Parker Authier & Ben Kepner

Prof. Foster

CE 420-03L Lab 5

Fall Semester 11/14/2019

Contents

[Objectives 3](#_Toc24618061)

[Hardware 3](#_Toc24618062)

[Software Design 4](#_Toc24618063)

[Flowcharts 4](#_Toc24618064)

[Power Consumption and Impact 5](#_Toc24618065)

[Program Source Code 6](#_Toc24618066)

[seven\_seg.c 6](#_Toc24618067)

[seven\_seg.h 9](#_Toc24618068)

[mian.c 11](#_Toc24618069)

[Disassembly Listing File 17](#_Toc24618070)

# Objectives

The objective of this lab was to use the working keypad code from a previous lab, and code that was used to provide numeric feedback through serial communication, to create a kitchen timer using a seven-segment display. Using system timers and provided code outlines, the timer function was implemented into the existing code. The use of the on-board speaker was also encouraged to act as emphasis to the alarm function of the timer. The inclusion of the speaker added a sound output to the already flashing seven-segment display.

# Hardware

The hardware used was the Microcomputer Systems II board, the expansion board designed for the class, and the 12 button membrane keypad.

# Software Design

## Flowcharts

Timer Logic Diagram

7-Seg Display Values

Seconds = 60

Minutes - 1

False

Minutes = 0?

False

True

Seconds = 0?

Set minutes & seconds

Alarm

Seconds - 1

Update minute displays

Update second displays

True

## Power Consumption and Impact

Power consumption is important to consider when using embedded systems to do tasks. For a device such as the kitchen timer that was created in this lab power consumption is critical. For a kitchen timer, or any other embedded system that is not in constant use, you would want to minimize the power consumption when the device is not in use to maximize the time that the device is functional before it needs to be serviced. One way that this can be done is by changing the different operating modes that the CPU runs in. The different operating modes use a different amount of power. Switching the CPU to a sleep mode that uses less power when the device is not in use is more efficient than keeping it in a higher power mode continuously. The CPU can be “woken” up from this low power mode by using interrupts.

# Program Source Code

## seven\_seg.c

|  |
| --- |
| // seven\_seg.c  //this is an incomplete program code for the seven segment device driver code  #include "seven\_seg.h" //assuming you named your header file this way  //Example -- initialize Timer1 to trigger interrupts every 1ms  void init\_Timer1(void) {  TMR1 = 0x0000; //clear timer register  PR1 = 0x9c; //set the period    //init interrupts  mT1SetIntPriority(5); //group priority set to 5  mT1ClearIntFlag();  // configure for multi-vectored mode and enable system interrupt  INTEnableSystemMultiVectoredInt();  mT1IntEnable(1); //enable T1 interrupts  T1CON = 0x8038; //enable timer, set prescaler to 1:8  }  void init7Seg(void)  {  init\_Timer1();  TRISEbits.TRISE0 = OUT;  TRISEbits.TRISE1 = OUT;  TRISEbits.TRISE2 = OUT;  TRISEbits.TRISE3 = OUT;  TRISEbits.TRISE4 = OUT;  TRISEbits.TRISE5 = OUT;  TRISEbits.TRISE6 = OUT;  TRISEbits.TRISE7 = OUT;    TRISDbits.TRISD2 = OUT;  TRISDbits.TRISD9 = OUT;    segPower = 1;  }  //void set7Seg(char numbers[4])  //{  // segValues[0] = numbers[0];  // segValues[1] = numbers[1];  // segValues[2] = numbers[2];  // segValues[3] = numbers[3];  //  //}  //Timer1 interrupt handler  //it is used to refresh the 7-segment displays every 1ms  void \_\_ISR(\_TIMER\_1\_VECTOR, IPL5SOFT) T1ISR(void)  {  mT1ClearIntFlag();    if(segPower == 1)  {  PORTE = segLookup[segValues[segState]];  //PORTECLR = ~ segLookup[segValues[segState]];  switch(segState)  {  case 0:  PORTDbits.RD2 = 0;  PORTDbits.RD9 = 0;  PORTESET = 0b10000000;  break;  case 1:  PORTDbits.RD2 = 1;  PORTDbits.RD9 = 0;  PORTECLR = 0b10000000;  break;  case 2:  PORTDbits.RD2 = 0;  PORTDbits.RD9 = 1;  break;  case 3:  PORTDbits.RD2 = 1;  PORTDbits.RD9 = 1;  break;  }  segState++;  if(segState > 3)  {  segState = 0;  }  }  else  {  PORTE = 0xFF;  }  } |

## seven\_seg.h

|  |
| --- |
| 1 //seven\_seg.h   2 //this is incomplete template shows the include section for the seven\_seg header file   3   4 #include <p32xxxx.h>  5   6 //suppress plib warnings   7 #define \_SUPPRESS\_PLIB\_WARNING   8 #define \_DISABLE\_OPENADC10\_CONFIGPORT\_WARNING  9  10 static char segLookup[] = {~0b00111111, ~0b00000110, ~0b01011011, ~0b01001111, ~0b01100110, ~0b01101101, ~0b01111101, ~0b00000111, ~0b01111111, ~0b01100111, 0xff}; 11  12  13 //static char segLookup[] = {0b00000001, 0b00000010, 0b00000100, 0b00001000, 0b00010000, 0b00100000, 0b01000000, 0b10000000, 0b11111111, 0b00000000}; 14  15 #include <plib.h> 16  17 volatile unsigned int segState = 0; 18 volatile unsigned int segValues[] = {0,0,0,0}; 19 volatile unsigned int segPower; 20  21 #define IN 1  22 #define OUT 0 |

## main.c

|  |
| --- |
| //provide complete documentation  #include <p32xxxx.h>  #include "keypad.h"  #include "uart.h"  #include "seven\_seg.h"  // configuration bit settings, Fcy=80MHz, Fpb=40MHz  #pragma config POSCMOD=XT, FNOSC=PRIPLL  #pragma config FPLLIDIV=DIV\_2, FPLLMUL=MUL\_20, FPLLODIV=DIV\_1  #pragma config FPBDIV=DIV\_2, FWDTEN=OFF, CP=OFF, BWP=OFF  volatile int minutes;  volatile int seconds;  volatile int state = 0;  #define PWM\_PWR(x) (x==1? (OC4CONSET = 0x8000) : (OC4CONCLR = 0x8000))  void init\_Timer4(void)  {  T4CON = 0x0; // Stop the timer and clear the control register,  T4CON = 0b01111000; // prescaler at 1:256,internal clock source, 32bit mode  TMR4 = 0x0; // Clear the timer register  PR4 = 0x2625a; // Load the period register  IPC5SET = 0x0000000D; // Set priority level = 3, subpriority level = 1  IFS0CLR = 0x00100000; // Clear the timer interrupt status flag  IEC0SET = 0x00100000; // Enable timer interrupts  T4CONSET = 0x8000; // Start the timer    }  void init\_PWM(void)  {  OC4CON = 0x0000; // Turn off the OC1 when performing the setup  OC4R = 0x0F00; // Initialize primary Compare register  OC4RS = 0x0F00; // Initialize secondary Compare register  OC4CON = 0x0006; // Configure for PWM mode without Fault pin  // enabled  PR2 = 0xffff; // Set period  // Configure Timer2 interrupt. Note that in PWM mode, the  // corresponding source timer interrupt flag is asserted.  // OC interrupt is not generated in PWM mode.  // IFS0CLR = 0x00000100; // Clear the T2 interrupt flag  // IEC0SET = 0x00000100; // Enable T2 interrupt  // IPC2SET = 0x0000001C; // Set T2 interrupt priority to 7  T2CONSET = 0x8000; // Enable Timer2  // OC4CONSET = 0x8000; // Enable OC1  }  void send\_PWM(signed int speed)  {  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* this function takes a speed value  \* in percentage from -100 to 100  \* and sets the pwm module accordingly  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/    OC1RS = ((speed \* 4) + 3000);  //CCP4RB = ((speed \* 4) + 3000);  //CCP4CON1Lbits.CCPON = 1; //Turn on MCCP module  }  //Timer4 interrupt handler  void \_\_ISR(\_TIMER\_5\_VECTOR, IPL5SOFT) T5ISR(void)  {  mT5ClearIntFlag();    switch(state)  {  case 0:  break;  case 1:  if(seconds == 0)  {  if(minutes == 0)  {  state = 2;  }  else  {  minutes--;  //new values for the minute display segments  segValues[0] = minutes / 10;  segValues[1] = minutes % 10;  //start seconds on 60 for end of minute  seconds = 60;  //updates the vlaues for the second display segments  segValues[2] = seconds / 10;  segValues[3] = seconds % 10;  }  }  else  {  seconds--;  //new values for the second display segments  segValues[2] = seconds / 10;  segValues[3] = seconds % 10;  }  break;    case 2:  segPower = ! segPower;  PWM\_PWR(segPower);  break;  }      }  int main(void) {  DDPCONbits.JTAGEN = 0;  //declare any local variables you may need  unsigned char number;    initKeypad();  initUART1();  init7Seg();  init\_Timer4();  init\_PWM();    UART1\_putstr("Hello, please enter keys on the keypad:");    //add your code here    //infinite loop  while (1)  {  number = readKey();  if(number != '\0')  switch(state)  {  case 0:    if(number == '\*')  break;    else if(number == '#')  {  state = 1;  }  else  {  UART1\_putchar(number);  segValues[0] = segValues[1];  segValues[1] = segValues[2];  segValues[2] = segValues[3];  segValues[3] = number -0x30;  minutes = (segValues[0] \* 10) + segValues[1];  seconds = (segValues[2] \* 10) + segValues[3];  }  break;  case 1:  if(number == '#')  {  state = 0;  }  case 2:  if(number == '#')  {  state = 0;  segPower = 1;  PWM\_PWR(0);  }  break;    }  }    } |

## Disassembly Listing File

|  |
| --- |
| Disassembly Listing for Lab05  Generated From:  /home/ben/Sync/school/CE420/Lab05/Lab05.X/dist/default/debug/Lab05.X.debug.elf  Nov 14, 2019 6:25:45 PM  --- /home/ben/Sync/school/CE420/Lab05/Lab05.X/uart.c --------------------------------------------------  1: /\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \*/  2: /\*\*  3: @Company  4: Kettering University  5:  6: @File Name  7: uart.c  8:  9: @Summary  10: Implements the functions that support the UART serial output communication  11:  12: @Description  13:  14: @Author  15: Girma Tewolde  16: @Last modification  17: 10/28/2019 @ 3:30 PM  18:  19: \*/  20: /\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \*/  21:  22: #include "uart.h"  23:  24: void initUART1(void)  25: {  9D000EAC 27BDFFF8 ADDIU SP, SP, -8  9D000EB0 AFBE0004 SW FP, 4(SP)  9D000EB4 03A0F021 ADDU FP, SP, ZERO  26: U1BRG = BRATE; //initialize the baud rate generator  9D000EB8 3C02BF80 LUI V0, -16512  9D000EBC 24030411 ADDIU V1, ZERO, 1041  9D000EC0 AC436040 SW V1, 24640(V0)  27: U1MODE = U\_ENABLE; //initialize the UART module  9D000EC4 3C02BF80 LUI V0, -16512  9D000EC8 34038008 ORI V1, ZERO, -32760  9D000ECC AC436000 SW V1, 24576(V0)  28: U1STA = U\_TXRX; //enable TX & RX  9D000ED0 3C02BF80 LUI V0, -16512  9D000ED4 24031400 ADDIU V1, ZERO, 5120  9D000ED8 AC436010 SW V1, 24592(V0)  29: }  9D000EDC 03C0E821 ADDU SP, FP, ZERO  9D000EE0 8FBE0004 LW FP, 4(SP)  9D000EE4 27BD0008 ADDIU SP, SP, 8  9D000EE8 03E00008 JR RA  9D000EEC 00000000 NOP  30:  31: void UART1\_putchar(uint8\_t c) { //send a character to UART1  9D000EF0 27BDFFF8 ADDIU SP, SP, -8  9D000EF4 AFBE0004 SW FP, 4(SP)  9D000EF8 03A0F021 ADDU FP, SP, ZERO  9D000EFC 00801021 ADDU V0, A0, ZERO  9D000F00 A3C20008 SB V0, 8(FP)  32: while (U1STAbits.UTXBF == 1); //wait until transmitter buffer becomes empty  9D000F04 00000000 NOP  9D000F08 3C02BF80 LUI V0, -16512  9D000F0C 8C426010 LW V0, 24592(V0)  9D000F10 30420200 ANDI V0, V0, 512  9D000F14 1440FFFC BNE V0, ZERO, 0x9D000F08  9D000F18 00000000 NOP  33: U1TXREG = c; //write character to TX data register  9D000F1C 93C30008 LBU V1, 8(FP)  9D000F20 3C02BF80 LUI V0, -16512  9D000F24 AC436020 SW V1, 24608(V0)  34: }  9D000F28 03C0E821 ADDU SP, FP, ZERO  9D000F2C 8FBE0004 LW FP, 4(SP)  9D000F30 27BD0008 ADDIU SP, SP, 8  9D000F34 03E00008 JR RA  9D000F38 00000000 NOP  35: void UART1\_putstr(uint8\_t s[]) { //send a null-terminated string to UART1  9D000F3C 27BDFFE8 ADDIU SP, SP, -24  9D000F40 AFBF0014 SW RA, 20(SP)  9D000F44 AFBE0010 SW FP, 16(SP)  9D000F48 03A0F021 ADDU FP, SP, ZERO  9D000F4C AFC40018 SW A0, 24(FP)  36: while (s[0] != 0) {  9D000F50 0B4003DE J 0x9D000F78  9D000F54 00000000 NOP  9D000F78 8FC20018 LW V0, 24(FP)  9D000F7C 90420000 LBU V0, 0(V0)  9D000F80 1440FFF5 BNE V0, ZERO, 0x9D000F58  9D000F84 00000000 NOP  37: UART1\_putchar(s[0]);  9D000F58 8FC20018 LW V0, 24(FP)  9D000F5C 90420000 LBU V0, 0(V0)  9D000F60 00402021 ADDU A0, V0, ZERO  9D000F64 0F4003BC JAL UART1\_putchar  9D000F68 00000000 NOP  38: s++;  9D000F6C 8FC20018 LW V0, 24(FP)  9D000F70 24420001 ADDIU V0, V0, 1  9D000F74 AFC20018 SW V0, 24(FP)  39: }  40: }  9D000F88 03C0E821 ADDU SP, FP, ZERO  9D000F8C 8FBF0014 LW RA, 20(SP)  9D000F90 8FBE0010 LW FP, 16(SP)  9D000F94 27BD0018 ADDIU SP, SP, 24  9D000F98 03E00008 JR RA  9D000F9C 00000000 NOP  41:  42: /\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  43: End of File  44: \*/  --- /home/ben/Sync/school/CE420/Lab05/Lab05.X/seven\_seg.c ---------------------------------------------  1: // seven\_seg.c  2: //this is an incomplete program code for the seven segment device driver code  3: #include "seven\_seg.h" //assuming you named your header file this way  4:  5: //Example -- initialize Timer1 to trigger interrupts every 1ms  6: void init\_Timer1(void) {  9D000B44 27BDFFE8 ADDIU SP, SP, -24  9D000B48 AFBF0014 SW RA, 20(SP)  9D000B4C AFBE0010 SW FP, 16(SP)  9D000B50 03A0F021 ADDU FP, SP, ZERO  7: TMR1 = 0x0000; //clear timer register  9D000B54 3C02BF80 LUI V0, -16512  9D000B58 AC400610 SW ZERO, 1552(V0)  8: PR1 = 0x9c; //set the period  9D000B5C 3C02BF80 LUI V0, -16512  9D000B60 2403009C ADDIU V1, ZERO, 156  9D000B64 AC430620 SW V1, 1568(V0)  9:  10: //init interrupts  11: mT1SetIntPriority(5); //group priority set to 5  9D000B68 3C02BF88 LUI V0, -16504  9D000B6C 2403001C ADDIU V1, ZERO, 28  9D000B70 AC4310A4 SW V1, 4260(V0)  9D000B74 3C02BF88 LUI V0, -16504  9D000B78 24030014 ADDIU V1, ZERO, 20  9D000B7C AC4310A8 SW V1, 4264(V0)  12: mT1ClearIntFlag();  9D000B80 3C02BF88 LUI V0, -16504  9D000B84 24030010 ADDIU V1, ZERO, 16  9D000B88 AC431034 SW V1, 4148(V0)  13: // configure for multi-vectored mode and enable system interrupt  14: INTEnableSystemMultiVectoredInt();  9D000B8C 0F4004B8 JAL INTEnableSystemMultiVectoredInt  9D000B90 00000000 NOP  15: mT1IntEnable(1); //enable T1 interrupts  9D000B94 3C02BF88 LUI V0, -16504  9D000B98 24030010 ADDIU V1, ZERO, 16  9D000B9C AC431064 SW V1, 4196(V0)  9D000BA0 3C02BF88 LUI V0, -16504  9D000BA4 24030010 ADDIU V1, ZERO, 16  9D000BA8 AC431068 SW V1, 4200(V0)  16: T1CON = 0x8038; //enable timer, set prescaler to 1:8  9D000BAC 3C02BF80 LUI V0, -16512  9D000BB0 34038038 ORI V1, ZERO, -32712  9D000BB4 AC430600 SW V1, 1536(V0)  17: }  9D000BB8 03C0E821 ADDU SP, FP, ZERO  9D000BBC 8FBF0014 LW RA, 20(SP)  9D000BC0 8FBE0010 LW FP, 16(SP)  9D000BC4 27BD0018 ADDIU SP, SP, 24  9D000BC8 03E00008 JR RA  9D000BCC 00000000 NOP  18: void init7Seg(void)  19: {  9D000BD0 27BDFFE8 ADDIU SP, SP, -24  9D000BD4 AFBF0014 SW RA, 20(SP)  9D000BD8 AFBE0010 SW FP, 16(SP)  9D000BDC 03A0F021 ADDU FP, SP, ZERO  20: init\_Timer1();  9D000BE0 0F4002D1 JAL init\_Timer1  9D000BE4 00000000 NOP  21: TRISEbits.TRISE0 = OUT;  9D000BE8 3C03BF88 LUI V1, -16504  9D000BEC 90626100 LBU V0, 24832(V1)  9D000BF0 7C020004 INS V0, ZERO, 0, 1  9D000BF4 A0626100 SB V0, 24832(V1)  22: TRISEbits.TRISE1 = OUT;  9D000BF8 3C03BF88 LUI V1, -16504  9D000BFC 90626100 LBU V0, 24832(V1)  9D000C00 7C020844 INS V0, ZERO, 1, 1  9D000C04 A0626100 SB V0, 24832(V1)  23: TRISEbits.TRISE2 = OUT;  9D000C08 3C03BF88 LUI V1, -16504  9D000C0C 90626100 LBU V0, 24832(V1)  9D000C10 7C021084 INS V0, ZERO, 2, 1  9D000C14 A0626100 SB V0, 24832(V1)  24: TRISEbits.TRISE3 = OUT;  9D000C18 3C03BF88 LUI V1, -16504  9D000C1C 90626100 LBU V0, 24832(V1)  9D000C20 7C0218C4 INS V0, ZERO, 3, 1  9D000C24 A0626100 SB V0, 24832(V1)  25: TRISEbits.TRISE4 = OUT;  9D000C28 3C03BF88 LUI V1, -16504  9D000C2C 90626100 LBU V0, 24832(V1)  9D000C30 7C022104 INS V0, ZERO, 4, 1  9D000C34 A0626100 SB V0, 24832(V1)  26: TRISEbits.TRISE5 = OUT;  9D000C38 3C03BF88 LUI V1, -16504  9D000C3C 90626100 LBU V0, 24832(V1)  9D000C40 7C022944 INS V0, ZERO, 5, 1  9D000C44 A0626100 SB V0, 24832(V1)  27: TRISEbits.TRISE6 = OUT;  9D000C48 3C03BF88 LUI V1, -16504  9D000C4C 90626100 LBU V0, 24832(V1)  9D000C50 7C023184 INS V0, ZERO, 6, 1  9D000C54 A0626100 SB V0, 24832(V1)  28: TRISEbits.TRISE7 = OUT;  9D000C58 3C03BF88 LUI V1, -16504  9D000C5C 90626100 LBU V0, 24832(V1)  9D000C60 7C0239C4 INS V0, ZERO, 7, 1  9D000C64 A0626100 SB V0, 24832(V1)  29:  30: TRISDbits.TRISD2 = OUT;  9D000C68 3C03BF88 LUI V1, -16504  9D000C6C 946260C0 LHU V0, 24768(V1)  9D000C70 7C021084 INS V0, ZERO, 2, 1  9D000C74 A46260C0 SH V0, 24768(V1)  31: TRISDbits.TRISD9 = OUT;  9D000C78 3C03BF88 LUI V1, -16504  9D000C7C 946260C0 LHU V0, 24768(V1)  9D000C80 7C024A44 INS V0, ZERO, 9, 1  9D000C84 A46260C0 SH V0, 24768(V1)  32:  33: segPower = 1;  9D000C88 24020001 ADDIU V0, ZERO, 1  9D000C8C AF82801C SW V0, -32740(GP)  34: }  9D000C90 03C0E821 ADDU SP, FP, ZERO  9D000C94 8FBF0014 LW RA, 20(SP)  9D000C98 8FBE0010 LW FP, 16(SP)  9D000C9C 27BD0018 ADDIU SP, SP, 24  9D000CA0 03E00008 JR RA  9D000CA4 00000000 NOP  35:  36: //void set7Seg(char numbers[4])  37: //{  38: // segValues[0] = numbers[0];  39: // segValues[1] = numbers[1];  40: // segValues[2] = numbers[2];  41: // segValues[3] = numbers[3];  42: //  43: //}  44:  45: //Timer1 interrupt handler  46: //it is used to refresh the 7-segment displays every 1ms  47: void \_\_ISR(\_TIMER\_1\_VECTOR, IPL5SOFT) T1ISR(void)  48: {  9D000CA8 415DE800 RDPGPR SP, SP  9D000CAC 401A7000 MFC0 K0, EPC  9D000CB0 401B6000 MFC0 K1, Status  9D000CB4 27BDFFE0 ADDIU SP, SP, -32  9D000CB8 AFBA001C SW K0, 28(SP)  9D000CBC 401A6002 MFC0 K0, SRSCtl  9D000CC0 AFBB0018 SW K1, 24(SP)  9D000CC4 AFBA0014 SW K0, 20(SP)  9D000CC8 7C1B7844 INS K1, ZERO, 1, 15  9D000CCC 377B1400 ORI K1, K1, 5120  9D000CD0 409B6000 MTC0 K1, Status  9D000CD4 AFBE000C SW FP, 12(SP)  9D000CD8 AFA40008 SW A0, 8(SP)  9D000CDC AFA30004 SW V1, 4(SP)  9D000CE0 AFA20000 SW V0, 0(SP)  9D000CE4 03A0F021 ADDU FP, SP, ZERO  49: mT1ClearIntFlag();  9D000CE8 3C02BF88 LUI V0, -16504  9D000CEC 24030010 ADDIU V1, ZERO, 16  9D000CF0 AC431034 SW V1, 4148(V0)  50:  51: if(segPower == 1)  9D000CF4 8F83801C LW V1, -32740(GP)  9D000CF8 24020001 ADDIU V0, ZERO, 1  9D000CFC 14620058 BNE V1, V0, 0x9D000E60  9D000D00 00000000 NOP  52: {  53: PORTE = segLookup[segValues[segState]];  9D000D04 8F838010 LW V1, -32752(GP)  9D000D08 3C02A000 LUI V0, -24576  9D000D0C 00031880 SLL V1, V1, 2  9D000D10 24420218 ADDIU V0, V0, 536  9D000D14 00621021 ADDU V0, V1, V0  9D000D18 8C430000 LW V1, 0(V0)  9D000D1C 3C02A000 LUI V0, -24576  9D000D20 24420238 ADDIU V0, V0, 568  9D000D24 00621021 ADDU V0, V1, V0  9D000D28 80420000 LB V0, 0(V0)  9D000D2C 00401821 ADDU V1, V0, ZERO  9D000D30 3C02BF88 LUI V0, -16504  9D000D34 AC436110 SW V1, 24848(V0)  54: //PORTECLR = ~ segLookup[segValues[segState]];  55:  56: switch(segState)  9D000D38 8F828010 LW V0, -32752(GP)  9D000D3C 24030001 ADDIU V1, ZERO, 1  9D000D40 10430019 BEQ V0, V1, 0x9D000DA8  9D000D44 00000000 NOP  9D000D48 2C430001 SLTIU V1, V0, 1  9D000D4C 14600009 BNE V1, ZERO, 0x9D000D74  9D000D50 00000000 NOP  9D000D54 24030002 ADDIU V1, ZERO, 2  9D000D58 10430021 BEQ V0, V1, 0x9D000DE0  9D000D5C 00000000 NOP  9D000D60 24030003 ADDIU V1, ZERO, 3  9D000D64 10430029 BEQ V0, V1, 0x9D000E0C  9D000D68 00000000 NOP  9D000D6C 0B40038E J 0x9D000E38  9D000D70 00000000 NOP  57: {  58: case 0:  59: PORTDbits.RD2 = 0;  9D000D74 3C03BF88 LUI V1, -16504  9D000D78 946260D0 LHU V0, 24784(V1)  9D000D7C 7C021084 INS V0, ZERO, 2, 1  9D000D80 A46260D0 SH V0, 24784(V1)  60: PORTDbits.RD9 = 0;  9D000D84 3C03BF88 LUI V1, -16504  9D000D88 946260D0 LHU V0, 24784(V1)  9D000D8C 7C024A44 INS V0, ZERO, 9, 1  9D000D90 A46260D0 SH V0, 24784(V1)  61: PORTESET = 0b10000000;  9D000D94 3C02BF88 LUI V0, -16504  9D000D98 24030080 ADDIU V1, ZERO, 128  9D000D9C AC436118 SW V1, 24856(V0)  62: break;  9D000DA0 0B40038E J 0x9D000E38  9D000DA4 00000000 NOP  63: case 1:  64: PORTDbits.RD2 = 1;  9D000DA8 3C03BF88 LUI V1, -16504  9D000DAC 946260D0 LHU V0, 24784(V1)  9D000DB0 24040001 ADDIU A0, ZERO, 1  9D000DB4 7C821084 INS V0, A0, 2, 1  9D000DB8 A46260D0 SH V0, 24784(V1)  65: PORTDbits.RD9 = 0;  9D000DBC 3C03BF88 LUI V1, -16504  9D000DC0 946260D0 LHU V0, 24784(V1)  9D000DC4 7C024A44 INS V0, ZERO, 9, 1  9D000DC8 A46260D0 SH V0, 24784(V1)  66: PORTECLR = 0b10000000;  9D000DCC 3C02BF88 LUI V0, -16504  9D000DD0 24030080 ADDIU V1, ZERO, 128  9D000DD4 AC436114 SW V1, 24852(V0)  67: break;  9D000DD8 0B40038E J 0x9D000E38  9D000DDC 00000000 NOP  68: case 2:  69: PORTDbits.RD2 = 0;  9D000DE0 3C03BF88 LUI V1, -16504  9D000DE4 946260D0 LHU V0, 24784(V1)  9D000DE8 7C021084 INS V0, ZERO, 2, 1  9D000DEC A46260D0 SH V0, 24784(V1)  70: PORTDbits.RD9 = 1;  9D000DF0 3C03BF88 LUI V1, -16504  9D000DF4 946260D0 LHU V0, 24784(V1)  9D000DF8 24040001 ADDIU A0, ZERO, 1  9D000DFC 7C824A44 INS V0, A0, 9, 1  9D000E00 A46260D0 SH V0, 24784(V1)  71: break;  9D000E04 0B40038E J 0x9D000E38  9D000E08 00000000 NOP  72: case 3:  73: PORTDbits.RD2 = 1;  9D000E0C 3C03BF88 LUI V1, -16504  9D000E10 946260D0 LHU V0, 24784(V1)  9D000E14 24040001 ADDIU A0, ZERO, 1  9D000E18 7C821084 INS V0, A0, 2, 1  9D000E1C A46260D0 SH V0, 24784(V1)  74: PORTDbits.RD9 = 1;  9D000E20 3C03BF88 LUI V1, -16504  9D000E24 946260D0 LHU V0, 24784(V1)  9D000E28 24040001 ADDIU A0, ZERO, 1  9D000E2C 7C824A44 INS V0, A0, 9, 1  9D000E30 A46260D0 SH V0, 24784(V1)  75: break;  9D000E34 00000000 NOP  76: }  77:  78: segState++;  9D000E38 8F828010 LW V0, -32752(GP)  9D000E3C 24420001 ADDIU V0, V0, 1  9D000E40 AF828010 SW V0, -32752(GP)  79: if(segState > 3)  9D000E44 8F828010 LW V0, -32752(GP)  9D000E48 2C420004 SLTIU V0, V0, 4  9D000E4C 14400007 BNE V0, ZERO, 0x9D000E6C  9D000E50 00000000 NOP  80: {  81: segState = 0;  9D000E54 AF808010 SW ZERO, -32752(GP)  9D000E58 0B40039B J 0x9D000E6C  9D000E5C 00000000 NOP  82: }  83: }  84: else  85: {  86: PORTE = 0xFF;  9D000E60 3C02BF88 LUI V0, -16504  9D000E64 240300FF ADDIU V1, ZERO, 255  9D000E68 AC436110 SW V1, 24848(V0)  87: }  88: }  9D000E6C 03C0E821 ADDU SP, FP, ZERO  9D000E70 8FBE000C LW FP, 12(SP)  9D000E74 8FA40008 LW A0, 8(SP)  9D000E78 8FA30004 LW V1, 4(SP)  9D000E7C 8FA20000 LW V0, 0(SP)  9D000E80 41606000 DI ZERO  9D000E84 000000C0 EHB  9D000E88 8FBA001C LW K0, 28(SP)  9D000E8C 8FBB0018 LW K1, 24(SP)  9D000E90 409A7000 MTC0 K0, EPC  9D000E94 8FBA0014 LW K0, 20(SP)  9D000E98 27BD0020 ADDIU SP, SP, 32  9D000E9C 409A6002 MTC0 K0, SRSCtl  9D000EA0 41DDE800 WRPGPR SP, SP  9D000EA4 409B6000 MTC0 K1, Status  9D000EA8 42000018 ERET  --- /home/ben/Sync/school/CE420/Lab05/Lab05.X/main.c --------------------------------------------------  1: //provide complete documentation  2:  3: #include <p32xxxx.h>  4: #include "keypad.h"  5: #include "uart.h"  6: #include "seven\_seg.h"  7:  8: // configuration bit settings, Fcy=80MHz, Fpb=40MHz  9: #pragma config POSCMOD=XT, FNOSC=PRIPLL  10: #pragma config FPLLIDIV=DIV\_2, FPLLMUL=MUL\_20, FPLLODIV=DIV\_1  11: #pragma config FPBDIV=DIV\_2, FWDTEN=OFF, CP=OFF, BWP=OFF  12:  13: volatile int minutes;  14: volatile int seconds;  15: volatile int state = 0;  16:  17: #define PWM\_PWR(x) (x==1? (OC4CONSET = 0x8000) : (OC4CONCLR = 0x8000))  18:  19: void init\_Timer4(void)  20: {  9D000000 27BDFFF8 ADDIU SP, SP, -8  9D000004 AFBE0004 SW FP, 4(SP)  9D000008 03A0F021 ADDU FP, SP, ZERO  21: T4CON = 0x0; // Stop the timer and clear the control register,  9D00000C 3C02BF80 LUI V0, -16512  9D000010 AC400C00 SW ZERO, 3072(V0)  22: T4CON = 0b01111000; // prescaler at 1:256,internal clock source, 32bit mode  9D000014 3C02BF80 LUI V0, -16512  9D000018 24030078 ADDIU V1, ZERO, 120  9D00001C AC430C00 SW V1, 3072(V0)  23: TMR4 = 0x0; // Clear the timer register  9D000020 3C02BF80 LUI V0, -16512  9D000024 AC400C10 SW ZERO, 3088(V0)  24: PR4 = 0x2625a; // Load the period register  9D000028 3C02BF80 LUI V0, -16512  9D00002C 3C030002 LUI V1, 2  9D000030 3463625A ORI V1, V1, 25178  9D000034 AC430C20 SW V1, 3104(V0)  25: IPC5SET = 0x0000000D; // Set priority level = 3, subpriority level = 1  9D000038 3C02BF88 LUI V0, -16504  9D00003C 2403000D ADDIU V1, ZERO, 13  9D000040 AC4310E8 SW V1, 4328(V0)  26: IFS0CLR = 0x00100000; // Clear the timer interrupt status flag  9D000044 3C02BF88 LUI V0, -16504  9D000048 3C030010 LUI V1, 16  9D00004C AC431034 SW V1, 4148(V0)  27: IEC0SET = 0x00100000; // Enable timer interrupts  9D000050 3C02BF88 LUI V0, -16504  9D000054 3C030010 LUI V1, 16  9D000058 AC431068 SW V1, 4200(V0)  28: T4CONSET = 0x8000; // Start the timer  9D00005C 3C02BF80 LUI V0, -16512  9D000060 34038000 ORI V1, ZERO, -32768  9D000064 AC430C08 SW V1, 3080(V0)  29:  30: }  9D000068 03C0E821 ADDU SP, FP, ZERO  9D00006C 8FBE0004 LW FP, 4(SP)  9D000070 27BD0008 ADDIU SP, SP, 8  9D000074 03E00008 JR RA  9D000078 00000000 NOP  31:  32: void init\_PWM(void)  33: {  9D00007C 27BDFFF8 ADDIU SP, SP, -8  9D000080 AFBE0004 SW FP, 4(SP)  9D000084 03A0F021 ADDU FP, SP, ZERO  34: OC4CON = 0x0000; // Turn off the OC1 when performing the setup  9D000088 3C02BF80 LUI V0, -16512  9D00008C AC403600 SW ZERO, 13824(V0)  35: OC4R = 0x0F00; // Initialize primary Compare register  9D000090 3C02BF80 LUI V0, -16512  9D000094 24030F00 ADDIU V1, ZERO, 3840  9D000098 AC433610 SW V1, 13840(V0)  36: OC4RS = 0x0F00; // Initialize secondary Compare register  9D00009C 3C02BF80 LUI V0, -16512  9D0000A0 24030F00 ADDIU V1, ZERO, 3840  9D0000A4 AC433620 SW V1, 13856(V0)  37: OC4CON = 0x0006; // Configure for PWM mode without Fault pin  9D0000A8 3C02BF80 LUI V0, -16512  9D0000AC 24030006 ADDIU V1, ZERO, 6  9D0000B0 AC433600 SW V1, 13824(V0)  38: // enabled  39: PR2 = 0xffff; // Set period  9D0000B4 3C02BF80 LUI V0, -16512  9D0000B8 3403FFFF ORI V1, ZERO, -1  9D0000BC AC430820 SW V1, 2080(V0)  40: // Configure Timer2 interrupt. Note that in PWM mode, the  41: // corresponding source timer interrupt flag is asserted.  42: // OC interrupt is not generated in PWM mode.  43: // IFS0CLR = 0x00000100; // Clear the T2 interrupt flag  44: // IEC0SET = 0x00000100; // Enable T2 interrupt  45: // IPC2SET = 0x0000001C; // Set T2 interrupt priority to 7  46:  47: T2CONSET = 0x8000; // Enable Timer2  9D0000C0 3C02BF80 LUI V0, -16512  9D0000C4 34038000 ORI V1, ZERO, -32768  9D0000C8 AC430808 SW V1, 2056(V0)  48: // OC4CONSET = 0x8000; // Enable OC1  49: }  9D0000CC 03C0E821 ADDU SP, FP, ZERO  9D0000D0 8FBE0004 LW FP, 4(SP)  9D0000D4 27BD0008 ADDIU SP, SP, 8  9D0000D8 03E00008 JR RA  9D0000DC 00000000 NOP  50:  51: void send\_PWM(signed int speed)  52: {  9D0000E0 27BDFFF8 ADDIU SP, SP, -8  9D0000E4 AFBE0004 SW FP, 4(SP)  9D0000E8 03A0F021 ADDU FP, SP, ZERO  9D0000EC AFC40008 SW A0, 8(FP)  53: /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  54: \* this function takes a speed value  55: \* in percentage from -100 to 100  56: \* and sets the pwm module accordingly  57: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  58:  59: OC1RS = ((speed \* 4) + 3000);  9D0000F0 8FC20008 LW V0, 8(FP)  9D0000F4 244202EE ADDIU V0, V0, 750  9D0000F8 00021080 SLL V0, V0, 2  9D0000FC 00401821 ADDU V1, V0, ZERO  9D000100 3C02BF80 LUI V0, -16512  9D000104 AC433020 SW V1, 12320(V0)  60: //CCP4RB = ((speed \* 4) + 3000);  61: //CCP4CON1Lbits.CCPON = 1; //Turn on MCCP module  62: }  9D000108 03C0E821 ADDU SP, FP, ZERO  9D00010C 8FBE0004 LW FP, 4(SP)  9D000110 27BD0008 ADDIU SP, SP, 8  9D000114 03E00008 JR RA  9D000118 00000000 NOP  63:  64: //Timer4 interrupt handler  65: void \_\_ISR(\_TIMER\_5\_VECTOR, IPL5SOFT) T5ISR(void)  66: {  9D00011C 415DE800 RDPGPR SP, SP  9D000120 401A7000 MFC0 K0, EPC  9D000124 401B6000 MFC0 K1, Status  9D000128 27BDFFD8 ADDIU SP, SP, -40  9D00012C AFBA0024 SW K0, 36(SP)  9D000130 401A6002 MFC0 K0, SRSCtl  9D000134 AFBB0020 SW K1, 32(SP)  9D000138 AFBA001C SW K0, 28(SP)  9D00013C 7C1B7844 INS K1, ZERO, 1, 15  9D000140 377B1400 ORI K1, K1, 5120  9D000144 409B6000 MTC0 K1, Status  9D000148 AFBE000C SW FP, 12(SP)  9D00014C AFA40008 SW A0, 8(SP)  9D000150 AFA30004 SW V1, 4(SP)  9D000154 AFA20000 SW V0, 0(SP)  9D000158 00001012 MFLO V0  9D00015C AFA20014 SW V0, 20(SP)  9D000160 00001810 MFHI V1  9D000164 AFA30010 SW V1, 16(SP)  9D000168 03A0F021 ADDU FP, SP, ZERO  67: mT5ClearIntFlag();  9D00016C 3C02BF88 LUI V0, -16504  9D000170 3C030010 LUI V1, 16  9D000174 AC431034 SW V1, 4148(V0)  68:  69: switch(state)  9D000178 8F828018 LW V0, -32744(GP)  9D00017C 24030001 ADDIU V1, ZERO, 1  9D000180 10430006 BEQ V0, V1, 0x9D00019C  9D000184 00000000 NOP  9D000188 24030002 ADDIU V1, ZERO, 2  9D00018C 1043006C BEQ V0, V1, 0x9D000340  9D000190 00000000 NOP  70: {  71: case 0:  72:  73: break;  9D000194 0B4000E1 J 0x9D000384  9D000198 00000000 NOP  74: case 1:  75:  76: if(seconds == 0)  9D00019C 8F828020 LW V0, -32736(GP)  9D0001A0 14400046 BNE V0, ZERO, 0x9D0002BC  9D0001A4 00000000 NOP  