**EE084IU**

**Micro-processing Systems**

**LAB 5:**

**UART, ADC AND TIMER APPLICATIONS**

**Student: DO MINH DUY**

**Student ID: ITITSB22029**

**Class:**

**Date: 18/12/2024**

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
* Know how to use UART serial interface
* Develop multi-tasks program for embedded system
* Design the embedded system hardware and software for specific application

I. PROCEDURE

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

I.1 TASK1 : Interface Microcontroller with PC using UART Pheripheral using polling method - C Programming

I.1.1 Problem Statement: Design an embedded systems using ATMEGA32 (working at 7.372800Mhz) communicate with PC by UART0 serial interface. The system has 8 LEDs connected to PORTC. When a system receiving new character from the UART port the system will display this character to Seven-segment LED in form Hexa Number, and toggle the relevant LED ( for example the character received is ‘0’ LED0 will be toggled, the character received is ‘1’ LED1 will be toggled and so on … the character received is ‘7’ LED7 will be toggled ). The system using polling method and UART setting with baud rate 19200, 8 bit, 1 stop bit, no parity. Write C programm to control the system.

I.1.2 Circuit Design:

Student Design the application circuit in Proteus

I.1.3 Theory:

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

- UART Control Registers

- UART initialization explanation for the task1 requirements ( baud rate 19200 8 bit, 1 stop bit, no parity )

I.1.4 Algorithm

1. Develop a function to initialize UART with baud rate 19200, 8 bit, 1 stop bit, no parity.

2. Develop a function to send a character

3. Develop an polling function to receive a character, when receiving a new character this function will toggle the relevant led.

4. Write main program

+ Setting up PORTC as output connect to 8 LEDs

+ Setting up PORTB as input connect to 8 Buttons

+ Enable UART receiver (Polling method)

+ While(1) loop :

- Call polling function to receive a character

I.1.5 Code

|  |  |
| --- | --- |
| Write general comments about the program here  /\*----------------------------------------------------------------  This program is a multi-tasks program to send and receive data to and from UART.  Author: xxxxxxxxx Date: xx.xx.20xx  ---------------------------------------------------------------------\*/  #include <avr/io.h>  char received\_data;  void Tx( unsigned char data ){  while ( !(UCSRA & (1<<UDRE)) ); // wait until UDR is empty  UDR = data; // Putting data into UDR, sends the data  }  unsigned char Rx(){  while ( !(UCSRA & (1<<RXC)) ); // wait for the Receive Complete (RXC) Flag  return UDR; // Get and return received data from buffer  }  void init\_UART(){  UCSRB = (1<<RXEN) |(1<<TXEN); // Enable USART Receiver and Transmitter  UCSRC = (1<<UCSZ1)|(1<<UCSZ0); // Mode3: Use 8-bit data  UBRRH = 23>>8; UBRRL = 23; // For 8 MHz Crystal and 1200 baud rate  }  int main()  {  DDRC= 0xFF;  PORTC=0x00;  DDRB = 0XFF;  x;  init\_UART();  while(1)  {    received\_data=Rx();  PORTB = received\_data;  switch(received\_data)  {  case '1': PORTC^=(1<<0);  break;  case '2': PORTC^=(1<<1);  break;  case '3': PORTC^=(1<<2);  break;  case '4': PORTC^=(1<<3);  break;  case '5': PORTC^=(1<<4);  break;  case '6': PORTC^=(1<<5);  break;  case '7': PORTC^=(1<<6);  break;  case '8': PORTC^=(1<<7);  break;  default:  PORTC=0x00;  break;  }  Tx(PORTB);  }  } | |
| Instruction | Comments |
|  |  |

I.1.6 Result

Check the designed circuit.

Check the the sending character in PC and check the led toggled led in PORTC.

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

I.2 TASK2 : Interface Microcontroller with PC using UART Pheripheral using polling method – AVR Assembly Programming

I.2.1 Problem Statement: Design an embedded systems using ATMEGA32 (working at 7.372800Mhz) communicate with PC by UART0 serial interface. The system has 8 LEDs connected to PORTC. When a system receiving new character from the UART port the system will display this character to Seven-segment LED in form Hexa Number, and toggle the relevant LED ( for example the character received is ‘0’ LED0 will be toggled, the character received is ‘1’ LED1 will be toggled and so on … the character received is ‘7’ LED7 will be toggled ). The system using polling method and UART setting with baud rate 1200, 8 bit, 1 stop bit, no parity. Write AVR Assembly programm to control the system.

I.2.3 Code

|  |  |
| --- | --- |
| Write general comments about the program here  /\*----------------------------------------------------------------  This program is an AVR Assembly program to send and receive data to and from UART.  Author: xxxxxxxxx Date: xx.xx.20xx  ---------------------------------------------------------------------\*/  .include "m32def.inc" ; //#include <avr/io.h>  .def received\_data=R20 ; // char received\_data;  main: ;//int main()  LDI R16,0xFF ;//DDRC= 0xFF;  OUT DDRC,R16  LDI R16,0x00 ;//PORTC=0x00;;  OUT PORTC,R16  LDI R16,0xFF ; //DDRB = 0XFF;  OUT DDRB,R16  LDI R16,0xFE ; //DDRD = 0XFE; tu dich  OUT DDRD,R16  call init\_UART ;//init\_UART();  While\_1: ; //while(1)  call Rx ; // received\_data=Rx();  OUT PORTB, received\_data ; //PORTB=received\_data;  call Tx ; Tx(received\_data);  ;//switch(received\_data)    case\_0: CPI received\_data,'0' ;//case '1':  BRNE case\_1  IN R16,PORTC ;// PORTC^=(1<<0); PORTC=PORTC^(1<<0);  LDI R17, (1<<0);  EOR R16,R17 ; // R16=R16^R17  OUT PORTC,R16  case\_1: CPI received\_data,'1' ;//case '1':  BRNE case\_2  IN R16,PORTC ;// PORTC^=(1<<1); PORTC=PORTC^(1<<1);  LDI R17, (1<<1);  EOR R16,R17 ; // R16=R16^R17  OUT PORTC,R16  case\_2: CPI received\_data,'2' ;//case '2':  BRNE case\_3  IN R16,PORTC ;// PORTC^=(1<<2); PORTC=PORTC^(1<<2);  LDI R17, (1<<2);  EOR R16,R17 ; // R16=R16^R17  OUT PORTC,R16  case\_3: CPI received\_data,'3' ;//case '3':  BRNE case\_4  IN R16,PORTC ;// PORTC^=(1<<3); PORTC=PORTC^(1<<3);  LDI R17, (1<<3);  EOR R16,R17 ; // R16=R16^R17  OUT PORTC,R16  case\_4: CPI received\_data,'4' ;//case '4':  BRNE case\_5  IN R16,PORTC ;// PORTC^=(1<<4); PORTC=PORTC^(1<<4);  LDI R17, (1<<4);  EOR R16,R17 ; // R16=R16^R17  OUT PORTC,R16 ;// CPI received\_data,'2' ;//case '2':  case\_5: CPI received\_data,'5' ;//case '5':  BRNE case\_6  IN R16,PORTC ;// PORTC^=(1<<5); PORTC=PORTC^(1<<5);  LDI R17, (1<<5);  EOR R16,R17 ; // R16=R16^R17  OUT PORTC,R16  case\_6: CPI received\_data,'6' ;//case '6':  BRNE case\_7  IN R16,PORTC ;// PORTC^=(1<<6); PORTC=PORTC^(1<<6);  LDI R17, (1<<6);  EOR R16,R17 ; // R16=R16^R17  OUT PORTC,R16  case\_7: CPI received\_data,'7'  BRNE While\_1  IN R16,PORTC ;// PORTC^=(1<<2); PORTC=PORTC^(1<<7);  LDI R17, (1<<7);  EOR R16,R17 ; // R16=R16^R17  OUT PORTC,R16 ;;  jmp While\_1      Tx: ;//void Tx( unsigned char data ){  wait\_tx: IN R16,UCSRA ; //while ( !(UCSRA & (1<<UDRE)) ); // wait until UDR is empty  LDI R17, (1<<UDRE);  AND R16,R17  BREQ wait\_rx ;; check Z=1 repeat wait\_tx  OUT UDR, received\_data ; UDR = received\_data; // Putting data into UDR, sends the data Tx(received\_data);  RET  Rx: ;//unsigned char Rx(){  wait\_rx: IN R16,UCSRA ; //while ( !(UCSRA & (1<<RXC)) ); // wait for the Receive Complete (RXC) Flag  LDI R17, (1<<RXC);  AND R16,R17  BREQ wait\_rx ;; check Z=1 repeat wait\_rx  IN received\_data,UDR ; //return UDR;  RET  init\_UART: ;//void init\_UART(){  LDI R16,(1<<RXEN) |(1<<TXEN) ; // UCSRB = (1<<RXEN) |(1<<TXEN); // Enable USART Receiver and Transmitter  OUT UCSRB, R16  LDI R16, (1<<UCSZ1)|(1<<UCSZ0) ;//UCSRC = (1<<UCSZ1)|(1<<UCSZ0); // Mode3: Use 8-bit data  OUT UCSRC,R16  LDI R16, HIGH(383) ;//UBRRH =0;  OUT UBRRH,R16  LDI R16,LOW(383) ; //UBRRL = 23; For 7.372800 MHz Crystal and 19200 baud rate  OUT UBRRL,R16  RET | |
| Instruction | Comments |
|  |  |

I.2.4 Result

Check the the sending character in PC and check the led toggled led in PORTC.

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

I.3 TASK3 : Interface Microcontroller with PC using UART Pheripheral using interrupt method - C Programming

I.3.1 Problem Statement: Design an embedded systems using ATMEGA32 (working at 11.0592Mhz) communicate with PC by UART serial interface. The system has 8 LEDs connected to PORTC. When a system receiving new character from the UART port the system will display this character to Seven-segment LED in form Hexa Number, and toggle the relevant LED ( for example the character received is ‘0’ LED0 will be toggled, the character received is ‘1’ LED1 will be toggled and so on … the character received is ‘7’ LED7 will be toggled ). The system using interrupt method and UART setting with baud rate 38400, 8 bit, 1 stop bit, no parity. Write C programm to control the system.

I.3.2 Circuit Design:

Student Design the application circuit in Proteus

I.3.3 Theory:

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

- UART Control Registers

- UART initialization explanation for the task1 requirements ( baud rate 19200 8 bit, 1 stop bit, no parity )

I.3.4 Algorithm

1. Develop a function to initialize UART with baud rate 38400, 8 bit, 1 stop bit, no parity.

2. Develop a function to send a character

3. Develop an Interrupt service routine to receive a character, when receiving a new character this function will toggle the relevant led.

4. Write main program

+ Setting up PORTC as output connect to 8 LEDs

+ Setting up PORTB as input connect to 8 Buttons

+ Enable UART receiver (Interrupt method)

+ While(1) loop :

- Call polling function to receive a character

I.3.5 Code

|  |  |
| --- | --- |
| Write general comments about the program here  /\*----------------------------------------------------------------  This program is a C program to send and receive data to and from UART.  Author: xxxxxxxxx Date: xx.xx.20xx  ---------------------------------------------------------------------\*/  #include <avr/io.h>  #include <avr/interrupt.h>  char received\_data;  void Tx( unsigned char data ){  while ( !(UCSRA & (1<<UDRE)) ); // wait until UDR is empty  UDR = data;  }  ISR (USART\_RXC\_vect){  received\_data = UDR;  PORTB = received\_data;  switch(received\_data)  {  case '1': PORTC^=(1<<0);  break;  case '2': PORTC^=(1<<1);  break;  case '3': PORTC^=(1<<2);  break;  case '4': PORTC^=(1<<3);  break;  case '5': PORTC^=(1<<4);  break;  case '6': PORTC^=(1<<5);  break;  case '7': PORTC^=(1<<6);  break;  case '8': PORTC^=(1<<7);  break;  default:  PORTC=0x00;  break;  }  Tx(received\_data);  }  void init\_s(){  DDRD = DDRD & 0xFE;  UCSRB = (1<<RXEN )|(1<<TXEN )|(1<<RXCIE); // Rx, Tx and Receive Interrupt Enable  UCSRC = (1<<UCSZ1)|(1<<UCSZ0); // 8-bit data  UBRRH = 17>>8; UBRRL = 17; // For 8 MHz Crystal and 1200 baud rate  sei();  }  int main(){  DDRC=0xFF;  PORTC=0x00;  DDRB = 0XFF;    init\_s();  while(1);  } | |
| Instruction | Comments |
|  |  |

I.3.6 Result

Check the designed circuit.

Check the the sending character in PC and check the led toggled led in PORTC.

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

I.4 TASK4 : Interface Microcontroller with PC using UART Pheripheral using Interupt method – AVR Assembly Programming

I.4.1 Problem Statement: Design an embedded systems using ATMEGA32 (working at 11.0592Mhz) communicate with PC by UART serial interface. The system has 8 LEDs connected to PORTC. When a system receiving new character from the UART port the system will display this character to Seven-segment LED in form Hexa Number, and toggle the relevant LED ( for example the character received is ‘0’ LED0 will be toggled, the character received is ‘1’ LED1 will be toggled and so on … the character received is ‘7’ LED7 will be toggled ). The system using interrupt method and UART setting with baud rate 57600, 8 bit, 1 stop bit, no parity. Write AVR Assembly program to control the system.

I.4.3 Code

|  |  |
| --- | --- |
| Write general comments about the program here  /\*----------------------------------------------------------------  This program is an AVR Assembly program to send and receive data to and from UART.  Author: xxxxxxxxx Date: xx.xx.20xx  ---------------------------------------------------------------------\*/ | |
| Instruction | Comments |
|  |  |

I.4.4 Result

Check the the sending character in PC and check the led toggled led in PORTC.

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

I.5 TASK5: UART Microcontroller inter-system connection

I.5.1 Problem Statement: Design two embedded systems using ATMEGA32 (working at 11.0592Mhz) communicate together by UART0 serial interface. Each system has a LCD to display received data and has 8 buttons and 8 LEDs. When button0 is pressed the system will send character ‘0’ to other system, and button1 is pressed the system will send character ‘1’ and so on … button7 is pressed the system will send character ‘7’. When a system receiving new character from the UART port the system will display this character to LCD and toggle the relevant LED ( for example the character received is ‘0’ LED0 will be toggled). The two system using baud rate 115200, 8 bit, 1 stop bit, no parity.

I.5.2 Circuit Design:

Student Design the application circuit in Proteus

I.5.3 Theory:

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

- UART Control Registers

- UART initialization explanation for the task1 requirements ( baud rate 115200 8 bit, 1 stop bit, no parity )

I.5.4 Algorithm

1. Develop a function to initialize UART with baud rate 9600, 8 bit, 1 stop bit, no parity.

2. Develop a function to send a character

3. Develop an interrupt function to receive a character, when receiving a new character this function will toggle the relevant led.

4. Write main program

+ Setting up PORTC as output connect to 8 LEDs

+ Setting up PORTB as input connect to 8 Buttons

+ Initialize LCD (2x16) which connect to PORTD working in 4 bit mode

+ Enable UART receiver interupt

+ While(1) loop :

- Check each of 8 buttons is pressed of not.

If any button is pressed, send the relevant character to UART.

I.5.5 Code

|  |  |
| --- | --- |
| Write general comments about the program here  /\*----------------------------------------------------------------  This program is a multi-tasks program to send and receive data to and from UART.  Author: xxxxxxxxx Date: xx.xx.20xx  ---------------------------------------------------------------------\*/ | |
| Instruction | Comments |
|  |  |

I.5.6 Result

Check the designed circuit.

Check the button pressed in one system and check the led toggled and the data display in other system.

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

I.6 TASK6 : Using UART and ADC

I.6.1 Problem Statement:

Design an embedded systems using ATMEGA32 (working at 7.372800Mhz) communicate with PC by UART0 serial interface. the system has a LCD to display received data and ADC Data. 8 ADC channels are connected to 8 LM35 sensors. When a system receiving new command character from the UART port the system will display command to LCD and get the relevant ADC channel then display this data to LCD ( for example when the character received is ‘0’ the ADC channel0 will be sampled and displayed to LCD, for example when the character received is ‘1’ the ADC channel1 will be sampled and displayed to LCD , …. for example when the character received is ‘0’ the ADC channel0 will be sampled and displayed to LCD,. The UART using baud rate 38400, 8 bit, 1 stop bit, no parity. ADC voltage reference is connected to 5V.

I.6.2 Circuit Design:

Student Design the application circuit in Proteus

I.6.3 Theory:

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

- UART Control Registers

- ADC Control Registers

- UART initialization explanation for the task3 requirements ( baud rate 38400, 8 bit, 1 stop bit, no parity )

- ADC Initilization

I.6.4 Algorithm

1. Develop a function to initialize UART with baud rate 38400, 8 bit, 1 stop bit, no parity.

3. Develop an interrupt function to receive a character and save in string buffer, when receiving a new character this function will check the string command and control the relevant ADC channel sampling.

4. Write main program

+ Initialize LCD (2x16) which connect to PORTD working in 4 bit mode

+ Enable UART receiver interrupt

+ Enable ADC interrupt with ADCMUX can be changed by UART receiver

interrupt

+ While(1) loop :

I.6.5 Code

|  |  |
| --- | --- |
| Write general comments about the program here  /\*----------------------------------------------------------------  This program is a multi-tasks program to receive data to and from PC using UART and control the ADC sampling  Author: xxxxxxxxx Date: xx.xx.20xx  ---------------------------------------------------------------------\*/ | |
| Instruction | Comments |
|  |  |

I.6.6 Result

Check the designed circuit.

Check the system operation.

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

I.7 TASK7 : Farm Watering Control system ( Advanced Question)

Design a simple digital clock using ATMEGA32 timer 1. ATmega32A microcontroller has a 16 bit timer, and we will be using that timer to count the seconds and develop a digital clock and display to LCD with the format hh:mm:ss. There are 3 buttons to adjust the digital clock. Button bt0 INT0 to adjust the hour digits, Button bt1 INT1 to adjust the minute digits, Button bt2 INT2 to adjust the second digits. And three button to adjust the Pumping time (format hh:mm:ss). When are clock value match the pumping time the pin ( with led indicator) will control the Pumping relay to turn on the pump for Pumping duration 10 seconds. The Pumping time, the clock and Pumping duration can be change by PC through UART interface ( Example command : A HH:MM:SS for changing Alarm time, or command C HH:MM:SS for changing clock, command P DD for changing Pumping duration). Write the control program for this circuit using C Language.

I.8 TASK8 : Farm Watering Control system ( Advanced Question)

Design a simple digital clock using ATMEGA32 timer 1. ATmega32A microcontroller has a 16 bit timer, and we will be using that timer to count the seconds and develop a digital clock and display to LCD with the format hh:mm:ss. There are 3 buttons to adjust the digital clock. Button bt0 INT0 to adjust the hour digits, Button bt1 INT1 to adjust the minute digits, Button bt2 INT2 to adjust the second digits. And three button to adjust the Pumping time (format hh:mm:ss). When are clock value match the pumping time the pin ( with led indicator) will control the Pumping relay to turn on the pump for Pumping duration 10 seconds. The system also control the the pump by monitoring the soil moisture sensor when the soil moisture is below a given soil moisture threshold, it will turn on the pump for Pumping duration 10 seconds.

The Pumping time, the clock and Pumping duration can be change by PC through UART interface ( Example command : A HH:MM:SS for changing Alarm time, or command C HH:MM:SS for changing clock, command P DD for changing Pumping duration).

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)
* Designed Circuit diagram
* 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: