**EE084IU**

**Micro-processing Systems**

**LAB 4:**

**ADC AND TIMERS, INTERUPTS**

**Student: Đỗ Minh Duy**

**Student ID: ITITSB22029**

**Class:**

**Date:**

I. LAB OBJECTIVES

After completion of this Lab, Students will:

* Know how to use timer and counter.
* Know how to use External Interrupts
* Interface ATmega32 with sensor using ADC
* Develop multi-tasks program for Microcontroller
* Design hardware and software for Microcontroller based control system application

I. PROCEDURE

Student Carry out all the Lab tasks and get Lab Instructor Signature check mark for each task.

I.1 TASK1: ADC interface using polling method

I.1.1 Problem Statement: Design an embedded system using ATMEGA32 working with 8Mhz.The system interface with an analog input at channel ADC3, AREF=5V, left justified data, CLK/128 (using a variable resistor to make a voltage divider) the data will be displayed in LCD connnected with PORTC (LCD 4-bit mode). Write C program use Polling method to get ADC data display to LCD.

I.1.2 Circuit Design:

Student Design the application circuit in Proteus for task 1

I.1.3 Theory:

Student review related theory knowledge about ADD and describe in the report, with following details:

- ADC Registers and initialization explanation for the task1 requirements

I.1.4 Code

|  |  |
| --- | --- |
| Write general comments about the program here  /\*----------------------------------------------------------------  This program is a multi-tasks program to get ADC Data using polling method  ---------------------------------------------------------------------\*/  // Reading ADC using polling  #include <avr/io.h> //standard AVR header  int main (void){  unsigned int data;  DDRB = 0xFF; // make Port B an output  DDRC = 0xFF; // make Port C an output  DDRD = 0xFF; // make Port D an output  DDRA = 0; // make Port A an input for ADC input  ADCSRA = 0x87; // make ADC enable and select CLK/128  // 5V Vref internal, right justified,  ADMUX = 0x43; // select ADC Channel 0  while (1){  ADCSRA |= (1<<ADSC); // start conversion  while ( (ADCSRA & (1<<ADIF) )== 0 );  // wait for conversion to finish  data = ADC;  PORTD = data; // give the low byte to PORTD  PORTC = data >> 8; // give the high byte to PORTB  PORTB = data >> 2; // if u need only 8-bit value  }  return 0; } | |
| Instruction | Comments |
|  |  |

I.1.5 Result

Check the designed circuit.

Check ADC data displayed in LCD.

|  |  |  |
| --- | --- | --- |
| For Office use only | Signature of Lab Instructor | Remarks |
| Circuit Design in Proteus |  |  |
| Code Running in Proteus correctly |  |  |

I.2 TASK2: ADC interface using inter method (Advanced Task):

I.2.1 Problem Statement: Design an embedded system using ATMEGA32 working with 8Mhz.The system interface with an analog input at channel ADC5, AREF=5V, right justified data, CLK/128 (using a variable resistor to make a voltage divider) the data will display Low byte in PORTD and High byte in PORTB. Write C program and Assemble program using ADC interrupt method to get ADC data and output to PORTD and PORTB .

I.2.2 Circuit Design:

Student Design the application circuit in Proteus for task 2

I.2.3 Theory:

Student review related theory knowledge about ADD and describe in the report, with following details:

