84
126
230
236
M8 , tapped 16 long
4 holes
Ø 20 , 8 holes , drill

รอก





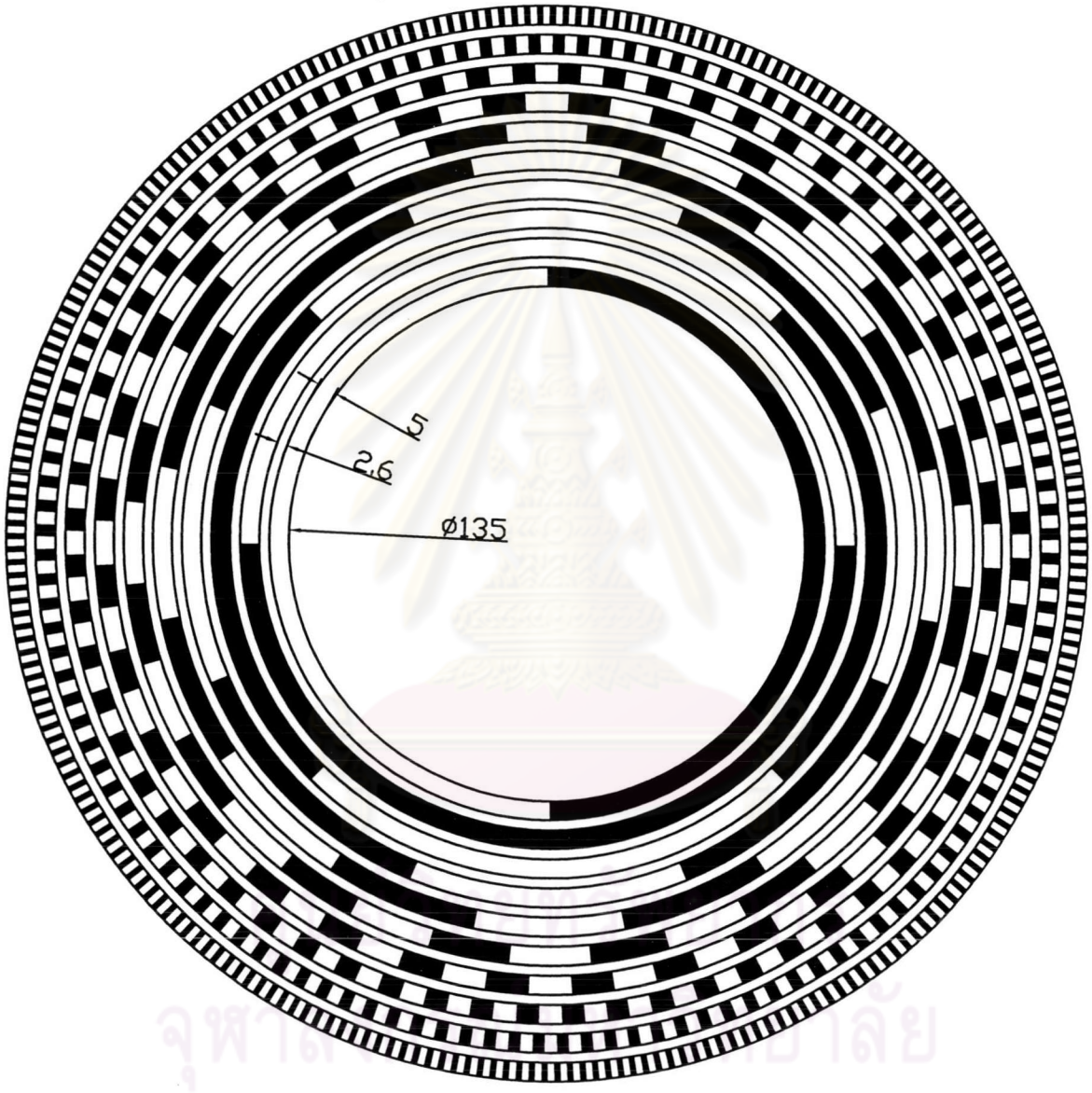


สปริง

ภาคผนวก ค

แบบรายละเอียดตัวเข้ารหัส

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย



รหัสเกรย์ความละเอียด 10 บิท

ภาคผนวก ง

รายละเอียดของบอร์ดไมโครคอนโทรลเลอร์

- CPU เบอร์ AT89S8252
- XTAL 18.432 MHz
- ROM 8 Kbyte Internal
- EEPROM 2 Kbyte
- มีคุณสมบัติ RS232
- มีชุด Power Supply
- มีชุด ISP Download

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

ภาคผนวก จ

คุณลักษณะของ CPU (AT89S8252)

- Compatible with MCS-51 TM Products
- 8 K of In-system Reprogrammable Downloadable Flash Memory
  - SPI Serial Interface for Program Downloading
  - Endurance : 1,000 Write/Erase Cycles
- 2 Kbytes EEPROM

  - Endurance : 100,000 Write/Erase Cycles
- 4.0 V to 6.0 V operating range
- Fully Static Operation : 0 Hz to 24 Hz
- Three-Level Program Memory Lock
- 256 x 8 Bit Internal Ram
- 32 Programmable I/O Line
- Nine Interrupt Sources
- Programmable UART Serial Channel
- SPI Serial Interface
- Low Power Idle and Power Down Modes
- Interrupt Recovery From Power Down
- Programmable Watchdog Timer
- Dual Data pointer
- Power Off Flag

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

ประวัติผู้เขียนวิทยานิพนธ์

นาย วิศิษฏ์ วิพัฒน์เกษมสุข เกิดวันที่ 30 มกราคม พ.ศ. 2523 จังหวัดนครราชสีมา สำเร็จการศึกษาวิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมเครื่องกล คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าพระนครเหนือ ในปีการศึกษา 2544 และเข้าศึกษาต่อในหลักสูตรวิศวกรรมศาสตรมหาบัณฑิต สาขาวิชาวิศวกรรมเครื่องกล คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย ในปีการศึกษา 2545

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย