KOCAELİ ÜNİVERSİTESİ

MÜHENDİSLİK FAKÜLTESİ

Mikroişlemciler 6 Nolu Proje

Musa Büyükçelebi

180208074

MSP430+Wi-Fi modül ile 2 farklı analog veriyi Thigspeak gönderme

Bölümü: Elektronik ve Haberleşme Mühendisliği

Dr. Ayhan Küçükmanisa

1. GİRİŞ

Bu proje, MSP430 mikrodenetleyici ve Wi-Fi modülü ile donatılmış bir sistem kullanarak, çevresel sensörlerden elde edilen iki farklı analog veriyi ThingSpeak bulut platformuna aktarmayı amaçlamaktadır. MSP430, düşük güç tüketimi ve yüksek hassasiyeti ile öne çıkan bir mikrodenetleyici olup, analog verileri doğru bir şekilde ölçmek için kullanılırken, Wi-Fi modülü ise kablosuz ağ bağlantısını sağlamaktadır.

Bu projede, MSP430 mikrodenetleyici, analog verileri okumak ve işlemek için kullanılacak. Sensörlerden alınan analog veriler, mikrodenetleyiciye bağlı olan ADC (Analog-Dijital Dönüştürücü) kanalları aracılığıyla ölçülür. Mikrodenetleyici, ölçülen değerleri dijital formata dönüştürür ve daha sonra Wi-Fi modülü aracılığıyla ThingSpeak bulut platformuna göndermek için uygun bir veri paketi oluşturur.

Wi-Fi modülü, kablosuz ağa bağlanarak Internet erişimini sağlar ve ThingSpeak API'sini kullanarak önceden oluşturulmuş bir kanala veri gönderimini gerçekleştirir. ThingSpeak platformunda, gönderilen veriler gerçek zamanlı olarak görüntülenebilir, kaydedilebilir ve analiz edilebilir.

Bu projenin potansiyel uygulama alanları arasında çevresel koşulların izlenmesi, hava kalitesi ölçümü, sıcaklık ve nem gibi parametrelerin takibi gibi çeşitli IoT tabanlı projeler yer almaktadır. Proje, mikrodenetleyici ve Wi-Fi modülü aracılığıyla basit ve etkili bir şekilde analog verilerin bulut platformuna aktarımını sağlamaktadır.