77: {  78: if(minutes == 0)  9D0001A8 8F828024 LW V0, -32732(GP)  9D0001AC 14400005 BNE V0, ZERO, 0x9D0001C4  9D0001B0 00000000 NOP  79: {  80: state = 2;  9D0001B4 24020002 ADDIU V0, ZERO, 2  9D0001B8 AF828018 SW V0, -32744(GP)  81:  82: }  83: else  84: {  85: minutes--;  9D0001C4 8F828024 LW V0, -32732(GP)  9D0001C8 2442FFFF ADDIU V0, V0, -1  9D0001CC AF828024 SW V0, -32732(GP)  86:  87: //new values for the minute display segments  88: segValues[0] = minutes / 10;  9D0001D0 8F828024 LW V0, -32732(GP)  9D0001D4 3C036666 LUI V1, 26214  9D0001D8 34636667 ORI V1, V1, 26215  9D0001DC 00430018 MULT 0, V0, V1  9D0001E0 00001810 MFHI V1  9D0001E4 00031883 SRA V1, V1, 2  9D0001E8 000217C3 SRA V0, V0, 31  9D0001EC 00621023 SUBU V0, V1, V0  9D0001F0 00401821 ADDU V1, V0, ZERO  9D0001F4 3C02A000 LUI V0, -24576  9D0001F8 AC430218 SW V1, 536(V0)  89: segValues[1] = minutes % 10;  9D0001FC 8F838024 LW V1, -32732(GP)  9D000200 3C026666 LUI V0, 26214  9D000204 34426667 ORI V0, V0, 26215  9D000208 00620018 MULT 0, V1, V0  9D00020C 00001010 MFHI V0  9D000210 00022083 SRA A0, V0, 2  9D000214 000317C3 SRA V0, V1, 31  9D000218 00821023 SUBU V0, A0, V0  9D00021C 00021040 SLL V0, V0, 1  9D000220 00022080 SLL A0, V0, 2  9D000224 00441021 ADDU V0, V0, A0  9D000228 00621023 SUBU V0, V1, V0  9D00022C 00401821 ADDU V1, V0, ZERO  9D000230 3C02A000 LUI V0, -24576  9D000234 24420218 ADDIU V0, V0, 536  9D000238 AC430004 SW V1, 4(V0)  90:  91: //start seconds on 60 for end of minute  92: seconds = 60;  9D00023C 2402003C ADDIU V0, ZERO, 60  9D000240 AF828020 SW V0, -32736(GP)  93: //updates the vlaues for the second display segments  94: segValues[2] = seconds / 10;  9D000244 8F828020 LW V0, -32736(GP)  9D000248 3C036666 LUI V1, 26214  9D00024C 34636667 ORI V1, V1, 26215  9D000250 00430018 MULT 0, V0, V1  9D000254 00001810 MFHI V1  9D000258 00031883 SRA V1, V1, 2  9D00025C 000217C3 SRA V0, V0, 31  9D000260 00621023 SUBU V0, V1, V0  9D000264 00401821 ADDU V1, V0, ZERO  9D000268 3C02A000 LUI V0, -24576  9D00026C 24420218 ADDIU V0, V0, 536  9D000270 AC430008 SW V1, 8(V0)  95: segValues[3] = seconds % 10;  9D000274 8F838020 LW V1, -32736(GP)  9D000278 3C026666 LUI V0, 26214  9D00027C 34426667 ORI V0, V0, 26215  9D000280 00620018 MULT 0, V1, V0  9D000284 00001010 MFHI V0  9D000288 00022083 SRA A0, V0, 2  9D00028C 000317C3 SRA V0, V1, 31  9D000290 00821023 SUBU V0, A0, V0  9D000294 00021040 SLL V0, V0, 1  9D000298 00022080 SLL A0, V0, 2  9D00029C 00441021 ADDU V0, V0, A0  9D0002A0 00621023 SUBU V0, V1, V0  9D0002A4 00401821 ADDU V1, V0, ZERO  9D0002A8 3C02A000 LUI V0, -24576  9D0002AC 24420218 ADDIU V0, V0, 536  9D0002B0 AC43000C SW V1, 12(V0)  96: }  97: }  98: else  99: {  100: seconds--;  9D0002BC 8F828020 LW V0, -32736(GP)  9D0002C0 2442FFFF ADDIU V0, V0, -1  9D0002C4 AF828020 SW V0, -32736(GP)  101:  102: //new values for the second display segments  103: segValues[2] = seconds / 10;  9D0002C8 8F828020 LW V0, -32736(GP)  9D0002CC 3C036666 LUI V1, 26214  9D0002D0 34636667 ORI V1, V1, 26215  9D0002D4 00430018 MULT 0, V0, V1  9D0002D8 00001810 MFHI V1  9D0002DC 00031883 SRA V1, V1, 2  9D0002E0 000217C3 SRA V0, V0, 31  9D0002E4 00621023 SUBU V0, V1, V0  9D0002E8 00401821 ADDU V1, V0, ZERO  9D0002EC 3C02A000 LUI V0, -24576  9D0002F0 24420218 ADDIU V0, V0, 536  9D0002F4 AC430008 SW V1, 8(V0)  104: segValues[3] = seconds % 10;  9D0002F8 8F838020 LW V1, -32736(GP)  9D0002FC 3C026666 LUI V0, 26214  9D000300 34426667 ORI V0, V0, 26215  9D000304 00620018 MULT 0, V1, V0  9D000308 00001010 MFHI V0  9D00030C 00022083 SRA A0, V0, 2  9D000310 000317C3 SRA V0, V1, 31  9D000314 00821023 SUBU V0, A0, V0  9D000318 00021040 SLL V0, V0, 1  9D00031C 00022080 SLL A0, V0, 2  9D000320 00441021 ADDU V0, V0, A0  9D000324 00621023 SUBU V0, V1, V0  9D000328 00401821 ADDU V1, V0, ZERO  9D00032C 3C02A000 LUI V0, -24576  9D000330 24420218 ADDIU V0, V0, 536  9D000334 AC43000C SW V1, 12(V0)  105: }  106: break;  9D0001BC 0B4000E1 J 0x9D000384  9D0001C0 00000000 NOP  9D0002B4 0B4000E1 J 0x9D000384  9D0002B8 00000000 NOP  9D000338 0B4000E1 J 0x9D000384  9D00033C 00000000 NOP  107:  108: case 2:  109: segPower = ! segPower;  9D000340 8F82801C LW V0, -32740(GP)  9D000344 2C420001 SLTIU V0, V0, 1  9D000348 304200FF ANDI V0, V0, 255  9D00034C AF82801C SW V0, -32740(GP)  110: PWM\_PWR(segPower);  9D000350 8F83801C LW V1, -32740(GP)  9D000354 24020001 ADDIU V0, ZERO, 1  9D000358 14620006 BNE V1, V0, 0x9D000374  9D00035C 00000000 NOP  9D000360 34038000 ORI V1, ZERO, -32768  9D000364 3C02BF80 LUI V0, -16512  9D000368 AC433608 SW V1, 13832(V0)  9D00036C 0B4000E0 J 0x9D000380  9D000370 00000000 NOP  9D000374 34038000 ORI V1, ZERO, -32768  9D000378 3C02BF80 LUI V0, -16512  9D00037C AC433604 SW V1, 13828(V0)  111: break;  9D000380 00000000 NOP  112: }  113:  114:  115:  116:  117:  118: }  9D000384 03C0E821 ADDU SP, FP, ZERO  9D000388 8FA20014 LW V0, 20(SP)  9D00038C 00400013 MTLO V0  9D000390 8FA30010 LW V1, 16(SP)  9D000394 00600011 MTHI V1  9D000398 8FBE000C LW FP, 12(SP)  9D00039C 8FA40008 LW A0, 8(SP)  9D0003A0 8FA30004 LW V1, 4(SP)  9D0003A4 8FA20000 LW V0, 0(SP)  9D0003A8 41606000 DI ZERO  9D0003AC 000000C0 EHB  9D0003B0 8FBA0024 LW K0, 36(SP)  9D0003B4 8FBB0020 LW K1, 32(SP)  9D0003B8 409A7000 MTC0 K0, EPC  9D0003BC 8FBA001C LW K0, 28(SP)  9D0003C0 27BD0028 ADDIU SP, SP, 40  9D0003C4 409A6002 MTC0 K0, SRSCtl  9D0003C8 41DDE800 WRPGPR SP, SP  9D0003CC 409B6000 MTC0 K1, Status  9D0003D0 42000018 ERET  119:  120: int main(void) {  9D0003D4 27BDFFE0 ADDIU SP, SP, -32  9D0003D8 AFBF001C SW RA, 28(SP)  9D0003DC AFBE0018 SW FP, 24(SP)  9D0003E0 03A0F021 ADDU FP, SP, ZERO  121: DDPCONbits.JTAGEN = 0;  9D0003E4 3C03BF81 LUI V1, -16511  9D0003E8 9062F200 LBU V0, -3584(V1)  9D0003EC 7C0218C4 INS V0, ZERO, 3, 1  9D0003F0 A062F200 SB V0, -3584(V1)  122: //declare any local variables you may need  123: unsigned char number;  124:  125: initKeypad();  9D0003F4 0F400170 JAL initKeypad  9D0003F8 00000000 NOP  126: initUART1();  9D0003FC 0F4003AB JAL initUART1  9D000400 00000000 NOP  127: init7Seg();  9D000404 0F4002F4 JAL init7Seg  9D000408 00000000 NOP  128: init\_Timer4();  9D00040C 0F400000 JAL init\_Timer4  9D000410 00000000 NOP  129: init\_PWM();  9D000414 0F40001F JAL init\_PWM  9D000418 00000000 NOP  130:  131: UART1\_putstr("Hello, please enter keys on the keypad:");  9D00041C 3C029D00 LUI V0, -25344  9D000420 24441314 ADDIU A0, V0, 4884  9D000424 0F4003CF JAL UART1\_putstr  9D000428 00000000 NOP  132:  133: //add your code here  134:  135: //infinite loop  136: while (1)  137: {  138: number = readKey();  9D00042C 0F4001AE JAL readKey  9D000430 00000000 NOP  9D000434 A3C20010 SB V0, 16(FP)  139: if(number != '\0')  9D000438 93C20010 LBU V0, 16(FP)  9D00043C 1040005E BEQ V0, ZERO, 0x9D0005B8  9D000440 00000000 NOP  140: switch(state)  9D000444 8F828018 LW V0, -32744(GP)  9D000448 24030001 ADDIU V1, ZERO, 1  9D00044C 10430048 BEQ V0, V1, 0x9D000570  9D000450 00000000 NOP  9D000454 24030002 ADDIU V1, ZERO, 2  9D000458 1043004A BEQ V0, V1, 0x9D000584  9D00045C 00000000 NOP  9D000460 10400003 BEQ V0, ZERO, 0x9D000470  9D000464 00000000 NOP  141: {  142: case 0:  143:  144: if(number == '\*')  9D000470 93C30010 LBU V1, 16(FP)  9D000474 2402002A ADDIU V0, ZERO, 42  9D000478 14620003 BNE V1, V0, 0x9D000488  9D00047C 00000000 NOP  145: break;  9D000480 0B40016E J 0x9D0005B8  9D000484 00000000 NOP  146:  147: else if(number == '#')  9D000488 93C30010 LBU V1, 16(FP)  9D00048C 24020023 ADDIU V0, ZERO, 35  9D000490 14620005 BNE V1, V0, 0x9D0004A8  9D000494 00000000 NOP  148: {  149: state = 1;  9D000498 24020001 ADDIU V0, ZERO, 1  9D00049C AF828018 SW V0, -32744(GP)  150: }  151: else  152: {  153: UART1\_putchar(number);  9D0004A8 93C20010 LBU V0, 16(FP)  9D0004AC 00402021 ADDU A0, V0, ZERO  9D0004B0 0F4003BC JAL UART1\_putchar  9D0004B4 00000000 NOP  154:  155: segValues[0] = segValues[1];  9D0004B8 3C02A000 LUI V0, -24576  9D0004BC 24420218 ADDIU V0, V0, 536  9D0004C0 8C430004 LW V1, 4(V0)  9D0004C4 3C02A000 LUI V0, -24576  9D0004C8 AC430218 SW V1, 536(V0)  156: segValues[1] = segValues[2];  9D0004CC 3C02A000 LUI V0, -24576  9D0004D0 24420218 ADDIU V0, V0, 536  9D0004D4 8C430008 LW V1, 8(V0)  9D0004D8 3C02A000 LUI V0, -24576  9D0004DC 24420218 ADDIU V0, V0, 536  9D0004E0 AC430004 SW V1, 4(V0)  157: segValues[2] = segValues[3];  9D0004E4 3C02A000 LUI V0, -24576  9D0004E8 24420218 ADDIU V0, V0, 536  9D0004EC 8C43000C LW V1, 12(V0)  9D0004F0 3C02A000 LUI V0, -24576  9D0004F4 24420218 ADDIU V0, V0, 536  9D0004F8 AC430008 SW V1, 8(V0)  158: segValues[3] = number -0x30;  9D0004FC 93C20010 LBU V0, 16(FP)  9D000500 2442FFD0 ADDIU V0, V0, -48  9D000504 00401821 ADDU V1, V0, ZERO  9D000508 3C02A000 LUI V0, -24576  9D00050C 24420218 ADDIU V0, V0, 536  9D000510 AC43000C SW V1, 12(V0)  159: minutes = (segValues[0] \* 10) + segValues[1];  9D000514 3C02A000 LUI V0, -24576  9D000518 8C420218 LW V0, 536(V0)  9D00051C 00021040 SLL V0, V0, 1  9D000520 00021880 SLL V1, V0, 2  9D000524 00431821 ADDU V1, V0, V1  9D000528 3C02A000 LUI V0, -24576  9D00052C 24420218 ADDIU V0, V0, 536  9D000530 8C420004 LW V0, 4(V0)  9D000534 00621021 ADDU V0, V1, V0  9D000538 AF828024 SW V0, -32732(GP)  160: seconds = (segValues[2] \* 10) + segValues[3];  9D00053C 3C02A000 LUI V0, -24576  9D000540 24420218 ADDIU V0, V0, 536  9D000544 8C420008 LW V0, 8(V0)  9D000548 00021040 SLL V0, V0, 1  9D00054C 00021880 SLL V1, V0, 2  9D000550 00431821 ADDU V1, V0, V1  9D000554 3C02A000 LUI V0, -24576  9D000558 24420218 ADDIU V0, V0, 536  9D00055C 8C42000C LW V0, 12(V0)  9D000560 00621021 ADDU V0, V1, V0  9D000564 AF828020 SW V0, -32736(GP)  161: }  162: break;  9D0004A0 0B40016E J 0x9D0005B8  9D0004A4 00000000 NOP  9D000568 0B40016E J 0x9D0005B8  9D00056C 00000000 NOP  163: case 1:  164: if(number == '#')  9D000570 93C30010 LBU V1, 16(FP)  9D000574 24020023 ADDIU V0, ZERO, 35  9D000578 14620002 BNE V1, V0, 0x9D000584  9D00057C 00000000 NOP  165: {  166: state = 0;  9D000580 AF808018 SW ZERO, -32744(GP)  167: }  168: case 2:  169: if(number == '#')  9D000584 93C30010 LBU V1, 16(FP)  9D000588 24020023 ADDIU V0, ZERO, 35  9D00058C 14620009 BNE V1, V0, 0x9D0005B4  9D000590 00000000 NOP  170: {  171: state = 0;  9D000594 AF808018 SW ZERO, -32744(GP)  172: segPower = 1;  9D000598 24020001 ADDIU V0, ZERO, 1  9D00059C AF82801C SW V0, -32740(GP)  173: PWM\_PWR(0);  9D0005A0 3C02BF80 LUI V0, -16512  9D0005A4 34038000 ORI V1, ZERO, -32768  9D0005A8 AC433604 SW V1, 13828(V0)  174: }  175: break;  9D0005AC 0B40016D J 0x9D0005B4  9D0005B0 00000000 NOP  9D0005B4 00000000 NOP  176:  177: }  178: }  9D000468 0B40010B J .LVL6  9D00046C 00000000 NOP  9D0005B8 0B40010B J .LVL6  9D0005BC 00000000 NOP  179:  180: }  --- /home/ben/Sync/school/CE420/Lab05/Lab05.X/keypad.c ------------------------------------------------  1:  2: #include "keypad.h"  3:  4: void initKeypad(void)  5: {  9D0005C0 27BDFFF8 ADDIU SP, SP, -8  9D0005C4 AFBE0004 SW FP, 4(SP)  9D0005C8 03A0F021 ADDU FP, SP, ZERO  6: DDPCONbits.JTAGEN = 0; // disable the JTAG port  9D0005CC 3C03BF81 LUI V1, -16511  9D0005D0 9062F200 LBU V0, -3584(V1)  9D0005D4 7C0218C4 INS V0, ZERO, 3, 1  9D0005D8 A062F200 SB V0, -3584(V1)  7: AD1PCFGSET = 0xFFFF; //set all PORTB as digital  9D0005DC 3C02BF81 LUI V0, -16511  9D0005E0 3403FFFF ORI V1, ZERO, -1  9D0005E4 AC439068 SW V1, -28568(V0)  8:  9: //configure columns as outputs  10: TRISBbits.TRISB11 = OUT;  9D0005E8 3C03BF88 LUI V1, -16504  9D0005EC 94626040 LHU V0, 24640(V1)  9D0005F0 7C025AC4 INS V0, ZERO, 11, 1  9D0005F4 A4626040 SH V0, 24640(V1)  11: TRISBbits.TRISB3 = OUT;  9D0005F8 3C03BF88 LUI V1, -16504  9D0005FC 94626040 LHU V0, 24640(V1)  9D000600 7C0218C4 INS V0, ZERO, 3, 1  9D000604 A4626040 SH V0, 24640(V1)  12: TRISBbits.TRISB15 = OUT;  9D000608 3C03BF88 LUI V1, -16504  9D00060C 94626040 LHU V0, 24640(V1)  9D000610 7C027BC4 INS V0, ZERO, 15, 1  9D000614 A4626040 SH V0, 24640(V1)  13:  14: //configure rows as inputs  15: TRISBbits.TRISB5 = IN;  9D000618 3C03BF88 LUI V1, -16504  9D00061C 94626040 LHU V0, 24640(V1)  9D000620 24040001 ADDIU A0, ZERO, 1  9D000624 7C822944 INS V0, A0, 5, 1  9D000628 A4626040 SH V0, 24640(V1)  16: TRISBbits.TRISB13 = IN;  9D00062C 3C03BF88 LUI V1, -16504  9D000630 94626040 LHU V0, 24640(V1)  9D000634 24040001 ADDIU A0, ZERO, 1  9D000638 7C826B44 INS V0, A0, 13, 1  9D00063C A4626040 SH V0, 24640(V1)  17: TRISBbits.TRISB1 = IN;  9D000640 3C03BF88 LUI V1, -16504  9D000644 94626040 LHU V0, 24640(V1)  9D000648 24040001 ADDIU A0, ZERO, 1  9D00064C 7C820844 INS V0, A0, 1, 1  9D000650 A4626040 SH V0, 24640(V1)  18: TRISBbits.TRISB9 = IN;  9D000654 3C03BF88 LUI V1, -16504  9D000658 94626040 LHU V0, 24640(V1)  9D00065C 24040001 ADDIU A0, ZERO, 1  9D000660 7C824A44 INS V0, A0, 9, 1  9D000664 A4626040 SH V0, 24640(V1)  19:  20: // set outputs to default value of on  21: column1 = 1;  9D000668 3C03BF88 LUI V1, -16504  9D00066C 94626050 LHU V0, 24656(V1)  9D000670 24040001 ADDIU A0, ZERO, 1  9D000674 7C827BC4 INS V0, A0, 15, 1  9D000678 A4626050 SH V0, 