|  |  |
| --- | --- |
| 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 | 06/16/2020 | |
| SUBMITTION DATE | 06/16/2020 | |
| 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 7: Comprehensive Experiment

1. Object of experiment

This experiment is a comprehensive experiment, its content covers the serial port and interrupt, timer/counter, modulus conversion, digital tube display, etc. Through the experiment can correct operation microcontroller part of the system software, programming used ATmega128 hardware, at the same time combined with a serial port and interrupt, timer/counter, modulus conversion, digital tube display at least more than two kinds of knowledge module design debugging program implementation modules work together.

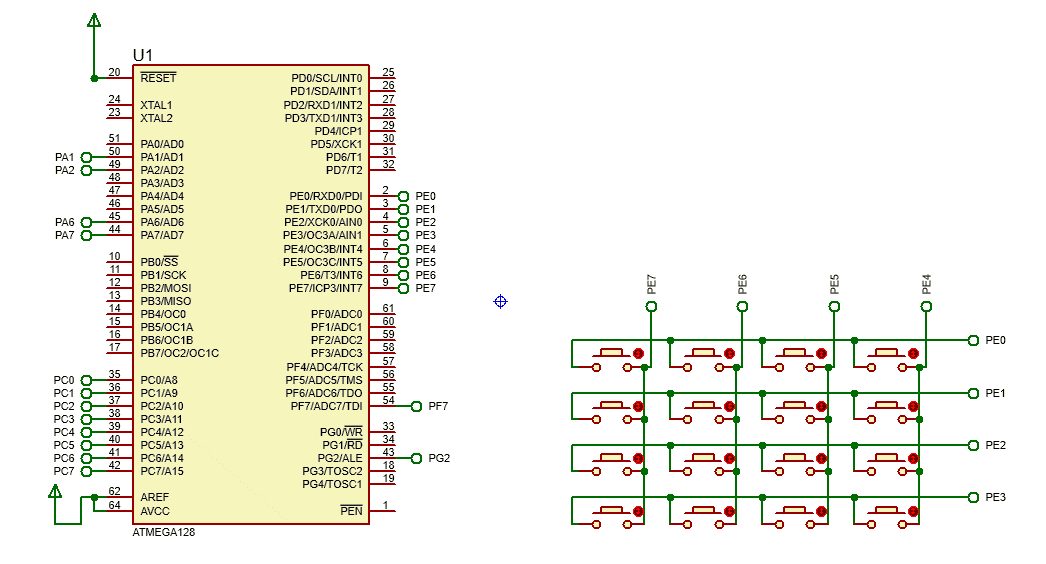
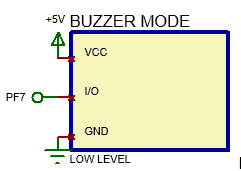
1. Experiment content
   1. basic experiment

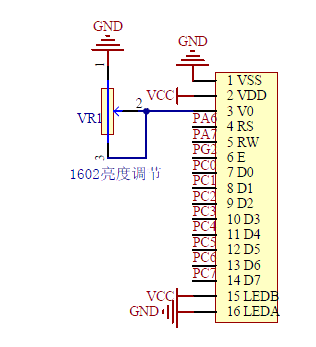
(1) Eight buttons on the development board are used to select tones, so that the buzzer can sound eight tones.

(2) The mode is changed by interrupting the control button, so that the button is changed from controlling the buzzer to controlling the buzzer to playing the complete song, which is stored in the single chip microcomputer in advance.

(3) Add songs played control function, when in the songs mode, use different buttons to control pause and continue to PLAY songs, and displayed on the digital tube tip PLAY and STOP.

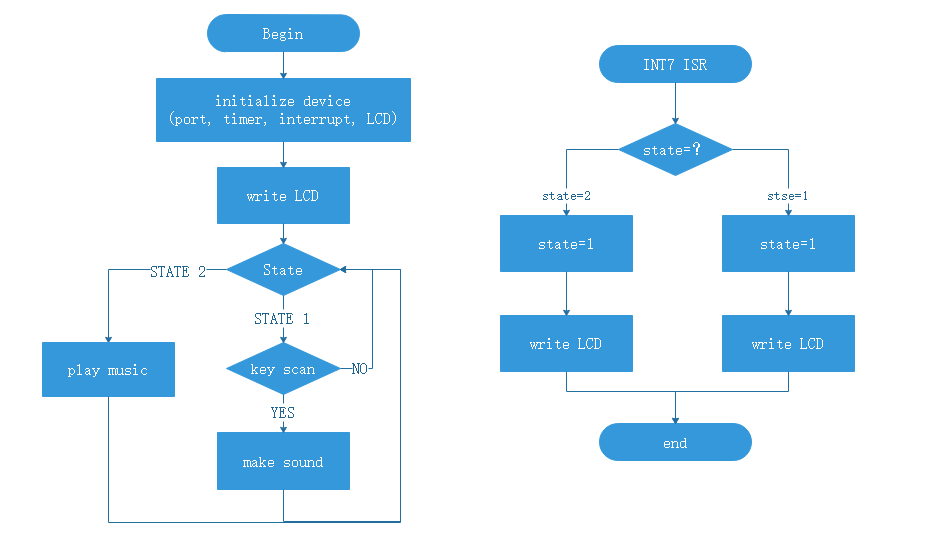
1. Experimental schematic diagram





**LCD1602**

1. Code and flow diagram
   * 1. flow diagram



* + 1. code

|  |
| --- |
| 1. #include <iom128v.h> 2. #include <macros.h> 3. #include "lcd\_1602.c" 5. #define L1 262 6. #define L2 294 7. #define L3 330 8. #define L4 349 9. #define L5 392 10. #define L6 440 11. #define L7 494 12. #define M1 523 13. #define M2 587 14. #define M3 659 15. #define M4 698 16. #define M5 784 17. #define M6 880 18. #define M7 988 19. #define H1 1047 20. #define H2 1175 21. #define H3 1319 22. #define H4 1397 23. #define H5 1568 24. #define H6 1760 25. #define H7 1967 27. unsigned **int** tune[]={0,M1,M2,M3,M4,M5,M6,M7,H1}; 28. unsigned **int** star[]= { 29. H1,8, H1,8, H5,8, H5,8, H6,8, H6,8, H5,16, 30. H4,8, H4,8, H3,8, H3,8, H2,8, H2,8, H1,16, 31. H5,8, H5,8, H4,8, H4,8, H3,8, H3,8, H2,16, 32. H5,8, H5,8, H4,8, H4,8, H3,8, H3,8, H2,16, 33. H1,8, H1,8, H5,8, H5,8, H6,8, H6,8, H5,16, 34. H4,8, H4,8, H3,8, H3,8, H2,8, H2,8, H1,16, 35. 0xFF  //as the end of music 36. }; 38. unsigned **char** state=1; 40. **void** init\_port(**void**){ 41. DDRF=0xFF; 42. PORTF=0xFF; 43. DDRE=0X0F; 44. PORTE=0XF0; 45. } 47. **void** init\_timer1(**void**){ 48. TIMSK=0x00;//disable timer1 49. TCCR1A|=(0<<WGM11)|(0<<WGM10); 50. TCCR1B|=(1<<WGM12);//CTC 51. TCCR1A|=(0<<COM1A1)|(0<<COM1A0);//OC1A NORMAL MODE OUTPUT 52. TCCR1B|=(0<<CS12)|(1<<CS11)|(0<<CS10);//8prescale 53. } 55. **void** init\_int(**void**){ 56. EICRB=0x80;//risng edge of INT7 57. EIMSK=0x80;//enable INT7 58. asm("sei"); //enable global 59. } 61. //to get key value from keybord 62. unsigned **char** key\_scan(**void**){ 63. unsigned **char** X,Y,key,key\_value=0; 64. DDRE=0X0F; 65. PORTE=0XF0; 66. **if**(PINE!=0XF0){ 67. delay\_ms(10); 68. **if**(PINE!=0XF0){ 69. X=PINE|0X0F; 70. DDRE=0X70; 71. PORTE=0X8F; 72. **if**(PINE!=0X8F){ 73. delay\_ms(10); 74. **if**(PINE!=0X8F){ 75. Y=PINE|0XF0; 76. key=X&Y; 77. **switch** ( key ) { 78. **case** 0xbe: key\_value=1; **break**; 79. **case** 0xde: key\_value=2; **break**; 80. **case** 0xee: key\_value=3; **break**; 81. **case** 0xbd: key\_value=4; **break**; 82. **case** 0xdd: key\_value=5; **break**; 83. **case** 0xed: key\_value=6; **break**; 84. **case** 0xbb: key\_value=7; **break**; 85. **case** 0xdb: key\_value=8; **break**; 86. **default**:key\_value=0; **break**; 87. } 88. } 89. } 90. } 91. } 92. **return** key\_value; 93. } 95. **void** music(unsigned **int** \*song){ 96. **while**(\*song!=0XFF){ 97. delay\_ms(5); 98. OCR1A=(unsigned **int**)(500000/(\*song)-1); 99. //OCR1=8M/(8\*2\*f)-1 100. TIMSK=0x10;//enable timer1 compture interrupt 101. song++;//get time 102. delay\_ms(52\*(\*song)); 103. song++;//get another tune 104. TIMSK=0x00;//disable timer1 105. **if** (state==1)**break**;//change state 106. **if** (key\_scan()==1){//key 1 to pause 107. LCD\_write\_str(11,0,"STOP"); 108. **while**(!(key\_scan()==2)); 109. //key 2 to play 110. } 111. LCD\_write\_str(11,0,"PLAY");//play 112. } 113. delay\_ms(1000); 114. } 116. **void** beat(unsigned **int** a){ 117. OCR1A=(unsigned **int**)(500000/a-1);//OCR1=8M/(8\*2\*f)-1 118. TIMSK=0x10;//enable timer1 compture interrupt 119. delay\_ms(400); 120. TIMSK=0x00;//disable timer1 121. } 123. #pragma interrupt\_handler timer1:iv\_TIM1\_COMPC 124. **void** timer1(**void**){ 125. PORTF^=BIT(7); 126. //inversing PF7 to make sound 127. } 129. #pragma interrupt\_handler int7:9 130. **void** int7(**void**){ 131. **if**(state==1){ 132. state=2; 133. LCD\_clear(); 134. LCD\_write\_str(0,0,"MODE:Music"); 135. LCD\_write\_str(0,1,"S1:Change mode"); 136. } 137. **else** **if**(state==2){ 138. state=1; 139. LCD\_clear(); 140. LCD\_write\_str(0,0,"MODE:E-organs"); 141. LCD\_write\_str(0,1,"S1:Change mode"); 142. } 143. } 145. **void** main(**void**){ 146. init\_port(); 147. init\_timer1(); 148. init\_int(); 149. LCD\_init();//initalize device 150. LCD\_clear(); 151. LCD\_write\_str(1,0,"MODE: E-organs"); 152. LCD\_write\_str(0,1,"S1:Change mode"); 153. **while**(1){ 154. **if**(state==1)//Play a single tone 155. **if**(key\_scan()) beat(tune[key\_scan()]); 156. **else** **if**(state==2) music(star); 157. //play music 158. } 159. } |

1. Experiment Result

Turn on the power, the system in Electric organs mode, if keys that correspond to tones are pressed, the MCU will make a corresponding sound. When S1 is pressed, the system will change into music mode, and the MCU will play “twinkle star ”, key 1 and key 2 control the music “pause ” and “play”. And if S1 is pressed again, the system will return to Electric organs mode.

1. Reflection

How can we use the same external interrupt button to switch different functions?

In the above program, I set a variable called "State" that is controlled by an external interrupt button. Also in the main function there will be a set of “if...else… ” structure to control what programs run with different “state ” values. Just like, when “state” equal to “1”, the system will show "MODE: E-organs" and run the function named “tune”. and if “state” equal to “2”, the system will show "MODE: Music" and run the function named “music” to play a song.

1. Experience

In this experiment, I tried to use LCD1602 as display, Timer1 and buzzer module to play music and a single scale, and matrix keyboard for keystroke control and external interrupt application.

Through this experiment, I have a more intuitive understanding of the external module and a deeper understanding of the priority of interrupt vectors.

At the same time, the logic of matrix keyboard was updated, so that it could respond to external interrupt no. 7 in addition to ordinary keystrokes.

In a word, In this experiment, I gained a deeper understanding of the monolithic integrated system and its application.