- ADC Registers and initialization explanation for the task1 requirements

I.2.4 Code

|  |  |
| --- | --- |
| Write general comments about the program here  /\*----------------------------------------------------------------  This program is a multi-tasks program to get ADC Data using polling method  ---------------------------------------------------------------------\*/  C code  #include <avr/io.h> //standard AVR header  #include <avr/interrupt.h>  ISR(ADC\_vect){  PORTD = ADCL; // give the low byte to PORTD  PORTC = ADCH; // give the high byte to PORTB  ADCSRA |= (1<<ADSC) ; // start conversion  }  int main (void){  DDRC = 0xFF; // make Port B an output  DDRD = 0xFF; // make Port D an output  DDRA = 0; // make Port A an input for ADC input  sei(); // enable global interrupts  // enable ADC and interrupt, and select CLK/128  ADCSRA= 0x8F;  // 2.56V Vref internal, right justified,  // select ADC0 chanel 0  ADMUX = 0x45;  ADCSRA |= (1<<ADSC) ; // start conversion  while (1); return 0;} // wait forever  Assembly code:  .include "m32def.inc"  .org 0x00000  jmp main  .org 0x0020  jmp my\_ADC\_interrupt  .org 0x2A  main:  LDI R16, 0xFF; // make Port B an output  OUT DDRC, R16; // make Port D an output    LDI R16, 0XFF  OUT DDRD, R16  LDI R17, 0x00  OUT DDRA, R17  // make Port A an input for ADC input  SEI ;sei(); // enable global interrupts  // enable ADC and interrupt, and select CLK/128  LDI R16, 0x8F  OUT ADCSR, R16 ;ADCSRA= 0x8F;  // 2.56V Vref internal, right justified,  // select ADC0 chanel 0  LDI R16, 0x45 ;ADMUX = 0x45;  OUT ADMUX, R16  IN R16, ADCSRA  LDI R17, (1<<ADSC)  OR R16, R17  OUT ADCSRA, R16  ;ADCSRA |= (1<<ADSC) ; // start conversion  while\_1:  jmp while\_1 ;while (1); return 0;} // wait forever  my\_ADC\_interrupt:  IN R16, ADCL;PORTD = ADCL; // give the low byte to PORTD  OUT PORTD, R16  IN R16, ADCH  OUT PORTC, R16    IN R16, ADCSRA  LDI R17, (1<<ADSC)  OR R16, R17  OUT ADCSRA, R16  RETI  ;PORTC = ADCH; // give the high byte to PORTB  ;ADCSRA |= (1<<ADSC) ; // start conversion | |
| Instruction | Comments |
|  |  |

I.2.5 Result

Check the designed circuit.

Check ADC data displayed in PORTD and PORTB.

|  |  |  |
| --- | --- | --- |
| For Office use only | Signature of Lab Instructor | Remarks |
| Circuit Design in Proteus |  |  |
| Code Running in Proteus correctly |  |  |

I.3 TASK3: ADC interface using Interrupt method

I.3.1 Problem Statement: Design an embedded system using ATMEGA32 working with 8Mhz.The system interface with an analog input at channel ADC1, ADC4, ADC6, AREF=2.56V, right justified data, CLK/128 (using a variable resistor to make a voltage divider) the data will be displayed in LCD. Write C program using ADC Interrupt method program to get ADC data. (Data of each ADC channel will be displayed once at a time with the interval of 1 second using TIMER1 interrupt), ADC result data of each channel will displayed in LCD once at a time.

I.3.2 Circuit Design:

Student Design the application circuit in Proteus for task 3

I.3.3 Theory:

Student review related theory knowledge about ADD and describe in the report, with following details:

- ADC Registers and initialization explanation for the task3 requirements

I.3.4 Code

|  |  |
| --- | --- |
| Write general comments about the program here  /\*----------------------------------------------------------------  This program is a multi-tasks program to get ADC Data using Interrupt method  ---------------------------------------------------------------------\*/  C code:    // Reading ADC using polling  #ifndef F\_CPU  #define F\_CPU 8000000UL // 16 MHz clock speed  #endif  #define D4 eS\_PORTC4  #define D5 eS\_PORTC5  #define D6 eS\_PORTC6  #define D7 eS\_PORTC7  #define RS eS\_PORTC0  #define EN eS\_PORTC1  #include <avr/io.h>  #include <util/delay.h>  #include "avr/interrupt.h"  #include "lcd.h"  #include <string.h>  #include <stdlib.h>  unsigned int ADC\_result;  int ADC\_channel=1;  char ACD\_lcd\_string[10];  double Volt\_result=0.0;  char buf\_lcd1[16];  ISR(ADC\_vect){  ADC\_result=ADC;  ADCSRA |= (1<<ADSC) ; // start conversion  }  void Timer1\_init()  { // Timer1\_init to generate 1 interrupt every 1 second interval  TCNT1 = 34286; // timer starts from 00  TCCR1A = 0x00;  TCCR1B = 0x04; //Run Timer1, Normal, Prescaler mode 1:256  TIMSK = (1<<TOIE1); // enable Timer1 overflow Interrupt Enable, overflow interrupt  sei (); // enable interrupts  }  ISR (TIMER1\_OVF\_vect){ // ISR for external interrupt 2  TCNT1 = 34286; // timer starts from 00  switch(ADC\_channel)  {  case 1: ADC\_channel=4;  ADMUX = 0xC4;  break;  case 4: ADC\_channel=6;  ADMUX = 0xC6;  break;  case 6: ADC\_channel=1;  ADMUX = 0xC1;  break;  }  }  int main (void){  DDRC=0xFF;  Lcd4\_Init();  Timer1\_init();    DDRA = 0; // make Port A an input for ADC input  sei(); // enable global interrupts  // enable ADC and interrupt, and select CLK/128  ADCSRA= 0x8F;  // 2.56V Vref internal, right justified,  // select ADC0 chanel 1  ADMUX = 0xC1;  ADCSRA |= (1<<ADSC) ; // start convers  while (1)  {  // wait for conversion to finish  sprintf(ACD\_lcd\_string,"ADC%d Result:%04d",ADC\_channel,ADC\_result);  Lcd4\_Set\_Cursor(1,1);  Lcd4\_Write\_String(ACD\_lcd\_string);  Volt\_result=(double)ADC\_result\*2.56/(double)1024;  dtostrf(Volt\_result, 6, 4,buf\_lcd1 );  Lcd4\_Set\_Cursor(2,1);  Lcd4\_Write\_String("Volt:");  Lcd4\_Write\_String(buf\_lcd1);  }  ;  return 0;  } // wait forever | |
| Instruction | Comments |
|  |  |

I.3.5 Result

Check the designed circuit.

Check ADC data displayed in LCD.

|  |  |  |
| --- | --- | --- |
| For Office use only | Signature of Lab Instructor | Remarks |
| Circuit Design in Proteus |  |  |
| Code Running in Proteus correctly |  |  |

I.4 TASK4: Using ADC interface to get data from LM35 Temperature sensor using C Language

I.4.1 Problem Statement: Design an embedded system using ATMEGA32 working with 8Mhz.The system interface with two LM35 analog sensor at ADC Channel2 and ADC channel ADC4 , AREF=2.56V, left justified data, CLK/1024 the data will display in LCD using PORTC (4 bit mode) . Write C pgrogram using Interrupt method to get ADC data every 1 second using Timer1 and displayed to LCD. The data will be displayed in LCD in form of voltage and tempurature.

I.3.2 Circuit Design:

Student Design the application circuit in Proteus for task 4

I.4.3 Theory:

Student review related theory knowledge about ADC and describe in the report, with following details:

- ADC Registers and initialization explanation for the task4 requirements

- LCD inerface

I.4.4 Code

|  |  |
| --- | --- |
| Write general comments about the program here  /\*----------------------------------------------------------------  This program is a multi-tasks program to get LM35 temperature sensor data using Interrupt method  ---------------------------------------------------------------------\*/ | |
| Instruction | Comments |
|  |  |

I.4.5 Result

Check the designed circuit.

Check ADC data displayed in LCD.

|  |  |  |
| --- | --- | --- |
| For Office use only | Signature of Lab Instructor | Remarks |
| Circuit Design in Proteus |  |  |
| Code Running in Proteus correctly |  |  |

I.5 TASK5: Using ADC interface to get data from LM35 Temperature sensor using Assembly Language (Advanced Task)

I.5.1 Problem Statement: Design an embedded system using ATMEGA32 working with 8Mhz.The system interface with two LM35 analog sensor at ADC Channel2 and ADC channel ADC4 , AREF=2.56V, left justified data, CLK/1024 the data will display in LCD using PORTC ( 8 bit mode) . Write Assembler pgrogram using Interrupt method to get ADC data every 1 second using Timer1 and displayed to LCD. The data will be displayed in LCD in form of voltage and tempurature.

