

Chapter 1: PICLAB-V2 Development Board System Introduction

1.1 Product Overview	2
1.2 Board Resource Introduction	3
1.3 Product Schematics	3

Chapter 2: MPLAB IDE Integrated Development Environment

2.1 MPLAB Installation	4
2.2 MPLAB Simple Application	4
2.2.1 Create a Simple Project	4

Chapter 3: Use of On-line Programmer/Debugger

3.1 Tools Programmer / debugger recommended	8
3.2 Typical Connection	9

Chapter 4: PICLAB-V2 System Functional Modules Details

4.1 All I/O External Output Module	10
4.2 Chip Socket and Clock Selection	10
4.3 Ext programmer / Debugger and Reset Button	11
4.4 Power Module	12
4.5 4*4Matrix Keyboard Module	13
4.6 RS232 Module	14
4.7 DS18B2 Module	15
4.8 SPI Communication Module	16
4.9 IIC Communication Module	17
4.10 LCD12864 and 1602 LCD Module	18
4.11A/D Converter Module	19
4.12 Remote Control Receiver & Decoder Module	19
4.13 Six-digital Display Module	20
4.14 8 play-in-turn Light Module	21
4.15 DS1307 Clock Module	22
4.16 Beeper Module	23
4.17 Independent Key and External Interrupt Module	24

Chapter 5: Practice of PICLAB-V2 Development Board 26**Appendix 1: Packing List and Contact** 31

Chapter 1 PICLAB-V2 Development Board System Introduction

1.1 Product Overview

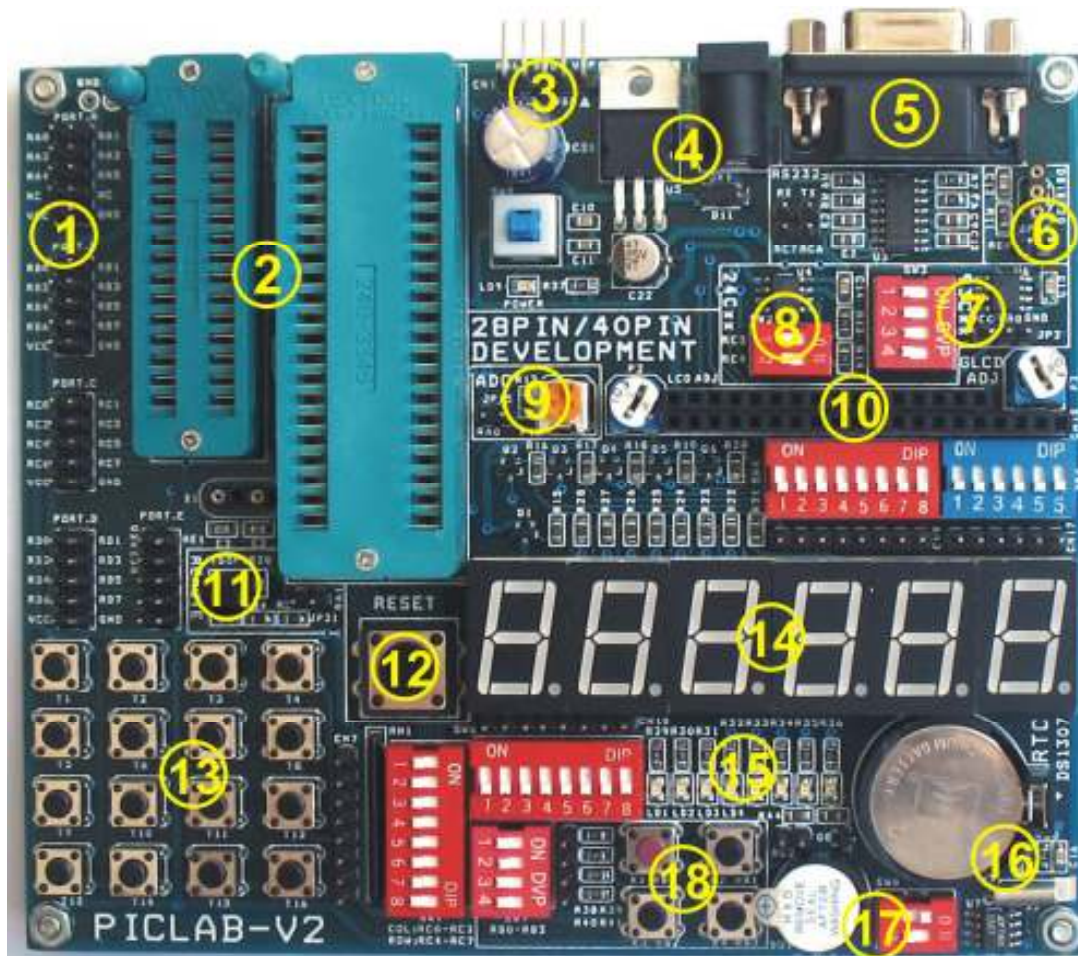
PICLAB-V2 PIC DEVELOPMENT BOARD (hereinafter referred to PICLAB-V2) is a multifunctional PIC microcontroller development platform which has been carefully designed and developed by Thien Minh Electronic Solutions Co., Ltd. based on many years' development experience and the original PICLAB-40 and EasyPIC development Board.

It integrated the common external resources and simulation interfaces. Associating with the data and a large number of examples of programs that our company provides, you will be able to fully master microcontroller programming technology in a shortest time. Particularly it is suitable for self learning for microcontroller beginners as well as electronic lovers.

PICLAB-V2 development board together with online debugger ICD2, PICKit2, PICKit3 produced by our company, or together with online debugger ICD2/PICKit2/PICKit3 produced by Microchip, can bring you a multiplier benefit.

The following points may illustrate how your choice was wise.

- Rich board resources
- Optimized modular design
- Superior production technology
- Rich supporting examples
- Low selling prices
- Comprehensive technical guidance
- Perfect after service



1.2 Board Resource Introduction

- | | |
|--|--|
| 1) I/O external output | 11) Remote control receiver & decoder module |
| 2) 40P/28P chip ZIF socket | 12) Reset button |
| 3) Ext programmer / Simulation interface | 13) 4*4 matrix keyboard module |
| 4) Power module | 14) Six-digital display module |
| 5) RS232 communication module | 15) 8 on-in-turn light module |
| 6) DS18B20 thermometer module | 16) DS1707 with battery backup module |
| 7) SPI 93Cxx communication module | 17) Beeper (Buzzer) module |
| 8) I2C 24Cxx communication module | 18) External interrupt input and Independent button module |
| 9) A/D converter module | |
| 10) LCD12864 and 1602 LCD module | |

1.3 Product Schematics

Schematic please see attached file in CD

Chapter 2 MPLAB IDE Integrated Development Environment

MPLAB IDE (hereinafter referred as MPLAB) is the powerful software integrated development environment provided by Microchip for its PIC microcontroller. It allows users to create, record, edit and comply programs of microcontrollers of PIC series on their own computer systems, and it even can achieves dynamic simulation and debugging and run like virtual exercises.

2.1 The installation of MPLAB

MPLAB is completely free software offered by Microchip. You can obtain the latest installation files through the following two ways.

- 1), Visit our website: www.tme.com.vn
- 2), Visit Microchip's website: www.microchip.com

After downloading the files, you only need to use compression/decompression software tools such as WINZIP to depress and release the files in your computer, and then run SETUP.EXE (or Install.exe) program, and follow a step-by-step installation guide (You may also do not need changes any of the settings, just click "Next") until completion of the installation.

2.2 MPLAB Simple Application

2.2.1 Create a Simple Project

Edit source codes

Click the MPLAB icon at your WINDOWS desktop, or choose Start → All application → Microchip → MPLAB IDE V.xx → MPLAB (Vx.x for MPLAB version) to start running MPLAB integrated environment. Shown as Figure 2 .1.

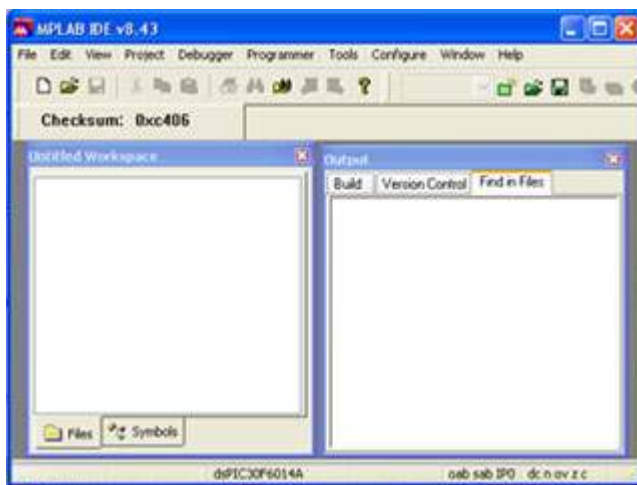


Figure 2-1 MPLAB main window

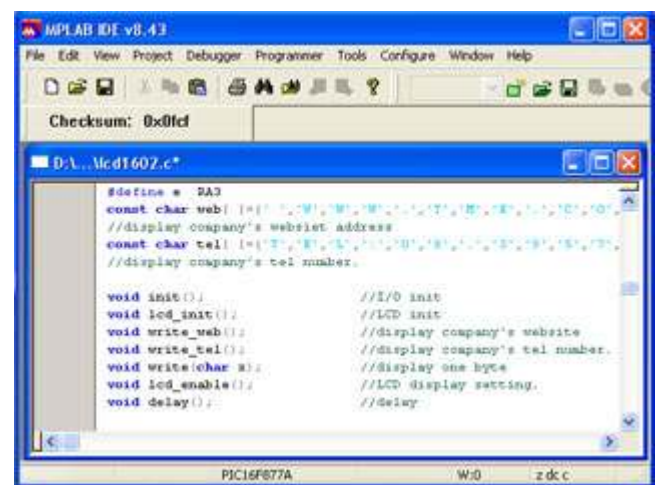


Figure 2-2 text editor window

Choose menu command File → new, MPLAB will automatically call MPLAB Editor (source editor), and the work area will have a text editor window, and you can complete input of source code. As shown in Figure 2-2.

Edit the source codes in "Text editor window", and then select the menu command File → Save to save to the source file to the specified directory, as shown in Figure 2-3.

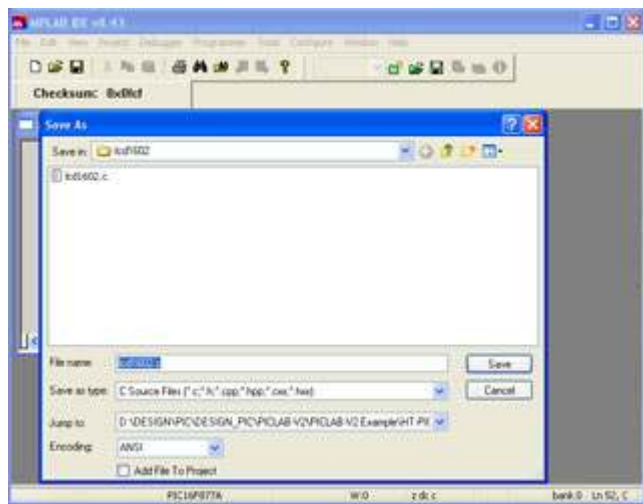


Figure 2-3 Save source codes



Figure 2-4 Project wizard welcome interface

Use the wizard to create project files

Step 1, Chose menu command Project → Project Wizard to come to the Welcome interface shown as Figure 2-4.

Step 2, directly click "Next", and select chip model, as shown in Figure 2-5.

Step 3, click "Next", select the appropriate compiler tools according to the source language and chip to be used, as shown in Figure 2-6.



Figure 2-5 Choose chip model

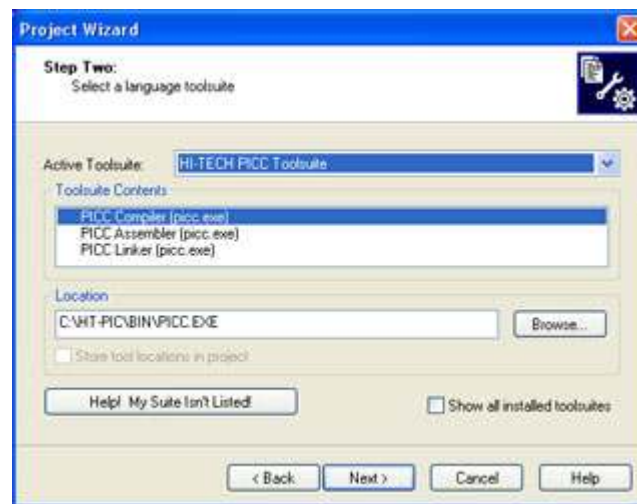


Figure 2-6 Choose compiler tools

Step 4, click "Next", choose the directory where the project is saved and complete the project name, as shown in Figure 2-7.

Step 5 click "Next", the add source codes to the project, as shown in Figure 2-8.

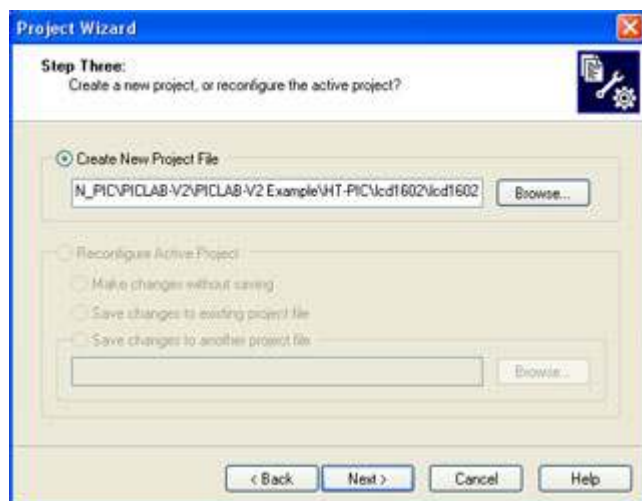


Figure 2-7 Choose directory to save project



Figure 2-8 source selection

Step 6: click "Next", as shown in Figure 2-9 to come to tips interface.



Figure 2-9 Tip interface

Step 7, directly click "Done", and exit the wizard.

So far, we have completed establishing a project the source. For more details, please refer to MPLAB Operation Manual.

2.2.2 Debugging the program

Based on the source code edited and project created in above, this section will show you a brief introduction about compiling and debugging a program.

Compile

Implement menu command Project → Build All and MPLAB will automatically call the tools mentioned in above Step 3 of establishment of project for you to compile this source code. When completing compile, the interface will be shown as Figure 2-10.

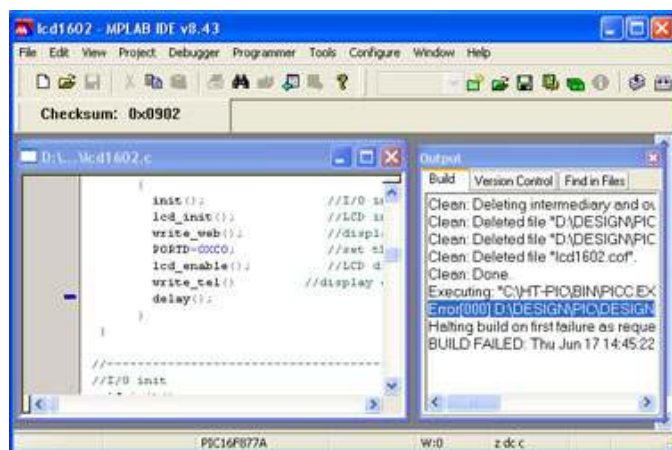


Figure 2-10 Source compiled results

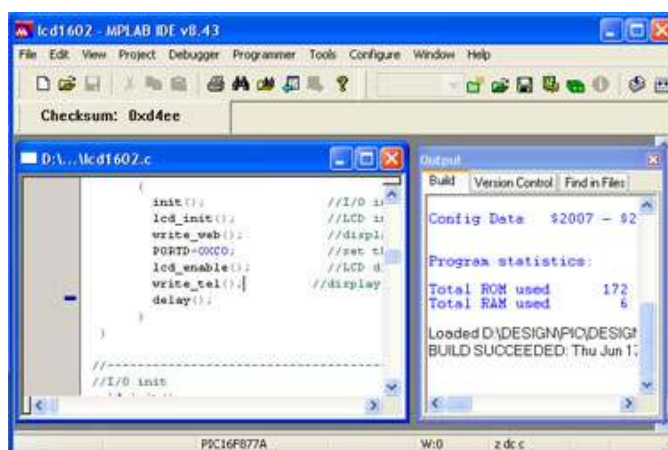


Figure 2-11 Compiling succeeds

From the output window of Figure 2-10, we can see the results that the program compiling failed because of a wrong source, double-click the message, and the cursor will automatically stay at the line where there's an error, and at the margin of most left of this line there's a "Green Arrow". Clearly, behind the jaw "write_tel ()" was missing the ";". Now correct the typo and compile it again, the results are shown in Figure 2-11.

If there is an error in the program, it will not generate the target HEX file, to get HEX file you need to correct all errors in the source codes.

Debugging

Debugging program is to test whether the program you designed is operational, whether it produces correct result as wanted, whether there are any defects in your design, whether the algorithm design is reasonable, and whether it can accurately control the various hardware resources, and whether it can obtain desired results.

Choose debugging tools

Select menu command Debugger → Select Tool, to select the simulator connected to the PC as the debugging tool, or you may select the software debugger which comes with the MPLAB software as the debugger for the target program. After choice, it will open the corresponding toolbar.

Observe debugging results

The internal storage area of PIC microcontroller can be divided into several sections: program memory, hardware stack, file registers, special function registers and EEPROM data memory. In the course of the operation of program, it will repeatedly read, write or modify the contents in the storage area. Therefore, we can observe the changes of content in storage area corresponding to the operation of program so as to understand the operation of program, and achieve the purpose of debugging. To open storage area we can choose the View menu commands, as shown in Figure 2-12.

Apart from the use of this storage area to observe the debugging process, we can also add the concerned specific modules to the observation window to monitor the results. Implementation the menu command View → Watch and the observation window will open, as shown in Figure 2-13

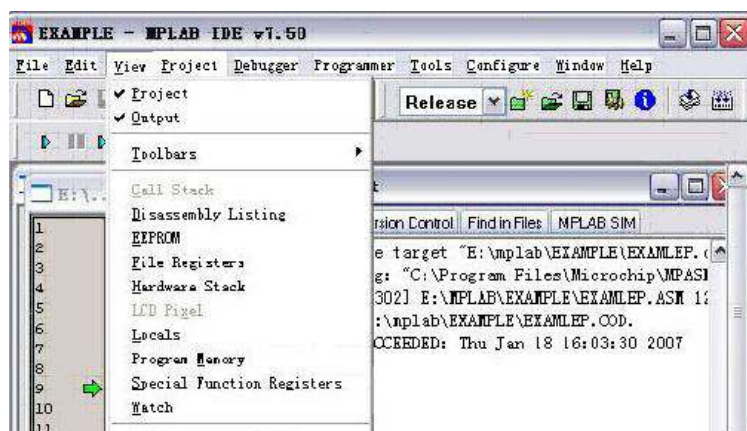


Figure 2-12 Menu command to open storage area

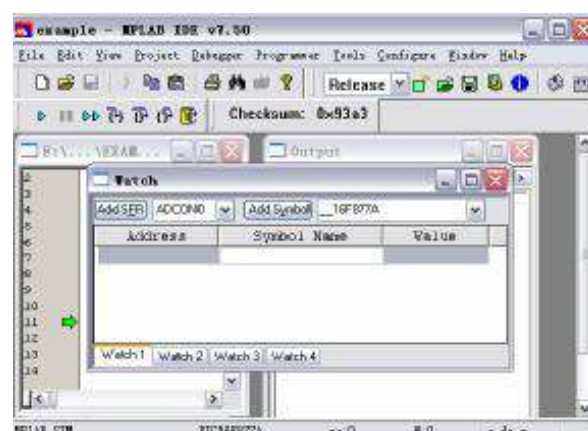


Figure 2-13 Observation window

Thus, we have introduced a simple use of MPLAB, and through the learning in this chapter, we should be able to complete the source code edit, compile and debug.

Note: For more information on the MPLAB please visits the website of MICROCHIP and downloads "MPLAB User Manual".

Chapter 3: Use of On-line Programmer/Debugger

This chapter briefly introduces the tool Programmer / debugger compatible with PICLAB- V2. For more detailed information, please refer its User Manual.

3.1 Tools Programmer / debugger recommended



Tools PICKIT2, PICKIT3 and ICD2 are functional similar of microchip and is fully compatible with PICLAB. View detailed its information in <http://www.tme.com.vn> or www.microchip.com

All Tools Communication via USB 2.0 and have a line ICSP the same order VPP, VDD, GND, PGD, PGC (and NC/AUX – not use). Like the entrance to PICLAB-V2

3.2 Typical connection:



Figure 3-1: PICLAB-V2 connect to PICKIT 2

PICLAB-V2 use a line ICSP for Programmer and debugger in order from right to left as follows: VPP, VDD, GND, PGD and PGC (VPP corresponding triangle stamp on the PICKIT 2 on Figure 3-1).

Chapter 4: PICLAB-V2 System Functional Module Details

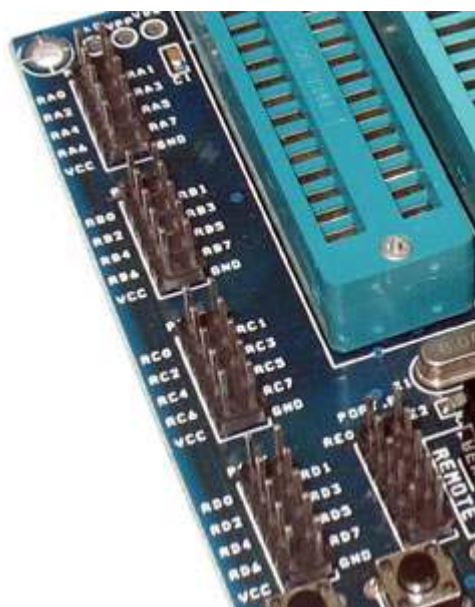
This chapter will describe in more detail on PICLAB-V2 development board and the functional modules by illustrating the schematics of the modules and the points that needs to pay attention to during usage. And in the CD-ROM that is provided along with the product, we have included a number of examples prepared by using such functional modules, and these examples cover the layout of the hardware and software, and the source code with detailed annotation in Vietnam as well as the executed results, which aids to the reference especially for beginners. As the experimental board is modular in design, the configuration of your project may diverse in large amount, and the output ports are all available for external resources. The hardware and program can be at reader's own will, so readers may draw inferences about other cases from one instance.

4.1 All I/O External Output Module

All I/O resources on PICLAB-V2 development board are designed for external output. Users can build their own circuit by taking advantage of the existing resources of PICLAB-V2. As shown in Figure 4-1.

This module consists of the following main components:

- 1) 40 pin chip all I/O (PORTA/B/C/D/E)
- 2) Serial programming voltage VPP.
- 3) Power supply VCC and GND



2) 28 pin chip socket.

3) The system clock selection

PICLAB-V2 can support all 40pin and 20pin PIC16FXXX and PIC18FXXX chips whose pins are compatible with PIC16F87X, as shown in Figure 4-2.

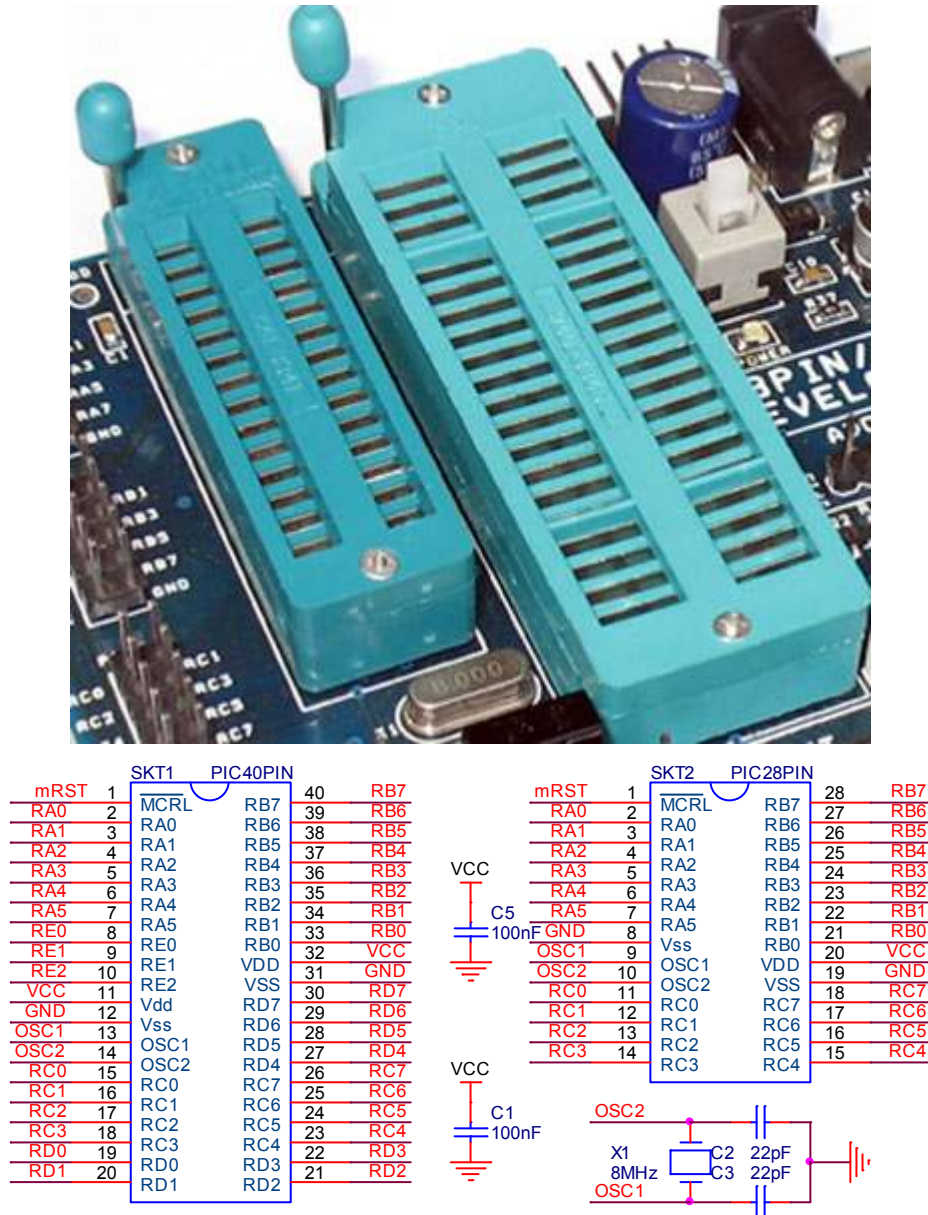


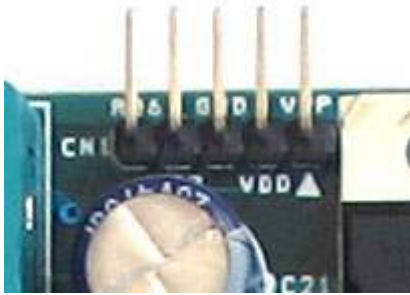
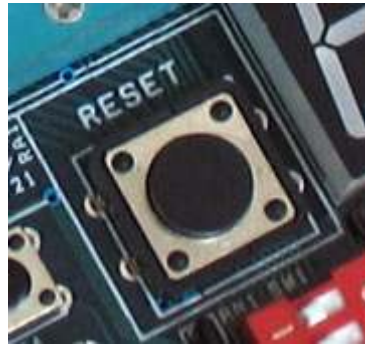
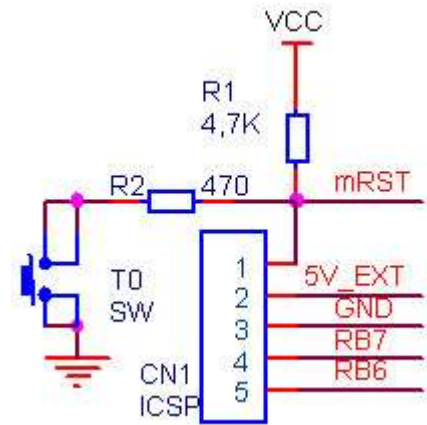
Figure 4-3 system clock selection schematic

4.3 Ext programmer / Debugger and Reset Button

Ext Programmer / Debugger are a line with standard ICSP communication to PICKit2, PICKit3 or ICD2... etc for Programmer, debugger or simulation. It is shown in Figure 4-6

Reset button is used to reset the MCU, as shown in Figure 4-7

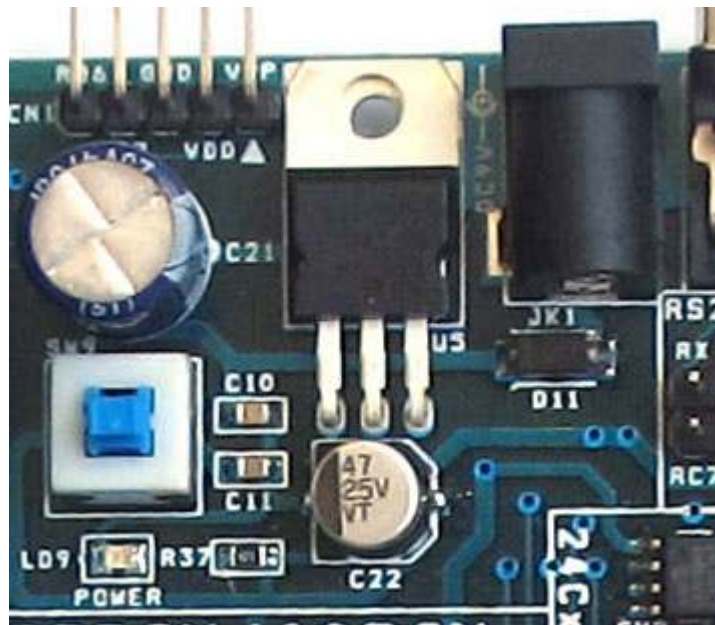
Ext, Programmer / Debugger and reset button schematics are shown in Figure 4-8.

**Figure 4-6** Ext Programmer/debug**Figure 4-7** Reset button**Figure 4-8** Schematics

4.4 Power Module

The Power Development Board adopted the one diode rectifier circuits, we need consider the polarity of external power, (+) inside, (-) outside, the output voltage range is 7-9 VDC and the DC power with current more than 200mA can be used directly. The module is shown in Figure 4-4.

Press the power switch and the whole board power supply is connected, while otherwise the power supply is disconnected.

**Figure 4-4** power modules

The module consists of the following major components:

- 1) Power input.
- 2) Rectifier, filter, regulator.

- 3) The power switch.
- 4) Power indicator LED

Power module schematic shown in Figure 4-5.

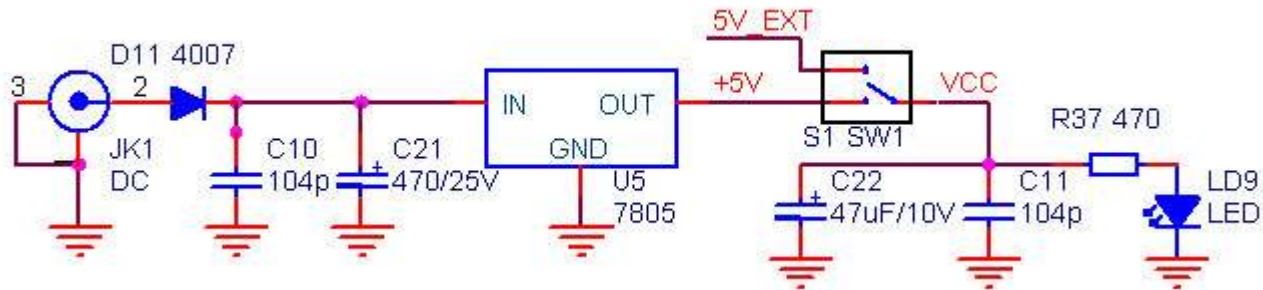


Figure 4-5 schematic power modules

4.5 4*4Matrix Keyboard Module

The module has the following main components:

- 1) 16 keys.
- 2) Coding switching.
- 3) Interface socket.
- 4) Up-pull resistance.

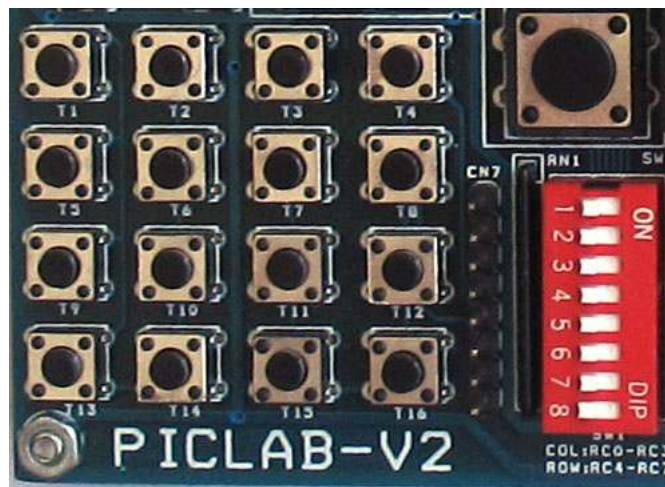


Figure 4-6 4x4 matrix keyboard

Descriptions on this module are as below:

- 1) 16 keys connected to the 8 pins in accordance with the mode 4x4 PORTC.
- 2) The module is controlled by the coding switch. When it is not used, it is recommended to disconnect the switches so as
- 3) Not to affect other modules.

- 4) You can practice the application by using the keyboards from other matrix through interface socket (coding switch Must be at a disconnected status).
- 5) There's a 10K up-pull resistor to ensure that the level of voltage keeps stable.
- 6) In the CD-ROM provided along with product, there're simple examples about "4x4 keyboard matrixex".

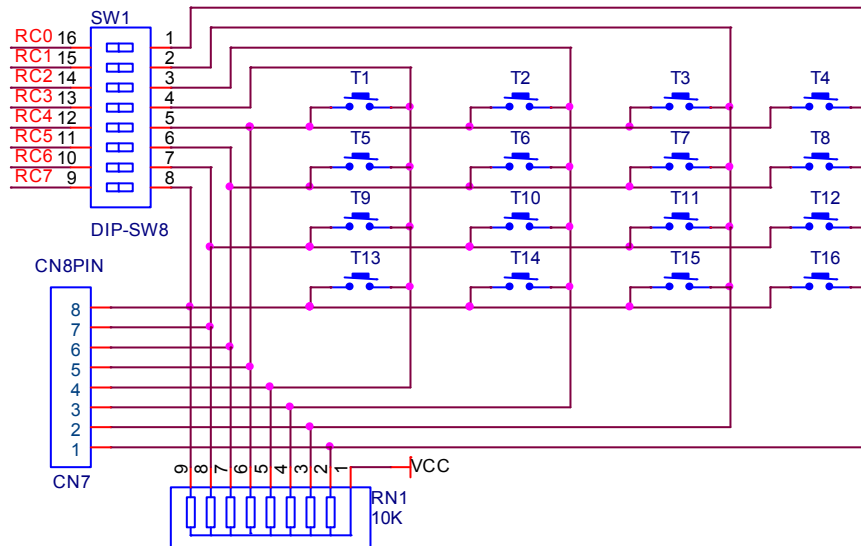


Figure 4-7 4x4 matrix keyboard schematic

4.6 RS232 Module

The modules mainly shows how for the MCU to communicate with external devices such as PC by using USART module, as shown in Figure 4-8.

The module consists of the following main components:

- 1) RS232 voltage level converter chip
- 2) RS232 communication port (9-pin serial port)
- 3) Two jumper wires JP22 and JP23

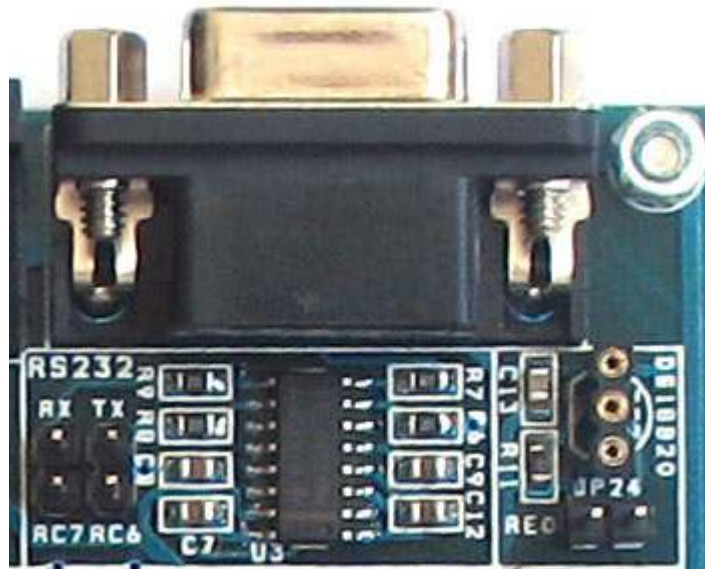
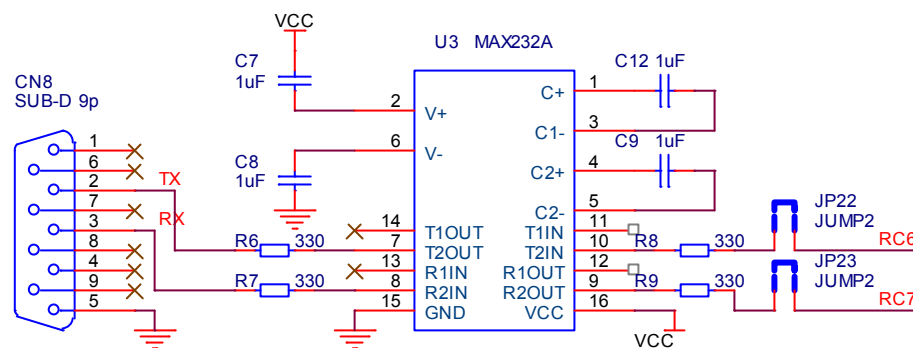


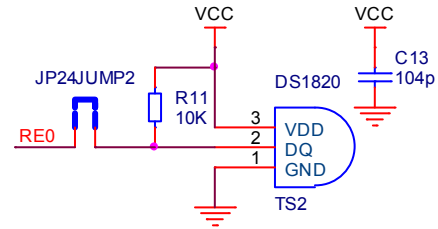
Figure 4-8 RS232 communication

This module is described as below:

- 1) The serial module connected through two jumper wires to the USART module of MCU at the interface of RC6 and RC7 ports.
- 2) Two jumper wires control the connection of serial module and MCU. When this module is in use, we must ensure that access jumper wire is in the connected state, when not in use, we must ensure that the jumper wire is in disconnected status.
- 3) CD-ROM provided along includes the examples about this module for reference.

The schematic of this module is shown in Figure 4-9.



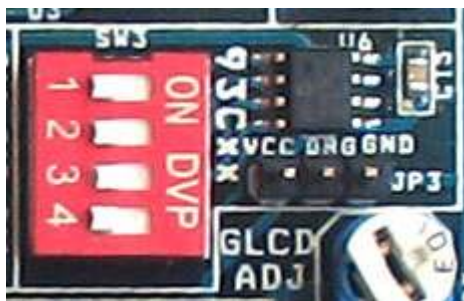
**Figure 4-10** DS18B20 Module**Figure 4-11** DS18B20 schematic

The module consists of the following major components:

- 1) Temperature sensor DS18B20 socket.
 - 2) Jumper wire JP24 (to be used as a switch).
- 1) Descriptions on this module:
- 1) DS18B20 is an optional product to be purchased.
 - 2) DS18B20 information included in the CD-ROM provided along with product.
 - 3) DS18B20 output by connecting jumper wire JP24 to the RE0 pin of the MCU.
 - 4) When this module is in use, we must ensure that access jumper wire is in the connected state, when not in use, we must ensure that the jumper wire is in disconnected status
 - 5) Examples about DS18B20 were included in the CD-ROM for reference

4.8 SPI Communication Module

This module mainly shows the SPI communication protocol through access to the external device EEPROM 93LCXXX, shown as Figure 4-12.

**Figure 4-10** 93LCXXX EEPROM

The module consists of the following components:

- 1) 93LCXXX EEPROM.
- 2) Coding switch.
- 3) ROM WORD SIZE selection jumper wire.

Descriptions on this module as below:

- 1) 93LCXXX EEPROM data are included in the CD-ROM.

- 2) The SPI communication port of 93LCXXX: SDI, SDO and SCL are connected to the RC5, RC4 and RC3 of the communication port of MCU through the coding switch, and the chip-select signal connects to the RC2 pin of the MCU, and therefore, it can be controlled by hardware.
- 3) When this module is in use, we must ensure that the coding switch is in the connected state, when not in use, we must ensure that it is in disconnected status so as not to affect the normal operation of other modules.
- 4) When using 93LCXXA (ROM SIZE is 8 bits), or 93LCXXB (ROM SIZE is 16 bits), the WORD SIZE jumper wire is non-functional. When using 93CXXC, jumper wire decides to choose WORD SIZE for 8 or 16 bits.
- 5) 93LC46B EEPROM related program examples were included in the CD-ROM for reference.

The schematic of this module is shown in Figure 4 -13.

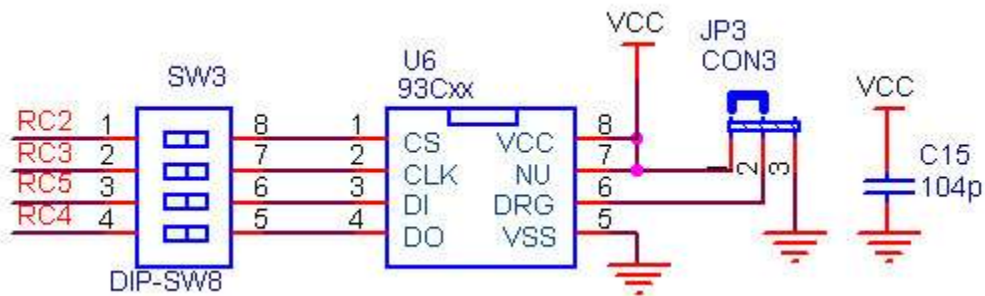


Figure 4-13 schematic of SPI Communication

4.9 I2C Communication Module

This module mainly shows the I2C protocol through access to the external device EEPROM 24CXX, as shown in Figure 4-14 and schematic shown in Figure 4-15

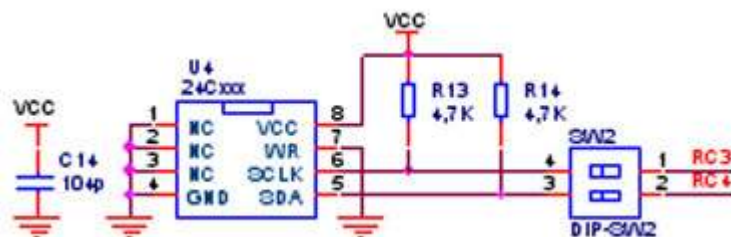
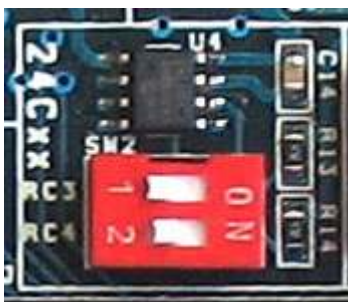


Figure 4-14 24CXX EEPROM

Figure 4-15 I2C communication schematic

The module has the following main components:

- 1) 24CXX EEPROM
- 2) Double bit coding switch

Descriptions on this module are as below:

- 1) Information about 24CXX EEPROM was included in the CD-ROM.

- 2) The I2C communication port SDA of 24XX is connected to the I2C communication port RC4 and RC3 of MCU CK to allocate yards and switch connected to the MCU so it can be controlled by hardware.
- 3) When this module is in use, we must ensure that the coding switch is in the connected state, when not in use, we must ensure that it is in disconnected status so as not to affect the normal operation of other modules.
- 4) Programs about read and write of 24C01B EEPROM is included in the CD-ROM for reference

4.10 LCD12864 and 1602 LCD Modules

This module major includes LCD12864 socket and 1602 LCD socket, as shown in Figure 4-16.



Figure 4-17 LCD12864 and 1602 LCD sockets

Descriptions of This module are as following:

- 1) LCD 12864 LCD and 1602 are using PORTA as the control bit, PORTD as data bit. Only the MCU which has
 - 1) PORTD can execute LCD display.
- 2) All the pins of this module are connected to the MCU directly, and there's no coding switch to control it. When the LCD is not in use, it is suggested that the LCD is taken off from the socket, while if it is in use, we should shut off other modules, or else the LCD will not display.
- 3) The development board can adjust the brightness of 1602 LCD or the contrast of LCD12864.
- 4) All LCD used in this development board are products type "WG12864" (with chip select CS1 and CS2). If you use other products, please confirm whether or not it is compatible.
- 5) Datasheet 1602 LCD and 12864 LCD are included in the CD-ROM.
- 6) Application examples about 1602 LCD and 12864 LCD are included in the CD-ROM.
- 7) Adjust VRs P2 and P3 to contrast of LCDs most clearly

Schematics of these two modules are shown in Figure 4-18.

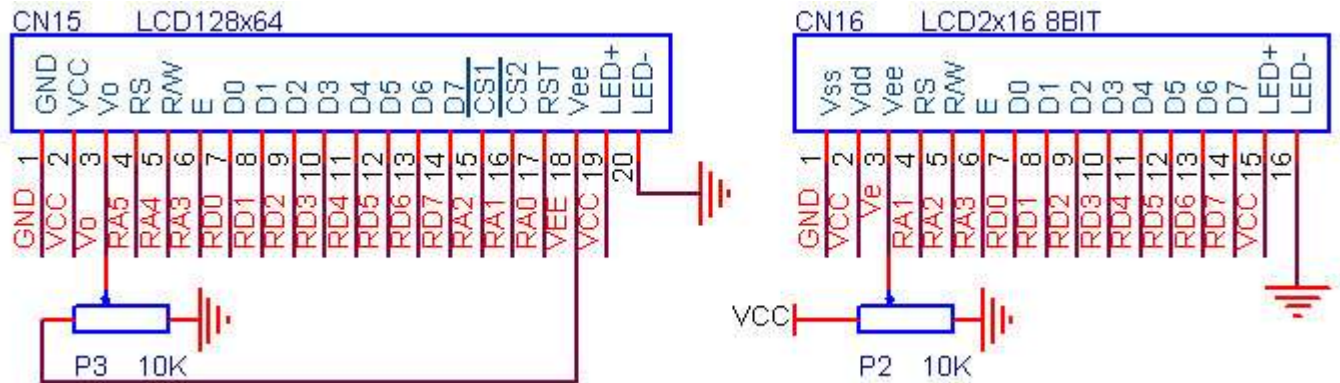


Figure 4-18 schematics of 12864LCD and LCD1602

4.11 A/D Converter Module

This module main executes the conversion from analog signal to digital signal.

Hardware and Schematic of this module is shown in Figure 4-19 and 4-20



Figure 4-19 A/D Converter Module

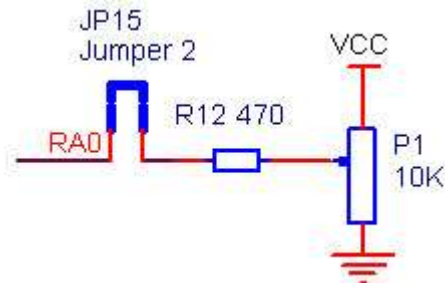


Figure 4-20 A/D converter schematic

This module mainly consists of the following components:

- 1) One 10K adjustable potentiometer.
- 2) Jumper wire JP15.

Descriptions of this module are as the following:

- 1) The potentiometer is connected to the RA0 port through jumper wire JP15.
- 2) When the A/D converter is in use, please make sure that the jumper wire is on while if it is not in use please sure it should be cut off so as not to affect other modules.
- 3) CD-ROM provided along with the product includes examples of A/D converter for reference.

4.12 Remote Control Receiver & Decoder Module

This module main executes the receiving and decoding regarding infrared remote-control, as shown in Figure 4-21, and schematic shown in Figure 4-22



Figure 4-21 Remote decoding module

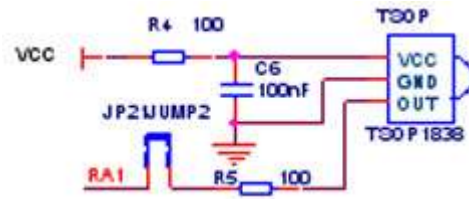


Figure 4-22 Remote decoder schematic

This module consists of the following major components:

- 1) Infrared remote control receiver
- 2) Interface Socket

Descriptions on this module are as the following:

- 1) Information about infrared remote encoding chips 6121 is included in the CD-ROM.
- 2) The output of the remote control receiver is connected to the RA1 port of MCU through a jumper wire.
- 3) When this module is in use, please make sure that the jumper wire is on while if it is not in use please making sure the jumper wire is disconnected.
- 4) The CD-ROM contains examples about remote decoding for reference.

4.13 Six-digital Display Module

This module main introduces the use of multi-digital LED display, as shown in Figure 4-23



Figure 4-23 digital LED control module

The module consists of the following major components:

- 1) 6 digital LED.
- 2) Bit control and segment control coding switch.
- 3) Driving circuit.
- 4) Interface socket.

Descriptions of this module are as the following:

- 1) The segment control of the digital LED is connected to the PORTD of MCU through coding switch.
- 2) The bit control of the digital LED is connected to the PORTA of MCU through coding switch.
- 3) When this module is in use, we must ensure that the coding switch is in the connected state, when not in use, we must ensure that it is in disconnected status so as not to affect the normal operation of other modules.
- 4) You may also achieve the display of digital LED by use of interface socket. (And at this point you must ensure that the coding switch is disconnected.)
- 5) The six digital LED are using a common anode.
- 6) The CD-ROM contains examples for reference. (One static display and one dynamic display examples are include respectively).

The schematic of this module is shown in Figure 4-24.

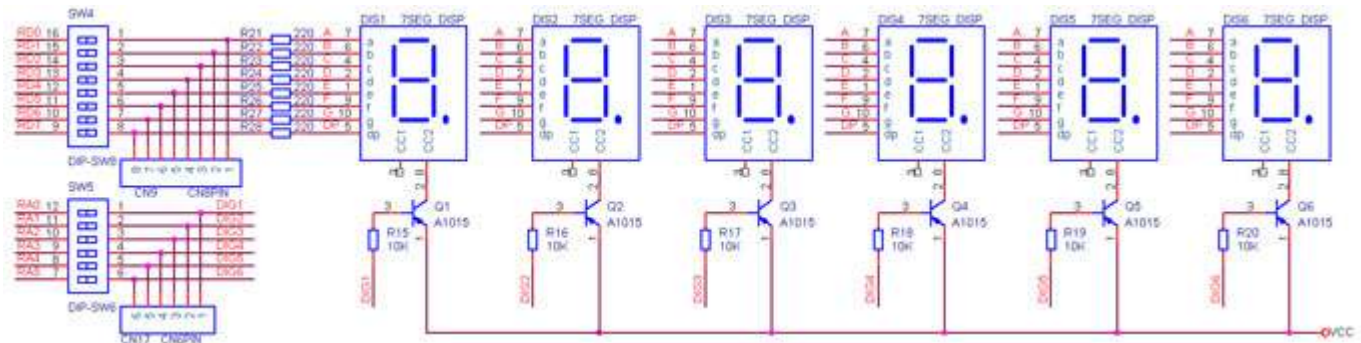


Figure 4-24 Digital LED schematic

4.14 Eight play-in-turn Light Module

This module is shown in Figure 4-25. It consists of the following main components:

- 1) 8 LED.
- 2) 8 coding switches.
- 3) Interface socket.



Figure 4-25 8 Play-in-turn Light Module

Descriptions of This module are as the following:

- 1) 8 LED are on when the I/O outputs HIGH voltage.

- 2) Each LED is controlled by a separate coding switch. When this module is not in use it is suggested that the corresponding bit be disconnected so as to avoid causing unnecessary disruption.
- 3) You may test the I/O ports of other MCU by using the interface socket.
- 4) The CD-ROM includes some simple examples of this module, which cover “lit each LED” a “Simple play-in-turn”, just for users’ reference.

The schematic of this module is shown in Figure 4-26.

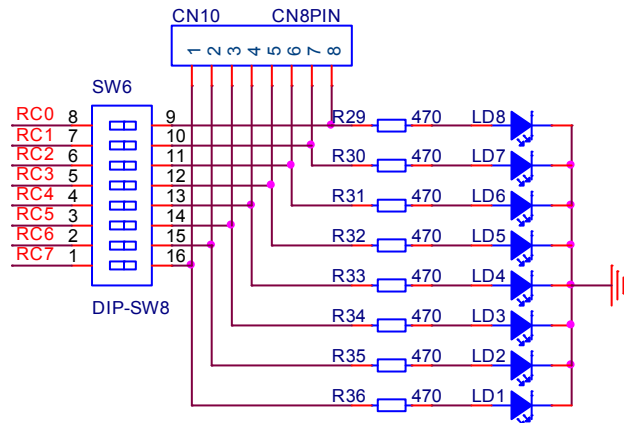


Figure 4-26 Schematic of play-in-turn

4.15 DS1307 RTC Module

This module mainly experiments the use of real-time clock chip DS1307 use, as shown in Figure 4-27



Figure 4-27 DS1307 clock module

This module consists of the following major components:

- 1) Clock chip DS1307
- 2) 3V button battery

- 3) Coding switch
- 4) 32.768K crystal oscillator
- 5) Interface socket

Descriptions of this module are as follows:

- 1) Datasheet of the chip DS1307 is included in the CD-ROM.
- 2) The communication ports of SCL, SDA are connected to the MCU at RC3 and RC4 through coding switch
- 3) When this module is in use, please make sure that the coding switches are on, and while it is not in use please make sure that the coding switches are off to avoid affecting the normal operation of other modules.
- 4) By using the interface socket, it is also possible to achieve the communication between DS1307 and other MCU chips. (Please make sure that the coding switches are off.)
- 5) Examples about access to DS1307 are included in the CD-ROM for reference.

The schematic of this module is shown in Figure 4-28.

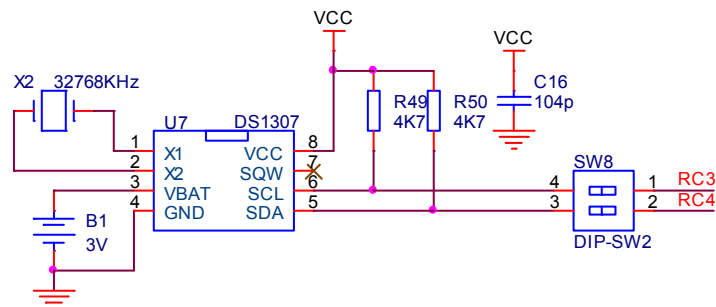


Figure 4-28 Schematic of DS1307 clock module

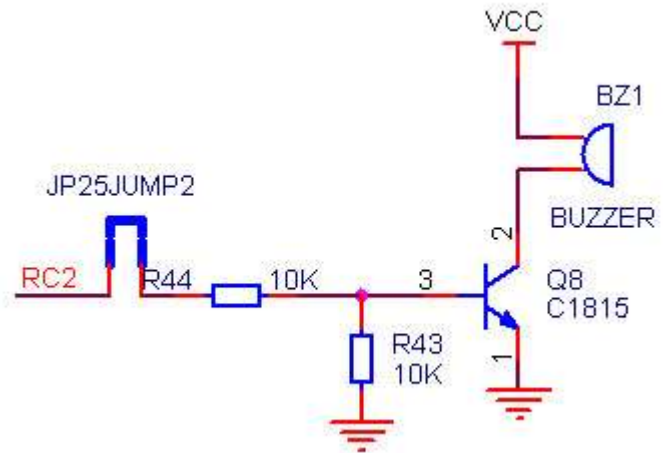
4.16 Beeper Module

This module mainly shows how to use MCU to control the beeper, as shown in Figure 4-29.

The schematic of this module is shown in Figure 4-29.

The main module of the following components:

- 1) Beeper
- 2) Jumper wire JP25

**Figure 4-29** Beeper module**Figure 4-30** Schematic of beeper

Descriptions about This module are as the following:

- 1) The beeper is connected to the RC2 pin of MCU through a jumper wire.
- 1) When this module is in use please make sure the jumper wire is connected while when it is not in use please make sure that the jumper wire is disconnected.
- 2) When the jumper wire JP25 is on connected status, we may hear a weak beeping even if there's no chip on the board, which is a normal phenomenon.
- 3) The CD-ROM includes programs of beeping for reference.

4.17 Independent Key and External Interrupt Module

This module is independent key and external interrupt module, as shown in Figure 4-31.

**Figure 4-31** Independent Key and External Interrupt Module

This module consists of the following main components:

- 1) Four independent buttons.
- 2) One four-bit coding switch.
- 3) Interface socket.

Descriptions of this module are as the following:

- 1) All keys are at LOW voltage when they are pressed down, and are at high-impedance when they are released (when they are in use, we should open the internal up-pull resistance of the MCU, so that when the keys are released they are at HIGH voltage.)
- 2) K1 not only can be used as ordinary key, but it can also be used to trigger external interrupt.
- 3) All the keys are connected to the Port B of the MCU, when they are in use please make sure that the coding switches are connected, and when they are not in use please make sure that the switches are not connected.
- 4) When using Port B as keys, it is suggested that the weak up-pull function of Port B be enabled, while other ports are used it is also suggested that up-pull resistors are added to the board.
- 5) You may also use the keys of other MCU by taking advantage of the interface socket.
- 6) The CD-ROM contains examples about the use of this module.

The schematic of this module is shown in Figure 4-32.

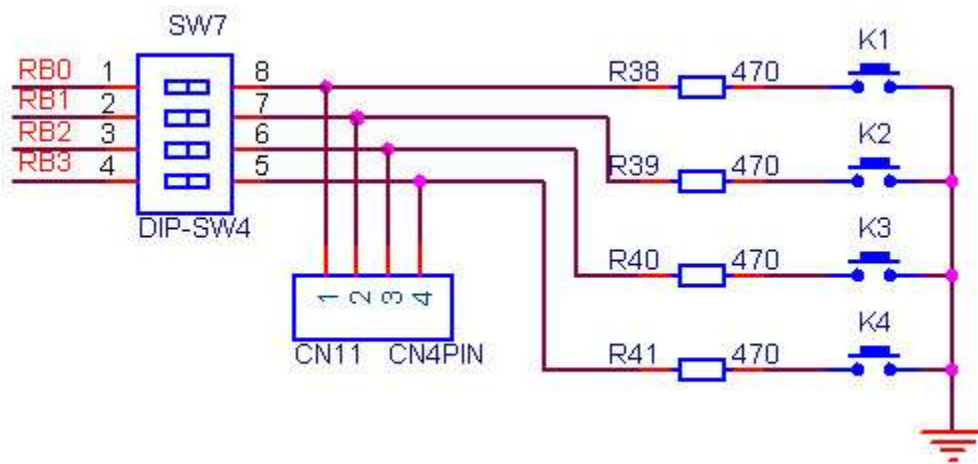


Figure 4-31 Keys and external interrupt module

Chapter 5 Practice of PICLAB-V2 Development Board

Actual practice purposes:

This chapter takes the "4 * 4 matrix keyboard module" and the "six digital control module" as the basis, and have systemic introduction about the use of PICLAB-V2 development board, including the use of MPLAB edit, compiling source and ICD2 debugging results.

Hardware layout:

- 1) 4x4 matrix keyboard controlling the coding switches SW1 which are all on.
- 2) The 1st and 2nd position of 6-bit digital LED coding switches are both on, while other bits are off (we use only the 5th and 6th digital LED to display.)
- 3) All the coding switches that control the six LED are ON.

Software planning:

- 1) In this software we ignore the jitter of keys, when the software detects LOW voltage, it would consider that there's a key press.
- 2) In this software, we do not consider the situations when several keys are pressed down simultaneously. When there are several keys being pressed, we consider that the key which has the smallest number is pressed, for example, when T1 and T2 are pressed at the same time, we only consider that K1 is pressed.
- 3) When there is no button being pressed, the two digital LED display as "FF", when a key is pressed, the LED will show the number of key that is pressed, for example, when T1 is pressed, the LED's display as "01", while T16 is pressed, LED's display as "16."

Prepare and compile the source codes:

Double-click MPLAB icon on the Desktop to run MPLAB programming environment.

- 1) In accordance with the method introduced in 2.2.1, edit a new source code, and save it as "KEY4x4.C".
- 2) In accordance with the method introduced in 2.2.1, establish a new engineering project, and complete the settings as below: select PIC16F877A in the second step as the target chip; select "HI-TECH PICC Tool suite" as the compiling tool in the third step; select "KEY4x4" as the new project name (the suffix can be omitted), and the directory for saving the project is the same as that of the source. After editing the source and establishing the project, MPLAB interface will be shown as Figure 5-1.

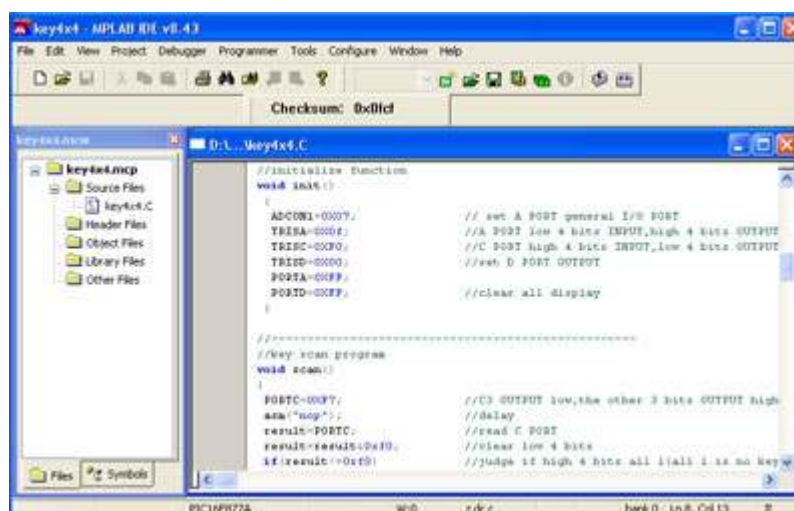



Figure 5-1 MPLAB interface after editing source code and establishing the project

- 3) Implement the menu command Project → Build ALL or the icon  in the tool bar to compile the source code. The compiled results will be shown as Figure 5-2. We can see from the figure as "BUILD SUCCEEDED", which indicates that the compiling is successful and it has created a KEY4x4.HEX file under the directory of the project (only compiling can it create such a file).

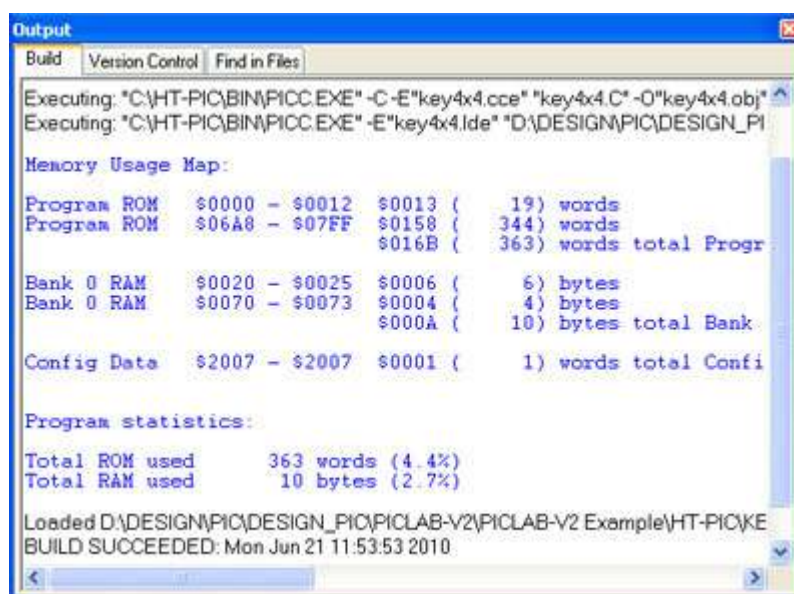


Figure 5-2 Source compiling results

It can be seen from the figure that there are several messages, suggesting that the registers used are not in the correct BANK, please carefully check the program to ensure that all the registers are in the right BANK (even if entirely correct, the information will still emerge, but it does not affect the implementation of results).

If there are ERROR[num] or WRNING[num] in the compiling results (num means the error or warning number), you may locate the cursor to the position that cause such errors or warnings by double clicking the message immerged, and correct the source code and re-compile it repeatedly until it shows the result as in Figure 5-2.

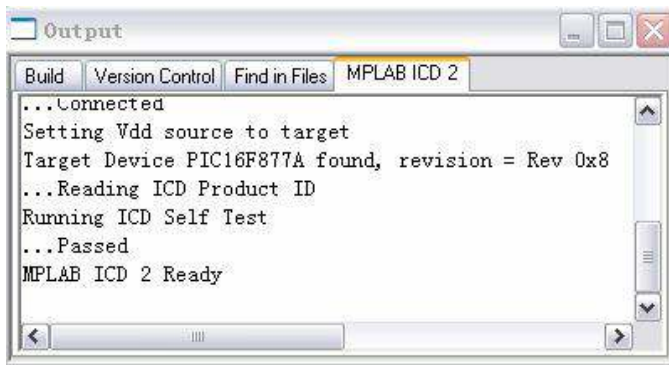
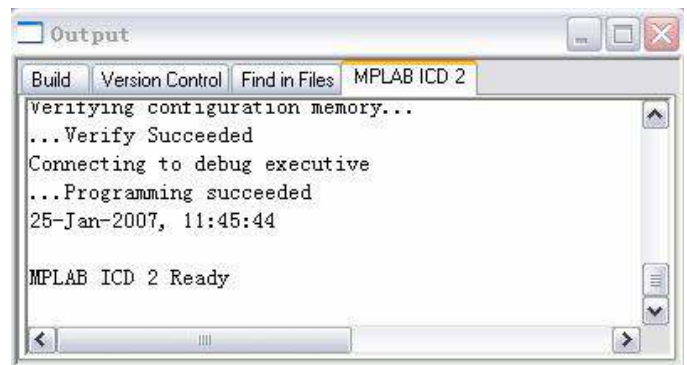
Using MCD2 to debug the source

- 1) In accordance with the method shown in Figure 3-2, connect the ICD2 to PC and PICLAB-V2 development board.
- 2) Implement the menu command Debugger → Select Tool to select MPLAB ICD 2 (same asMCD2) as a debugging tool.
- 3) Implement the menu command Debugger → Settings to select the USB as the communication port, and the board will use its own power supply (i.e. disable "ICD2 power from the target board". It is shown in Figure 5-3.

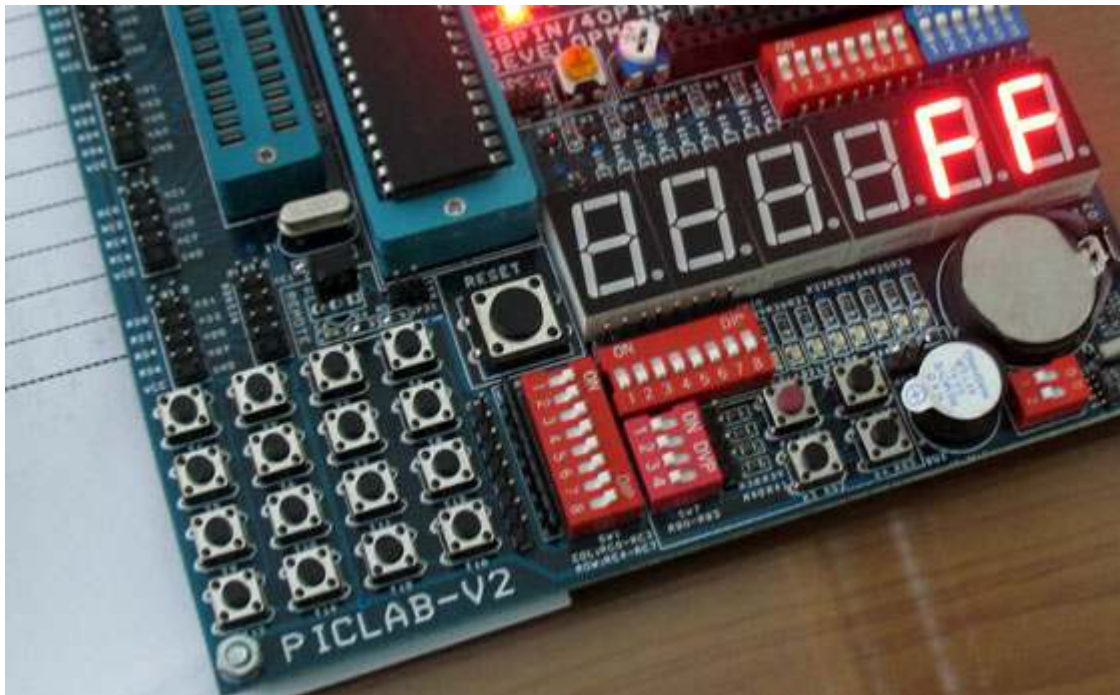


Figure 5-3 Set up communication ports and power supply for target board

- 4) Implement menu command Debugger → Connect or icon in the tool bar to connect ICD2 and PICLAB-V2 development board, and when it is successfully connected it will shown as in Figure 5-4 with a message.
- 5) Implement menu command Debugger → Program or the icon in the tool bar to program/burn the objective codes to the target MCU chip (Note: now the MCU cannot run on offline mode but has to be run under debugging mode), after programming it will show a message as in Figure 5-5.
- 6) Implement menu command Debugger → Run or toolbar icon, we can see the display of "FF" in the PICLAB-V2 development board, shown as Figure 5-6; Now if any key is pressed the LED will display a the number of that key, for example if T15 is pressed, the LED will display "15", shown as Figure 5-7.

**Figure 5-4** Connecting information**Figure 5-5** Programming Information

- 7) After several debug, we have basically achieved the desired results, and the debug completed. Disconnect the connections.

**Figure 5-6** Result when no key is pressed

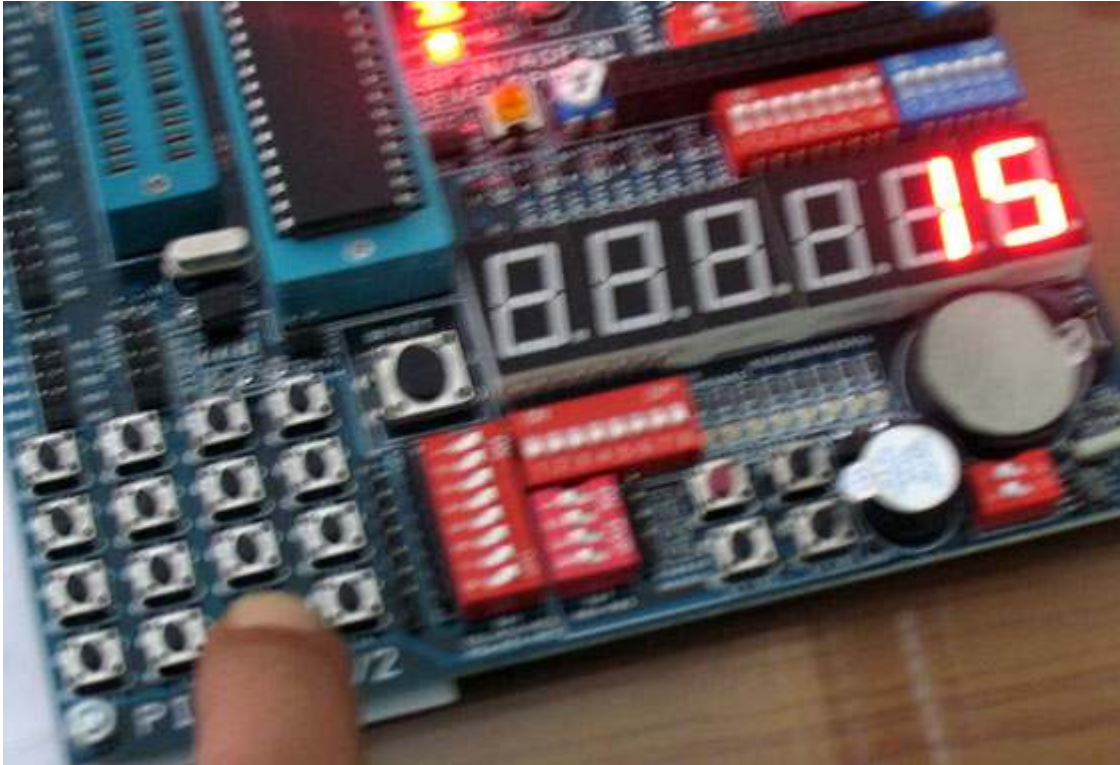


Figure 5-7 Result when T15 is pressed

Programming the target MCU using MCD2

The source codes prepared as described above has passed the debug by using MCD2, and we have achieved our desired goal, now we may use ICD2 to burn/program the KEY4x4.HEX file into the target MCU chip in order that the MCU can run offline.

- 1) Connect the ICD2 and the PICLAB-V2 development board in accordance with the method shown in Figure 3-2.
- 2) Implement the menu command Programmer → Select Too to select MPLAB ICD2 (or PICKit 2, PICKit3...) as a debugging tool.
- 3) Set up the configured bits and other parameters according to the specific conditions.
- 4) Implement the menu command Programmer → Program, to program/burn the KEY4x4.HEX file into the target MCU chip.
- 5) Disconnect PICLAB-V2 and ICD2 to see the results of MCU execution.

Appendix 1: Packing List and Contact

When you receive the products, please check the box to see if all accessories are complete. This product should include the following components:

- PICLAB-V2 development board, 1 pc;
- PIC16F877AMCU, 1 pc;
- Serial cable, 1 pc;
- Manual and CD-ROM, 1 copy respectively;

Thien Minh Electronic Solutions Co., Ltd (TMe)

Web: www.tme.com.vn – www.tme.vn

Email: tminh@tme.com.vn

Address: 173 Tan Phuoc Street, Ward 6, District 10, Ho Chi Minh City, Vietnam

Tel: +84.8.39573224