18200031-Trần Đăng Khoa

**BÁO CÁO UART**

**Đề:**

Thực hiện viết chương trình C để cấu hình cho PIC24F thực hiện giao tiếp UART với Vitual Terminal để nhận lệnh.

+ Lệnh từ Vitual Terminal là "temperature" thì PIC24F sẽ đọc nhiệt độ và hiển thị lên console.

+ Lệnh từ Vitual Terminal là "potetiometer" thì PIC24F sẽ đọc giá trị của biển trở và hiển thị Vout lên console.

**Sơ đồ:**

A picture containing chart

Description automatically generated

Diagram, schematic

Description automatically generated

Cảm biến nhiệt độ kết nối qua RB4, Biến trở kết nối qua RB5

**Code:**

**#include <xc.h>**

**#include "p24FJ128GA010.h"**

**#include <stdio.h>**

**#include <string.h>**

**#define CTS \_RF12 // Clear To Send, input, HW handshake**

**#define RTS \_RF13 // Request To Send, output, HW handshake**

**#define BRATE 68**

**#define U\_ENABLE 0x8008**

**#define U\_TX 0x0400**

**#define BACKSPACE 0x8**

**#define BUF\_SIZE 128**

**char tr[BUF\_SIZE];**

**int index = 0;**

**int flag = 0;**

**float temperature = 0.0;**

**float potentiometer = 0.0;**

**void initU2( void){**

**U2BRG = BRATE;**

**U2MODE = U\_ENABLE;**

**U2MODEbits.BRGH = 1;**

**U2STA = U\_TX;**

**\_TRISF13 = 0;**

**\_TRISF12 = 1;**

**RTS = 1;**

**} // initU2**

**int putU2( int c) {**

**while (CTS);**

**while ( U2STAbits.UTXBF);**

**U2TXREG = c;**

**return c;**

**} // putU2**

**void putsU2( char \*s) {**

**while( \*s)**

**putU2( \*s++);**

**} // putsU2**

**void U2\_Interrupt\_Init(){**

**U2STAbits.URXISEL = 0;**

**\_U2RXIP = 4;**

**\_U2RXIF = 0;**

**\_U2RXIE = 1;**

**}**

**void \_ISR \_U2RXInterrupt(void){**

**\_U2RXIF = 0;**

**char c = U2RXREG;**

**tr[index] = c;**

**putU2(tr[index]);**

**index++;**

**if(index > BUF\_SIZE)**

**index = 0;**

**if(c == '\r'){**

**if (strcmp(tr,"hello\r") == 0){**

**putsU2("My name is Khoa\n\r");**

**}else if (strcmp(tr,"temperature\r") == 0){**

**initTemperature();**

**putsU2("Nhiet do: \r\n");**

**int temp = (int)(temperature\*10);**

**putU2(temp/100+48);**

**putU2((temp%100)/10+48);**

**putU2('.');**

**putU2((temp%100)%10+48);**

**putsU2("doC\r\n");**

**}else if(strcmp(tr,"potentiometer\r")==0){**

**putsU2("Vout cua bien tro: \r\n");**

**initPotentiometer();**

**int v = (int)(potentiometer\*100);**

**putU2(v/100+48);**

**putU2('.');**

**putU2((v%100)/10+48);**

**putU2((v%100)%10+48);**

**putsU2("doC\r\n");**

**}**

**else{**

**putsU2("I don't know\n\r");**

**}**

**for(index = 0; index < BUF\_SIZE; index++)**

**tr[index] = '\0';**

**index = 0;**

**}**

**PORTA = index;**

**}**

**void initTemperature() {**

**AD1PCFGbits.PCFG4 = 0;**

**AD1CON1 = 0x00E0;**

**AD1CSSL = 0;**

**AD1CON2 = 0;**

**AD1CON3 = 0x1F02;**

**AD1CON1bits.ADON = 1;**

**int code;**

**code = readADC(4);**

**temperature = (5\*(float)(code)\*1000/1024-500)/10;**

**}**

**void initPotentiometer() {**

**AD1PCFGbits.PCFG5 = 0;**

**AD1CON1 = 0x00E0;**

**AD1CSSL = 0;**

**AD1CON2 = 0;**

**AD1CON3 = 0x1F02;**

**AD1CON1bits.ADON = 1;**

**int code;**

**code = readADC(5);**

**potentiometer = (5\*(float)(code)/1024);**

**}**

**int readADC(int ch) {**

**AD1CHS = ch; // 1. select analog input channel**

**AD1CON1bits.SAMP = 1; // 2. start sampling**

**while (!AD1CON1bits.DONE); // 3. wait for the conversion to complete**

**return ADC1BUF0; // 4. read the conversion result**

**} // readADC**

**int main() {**

**char s[BUF\_SIZE];**

**TRISA = 0;**

**initU2();**

**putsU2( "THUC HANH VI DIEU KHIEN!\n\r");**

**U2\_Interrupt\_Init();**

**//initADC();**

**while(1){**

**}**

**}**

Các hàm **init** để cấu hình.

Hàm **\_ISR \_U2RXInterrupt(void)** đượcthực thi khi ngắt UART xảy ra.

Hàm **initTemperature()** cấu hình và đọc ADC qua kênh 4

Hàm **initPotentiometer()** cấu hình và đọc ADC qua kênh 5

**Lưu ý:** phải tính các giá trị temperature và potentiometer trong hàm init của nó, Vì hàm này được gọi trong hàm ngắt UART (**\_ISR \_U2RXInterrupt**). Nếu đặt ngoài sẽ không đọc được.

**Hình demo:**

Schematic

Description automatically generated with medium confidence

Diagram, schematic

Description automatically generated