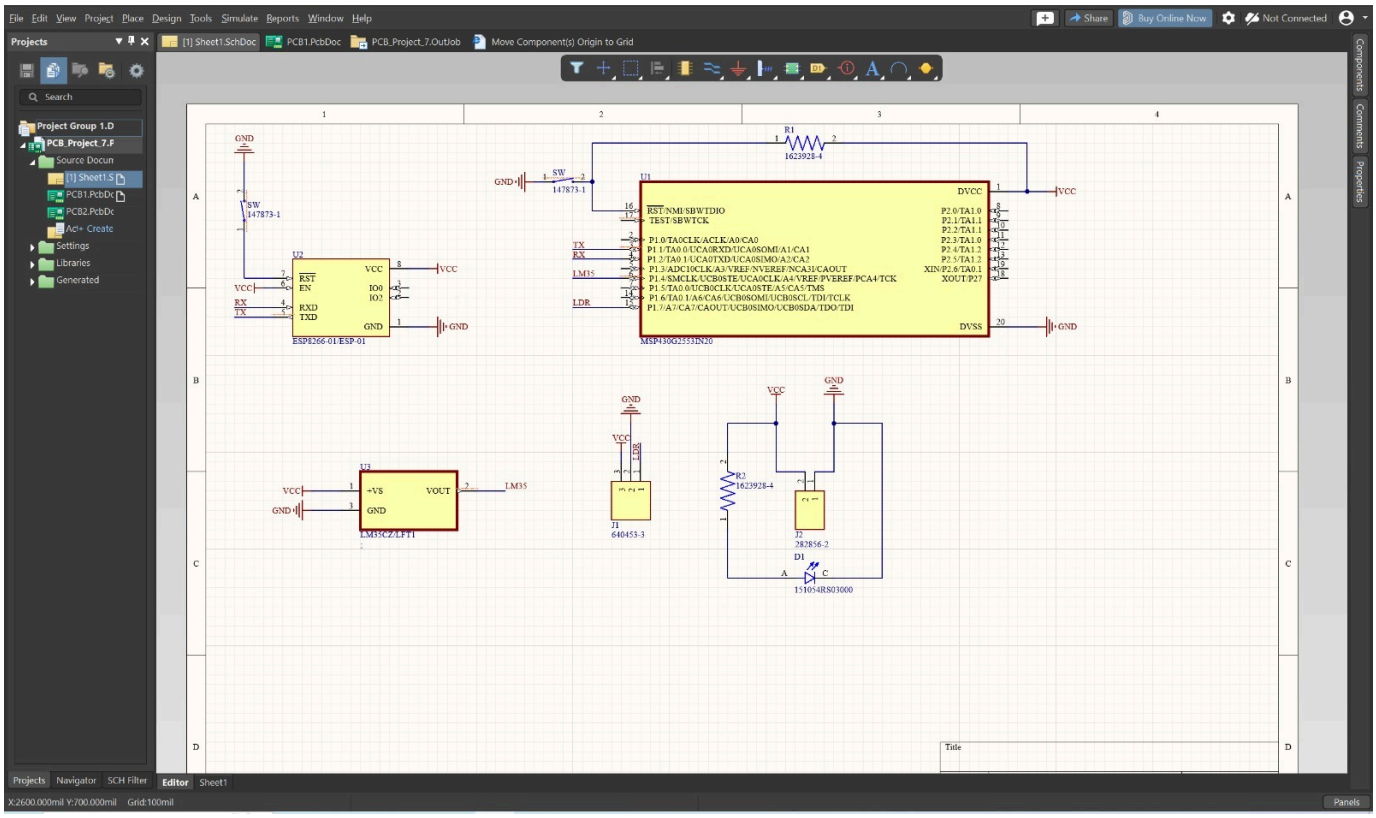
2. PROJENİN AMACI

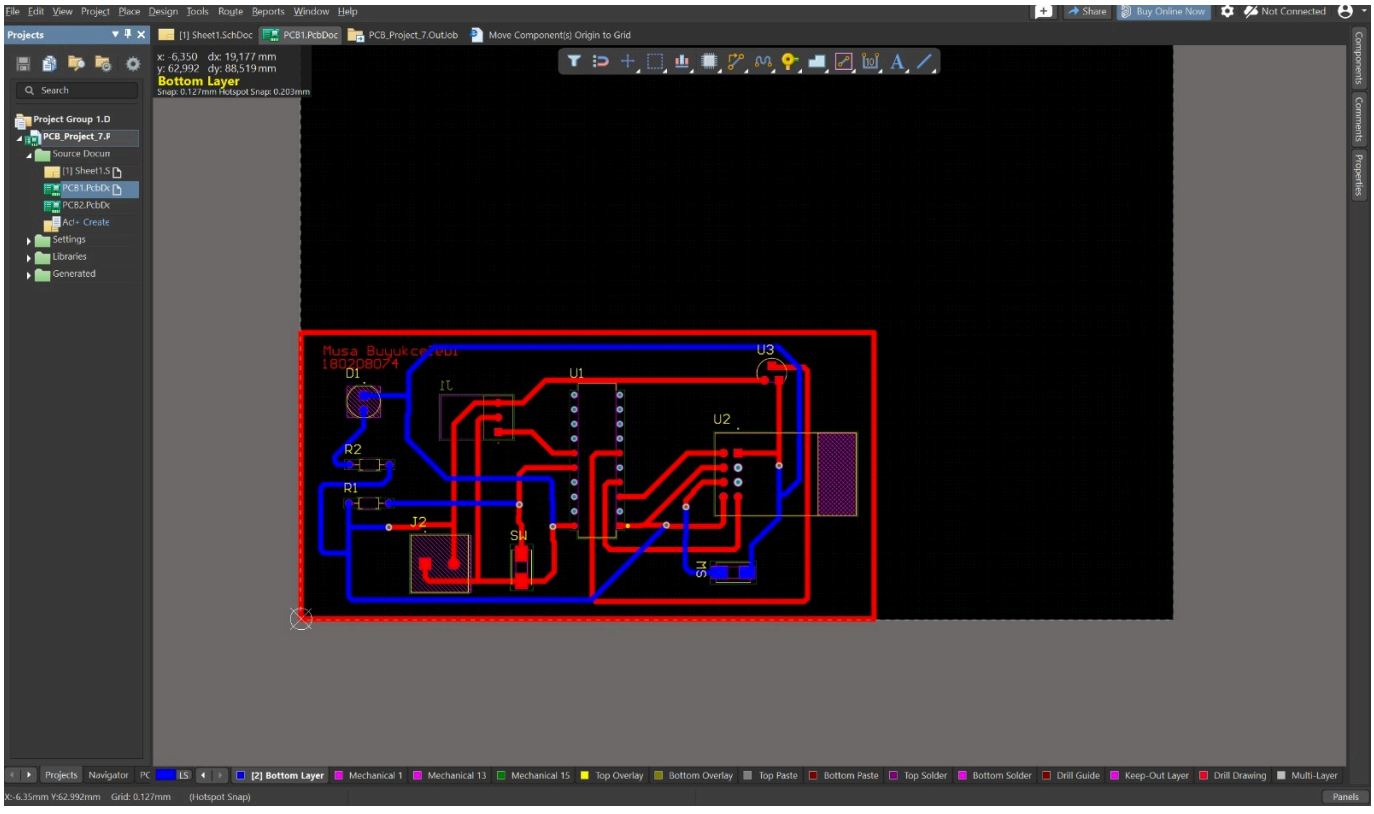
Bu proje, MSP430 mikrodenetleyici ve Wi-Fi modülü kullanarak iki farklı analog veriyi ThingSpeak bulut platformuna göndermeyi hedeflemektedir. Amacı, çevresel sensörlerden elde edilen analog verileri güvenilir bir şekilde ölçmek, bu verileri kablosuz olarak bir bulut platformuna iletmek ve ThingSpeak platformunda gerçek zamanlı olarak izlemek ve analiz etmektir. Bu sayede, veri odaklı kararlar almak, uzaktan izleme ve kontrol sağlamak veya IoT uygulamaları için kullanılan veri toplama sistemleri oluşturmak gibi birçok uygulama alanında kullanılabilir, verilerin toplanması ve analiz edilmesi sağlanır.

2. PROJENİN MALZEMELERİ

- 1 adet msp430G2553
- 1 adet esp8266
- 1 adet ldr sensör
- 1 adet lm35
- 2 adet buton
- 1 adet led
- 1 adet 10k direnç
- 1 adet 1k direnç

Altium


File Edit View Project Place Design Tools Simulate Reports Window Help
Projects
[1] Sheet1.SchDoc
PCB1.PcbDoc
PCB_Project_7.OutJob
Move Component(s) Origin to Grid
Project Group 1.D
PCB_Project_7.F
Source Docun
[1] Sheet1.S
PCB1.PcbDc
PCB2.PcbDc
Act+ Create
Settings
Libraries
Generated
R1
1623928-4
SW
147873-1
U1
MSP430G2553IN20
U2
ESP8266-01/ESP-01
U3
LM35CZ/LFT1
J1
640453-3
J2
282856-2
D1
151034RS03000
LDR
LM35
VCC
GND
Title
Editor
Sheet1


File Edit View Project Place Design Tools Route Reports Window Help
Projects
[1] Sheet1.SchDoc
PCB1.PcbDoc
PCB_Project_7.OutJob
Move Component(s) Origin to Grid
x: -6,350 dx: 19,177 mm
y: 62,992 dy: 88,519 mm
Bottom Layer
Snap: 0.127mm Hotspot Snap: 0.203mm
Musa Buyukce
180208074
D1
U1
U2
U3
R1
R2
J1
J2
SW
[2] Bottom Layer
Mechanical 1
Mechanical 13
Mechanical 15
Top Overlay
Bottom Overlay
Top Paste
Bottom Paste
Top Solder
Bottom Solder
Drill Guide
Keep-Out Layer
Drill Drawing
Multi-Layer

Kod

```
#include <msp430.h>

#define MAX_SEND_LENGTH     3
#define BASE_URL_LDR            "GET /update?
key=23KT0FP5RL63VOYW&field3="
#define BASE_URL_TMP            "GET /update?
key=23KT0FP5RL63VOYW&field1="
#define WIFI_NETWORK_SSID   "musa"
#define WIFI_NETWORK_PASS  "11111111"
#define ENABLE_CONFIG_WIFI 1
void initUART(void); // UART configuration
void initADC_LDR(void); // ADC Configuration for LDR Sensor
void resetADC_LDR(void); // Reset ADC Configuration to Defult
void initADC_TMP(void); // ADC Configuration for TMP Sensor
void resetADC_TMP(void); // Reset ADC Configuration to Defult
void initUploadTimer(void); // Timer0_A0 Configuration
void putc(const unsigned c); // Output char to UART
void TX(const char *s); // Output string to UART
void crlf(void); // CarriageReturn and LineFeed (Which ends the line and starts a
new line - sends the message to ESP8266)
void WifiConfigureNetwork(void); // Wifi Network Configuration
void WiFiSendMessage_LDR_TMP(int data); // Send LDR data
void WiFiSendLength(int data); // Send the Length of the HTTP Request -- Defult
50 digit
char singleDigitToChar(unsigned int digit); // Converts Single to chart to be sent by
the Function putc();

int LDR_Result ;
int LDR ;
int TMP_Result;
int TMP;
int sensorReadingLDR = 0;
int sensorReadingTMP = 0;
```

```
int main(void)
{
   WDTCTL = WDTPW + WDTHOLD;
   if (CALBC1_1MHZ == 0xFF)
   {
      while(1);
   }
   initUART();
  #if defined(ENABLE_CONFIG_WIFI)
      WifiConfigureNetwork();
  #endif
   initUploadTimer();
   __bis_SR_register(LPM0_bits + GIE);

}


void initUploadTimer(void)
{
   TA0CCTL0 = CCIE;
   TA0CTL = TASSEL_2 + MC_1 + ID_3;
   TA0CCR0 = 62500;
}
void initADC_LDR(void)
{
   ADC10CTL0 =  ADC10ON + ADC10IE;
   ADC10CTL1 = INCH_4;
   ADC10AE0 = BIT4;
}
void resetADC_LDR(void)
{
   ADC10CTL0 &= ~ENC ;
   ADC10CTL0 = 0 ;   ADC10CTL1 = 0 ; ADC10AE0 =  0 ;
}
void initADC_TMP(void)
{
   ADC10CTL0 = ADC10ON + ADC10IE;
   ADC10CTL1 = INCH_7 ;
   ADC10AE0 = BIT7;
}
```

```
void resetADC_TMP(void)
{
  ADC10CTL0 &= ~ENC ;
  ADC10CTL0 = 0 ;   ADC10CTL1 = 0 ; ADC10AE0 =  0 ;
}

void WifiConfigureNetwork(void)
{
      TX("AT+RST");                              // Reset the ESP8266
      crlf();
      __delay_cycles(2000000);
      TX("AT+CWMODE_DEF=1");          // Set mode to client (save to flash)



      crlf();
      __delay_cycles(2000000);    // Delay
      TX("AT+CWJAP_DEF=\"");
      TX(WIFI_NETWORK_SSID);
      TX("\",\"");
      TX(WIFI_NETWORK_PASS);          // Connect to WiFi network
      TX("\"");
      crlf();
      __delay_cycles(5000000);
}
// Transmits the TCP request
void WiFiSendTCP()
{
  // TCP Request to ThingSpeak  AT+CIPSTART="TCP","api.thingspeak.com",80
  TX("AT+CIPSTART=");
  while (!(IFG2&UCA0TXIFG));
  UCA0TXBUF = "";                // Transmit a byte
  TX("TCP");
  while (!(IFG2&UCA0TXIFG));
  UCA0TXBUF = "";
  TX(",");
  while (!(IFG2&UCA0TXIFG));
  UCA0TXBUF = "";
  TX("api.thingspeak.com");
  while (!(IFG2&UCA0TXIFG));
  UCA0TXBUF = "";
```

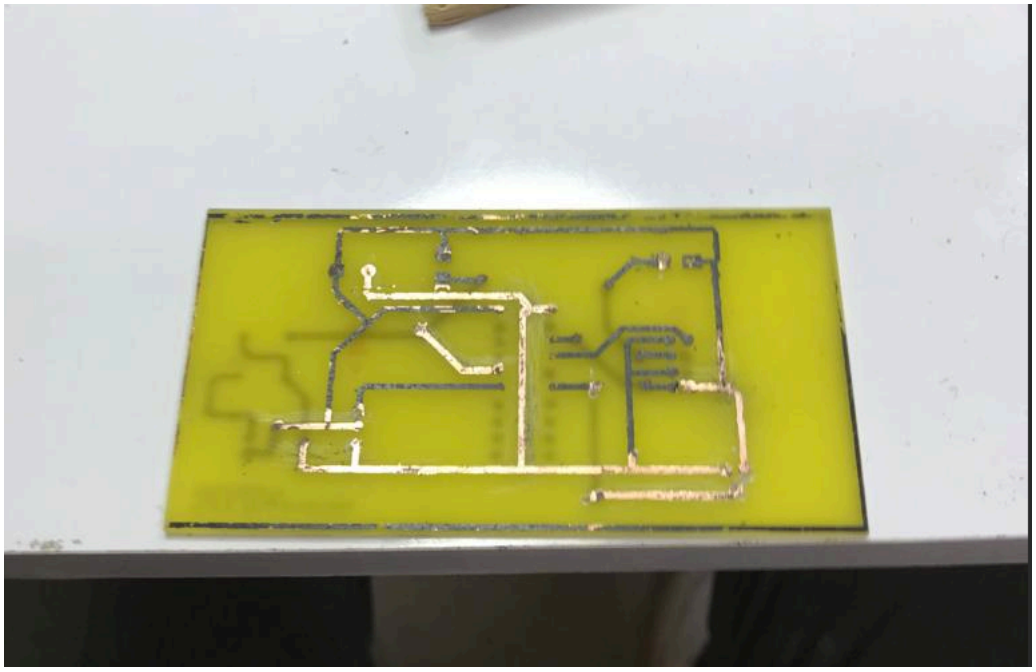
```
    TX(",");
    TX("80");
    crlf();
}

char singleDigitToChar(unsigned int digit)
{
    char returnDigit = '0';
    if(digit == 0) returnDigit = '0';
    else if(digit == 1) returnDigit = '1';  else if(digit == 2) returnDigit = '2'; else
if(digit == 3) returnDigit = '3';
    else if(digit == 4) returnDigit = '4';  else if(digit == 5) returnDigit = '5'; else
if(digit == 6) returnDigit = '6';
    else if(digit == 7) returnDigit = '7';  else if(digit == 8) returnDigit = '8'; else
if(digit == 9) returnDigit = '9';
    else returnDigit = '0'; return returnDigit;
}



void WiFiSendLength(int data)
{
    TX("AT+CIPSEND=50");
    crlf();
}
void WiFiSendMessage_LDR_TMP(int data )
{
  if(data==LDR){
    TX(BASE_URL_LDR);
    }

  if(data==TMP){
    TX(BASE_URL_TMP);
  }
    unsigned int i;
    unsigned int reverseIndex = MAX_SEND_LENGTH - 1;
    char reverseBuffer[MAX_SEND_LENGTH - 1];
    for(i = 0; i < MAX_SEND_LENGTH; i++) reverseBuffer[i] = 0;
    while(data > 0)
    {
        int operatedDigit ;
```

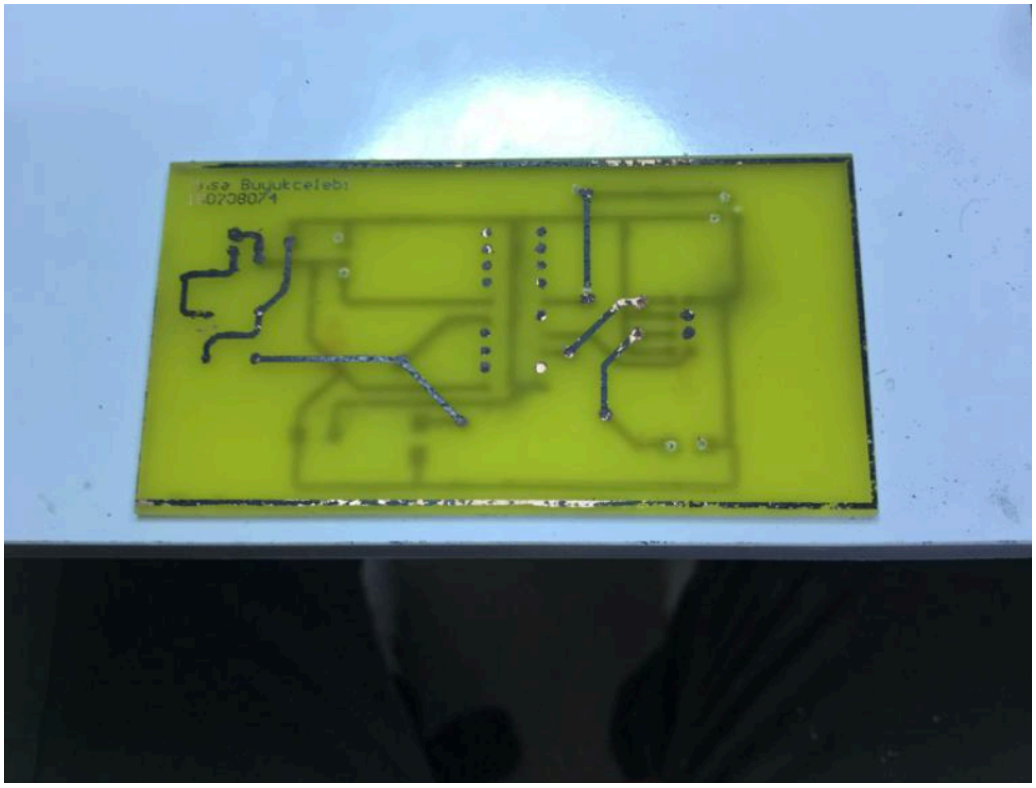
```
        operatedDigit = data % 10 ;
        reverseBuffer[reverseIndex] = singleDigitToChar(operatedDigit);
        reverseIndex--;
        data /= 10;
    }
    for(i = 0; i < MAX_SEND_LENGTH; i++)
    {
        if(reverseBuffer[i] != 0) putc(reverseBuffer[i]);
    }
    crlf();
}


#pragma vector=ADC10_VECTOR
__interrupt void ADC10_ISR(void)
{
    __bic_SR_register_on_exit(CPUOFF);
}

#pragma vector=TIMER0_A0_VECTOR
__interrupt void Timer0_A0 (void)
{

        initADC_LDR();
        __delay_cycles(1000000);
        ADC10CTL0 |= ENC + ADC10SC;
        sensorReadingLDR = ADC10MEM ;
        LDR_Result = sensorReadingLDR;

        LDR = LDR_Result;
        if ( LDR_Result > 0 && LDR_Result < 100 ) { LDR = 5 ; }
        else if ( LDR_Result > 100 && LDR_Result < 250 ) { LDR = 4 ; }
        else if ( LDR_Result > 250 && LDR_Result < 450 ) { LDR = 3 ; }
        else if ( LDR_Result > 450 && LDR_Result < 650 ) { LDR = 2 ; }
        else if ( 650 < LDR_Result ) { LDR = 1 ;}

        WiFiSendTCP();
        __delay_cycles(500000);
        WiFiSendLength(LDR);
        __delay_cycles(500000);
```

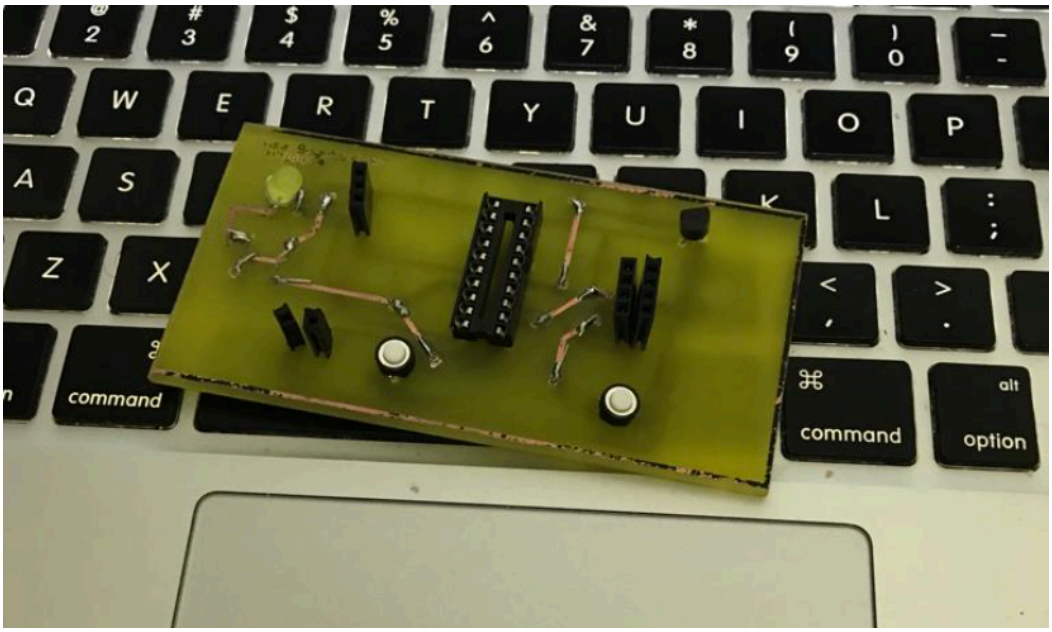
```
        for ( int x = 0 ; x < 8 ; x++){
          WiFiSendMessage_LDR_TMP(LDR);}
        __delay_cycles(500000);
        sensorReadingLDR = 0 ; LDR_Result = 0 ; LDR = 0 ;



        __delay_cycles(3000000);
        resetADC_LDR();
        initADC_TMP();
        __delay_cycles(1000000);
        sensorReadingTMP = ADC10MEM ;
        TMP_Result = sensorReadingTMP;
          TMP = (TMP_Result * 300) / 1024;
        WiFiSendTCP();
        __delay_cycles(500000); // 0.5s delay
        WiFiSendLength(TMP);
        __delay_cycles(500000); // 0.5s delay
        for ( int x = 0 ; x < 8 ; x++ ){
          WiFiSendMessage_LDR_TMP(TMP);}
        __delay_cycles(500000); // 0.5s delay
        resetADC_TMP();
        __delay_cycles(3000000); // 2 s delay
        sensorReadingTMP = 0 ; TMP_Result = 0 ; TMP = 0 ;
}



void putc(const unsigned c)
{
   while (!(IFG2&UCA0TXIFG)); // USCI_A0 TX buffer ready?
   UCA0TXBUF = c;
}

void TX(const char *s)
{
   while(*s) putc(*s++);
}

void crlf(void)
{
```