I.5.2 Circuit Design:

Student Design the application circuit in Proteus for task 5

I.5.3 Theory:

Student review related theory knowledge about ADC and describe in the report, with following details:

- ADC Registers and initialization explanation for the task5 requirements

- LCD inerface

I.5.4 Code

|  |  |
| --- | --- |
| Write general comments about the program here  /\*----------------------------------------------------------------  This program is a multi-tasks program to get LM35 temperature sensor data using Interrupt method  ---------------------------------------------------------------------\*/ | |
| Instruction | Comments |
|  |  |

I.5.5 Result

Check the designed circuit.

Check ADC data displayed in LCD.

|  |  |  |
| --- | --- | --- |
| For Office use only | Signature of Lab Instructor | Remarks |
| Circuit Design in Proteus |  |  |
| Code Running in Proteus correctly |  |  |

I.6 TASK6 : TIMERS, ADC and EXTERNAL INTERUPT USING C LANGUAGE

I.6.1 Problem Statement: Design an embedded system using ATMEGA32 working with 8Mhz and multi-tasks program to get data from 8 ADC channels from 8 LM35 temperature sensors and display to LCD and control relays. Using TIMER1\_OVF\_vect interrupt to set up time interval for getting data (1 sample data / 1 second). The data getting of ADC conversion is altered from ADC channel0 to ADC channel7 and repeat to channel0 again. The ADC data from 8 channels will be used to control 8 relays connected to 8 pins of PORTC. When data of any channel is smaller than a predefined threshold value the appropriate relay will be turned on, otherwise it is turned off. The predefined threshold value is set up by INT0 and INT1.

|  |  |  |
| --- | --- | --- |
| INTERUPT  Vectors | Function | Frequency |
| 1. TIMER1\_OVF\_vect 2. (first event) | - Get data from ADC channel0  - Set ADCMUX to prepare for getting ADC data for next channel ( channel1) | 1 Hhz |
| 1. TIMER1\_OVF\_vect 2. (second event) | - Get data from ADC channel1  - Set ADCMUX to prepare for getting ADC data for next channel ( channel2) | 1 Hhz |
| … |  |  |
| 1. TIMER1\_OVF\_vect 2. (6th event) | - Get data from ADC channel6  - Set ADCMUX to prepare for getting ADC data for next channel ( channel7) | 1 Hhz |
| 1. TIMER1\_OVF\_vect 2. (7th event) | - Get data from ADC channel7  - Set ADCMUX to prepare for getting ADC data for next channel ( channel0) | 1 Hhz |

|  |  |
| --- | --- |
| Interrupt Vector | Function |
| INT0\_vect | Activated by falling edge to increase ADC threshold value. |
| INT1\_vect | Activated by falling edge to decrease crease ADC threshold value. |

I.6.2 Circuit Design:

Student Design the application circuit in Proteus

I.6.3 Theory:

Student review related theory knowledge about ADC and ADC interupt

I.6.4 Algorithm

1. Develop function to setup TIMER0\_OVF\_vect with frequency 1Hz

- getting data from current ADC channel

- Set up ADCMUX to prepare for getting next ADC channel

+ if current channel is ADC channel7

Set up Next ADC channel is channel0.

2. Develop ISR(INT0\_vect) function to increase ADC threshold value

3. Develop ISR(INT1\_vect) function to decrease ADC threshold value.

4. set up INT0 working with falling edge activation mode.

5. set up INT1 working with falling edge activation mode.