24656(V1)  22: column2 = 1;  9D00067C 3C03BF88 LUI V1, -16504  9D000680 94626050 LHU V0, 24656(V1)  9D000684 24040001 ADDIU A0, ZERO, 1  9D000688 7C8218C4 INS V0, A0, 3, 1  9D00068C A4626050 SH V0, 24656(V1)  23: column3 = 1;  9D000690 3C03BF88 LUI V1, -16504  9D000694 94626050 LHU V0, 24656(V1)  9D000698 24040001 ADDIU A0, ZERO, 1  9D00069C 7C825AC4 INS V0, A0, 11, 1  9D0006A0 A4626050 SH V0, 24656(V1)  24: }  9D0006A4 03C0E821 ADDU SP, FP, ZERO  9D0006A8 8FBE0004 LW FP, 4(SP)  9D0006AC 27BD0008 ADDIU SP, SP, 8  9D0006B0 03E00008 JR RA  9D0006B4 00000000 NOP  25:  26: char readKey(void)  27: {  9D0006B8 27BDFFE0 ADDIU SP, SP, -32  9D0006BC AFBF001C SW RA, 28(SP)  9D0006C0 AFBE0018 SW FP, 24(SP)  9D0006C4 03A0F021 ADDU FP, SP, ZERO  28: char returnVal = NULL;  9D0006C8 A3C00010 SB ZERO, 16(FP)  29:  30: //check first column  31: column1 = 0;  9D0006CC 3C03BF88 LUI V1, -16504  9D0006D0 94626050 LHU V0, 24656(V1)  9D0006D4 7C027BC4 INS V0, ZERO, 15, 1  9D0006D8 A4626050 SH V0, 24656(V1)  32: if(row1 == 0)  9D0006DC 3C02BF88 LUI V0, -16504  9D0006E0 8C426050 LW V0, 24656(V0)  9D0006E4 30420020 ANDI V0, V0, 32  9D0006E8 1440000A BNE V0, ZERO, 0x9D000714  9D0006EC 00000000 NOP  33: {  34: msDelay(12);  9D0006F0 2404000C ADDIU A0, ZERO, 12  9D0006F4 0F40027A JAL msDelay  9D0006F8 00000000 NOP  35: Debounce();  9D0006FC 0F400299 JAL Debounce  9D000700 00000000 NOP  36: returnVal = '1';  9D000704 24020031 ADDIU V0, ZERO, 49  9D000708 A3C20010 SB V0, 16(FP)  9D00070C 0B4001ED J 0x9D0007B4  9D000710 00000000 NOP  37: }  38: else if(row2 == 0)  9D000714 3C02BF88 LUI V0, -16504  9D000718 8C426050 LW V0, 24656(V0)  9D00071C 30422000 ANDI V0, V0, 8192  9D000720 1440000A BNE V0, ZERO, 0x9D00074C  9D000724 00000000 NOP  39: {  40: msDelay(12);  9D000728 2404000C ADDIU A0, ZERO, 12  9D00072C 0F40027A JAL msDelay  9D000730 00000000 NOP  41: Debounce();  9D000734 0F400299 JAL Debounce  9D000738 00000000 NOP  42: returnVal = '4';  9D00073C 24020034 ADDIU V0, ZERO, 52  9D000740 A3C20010 SB V0, 16(FP)  9D000744 0B4001ED J 0x9D0007B4  9D000748 00000000 NOP  43: }  44: else if(row3 == 0)  9D00074C 3C02BF88 LUI V0, -16504  9D000750 8C426050 LW V0, 24656(V0)  9D000754 30420002 ANDI V0, V0, 2  9D000758 1440000A BNE V0, ZERO, 0x9D000784  9D00075C 00000000 NOP  45: {  46: msDelay(12);  9D000760 2404000C ADDIU A0, ZERO, 12  9D000764 0F40027A JAL msDelay  9D000768 00000000 NOP  47: Debounce();  9D00076C 0F400299 JAL Debounce  9D000770 00000000 NOP  48: returnVal = '7';  9D000774 24020037 ADDIU V0, ZERO, 55  9D000778 A3C20010 SB V0, 16(FP)  9D00077C 0B4001ED J 0x9D0007B4  9D000780 00000000 NOP  49: }  50: else if(row4 == 0)  9D000784 3C02BF88 LUI V0, -16504  9D000788 8C426050 LW V0, 24656(V0)  9D00078C 30420200 ANDI V0, V0, 512  9D000790 14400008 BNE V0, ZERO, 0x9D0007B4  9D000794 00000000 NOP  51: {  52: msDelay(12);  9D000798 2404000C ADDIU A0, ZERO, 12  9D00079C 0F40027A JAL msDelay  9D0007A0 00000000 NOP  53: Debounce();  9D0007A4 0F400299 JAL Debounce  9D0007A8 00000000 NOP  54: returnVal = '\*';  9D0007AC 2402002A ADDIU V0, ZERO, 42  9D0007B0 A3C20010 SB V0, 16(FP)  55: }  56: //reset column  57: column1 = 1;  9D0007B4 3C03BF88 LUI V1, -16504  9D0007B8 94626050 LHU V0, 24656(V1)  9D0007BC 24040001 ADDIU A0, ZERO, 1  9D0007C0 7C827BC4 INS V0, A0, 15, 1  9D0007C4 A4626050 SH V0, 24656(V1)  58:  59: //check second column  60: column2 = 0;  9D0007C8 3C03BF88 LUI V1, -16504  9D0007CC 94626050 LHU V0, 24656(V1)  9D0007D0 7C0218C4 INS V0, ZERO, 3, 1  9D0007D4 A4626050 SH V0, 24656(V1)  61: if(row1 == 0)  9D0007D8 3C02BF88 LUI V0, -16504  9D0007DC 8C426050 LW V0, 24656(V0)  9D0007E0 30420020 ANDI V0, V0, 32  9D0007E4 1440000A BNE V0, ZERO, 0x9D000810  9D0007E8 00000000 NOP  62: {  63: msDelay(12);  9D0007EC 2404000C ADDIU A0, ZERO, 12  9D0007F0 0F40027A JAL msDelay  9D0007F4 00000000 NOP  64: Debounce();  9D0007F8 0F400299 JAL Debounce  9D0007FC 00000000 NOP  65: returnVal = '2';  9D000800 24020032 ADDIU V0, ZERO, 50  9D000804 A3C20010 SB V0, 16(FP)  9D000808 0B40022C J 0x9D0008B0  9D00080C 00000000 NOP  66: }  67: else if(row2 == 0)  9D000810 3C02BF88 LUI V0, -16504  9D000814 8C426050 LW V0, 24656(V0)  9D000818 30422000 ANDI V0, V0, 8192  9D00081C 1440000A BNE V0, ZERO, 0x9D000848  9D000820 00000000 NOP  68: {  69: msDelay(12);  9D000824 2404000C ADDIU A0, ZERO, 12  9D000828 0F40027A JAL msDelay  9D00082C 00000000 NOP  70: Debounce();  9D000830 0F400299 JAL Debounce  9D000834 00000000 NOP  71: returnVal = '5';  9D000838 24020035 ADDIU V0, ZERO, 53  9D00083C A3C20010 SB V0, 16(FP)  9D000840 0B40022C J 0x9D0008B0  9D000844 00000000 NOP  72: }  73: else if(row3 == 0)  9D000848 3C02BF88 LUI V0, -16504  9D00084C 8C426050 LW V0, 24656(V0)  9D000850 30420002 ANDI V0, V0, 2  9D000854 1440000A BNE V0, ZERO, 0x9D000880  9D000858 00000000 NOP  74: {  75: msDelay(12);  9D00085C 2404000C ADDIU A0, ZERO, 12  9D000860 0F40027A JAL msDelay  9D000864 00000000 NOP  76: Debounce();  9D000868 0F400299 JAL Debounce  9D00086C 00000000 NOP  77: returnVal = '8';  9D000870 24020038 ADDIU V0, ZERO, 56  9D000874 A3C20010 SB V0, 16(FP)  9D000878 0B40022C J 0x9D0008B0  9D00087C 00000000 NOP  78: }  79: else if(row4 == 0)  9D000880 3C02BF88 LUI V0, -16504  9D000884 8C426050 LW V0, 24656(V0)  9D000888 30420200 ANDI V0, V0, 512  9D00088C 14400008 BNE V0, ZERO, 0x9D0008B0  9D000890 00000000 NOP  80: {  81: msDelay(12);  9D000894 2404000C ADDIU A0, ZERO, 12  9D000898 0F40027A JAL msDelay  9D00089C 00000000 NOP  82: Debounce();  9D0008A0 0F400299 JAL Debounce  9D0008A4 00000000 NOP  83: returnVal = '0';  9D0008A8 24020030 ADDIU V0, ZERO, 48  9D0008AC A3C20010 SB V0, 16(FP)  84: }  85: //reset column  86: column2 = 1;  9D0008B0 3C03BF88 LUI V1, -16504  9D0008B4 94626050 LHU V0, 24656(V1)  9D0008B8 24040001 ADDIU A0, ZERO, 1  9D0008BC 7C8218C4 INS V0, A0, 3, 1  9D0008C0 A4626050 SH V0, 24656(V1)  87:  88: ////check third column  89: column3 = 0;  9D0008C4 3C03BF88 LUI V1, -16504  9D0008C8 94626050 LHU V0, 24656(V1)  9D0008CC 7C025AC4 INS V0, ZERO, 11, 1  9D0008D0 A4626050 SH V0, 24656(V1)  90: if(row1 == 0)  9D0008D4 3C02BF88 LUI V0, -16504  9D0008D8 8C426050 LW V0, 24656(V0)  9D0008DC 30420020 ANDI V0, V0, 32  9D0008E0 1440000A BNE V0, ZERO, 0x9D00090C  9D0008E4 00000000 NOP  91: {  92: msDelay(12);  9D0008E8 2404000C ADDIU A0, ZERO, 12  9D0008EC 0F40027A JAL msDelay  9D0008F0 00000000 NOP  93: Debounce();  9D0008F4 0F400299 JAL Debounce  9D0008F8 00000000 NOP  94: returnVal = '3';  9D0008FC 24020033 ADDIU V0, ZERO, 51  9D000900 A3C20010 SB V0, 16(FP)  9D000904 0B40026B J 0x9D0009AC  9D000908 00000000 NOP  95: }  96: else if(row2 == 0)  9D00090C 3C02BF88 LUI V0, -16504  9D000910 8C426050 LW V0, 24656(V0)  9D000914 30422000 ANDI V0, V0, 8192  9D000918 1440000A BNE V0, ZERO, 0x9D000944  9D00091C 00000000 NOP  97: {  98: msDelay(12);  9D000920 2404000C ADDIU A0, ZERO, 12  9D000924 0F40027A JAL msDelay  9D000928 00000000 NOP  99: Debounce();  9D00092C 0F400299 JAL Debounce  9D000930 00000000 NOP  100: returnVal = '6';  9D000934 24020036 ADDIU V0, ZERO, 54  9D000938 A3C20010 SB V0, 16(FP)  9D00093C 0B40026B J 0x9D0009AC  9D000940 00000000 NOP  101: }  102: else if(row3 == 0)  9D000944 3C02BF88 LUI V0, -16504  9D000948 8C426050 LW V0, 24656(V0)  9D00094C 30420002 ANDI V0, V0, 2  9D000950 1440000A BNE V0, ZERO, 0x9D00097C  9D000954 00000000 NOP  103: {  104: msDelay(12);  9D000958 2404000C ADDIU A0, ZERO, 12  9D00095C 0F40027A JAL msDelay  9D000960 00000000 NOP  105: Debounce();  9D000964 0F400299 JAL Debounce  9D000968 00000000 NOP  106: returnVal = '9';  9D00096C 24020039 ADDIU V0, ZERO, 57  9D000970 A3C20010 SB V0, 16(FP)  9D000974 0B40026B J 0x9D0009AC  9D000978 00000000 NOP  107: }  108: else if(row4 == 0)  9D00097C 3C02BF88 LUI V0, -16504  9D000980 8C426050 LW V0, 24656(V0)  9D000984 30420200 ANDI V0, V0, 512  9D000988 14400008 BNE V0, ZERO, 0x9D0009AC  9D00098C 00000000 NOP  109: {  110: msDelay(12);  9D000990 2404000C ADDIU A0, ZERO, 12  9D000994 0F40027A JAL msDelay  9D000998 00000000 NOP  111: Debounce();  9D00099C 0F400299 JAL Debounce  9D0009A0 00000000 NOP  112: returnVal = '#';  9D0009A4 24020023 ADDIU V0, ZERO, 35  9D0009A8 A3C20010 SB V0, 16(FP)  113: }  114: //reset column  115: column3 = 1;  9D0009AC 3C03BF88 LUI V1, -16504  9D0009B0 94626050 LHU V0, 24656(V1)  9D0009B4 24040001 ADDIU A0, ZERO, 1  9D0009B8 7C825AC4 INS V0, A0, 11, 1  9D0009BC A4626050 SH V0, 24656(V1)  116: msDelay(12);  9D0009C0 2404000C ADDIU A0, ZERO, 12  9D0009C4 0F40027A JAL msDelay  9D0009C8 00000000 NOP  117: //returnVal = '\0';  118:  119: return returnVal;  9D0009CC 83C20010 LB V0, 16(FP)  120: }  9D0009D0 03C0E821 ADDU SP, FP, ZERO  9D0009D4 8FBF001C LW RA, 28(SP)  9D0009D8 8FBE0018 LW FP, 24(SP)  9D0009DC 27BD0020 ADDIU SP, SP, 32  9D0009E0 03E00008 JR RA  9D0009E4 00000000 NOP  121: /\*  122: //setup the time delay  123: void msDelay(unsigned int ms)  124: {  125: //figures out how many clock cycles it will go through in ms miliseconds  126: unsigned int tick = ms \* (ONE\_SEC\_TICK / 1000);  127:  128: //set the core timer count to 0  129: \_CP0\_SET\_COUNT(0);  130:  131: //waits until the core timer reaches the target value  132: while(tick > \_CP0\_SET\_COUNT());  133: }\*/  134:  135: //generate time delay for the specified amount of milliseconds  136: void msDelay(unsigned int ms)  137: {  9D0009E8 27BDFFF0 ADDIU SP, SP, -16  9D0009EC AFBF000C SW RA, 12(SP)  9D0009F0 AFBE0008 SW FP, 8(SP)  9D0009F4 03A0F021 ADDU FP, SP, ZERO  9D0009F8 AFC40010 SW A0, 16(FP)  138: // Convert ms microseconds into how many clock ticks it will take  139: unsigned int ticks = ms \* ( ONE\_SEC\_TICK /1000);  9D0009FC 8FC40010 LW A0, 16(FP)  9D000A00 00801821 ADDU V1, A0, ZERO  9D000A04 00031080 SLL V0, V1, 2  9D000A08 00401821 ADDU V1, V0, ZERO  9D000A0C 00031140 SLL V0, V1, 5  9D000A10 00431023 SUBU V0, V0, V1  9D000A14 00441021 ADDU V0, V0, A0  9D000A18 00021880 SLL V1, V0, 2  9D000A1C 00431021 ADDU V0, V0, V1  9D000A20 00021180 SLL V0, V0, 6  9D000A24 AFC20000 SW V0, 0(FP)  140: // ms \*= TIMER\_FREQ / 1000000 ; // Core Timer updates every 2 ticks  141:  142: \_CP0\_SET\_COUNT(0); // Set Core Timer count to 0  9D000A28 00001021 ADDU V0, ZERO, ZERO  9D000A2C 40824800 MTC0 V0, Count  9D000A30 000000C0 EHB  143:  144: while (ticks > \_CP0\_GET\_COUNT()); // Wait until Core Timer count reaches the number we calculated earlier  9D000A34 00000000 NOP  9D000A38 40034800 MFC0 V1, Count  9D000A3C 8FC20000 LW V0, 0(FP)  9D000A40 0062102B SLTU V0, V1, V0  9D000A44 1440FFFC BNE V0, ZERO, 0x9D000A38  9D000A48 00000000 NOP  145: }  9D000A4C 03C0E821 ADDU SP, FP, ZERO  9D000A50 8FBF000C LW RA, 12(SP)  9D000A54 8FBE0008 LW FP, 8(SP)  9D000A58 27BD0010 ADDIU SP, SP, 16  9D000A5C 03E00008 JR RA  9D000A60 00000000 NOP  146: void Debounce (void)  147: {  9D000A64 27BDFFF8 ADDIU SP, SP, -8  9D000A68 AFBE0004 SW FP, 4(SP)  9D000A6C 03A0F021 ADDU FP, SP, ZERO  148: column1 = 0;  9D000A70 3C03BF88 LUI V1, -16504  9D000A74 94626050 LHU V0, 24656(V1)  9D000A78 7C027BC4 INS V0, ZERO, 15, 1  9D000A7C A4626050 SH V0, 24656(V1)  149: column2 = 0;  9D000A80 3C03BF88 LUI V1, -16504  9D000A84 94626050 LHU V0, 24656(V1)  9D000A88 7C0218C4 INS V0, ZERO, 3, 1  9D000A8C A4626050 SH V0, 24656(V1)  150: column3 = 0;  9D000A90 3C03BF88 LUI V1, -16504  9D000A94 94626050 LHU V0, 24656(V1)  9D000A98 7C025AC4 INS V0, ZERO, 11, 1  9D000A9C A4626050 SH V0, 24656(V1)  151:  152: while ((row1 == 0) || (row2 == 0) || (row3 == 0) || (row4 == 0))  9D000AA0 00000000 NOP  9D000AA4 3C02BF88 LUI V0, -16504  9D000AA8 8C426050 LW V0, 24656(V0)  9D000AAC 30420020 ANDI V0, V0, 32  9D000AB0 1040FFFC BEQ V0, ZERO, 0x9D000AA4  9D000AB4 00000000 NOP  9D000AB8 3C02BF88 LUI V0, -16504  9D000ABC 8C426050 LW V0, 24656(V0)  9D000AC0 30422000 ANDI V0, V0, 8192  9D000AC4 1040FFF7 BEQ V0, ZERO, 0x9D000AA4  9D000AC8 00000000 NOP  9D000ACC 3C02BF88 LUI V0, -16504  9D000AD0 8C426050 LW V0, 24656(V0)  9D000AD4 30420002 ANDI V0, V0, 2  9D000AD8 1040FFF2 BEQ V0, ZERO, 0x9D000AA4  9D000ADC 00000000 NOP  9D000AE0 3C02BF88 LUI V0, -16504  9D000AE4 8C426050 LW V0, 24656(V0)  9D000AE8 30420200 ANDI V0, V0, 512  9D000AEC 1040FFED BEQ V0, ZERO, 0x9D000AA4  9D000AF0 00000000 NOP  153: {  154: //do nothing  155: }  156: //reset all columns  157: column1 = 1;  9D000AF4 3C03BF88 LUI V1, -16504  9D000AF8 94626050 LHU V0, 24656(V1)  9D000AFC 24040001 ADDIU A0, ZERO, 1  9D000B00 7C827BC4 INS V0, A0, 15, 1  9D000B04 A4626050 SH V0, 24656(V1)  158: column2 = 1;  9D000B08 3C03BF88 LUI V1, -16504  9D000B0C 94626050 LHU V0, 24656(V1)  9D000B10 24040001 ADDIU A0, ZERO, 1  9D000B14 7C8218C4 INS V0, A0, 3, 1  9D000B18 A4626050 SH V0, 24656(V1)  159: column3 = 1;  9D000B1C 3C03BF88 LUI V1, -16504  9D000B20 94626050 LHU V0, 24656(V1)  9D000B24 24040001 ADDIU A0, ZERO, 1  9D000B28 7C825AC4 INS V0, A0, 11, 1  9D000B2C A4626050 SH V0, 24656(V1)  160: }  9D000B30 03C0E821 ADDU SP, FP, ZERO  9D000B34 8FBE0004 LW FP, 4(SP)  9D000B38 27BD0008 ADDIU SP, SP, 8  9D000B3C 03E00008 JR RA  9D000B40 00000000 NOP |