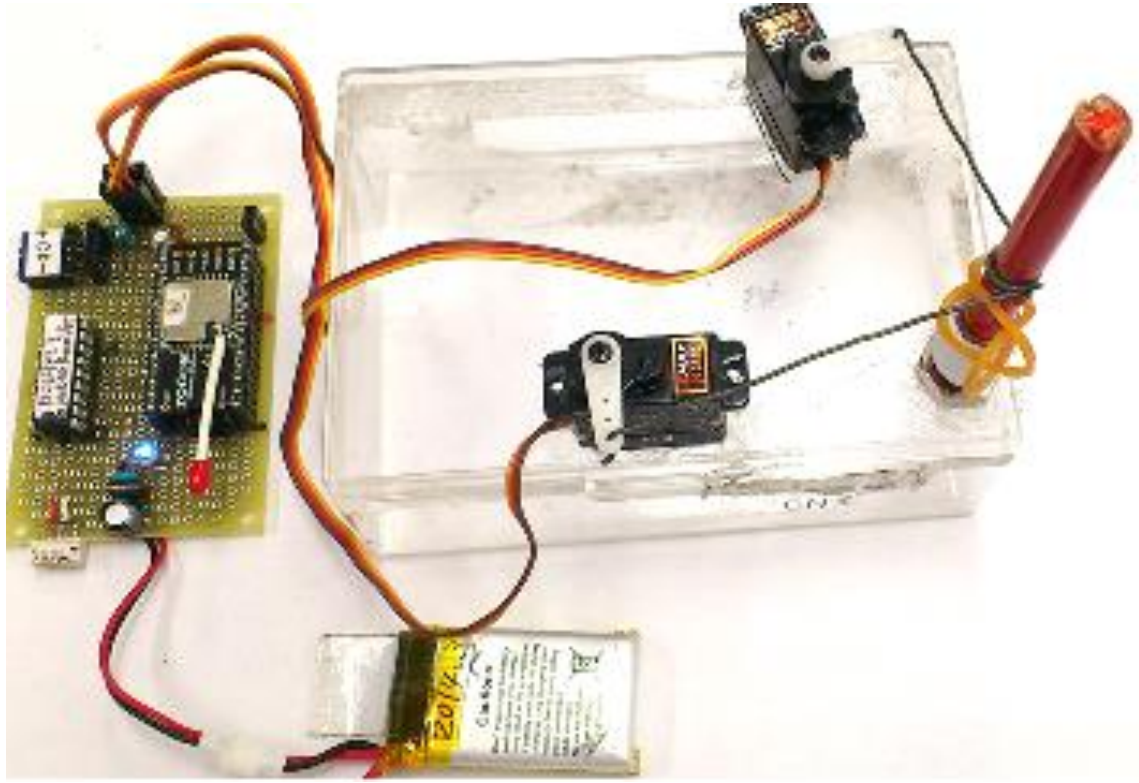
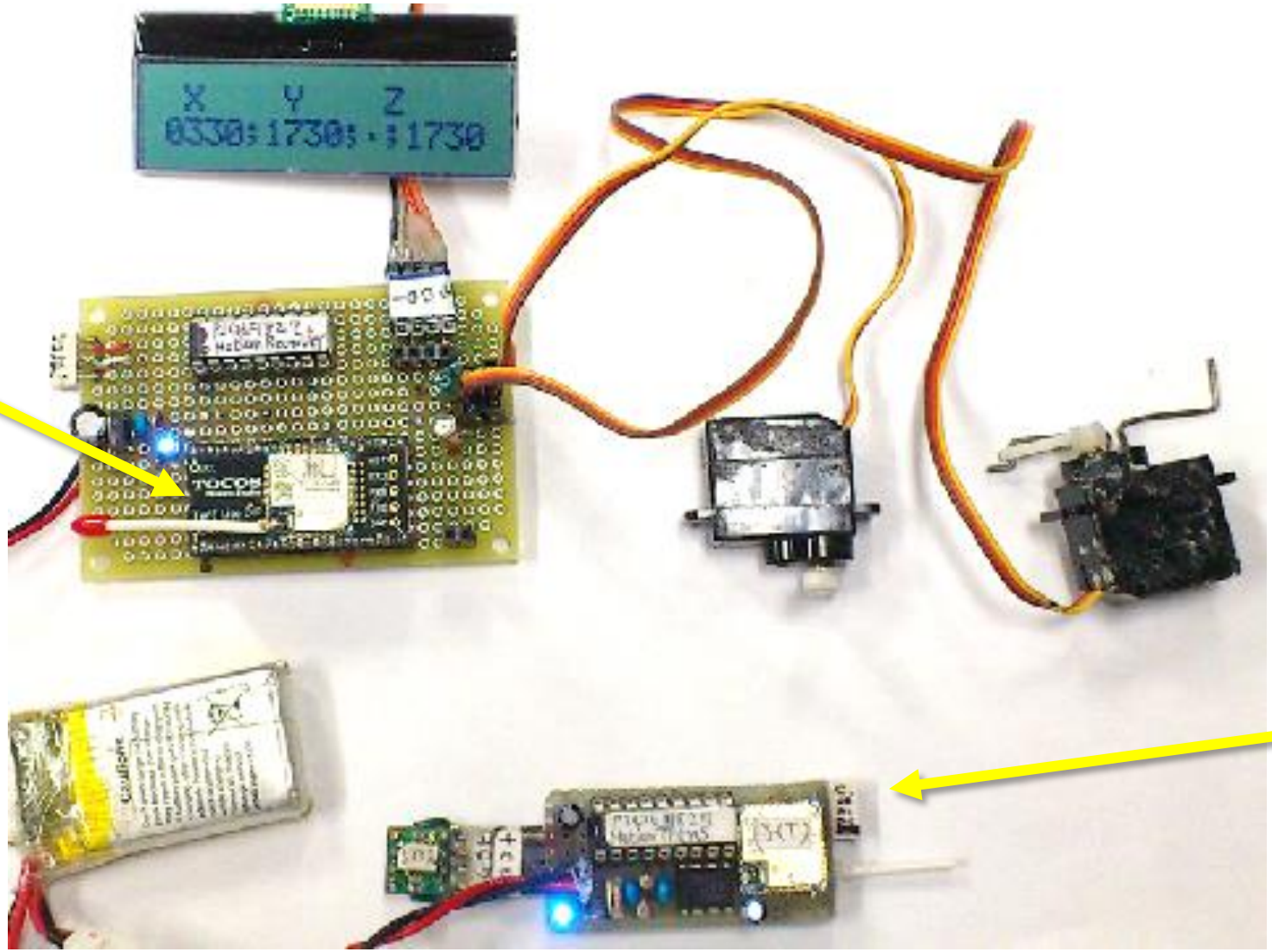
ワイヤレスモーションレシーバー



LCDなし

LCD付き：AQM1602A
磁束密度（ガウス），磁気偏角表示

ワイヤレスモーション　トランスミッター　＆　レシーバー



レシーバー

トランスミッター

ワイヤレスモーションレシーバー

PICKit3 ICSPコネクタ(5P)

PIC 16F1827

I2Cコネクタ

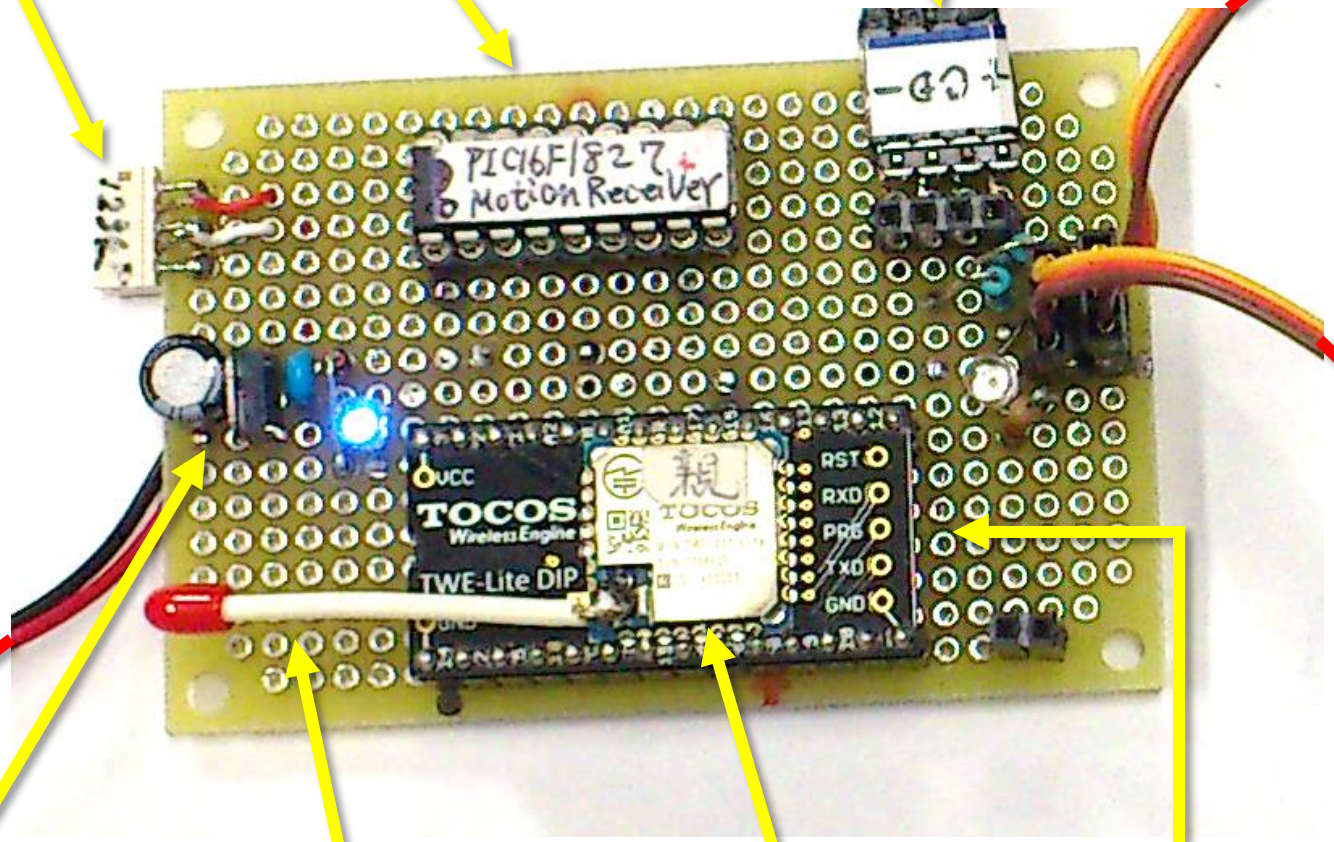
CH1
ラジコンサーボ

CH2
ラジコンサーボ

2つ目のPICは無線モジュールの下に配置し、ICSPは2つとも並列に配線した。プログラム時は、該当のPICのみ差しておく。

シリアルポートUARTも、2つのPICに並列に配線した。
チャンネルの分離は、プログラムで対応する。

サーボ信号の生成は，x, y 軸のデータをそれぞれサーボ信号（パルス幅：位置）に変換する．単純な構成とするために，ここでは各チャンネル専用のPICを配置した．



Lipoバッテリー

3.3Vレギュレータ
TA48M033S

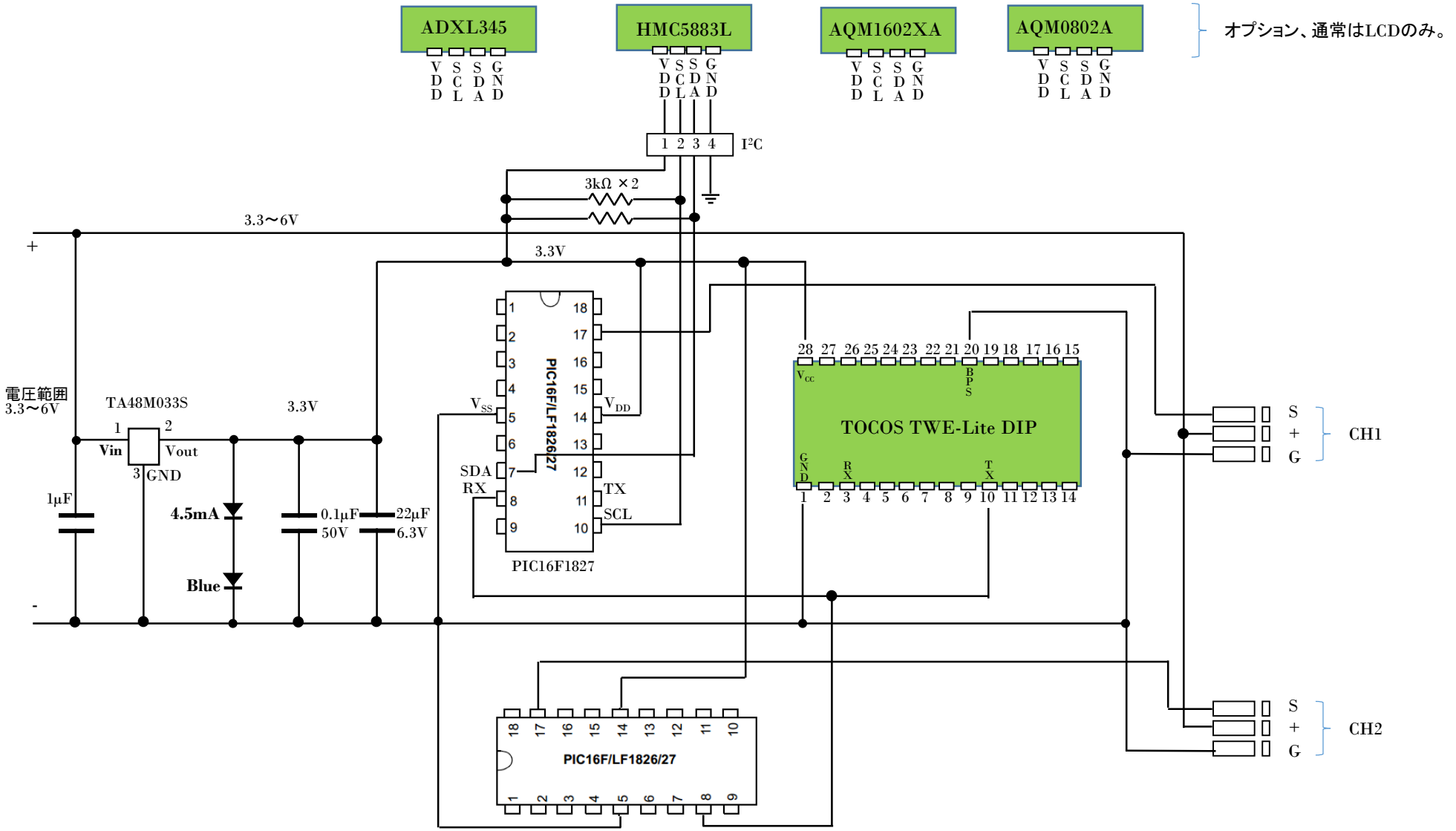
アンテナ
（空中線：無線技術用語）

無線モジュール
TOCOS　Wireless Engine
TWE-001-Lite

PIC 16F1827

（無線モジュールの下）

ワイヤレスモーションレシーバー　回路図



ワイヤレスモーションレシーブプログラム （サーボCH1の例）

```
// 2015.10. 24, 27  Motion Receiver CH1
// ===> LCD AQM1602XA  "Akizuki"
// Receive from "Motion Transmitter" by UART of TWE-Lite  M.T
#include<16F1827.h>
#include<math.h>
//#device ADC=10
#fuses INTRC_IO,NOWDT,NOPROTECT,NOMCLR,BROWNOUT
                      //
#use delay(clock= 16000000)
#use i2c(MASTER,SDA=PIN_B1,SCL=PIN_B4,FAST,NOFORCE_SW)
#use RS232(PARITY=N, BAUD=38400,XMIT=PIN_B5, RCV=PIN_B2)

// AQM016702XA
#define LCD_ADD_AQM       0x7C   // AQM0802 Slave Address
// AQM W/R Mode val
#define LCD_CMD_AQM       0x80   // Instruction Write Mode
#define LCD_DAT_AQM       0xC0   // Data Write Mode
#define line_1_AQM        0x00   // first line
#define line_2_AQM        0xC0   // second line = 0x80 + 0x40

void LCD_com_AQM(unsigned char cmd);
void LCD_Clear_AQM();
void LCD_Setline_AQM(unsigned char line);
void LCD_init_AQM();
void LCD_data_AQM(unsigned char data);
void LCD_space_AQM(int8 n);

// Ring Buffer
#define MAX 200