14. Write main program

+ Setting up PORTC as output

+ Initialize LCD (16x2) 4 bit using PORTD

+ Call functions to setup timer

+ Enable all defined interrupts

+ While (1) loop:

- display current value ADC data in first LCD line

- display current ADC threshold value

I.6.5 Code

|  |  |
| --- | --- |
| Write general comments about the program here  /\*----------------------------------------------------------------  This program is a multi-tasks program to get data from 8 ADC channels from 8 LM35 temperature sensors and display to LCD and control relay.  Author: xxxxxxxxx Date: xx.xx.20xx  ---------------------------------------------------------------------\*/ | |
| Instruction | Comments |
|  |  |

I.7 TASK7 (Advanced Task): TIMERS, ADC and EXTERNAL INTERUPT USING AVR ASSEMBLY PROGRAMMING LANGUAGE

I.7.1 Problem Statement: Design an embedded system using ATMEGA32 working with 8Mhz and multi-tasks program to get data from 8 ADC channels from 8 LM35 temperature sensors and display to LCD and control relays. Using TIMER1\_OVF\_vect interrupt to set up time interval for getting data (1 sample data / 1 second). The data getting of ADC conversion is altered from ADC channel0 to ADC channel7 and repeat to channel0 again. The ADC data from 8 channels will be used to control 8 relays connected to 8 pins of PORTC. When data of any channel is smaller than a predefined threshold value the appropriate relay will be turned on, otherwise it is turned off. The predefined threshold value is set up by INT0 and INT1.

I.7.2 Circuit Design:

Student Design the application circuit in Proteus

I.7.3 Theory:

Student review related theory knowledge about ADC and ADC interupt

I.7.4 Algorithm

1. Develop function to setup TIMER0\_OVF\_vect with frequency 1Hz

- getting data from current ADC channel

- Set up ADCMUX to prepare for getting next ADC channel

+ if current channel is ADC channel7

Set up Next ADC channel is channel0.

2. Develop ISR(INT0\_vect) function to increase ADC threshold value

3. Develop ISR(INT1\_vect) function to decrease ADC threshold value.

4. set up INT0 working with falling edge activation mode.

5. set up INT1 working with falling edge activation mode.

14. Write main program

+ Setting up PORTC as output

+ Initialize LCD (16x2) 4 bit using PORTD

+ Call functions to setup timer

+ Enable all defined interrupts

+ While (1) loop:

- display current value ADC data in first LCD line

- display current ADC threshold value

I.7.5 Code

|  |  |
| --- | --- |
| Write general comments about the program here  /\*----------------------------------------------------------------  This program is a multi-tasks program to get data from 8 ADC channels from 8 LM35 temperature sensors and display to LCD and control relay.  Author: xxxxxxxxx Date: xx.xx.20xx  ---------------------------------------------------------------------\*/ | |
| Instruction | Comments |
|  |  |

I.7.6 Result

Check the designed circuit.

Check ADC current data display and ADC threshold value display in LCD.

Press INT0 and INT1 to change ADC threshold value and change LM35 sensor data and check the status of relays.

|  |  |  |
| --- | --- | --- |
| For Office use only | Signature of Lab Instructor | Remarks |
| Circuit Design in Proteus |  |  |
| Code Running in Proteus correctly |  |  |

II. REPPORT

After finish all tasks student analyze the laboratory results and submit your report in blackboard. The report for each task should include

* Theory ( Related theory)
* Circuit diagram in proteus
* Algorithm
* Code with clear comments

The format of the report should follow the Lab report guideline of School of Electrical Engineering

**GRADING GUIDELINE FOR LAB REPORT**

Name of Student: ID:

Subject: Lab Number:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Number** | **Content** |  | **Score** | **Comment** |
| 1 | **Format (max 9%)** | |  |  |
| * Font type | Yes No |  |
| * Font size | Yes No |  |
| * Lab title | Yes No |  |
| * Page number | Yes No |  |
| * Table of contents | Yes No |  |
| * Header/Footer | Yes No |  |
| * List of figures (if exists) | Yes No |  |
| * List of tables (if exists) | Yes No |  |
| * Lab report structure | Yes No |  |
| 2 | **English Grammar and Spelling (max 6%)** | |  |  |
| * Grammar | Yes No |  |
| * Spelling | Yes No |  |
| 3 | **Data and Result Analysis (max 85%)**   * Running result check * Report contents * Code with clear comments | |  |  |
| **Total Score** | |  | |  |

Signature:

Date: