**TIỂU LUẬN THIẾT KẾ HỆ THỐNG NHÚNG**

**Họ và tên: Nguyễn Duy Tý**

**Lớp : Kĩ thuật ĐK&TĐH**

**Mssv : 1755252021600043**

**Sbd : 50**

**NỘI DUNG CÁC CÂU HỎI:**

**Câu 1**. Sử dụng vi điều khiển AT89C52, mô phỏng trên phần mềm Proteus và lập

trình thực hiện các nhiệm vụ sau:

1. Hiển thị số 00 lên 2 LED 7 thanh nối vào cổng P2 theo phương pháp quét LED;

2. Tăng số đếm sau mỗi 500ms, nếu số đếm bằng “SBD+20” thì dừng lại (sử dụng

timer để định thời gian).

**Câu 2.** Sử dụng vi điều khiển AT89C52, mô phỏng trên phần mềm Proteus và lập

trình thực hiện các nhiệm vụ sau:

1. Cấu hình ngắt ngoài INT0 ở chế độ ngắt sườn xuống;

2. Đếm số lần nút bấm nút nối vào chân INT0 được bấm, hiển thị kết quả lên 2 LED7

thanh nối vào cổng P2 theo phương pháp quét LED (nếu số lần bấm bằng “SBD+10”

thì quay về 0).

**Câu 3.** Sử dụng vi điều khiển AT89C52, thực hiện các nhiệm vụ sau:

1. Vẽ sơ đồ mô phỏng trên Proteus ghép nối với LCD theo chế độ 4bit, hiển thị họ tên, mã số sinh viên lên LCD.

2. Vẽ sơ đồ mô phỏng trên Proteus, lập trình hiển thị “Họ tên và mã số sinh viên” qua chuẩn truyền thông UART;

**Câu 4.** Sử dụng vi điều khiển AT89C52, vẽ sơ đồ mô phỏng trên Proteus ghép nối với Led D1 qua cổng P1.2, BUTTON B1 qua cổng P1.3. Sử dụng hệ điều hành RTX51 lập trình ngắt USART, tast BUTTON, tast LED. Thực hiện gửi “signal” từ ngắt USART và task BUTTON đến tast LED. Task LED thực hiện đảo trạng thái của Led D1 khi nhận được tín hiệu task khác gửi tới.

**Câu 5.** Hãy trình bày:

1. Trình bày quy trình thiết kế hệ thống nhúng sử dụng vi điều khiển.

2. Hệ điều hành thời gian thực (RTOS). Ưu điểm, nhược điểm và ứng dụng của hệ điều hành thời gian thực trong thiết kế các hệ thống nhúng.

**BÀI LÀM**

**Câu 1:**

**1.1.**

**Code:**

#include <REGX52.H>

char so[ ]= {0x40,0x79,0x24,0x30,0x19,0x12,0x02,0x78,0x00,0x10,0x89,0x06,0xC7,0x40};

#define led1 P3\_0

#define led2 P3\_1

#define sang 0

#define tat 1

char i;

void delay(int time){

while(time--);

}

void main(){

led1 = led2 = tat;

while(1){

led1 = sang;

P2 = so[0];

delay(1000);

led1 = tat;

led2 = sang;

P2 = so[0];

delay(1000);

led2 = tat;

}

}

**Mô phỏng proteus:**

Chart, box and whisker chart

Description automatically generated

**1.2.**

**Code:**

#include <REGX52.H>

char so[ ]= {0x40,0x79,0x24,0x30,0x19,0x12,0x02,0x78,0x00,0x10,0x89,0x06,0xC7,0x40};

char i;

int dem;

usigned char chuc, donvi;

#define led1 P3\_0

#define led2 P3\_1

#define sang 0

#define tat 1

void delay(int time){

while(time--);

TMOD=0x01;

TH0=0xfc;

TL0=0x18;

TR0=1;

while(!TF0);

TF0=0;

TR0=0;

}

void main(){

EA=1;

EX0=1;

led1 = led2 = tat;

while(1){

for (dem>=0;dem<=70;dem++){

//tach chu so

chuc = dem/10;

donvi = dem%10;

for (i=0;i<=20;i++){

led1 = sang;

P2 = so[chuc];

delay(500);

led1 = tat;

led2 = sang;

P2 = so[donvi];

delay(500);

led2 = tat;

}

}

}

}

**Mô phỏng proteus:**

Chart, box and whisker chart

Description automatically generated

**Câu 2:**

**2.1.**

**Code:**

//NGUYENDUYTY

#include <REGX52.H>

void delay\_ms(int ms){

while(ms--){

TH0 = 0Xfc;

TL0 = 0x18;

TR0 = 1;

while(!TF0);

TF0 = 0;

TR0 = 0;

}

}

void main(){

EX0 = 1; //cho phep ngat ngoai 0

IT0 = 1; //chon kieu ngat theo suon

EA = 1; //cho phep ngat toan cuc

while(1){

P2 = 0;

delay\_ms(1000);

P2 = 0xff;

delay\_ms(1000);

}

}

void ngat() interrupt 0{

long a = 50000;

P1\_3 = 0;

while(a--){};

P1\_3 = 1;

}

**Mô phỏng Proteus:**

Chart

Description automatically generated

Chart

Description automatically generated

**2.2.**

**Code:**

//NGUYENDUYTY

#include <REGX52.H>

#define led1 P3\_0

#define led2 P3\_1

#define sang 0

#define tat 1

sbit up=P3^2;

char so[ ]={0x40,0x79,0x24,0x30,0x19,0x12,0x02,0x78,0x00,0x10};

char count;

unsigned char chuc,donvi;

void delay\_ms(int time){

while(time--){

TMOD=0x01;

TH0=0xfc;

TL0=0x18;

TR0=1;

while(!TF0);

TF0=0;

TR0=0;

}

}

void tang()interrupt 0{

count++;

if(count>60) count=0;

}

void main (){

EA=1;

EX0=1;

IT0=1;

EX1=1;

IT1=1;

while(1){

chuc=count/10;

donvi=count%10;

led1 =sang; P2=so[chuc]; delay\_ms(10); led1 =tat;

led2 =sang; P2=so[donvi]; delay\_ms(10); led2 =tat;

}

}

**Mô phỏng proteus:**

Chart

Description automatically generated

Chart

Description automatically generated

**Câu 3:**

**3.1.**

**Code:**

#include <REGX52.H>

#define LCD\_RS P1\_0

#define LCD\_RW P1\_1

#define LCD\_EN P1\_2

#define LCD\_D4 P0\_4

#define LCD\_D5 P0\_5

#define LCD\_D6 P0\_6

#define LCD\_D7 P0\_7

void delay\_us(unsigned int t){

unsigned int i;

for(i=0;i<t;i++);

}

void delay\_ms(unsigned int t){

unsigned int i,j;

for(i=0;i<t;i++)

for(j=0;j<125;j++);

}

void delay(long time){

while (time--);

}

/\*\*\*\*\*\*\*\*\*\*chuong trinh giao tiep LCD 16x2 4bit\*\*\*\*\*\*\*\*\*\*/

void LCD\_Enable(void){

LCD\_EN =1;

delay\_us(3);

LCD\_EN =0;

delay\_us(50);

}

// ham gui 4bit du lieu ra LCD

void LCD\_Send4Bit(unsigned char Data){

LCD\_D4=Data & 0x01;

LCD\_D5=(Data>>1)&1;

LCD\_D6=(Data>>2)&1;

LCD\_D7=(Data>>3)&1;

}

// ham gui 1 lenh cho LCD

void LCD\_SendCommand(unsigned char command){

LCD\_Send4Bit(command >>4);

LCD\_Enable();

LCD\_Send4Bit(command);

LCD\_Enable();

}

void LCD\_Clear(){ //ham xoa man hinh LCD

LCD\_SendCommand(0x01);

delay\_us(10);

}

//ham khoi tao LCD

void LCD\_Init(){

LCD\_Send4Bit(0x00);

delay\_ms(20);

LCD\_RS=0;

LCD\_RW=0;

LCD\_Send4Bit(0x03);

LCD\_Enable();

delay\_ms(5);

LCD\_Enable();

delay\_us(100);

LCD\_Enable();

LCD\_Send4Bit(0x02);

LCD\_Enable();

LCD\_SendCommand(0x28);

LCD\_SendCommand(0x0c);

LCD\_SendCommand(0x06);

LCD\_SendCommand(0x01);

}

void LCD\_Gotoxy(unsigned char x, unsigned char y){

unsigned char address;

if(!y)address=(0x80+x);

else address= (0xc0+x);

delay\_us(1000);

LCD\_SendCommand(address);

delay\_us(50);

}

void LCD\_PutChar(unsigned char Data){

LCD\_RS=1;

LCD\_SendCommand(Data);

LCD\_RS=0;

}

void LCD\_Puts(char\*s){

while (\*s){

LCD\_PutChar(\*s);

s++;

}

}

void main(){

LCD\_Init();

LCD\_Puts(" hello");

delay\_ms(1000);

LCD\_Clear();

LCD\_Gotoxy(0,0);

LCD\_Puts("nguyen duy ty");

delay\_ms(1000);

LCD\_Gotoxy(0,1);

LCD\_Puts("1755252021600043");

while(1);

}

**Mô phỏng proteus:**

Chart

Description automatically generated

**3.2.**

**Code:**

#include <REGX52.H>

sbit rs=P2^0;

sbit rw=P2^1;

sbit e=P2^2;

void delay(unsigned int);

void cmd1(unsigned char);

void dat1(unsigned char);

void cmd(unsigned char);

void dat(unsigned char);

void main(void)

{

unsigned char mybyte;

cmd(0x28);

cmd(0x01);

cmd(0x0c);

cmd(0x80);

cmd(0x06);

TMOD=0X20; //su dung Time1 mode2

TH1=0xFd;

SCON=0X50;

TR1=1;

while(1)

{//lap lai mai mai

while(RI==0);

mybyte=SBUF;

dat(mybyte);

RI=0;

if(mybyte==0x08)

cmd(0x01);

if(mybyte==0x0d)

cmd(0xc0);

}

}

void delay(unsigned int t)

{

unsigned int i,j;

e=1;

for(i=0;i<t;i++);

for(j=0;j<1275;j++);

e=0;

}

void cmd(unsigned char a)

{

unsigned char x;

x=a&0xf0;

cmd1(x);

x=(a<<4)&0xf0;

cmd1(x);

}

void dat(unsigned char a)

{

unsigned char x;

x=a&0xf0;

dat1(x);

x=(a<<4)&0xf0;

dat1(x);

}

void cmd1(unsigned char ch)

{

P2=ch;

rs=0;

rw=0;

delay(10);

}

void dat1(unsigned char ch)

{

P2=ch;

rs=1;

rw=0;

delay(10);

}

**Mô phỏng proteus:**

Diagram, schematic

Description automatically generated

**Câu 4:**

**Code:**

#include <REGX52.H>

#include <RTX51TNY.H> //Su dung thu vien RTX51 Tiny Real-Time

#define INIT 0 //INIT = 0

#define DO 1 //DO = 1

#define BUTT 2 //BUTTTON = 2

sbit LED\_DO = P1^2; //chan LED\_DO

sbit BUTTON = P1^3; //chan BUTTON

void USART(void) interrupt 4 //Ngat nhan USART

{

if(RI) //Flag nhan duoc ki tu

{ //Clear flag

RI=0; //Nhan ki tu

isr\_send\_signal(DO); //Gui signal cho task DO

}

}

//=========Ham Start up==========

void Startup(void) \_task\_ INIT

{

SCON=0x52; //USART che do 1

TMOD=0x21; //Timer 1 mode 2

TH1=TL1=-3; //baudrate 9600

TR1=1;

IE=0x90; //Ngat USART

os\_create\_task (DO); //Tao Task\_Led\_Do

os\_create\_task (BUTT); //Tao Task BUTTON

os\_delete\_task (INIT); //Xoa Task hien tai (Task 0)

}

void Task\_Led\_Do(void) \_task\_ DO

{

while(1)

{

os\_wait2(K\_SIG ,50); //Cho signal voi time out 50 ticks

LED\_DO ^= 1; //Dao trang thai Led Do

}

}

void Task\_BUTTON(void) \_task\_ BUTT

{

while(1)

{

if(BUTTON == 0) //Nhan nut nhan = 0

{

os\_send\_signal(DO); //Gui signal cho task DO

while(BUTTON==0); //Cho nut nhan = 1(Chong nhieu)

}

os\_wait2(K\_TMO, 10); //Cho 10 ticks = 100ms

}

}

**Mo phỏng proteus:**

A picture containing diagram

Description automatically generated

A picture containing graphical user interface

Description automatically generated

**Câu 5:**

**5.1.**

Các kỹ sư xây dựng hệ thống nhúng luôn phải đối mặt với nhiều khó khăn trong quá trình thiết kế hệ thống. Từ việc xác định phân bổ giữa phần cứng và phần mềm cho đến việc tính toán để thiết kế đạt những mục tiêu về hiệu năng và chi phí. Việc xây dựng hệ thống nhúng ngày càng trở lên phức tạp vì yêu cầu ngày càng cao. Họ phải xây dựng những hệ thống ngày càng thông minh hơn, có nhiều chức năng hơn nhưng lại phải được gói gọn trong một không gian nhỏ hơn, ít tiêu thụ điện năng hơn, thời gian sản xuất nhanh hơn và chi phí cho hệ thống giảm. Có lẽ bởi vì liên quan đến sự kết hợp của ít nhất hai nguyên tắc phức tạp và sáng tạo: kỹ nghệ phần mềm và thiết kế logic. Vấn đề này thường liên quan đến xây dựng một cái mới mà mọi người chưa từng xây dựng bao giờ hay đơn giản là có quá nhiều sự lựa chọn như: sử dụng bộ vi xử lý nào cho phù hợp, cách sắp xếp các bus, triển khai bằng ngôn ngữ lập trình nào, có sử dụng hệ điều hành hay không, môi trường phát triển …làm cho người phát triển nhiều lúc không biết bắt đầu từ đâu. Không giống với việc thiết kế các ứng dụng phần mềm trên máy tính, việc thiết kế một hệ thống nhúng phải thực hiện thiết kế cả phần cứng và phần mềm một cách song song. Mặc dù không phải lúc nào cũng vậy nhưng thực tế cho thấy đây là cách thức tiếp cận việc thiết kế hệ thống nhúng một cách hiệu quả và ảnh hưởng sâu sắc đến việc xây dựng hệ thống. Quy trình thiết kế hệ thống nhúng bao gồm các bước sau:

• Xác định yêu cầu sản phẩm

• Phân bổ phần cứng phần mềm

• Thiết kế phần cứng/phần mềm nhúng

• Tích hợp phần cứng/phần mềm

• Kiểm tra

• Bảo trì và nâng cấp

**5.2.**

**Ưu điểm:**

- Thay đổi bất kỳ tác vụ nào trong Round Robin hoặc lập lịch theo hàng đợi đều có một nhược điểm là ảnh hưởng đến tổng thể toàn bộ các tác vụ.

- Thay đổi tới tác vụ có độ ưu tiên thấp hơn trong RTOS không ảnh hưởng đến thời gian đáp ứng của các tác vụ có độ ưu tiên cao hơn.

- RTOS được sử dụng rộng rãi và là giải pháp thật sự cần thiết cho các hệ thống yêu cầu ràng buộc về thời gian đáp ứng (ràng buộc thời gian cứng, mềm)

**Nhược điểm:**

- RTOS cần thêm một số thời gian để xử lý các thông tin về tác vụ trước và sau khi đưa nó vào xử lý trong CPU nên hiệu suất sử dụng bị ảnh hưởng (do yêu cầu tính ràng buộc về thời gian nên độ phức tạp cao và xử lý cần đảm bảo độ an toàn). Chi phí cao khi mua các sản phẩm thương mại.

**Ứng dụng:**

**-** Hệ điều hành thời gian thực **(**RTOS**)**  là hệ điều hành đa nhiệm dành cho những ứng dụng thời gian thực. Những ứng dụng này bao gồm hệ thống nhúng, robot công nghiệp, hệ thống điều khiển công nghiệp hay các phương tiện nghiên cứu khoa học. Đây là những phần mềm điều khiển chuyên dụng thường được dùng trong những ứng dụng điện toán nhúng có tài nguyên bộ nhớ hạn chế và yêu cầu ngặt nghèo về thời gian đáp ứng tức thời, tính sẵn sàng cao và khả năng tự kiểm soát một cách chính xác.