|  |  |
| --- | --- |
| MECHATRONICS INTL-PROGRAM | ASSIGNMENT COVERSHEET |

|  |  |  |
| --- | --- | --- |
| STUDENT DETAILS | | |
| STUDENT NAME | 于航 | |
| STUDENT NAME (PINYIN) | YU HANG | |
| STUDENT ID | 17222054 | |
| CLASS NUMBER | 1702 | |
| ASSIGNMENT DETAILS | | |
| COURSE NAME |  | |
| COURSE NUMBER |  | |
| ASSIGNMENT NAME |  | |
| LECTURER/TUTOR |  | |
| SUBMISSION DETAILS | | |
| DATE DUE | 05/26/2020 | |
| SUBMITTION DATE | 05/26/2020 | |
| STAFF ONLY | | |
| RECEIVED BY |  | DATE/TIME |
| MARKER |  | GRADE |

Declaration and Acknowledgement

By submitting this, I declare that:

1. This assignment meets all the requirements of the subject as student in the relevant subject outline, which I have read.
2. (a) This assessment item is entirely my own work, except where I have included fully-documented references to the work of others.

(b) The material contained in this assessment item has not previously been submitted for assessment.

1. I acknowledge that:

(a) The marker of this assessment item may, for the purpose of assessing this assignment, reproduce this assignment and provide a copy to another member of academic staff.

(b) If required to do so, I will provide an electronic copy of this assessment item to the marker.

1. I am aware that late submission without an authorized extension from the subject coordinator may incur a penalty.

Please note: Assignments are not to be submitted by fax and must be submitted during Lectures/tutorials/laboratories or directly to the academic. Only under special circumstances will the Administrative Staff collect assignments

Experiment 2: USART serial communication

1. Object of experiment
   1. be able to correctly explain the principle of serial communication, analyze the serial communication mode, and calculate UBRR;
   2. be able to further design, write and debug serial communication program and its extension program through experiments.
2. Equipment
   1. PC
   2. ATmega128 development board with CH340
3. Experiment content
   1. basic experiment

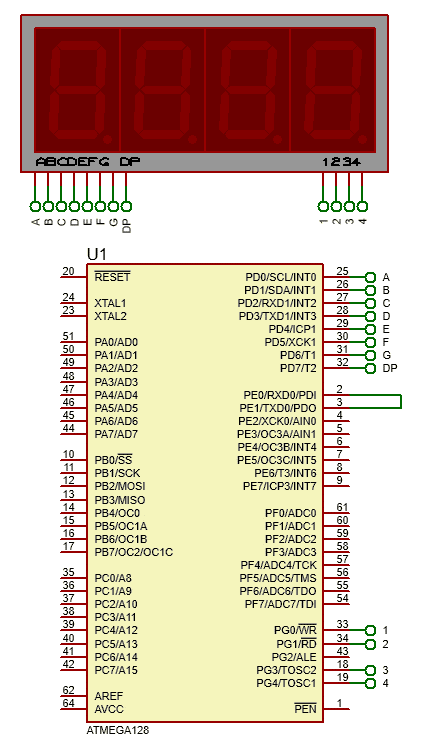
Two comprehensive experiments by using USART0, and the serial port sends and receives data in query mode.

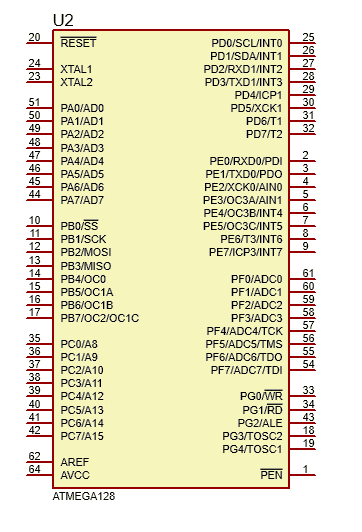
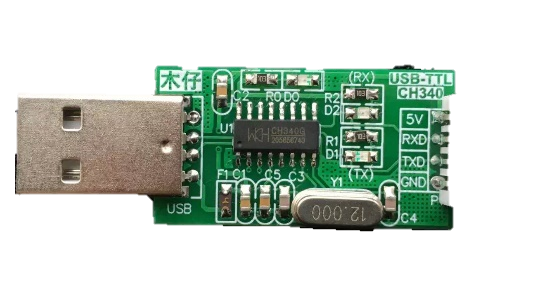
USART0: sends one byte of data every second. Each time, the data is added to 1. Receive data, and display the received data in LED.

Internal crystal frequency 8MHz data transmission baud rate: 9600, 8-bit data at double speed, 1-bit stop, asynchronous, no check. Query mode transfer.

* 1. extended experiment

Using the USART of ATmega128 to establish the RS 232 connection communication with COM port of PC, press the characters of the keyboard of PC, transmit to the USART0 of ATmega128 through the serial port, send USART0 to PC and display it on the screen of PC.

1. Experimental schematic diagram



Circuit for basic experiment Circuit for extended experiment

1. Code and flow diagram
   1. basic experiment I
      1. flow diagram



* + 1. code

|  |
| --- |
| 1. #include<iom128v.h> 2. #include <macros.h> 4. #define fosc 8000000 5. #define baud 9600 7. unsigned **char** tab[10]={0xC0, 0xF9, 0xA4, 0xB0, 0x99, 8. 0x92, 0x82, 0xF8,0x80, 0x90}; 9. unsigned **char** i,data; 11. //to get 1 us delay in 8MHz crystal 12. **void** delay\_us(unsigned **int** microsecond){ 13. **do**{ 14. microsecond--; 15. }**while** (microsecond>1); 16. } 18. // to get 1 ms delay 19. **void** delay\_ms(unsigned **int** millisecond){ 20. **while** (millisecond--){ 21. delay\_us(999); 22. } 23. } 25. **void** init\_usart(**void**){ 26. UCSR0A = 0x02;//double speed 27. UCSR0B = 0x00; 28. UCSR0C = 0x86;//asynchronous, no parity, 1 stop bit,8 data bits 29. UBRR0L = (fosc/(8\*baud)-1)%256;//Baud Rate initialization 30. UBRR0H = (fosc/(8\*baud)-1)/256; 31. UCSR0B = 0x18;//enable receive and trasmitter 32. } 34. **void** send(unsigned **char** a){ 35. UDR0 = a;//load data to UDR for transmission 36. **while**(!(UCSR0A&BIT(UDRE0)));//wait for UDRE flag 37. } 39. **void** init\_port(**void**){ 40. DDRD  = 0XFF; 41. PORTD = 0XFF; 42. PORTG = 0x1B; 43. DDRG  = 0x1B; 44. } 46. unsigned **char** receive(**void**){ 47. **while**(!(UCSR0A&BIT(RXC0)));//wait for RXC flag 48. **return** UDR0;//retrieve data from UDR and return 49. } 51. **void** main(**void**){ 52. init\_port(); 53. init\_usart(); 54. **while**(1){ 55. **for**(i=0;i<10;i++){ 56. send(i); 57. data=receive(); 58. PORTD=tab[data]; 59. delay\_ms(1000); 60. } 61. } 62. } |

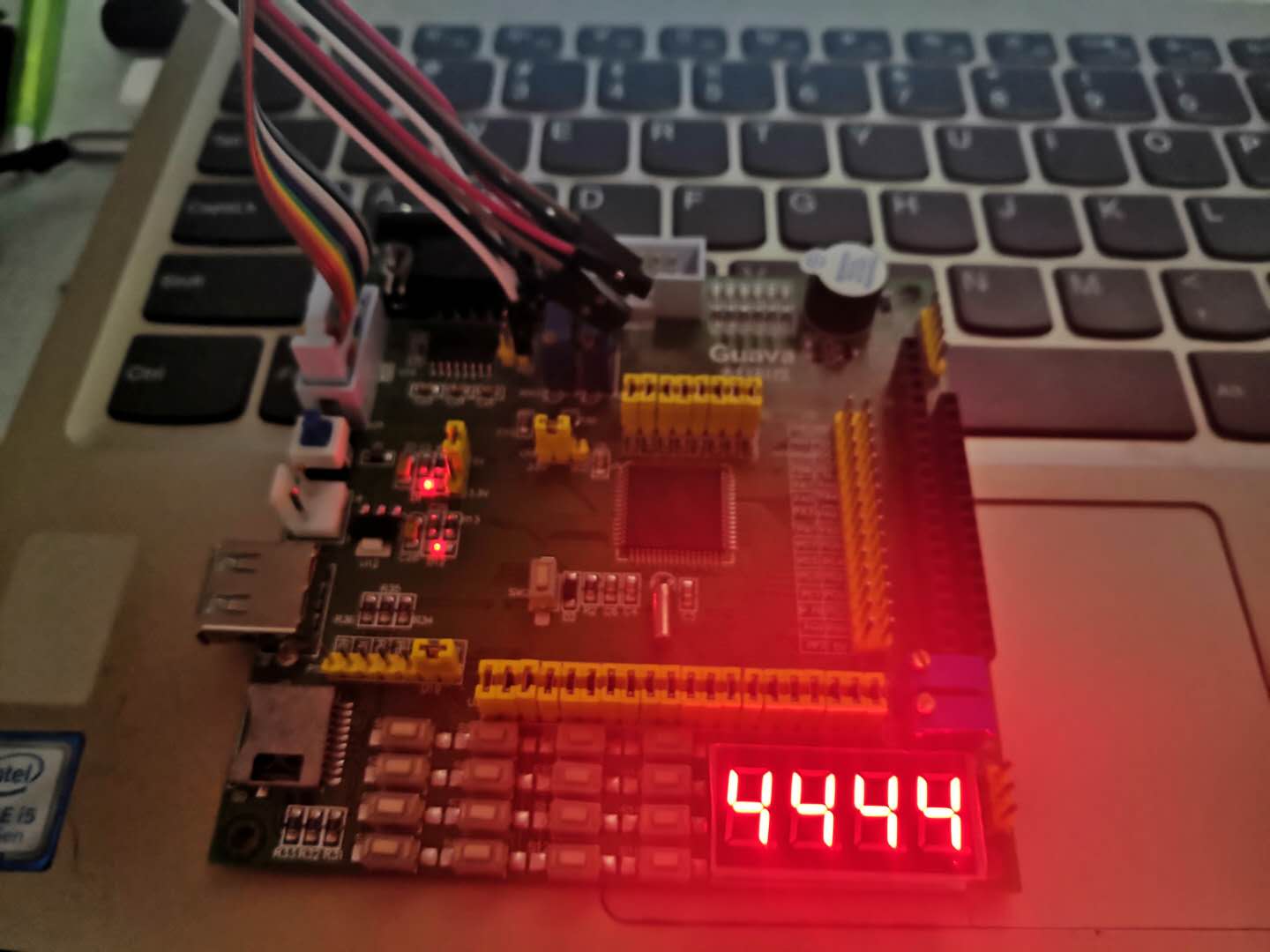
* 1. extended experiment
     1. flow diagram



* + 1. code

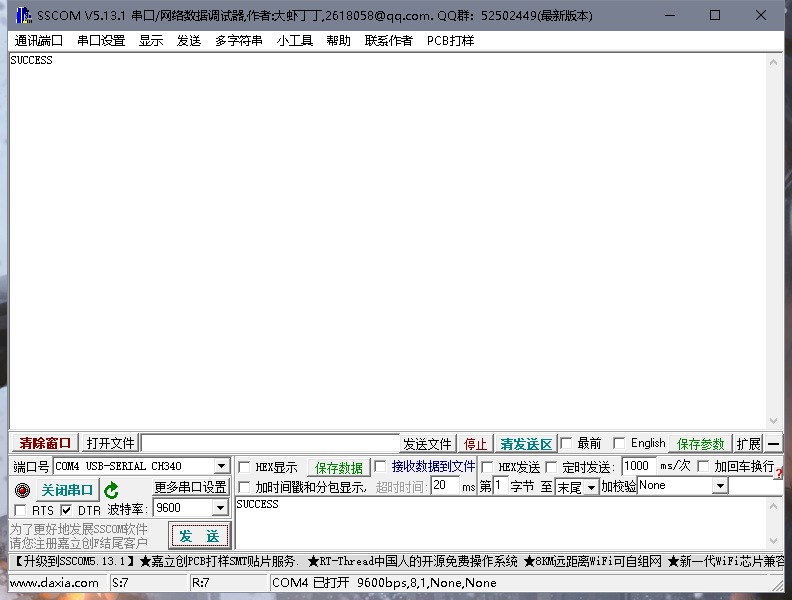
|  |
| --- |
| 1. #include<iom128v.h> 2. #include <macros.h> 3. #define fosc 8000000 4. #define baud 9600 6. unsigned **char** rdata; 8. //to get 1 us delay in 8MHz crystal 9. **void** delay\_us(unsigned **int** microsecond){ 10. **do**{ 11. microsecond--; 12. }**while** (microsecond>1); 13. } 15. // to get 1 ms delay 16. **void** delay\_ms(unsigned **int** millisecond){ 17. **while** (millisecond--){ 18. delay\_us(999); 19. } 20. } 22. **void** init\_usart(**void**){ 23. UCSR0A = 0x02;//double speed 24. UCSR0B = 0x18;//enable receive and trasmitter 25. UCSR0C = 0x86;//asynchronous, no parity, 1 stop bit,8 data bits 26. UBRR0L = (fosc/(8\*baud)-1)%256;//Baud Rate initialization 27. UBRR0H = (fosc/(8\*baud)-1)/256; 28. } 30. **void** send(unsigned **char** a){ 31. UDR0 = a;//load data to UDR for transmission 32. **while**(!(UCSR0A&BIT(UDRE0)));//wait for UDRE flag 33. }  36. unsigned **char** receive(**void**){ 37. **while**(!(UCSR0A&BIT(RXC0)));//wait for RXC flag 38. **return** UDR0;//retrieve data from UDR and return 39. } 41. **void** main (**void**){ 42. init\_usart(); 43. **while**(1){ 44. rdata=receive(); 45. send(rdata); 46. } 47. } |

1. Experiment Result
   1. Basic experiment



The seven-segment digital tube displays a sequence of Numbers from 0 to 910, with a time interval of about 1s

* 1. Extended experiment



The serial debugging assistant sends a data to the ATmega128, the ATmega128 will return the data to the computer, and display the data in the serial debugging assistant.

1. Reflection

To initial the USART, we should set 5 registers: UCSRA, UCSRB,UCSRC, UBRRL and UBRRH. we always need to set speed, data bits, stop bit, parity bit, enable receive and transmitter. which are shown following.





U2X: Double the USART Transmission Speed 1 to double the transmission speed

RXEN: Receiver Enable1 to enable USART receiver: Pin D.0 = RXD pin

TXEN: Transmitter Enable1 to enable USART transmitter: Pin D.1 = TXD pin

UMSEL: USART Mode Select To select USART modes: 0 asynchronous, 1 synchronous

UPM1:0: Parity Mode To select parity mode: 00 no parity, 10 even parity, 11 odd parity, 01 Reserved

USBS: Stop Bit Select To select stop bit mode: 0 1 stop bit, 1 2 stop bits

UCSZ2:0: Character Size

When we write the sending program，we should put the data into UDR and monitor UDRE, which will be set 1 when UDRE is empty. then the data is send successfully.

When we write the receiving program, we should monitor RXD which will be set 1 when the data is received successfully.

1. Experience

In this experiment, at the beginning of writing code, although the program is compiled through, but in the operation of the MCU, the MCU will constantly send information to the computer. After checking, “&” was misused as “&&”. And that leads to the fact that when you run the sending or receiving program, the discriminant condition is always true, which leads to errors.

“&&” is logical operation while “&” is bit operation，we need to distinguish between the two operators, when writing a program.