```
//  AD8307 RF Power Meter V1.1  For Arduino UNO or MEGA
//  Orinal for i2cLCD by nobcha ©  04/23/2021
//  Connect AD8307 output on A1 port
//  added paralel LCD of K&L shield by nobcha ©  06/20/2021
//  DFROBOT Key function for ATT setting
//  UP:ATT++, DOWN:ATT--, SELECT:ATT set in EEPROM
//  Thanks JM1DPL@DOYOYON-GENSHOU  AD8307 slice number
// https://doyoyon.com/blog-entry-651.html
//  Thanks JH1UMV of Hanrangen for calculation
// https://lowreal.net/2016/03/14/1
//

#include <Arduino.h>

#include <LiquidCrystal.h>
#include <EEPROM.h>
#define ATT_AD 0 // EEPROM address for ATT data as int

#define btnRIGHT  1
#define btnUP     2
#define btnDOWN   3
#define btnLEFT   4
#define btnSELECT 5
#define btnNONE   0

#define KEY       0
#define MEASURE   1                    //

int ATT_val;
static const float SLOPE = 26.0;       //  I refered DOYOYON-GENSHOU
static const float INTERCEPT = -85.4;  //

int mode;
int key_value;
int old_key_value;
int key_accepted;
long Timer_LED = 2000;
long Old_millis;
long Timer_MEASURE = 500;
long Timer_KEY = 0;

// select the pins used on the LCD panel
LiquidCrystal lcd(8, 9, 4, 5, 6, 7);

String formatWatts(const float watts) {
  if (watts < 1e-3) {
    return String(watts * 1e6) + "uW";
  } else
  if (watts < 1e-1) {
```

```
    return String(watts * 1e3) + "mW";
  } else {
    return String(watts) + "W ";
  }
}

// Function SW determining by A0 port voltage
/Vcc-2k-btnRIGHT:GND,-btnUP:330,-btnDOWN:620,-btnLEFT:1k,-btnSELECT:3.3k
int function_key(void) {
  int adc_key_in = analogRead(A0); // define sw as voltage
  if (adc_key_in < 50)   return btnRIGHT;
  else if (adc_key_in < 250)  return btnUP;      // 2
  else if (adc_key_in < 450)  return btnDOWN;    // 3
  else if (adc_key_in < 650)  return btnLEFT;
  else if (adc_key_in < 850)  return btnSELECT;  // 5
  else return btnNONE;
}

void Timer(void){                                // Timing monitor
  long cor_time = millis() - Old_millis;         // Timer_value<=0 : time up
  if (Timer_LED > 0) Timer_LED = Timer_LED - cor_time;
  if (Timer_MEASURE > 0) Timer_MEASURE = Timer_MEASURE - cor_time;
  if (Timer_KEY > 0)   Timer_KEY = Timer_KEY - cor_time;
  Old_millis = millis();
}


void setup() {
  Serial.begin(9600);
  Serial.print("AD8307 RF Power Meter V1.1 ");

  pinMode(13, OUTPUT);           // Heartbeat LED

  lcd.begin(16, 2);              // start LCD 1602
  lcd.setCursor(0,0);
  lcd.print("AD8307RF-M V1.1 "); // print a opening message

// EEPROM data recover for ATT data
  EEPROM.get( ATT_AD, ATT_val);  // ATT data is recovered from EEPROM
  if (ATT_val > 30){             // If ATT_val >30 then error
    Serial.print("ATT EEPROM ERROR");
    ATT_val = 0;
    EEPROM.put( ATT_AD, ATT_val ); //
  }
  delay(1000);
  Serial.print("ATT=");
  Serial.print(ATT_val);
  lcd.setCursor(0, 1);
  lcd.print("ATT=");
```

```
 lcd.print(ATT_val);

  Old_millis = millis();          //  reference for timer
  delay(500);
}

boolean k = 0;                        // Heartbeat setting

void loop() {
  Timer();
  if (Timer_LED <= 0){
    k = !k;   digitalWrite(13, k);     // LED on/off
    Timer_LED = 2000;
  }
  if(Timer_KEY <= 0){                 // to avoid key chatter
    key_value = function_key() ;
    if(key_value == 0) old_key_value = 0;
    Timer_KEY = 100;
  }

  if ( (key_value != 0) && (old_key_value != key_value) ){ // Key accepted
    old_key_value = key_value;
    mode=KEY;
    key_accepted = key_value;
   Serial.print("Key in = "); Serial.println(key_value);
    lcd.setCursor(0, 1);  lcd.print("               ");
  }

  if( mode == KEY){
    switch (key_accepted ){           // 4
     case btnSELECT:{
       EEPROM.put( ATT_AD, ATT_val ); //
       lcd.setCursor(0, 1);
      lcd.print("ATT");  lcd.print(ATT_val); lcd.print("  ");
       mode=MEASURE;
       break;
     }
     case btnUP:{                     // Increment ATT_value 1
       ATT_val ++;
       if( ATT_val >30) ATT_val = 30; // Maxmum is 30
       break;
     }
     case btnDOWN:{                   // Decrement ATT_value  2
       ATT_val --;
       if( ATT_val < 0) ATT_val = 0;
       break;
     }
    }
    lcd.setCursor(10, 0);
```

```
    lcd.print(" ATT");
    lcd.print(ATT_val);   lcd.print("  ");

    key_accepted = 0;
  }
  if((mode==MEASURE) && (Timer_MEASURE <= 0)){
    Timer_MEASURE = 1000;
    uint16_t adc_raw = analogRead(1);  // Get AD8307 output voltage via ADC
    Serial.print(" ADC value = "); Serial.print(adc_raw);
    float adc = static_cast<float>(adc_raw) / 1024 * 5;  // Operating on 5V

    Serial.print(" ADC Voltage = "); Serial.println(adc * 1000);

    float dBm = (adc * 1000 / SLOPE) + INTERCEPT - float(ATT_val);
                                                          // from voltage to dBm
    Serial.print(" dBm = "); Serial.print(round(dBm));

    float watts = pow(10, dBm / 10) / 1000;               // from dBm to wattage
    Serial.print("  mW = "); Serial.println(watts * 1000);

    lcd.setCursor(0, 1);
    if (adc_raw <30 ) lcd.print("<<");   //  If adc is less than 30, error
increasing
    lcd.print(round(dBm));
    lcd.print("dBm ");

   lcd.print(formatWatts(watts));

    lcd.setCursor(10, 0);
    lcd.print(" ATT");
    lcd.print(ATT_val);    lcd.print("  ");
  }

}
```