static int16 tail, head;
static char  buff[MAX];
static char  val_cnt;
static int16 val[3];
static char  CH;

#int_rda
void isr_rcv()
{
  char C;   // enqueue

  C = (0x7f & getc());
  buff[tail++] = C;
  if(tail>=MAX){
    tail = 0;
  }
}

//  ========================
void main(){
  char C;
  char title1[17] = "Motion Receiver";
  char title2[17] = " X   Y   Z";
  char cnt;

  LCD_init_AQM();
  LCD_Setline_AQM(line_1_AQM);
  for(cnt=0 ; cnt < 16; cnt++){
    LCD_data_AQM(0x30 + cnt);
  }
  delay_ms(900);
  LCD_Setline_AQM(line_2_AQM);
  for(cnt=0 ; cnt < 16; cnt++){
    LCD_data_AQM(0x40 + cnt);
  }
  delay_ms(900);
  LCD_Clear_AQM();
  LCD_Setline_AQM(line_1_AQM);
  for(cnt=0 ; cnt < 16; cnt++){
    LCD_data_AQM( title1[cnt]);
  }
  delay_ms(1000);
  LCD_Clear_AQM();
  LCD_Setline_AQM(line_1_AQM);
  for(cnt=0 ; cnt < 12; cnt++){
    LCD_data_AQM( title2[cnt]);
  }
  //delay_ms(2000);
  tail = head  = 0;
  val_cnt = 0;
  val[0]=val[1]=val[2]=0;
  CH = 0;    //Servo channel = CH1

  LCD_Setline_AQM(line_2_AQM);
  enable_interrupts(INT_RDA);
  enable_interrupts(GLOBAL);

  while(1) {
  // dequeue
    if(tail != head){
      C = buff[head++];
      if(head >= MAX){
        head = 0;
      }
      LCD_data_AQM(C);
      if(C==0x0d){
        LCD_Setline_AQM(line_2_AQM);
        output_high(PIN_A7);
        delay_us(1000);
        delay_us((val[CH]&0x7ff)/2);
        output_low(PIN_A7);
        val_cnt = 0;
        val[0]=val[1]=val[2]=0;
      }
      else{
        if( (0x30<=C)&&(C<=0x39)){
          val[val_cnt] = val[val_cnt]*10 + (C-0x30);
        }
        if(C==';'){ val_cnt++ ; }
      }
    }
  }
}

// AQM0802 & AQM1602XA //
void LCD_com_AQM(unsigned char cmd){
  int16 u_t=20;
  i2c_start();          //delay_ms(1);
  i2c_write(LCD_ADD_AQM); delay_us(u_t);    // Slave Address
  i2c_write(LCD_CMD_AQM); delay_us(u_t);    // Command Mode
  i2c_write(cmd);       delay_us(u_t);      // Send Command
  i2c_stop();           //delay_ms(1);
}
void LCD_Clear_AQM(){
  LCD_com_AQM(0x01);
  delay_ms(20);
}
void LCD_Setline_AQM(unsigned char line) {
  LCD_com_AQM(line);
}
void LCD_init_AQM(){
  delay_ms(60);
  LCD_com_AQM(0x38);     // Function set
  delay_us(30);
  LCD_com_AQM(0x39);     // Function set
  delay_us(30);
  LCD_com_AQM(0x14);     // Internal OSC frequency
  delay_us(30);
  LCD_com_AQM(0x70);     // Contrast
  delay_us(30);
  LCD_com_AQM(0x56);     // Power/ICON/Contrast Control
  delay_us(30);
  LCD_com_AQM(0x6c);     // Follow Control
  delay_ms(300);
  LCD_com_AQM(0x38);     // Function set
  delay_us(30);
  LCD_com_AQM(0x0c);     // Display ON/OFF Control
  delay_us(30);
  LCD_com_AQM(0x01);     // Clear Display
  delay_ms(2);
}
void LCD_data_AQM(unsigned char data){
  int16 u_t=26;

  i2c_start();          //delay_us(100);  //delay_ms(1);
  i2c_write(LCD_ADD_AQM); delay_us(u_t);    // Slave Address
  i2c_write(LCD_DAT_AQM); delay_us(u_t);    // Data Out Code
  i2c_write(data);      delay_us(u_t);      // Send Command
  i2c_stop();           //delay_us(100);  //delay_ms(1);
}
void LCD_space_AQM(int8 n){
  int8 cnt;
  cnt = 0;
  while(cnt++ < n){
    LCD_data_AQM(' ');
  }
}
```