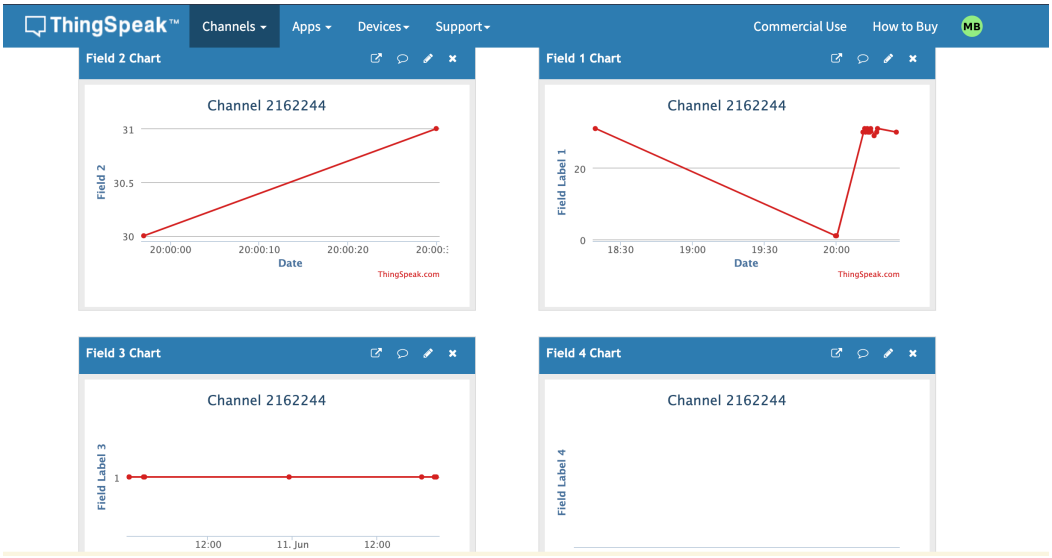
```
    TX("\r\n");
}
void initUART(void)
{
    DCOCTL = 0;                         // Select lowest DCOx and MODx settings
    BCSCTL1 = CALBC1_1MHZ;                  // Set DCO
    DCOCTL = CALDCO_1MHZ;
    P1DIR = 0xFF;                       // All P1.x outputs
    P1OUT = 0;                          // All P1.x reset
    P1SEL = BIT1 + BIT2 + BIT4;             // P1.1 = RXD, P1.2=TXD
    P1SEL2 = BIT1 + BIT2;                   // P1.4 = SMCLK, others GPIO
    P2DIR = 0xFF;                       // All P2.x outputs
    P2OUT = 0;                          // All P2.x reset
    UCA0CTL1 |= UCSSEL_2;                   // SMCLK
    UCA0BR0 = 8;                        // 1MHz 115200
    UCA0BR1 = 0;                        // 1MHz 115200
    UCA0MCTL = UCBRS2 + UCBRS0;             // Modulation UCBRSx = 5
    UCA0CTL1 &= ~UCSWRST;                   // Initialize USCI state machine
}
```

Projenin Yapım Aşamaları






command
command
option


ThingSpeak™
Channels
Apps
Devices
Support
Commercial Use
How to Buy
MB
Field 2 Chart
Channel 2162244
Field 2
31
30.5
30
20:00:00
20:00:10
20:00:20
Date
ThingSpeak.com
Field 1 Chart
Channel 2162244
Field Label 1
20
0
18:30
19:00
19:30
20:00
Date
ThingSpeak.com
Field 3 Chart
Channel 2162244
Field Label 3
1
12:00
11. Jun
12:00
Field 4 Chart
Channel 2162244
Field Label 4