Assembly Programming and Interfacing

(Lecture notes for Experiments)

Author: Yin LU

Affiliation: The Computer Science School of NWPU

Date: 2021-4-3

Catalogue

目录

[Chapter 1 Knowing the Applications 3](#_Toc35275)

[1. Emu8086, the Intel 8086 emulator 3](#_Toc35276)

[2. VMWare and Windows XP virtual machine 8](#_Toc35277)

[3. EV-Capture(EV-录屏) 18](#_Toc35278)

[4. Prepare your experiment report 20](#_Toc35279)

[Chapter 2 Experiment 1 Brunch and Loop structure 22](#_Toc35280)

[1. Assignments 22](#_Toc35281)

[2. Experiment Preparation 23](#_Toc35282)

[3. Experiment Circuit Scheme 23](#_Toc35283)

[4. Experiment Process 23](#_Toc35284)

[Chapter 3 Experiment 2 Simple IO and Lantern Control 25](#_Toc35285)

[1. Assignments 25](#_Toc35286)

[2. Experiment Preparation 28](#_Toc35287)

[3. Experiment Circuit Scheme 28](#_Toc35288)

[4. Experiment Process 28](#_Toc35289)

Chapter 4 Experiment 3 Display devices Interfacing................................................................31 Chapter 5 Experiment 4............................................................................错误！未定义书签。

# Chapter 1 Knowing the Applications

## 1. Emu8086, the Intel 8086 emulator

(what is)

Emu8086 is Intel x86 processor emulator designed for x86 assembly language programming and debugging. It integrates the source code editor, the x86 masm assembler, and the the x86 debugger in a GUI application, also a x86 virtual processor on the back. With the help of the pure software virtual processor, emu8086 can run a program designed for the Intel 8086/8088 processor. And moreover, if the executable program is generated by its own assembler, emu8086 will support single step forward debugging.

The main window of Emu8086 is its source code editor by default (see figure 1.1). When you

press the [emulate] button on the tool bar, the emulation window together with the source code view window appear (figure 1.2). The x86 emulator provides machine code view,

register view, and the buttons on the bottom toolbar will open other addition view windows including command line screen, memory view, variable view, stack view and flag register view. All these view windows will help the programmer to debug their assembly language program.

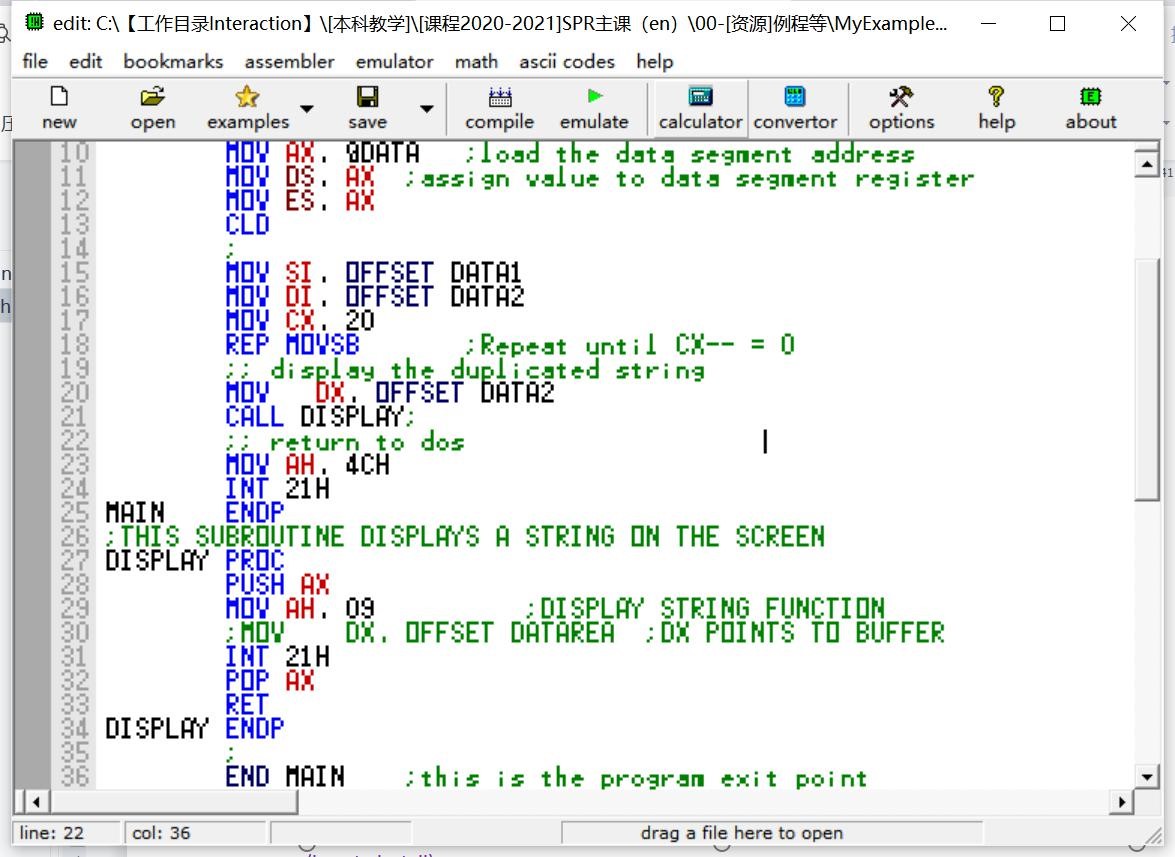


Figure1.1 The main window of Emu8086

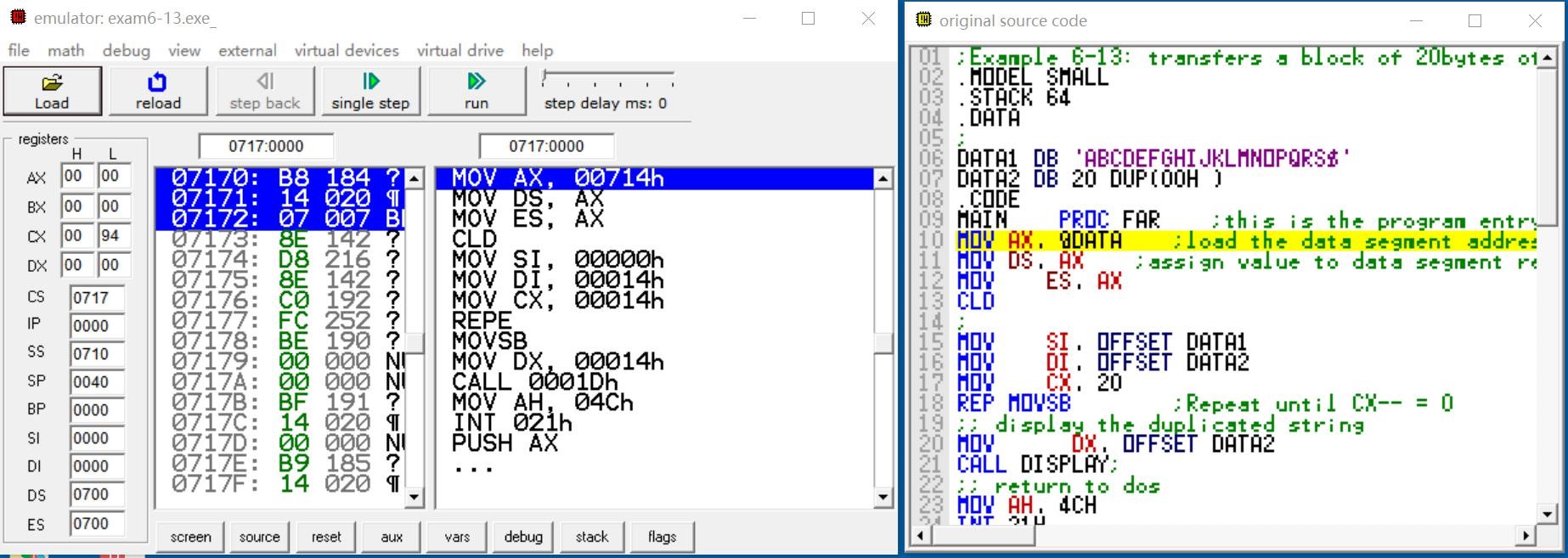


Figure1.2 The emulation and source code view windows

### 1.1 How to install the Emu8086 program

Most Emu8086 installation program are provided as a self-extraction and auto-start package

file, like this . Double click the package file, and start install process.

Click the [next] button to take the default settings, and the Emu8086 program will

be installed into your C: disk like this .

After the installation, a short cut group will be created in your starting menu (see Figure1.3).

You can click the [emu8086] menu to start the program.

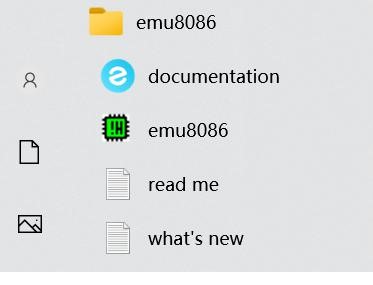


Figure1.3 emu8086 short cut group

### 1.2 How to program with Emu8086

Start the emu8086 program, then click the [new] botton to start a project (figure 1.4). Choose [BOOT template \ empty workspace] in the next popup dialog, then press [ok]  button to open a new workspace for a boot-able program. The source code editor window appears, and you can type in your program source code (figure 1.5 shows a program to display a down-counting number with the a virtual LED displayer).



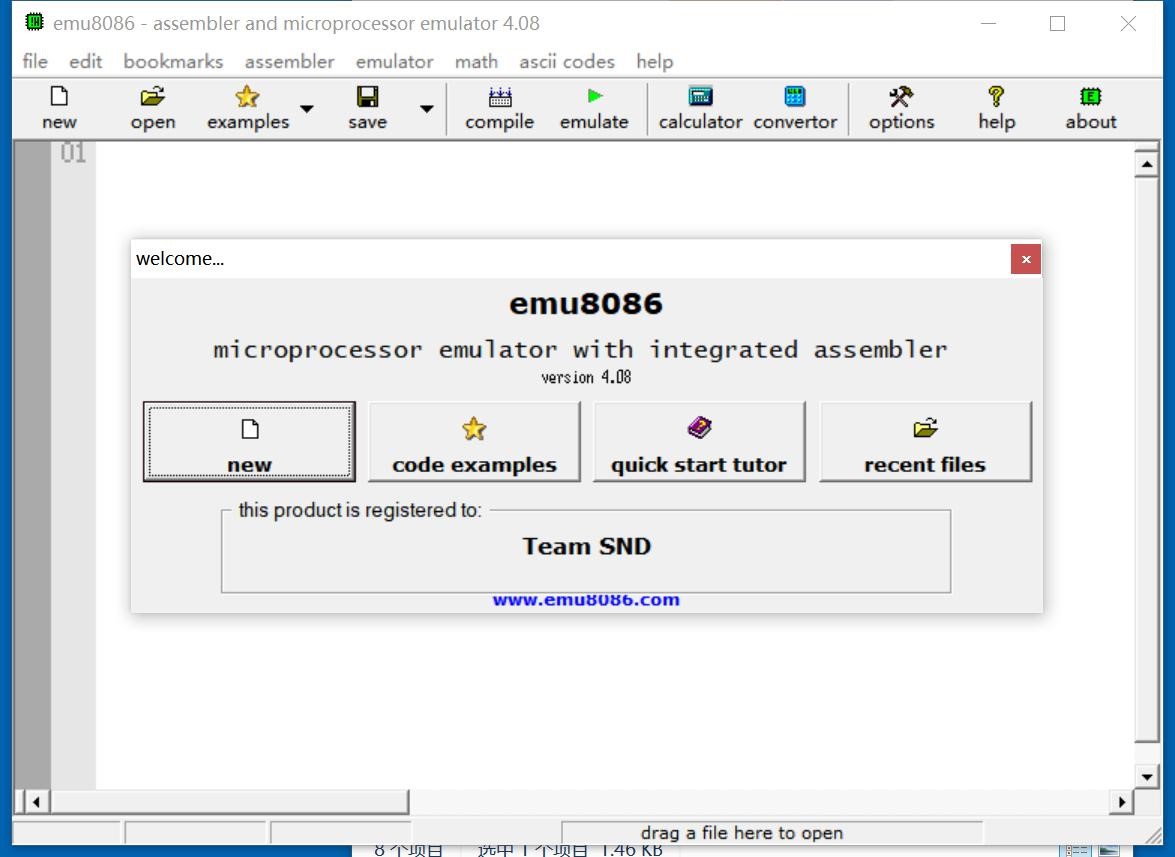


Figure 1.4 The emu8086 starting window

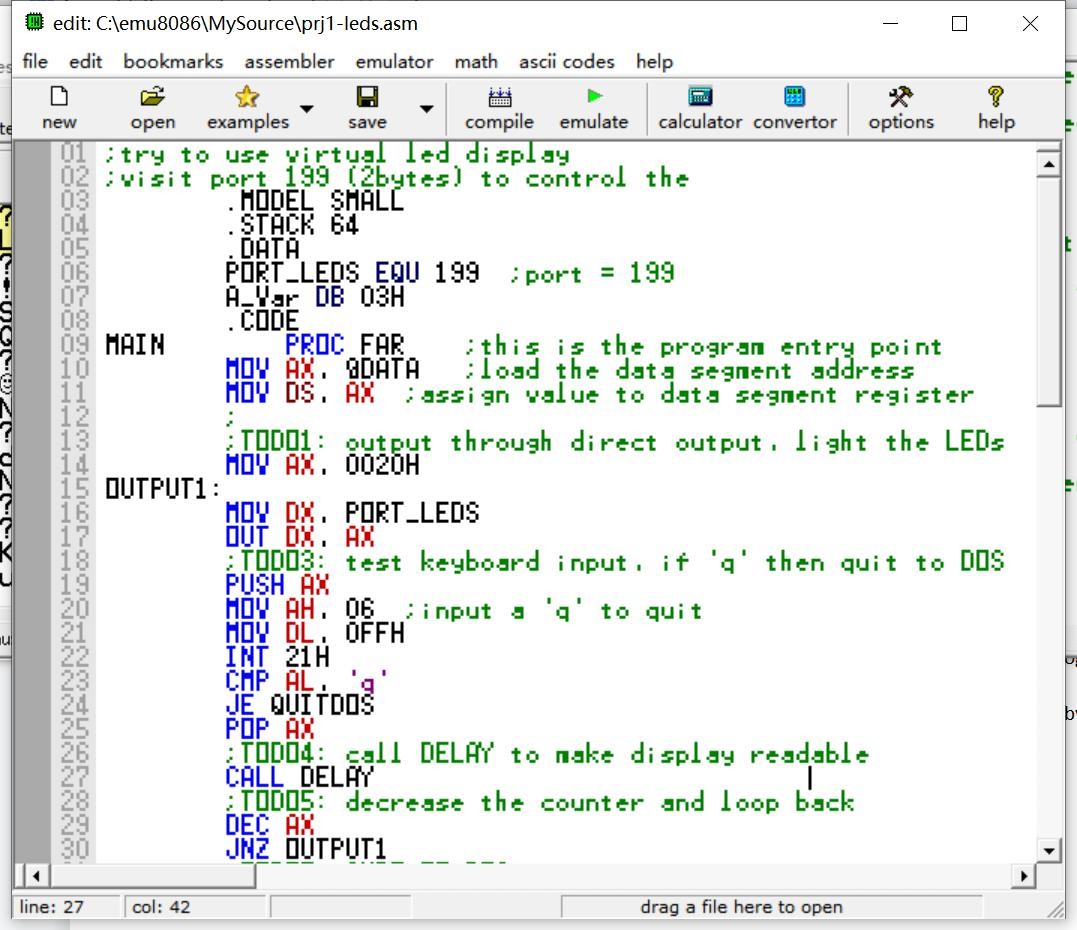


Figure1.5 The source code editor window

Remember to press the [save] button to save your source code in the hard disk,

otherwise you will lose it when quit the program.

### 1.3 How to debug with Emu8086

Click the [emulate]  button on the toolbar, to start the debug windows, including the emulation and source code view window. Then debug window below (figure 1.6 )

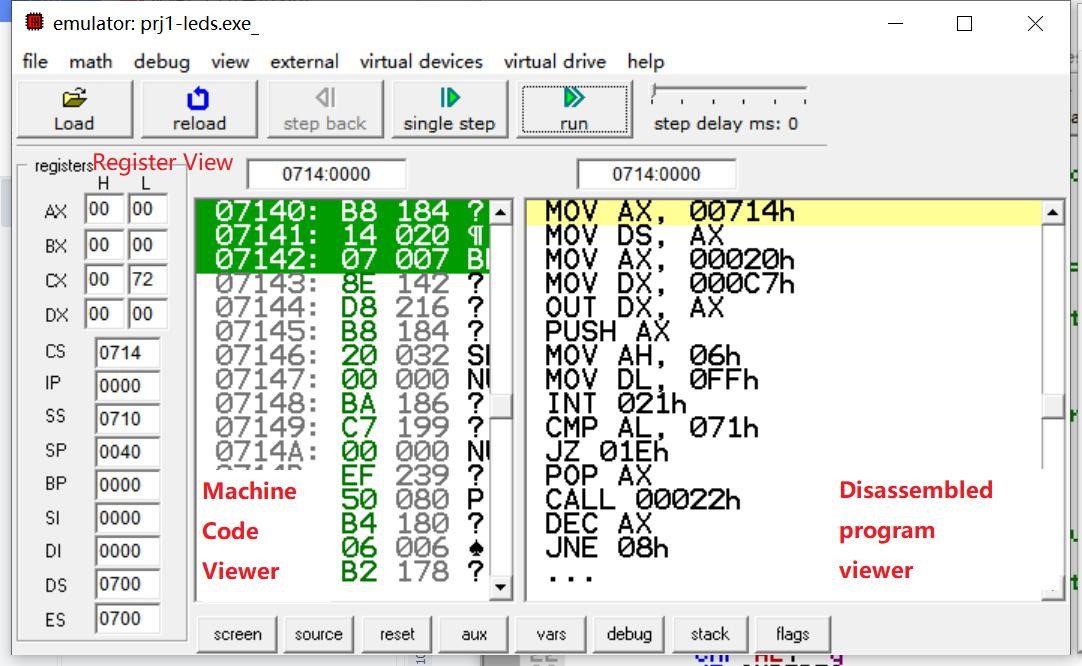
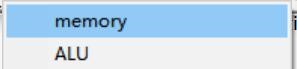


Figure1.6 The debug window

Use the [single step] button to run your program single direction forward. Each time the debugger halts, you can check the effect your program do to registers in the register view.

If you want to check the variable, press the [aux]  button, and click [memory]

 menu item in the popup menu. A memory view dialog will appear (figure 1.7). Beware to change the display beginning address to the start of your

program  , which is in the first line of your program

 in the disassembled program view. And in this example, it is 00714H

(0714:0000 logical address).

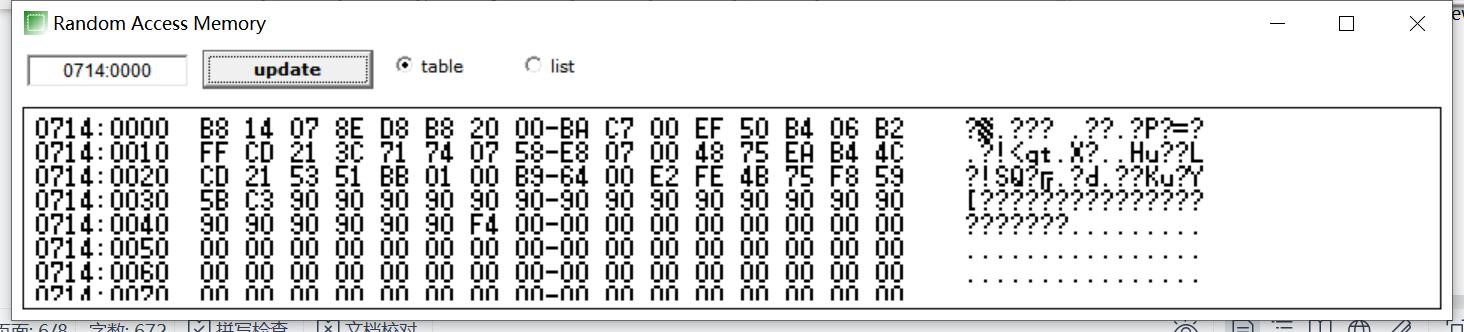


Figure1.7 the memory view dialog

An other way to check the output of your program, is to open the variable view dialog

(figure 1.8 (A)) by touching the [vars]  button. And the flag register can be verified by

open

the

flag

register

view(figure

1.8

(

B

))

by

pressing

the

[

flags

]



button.

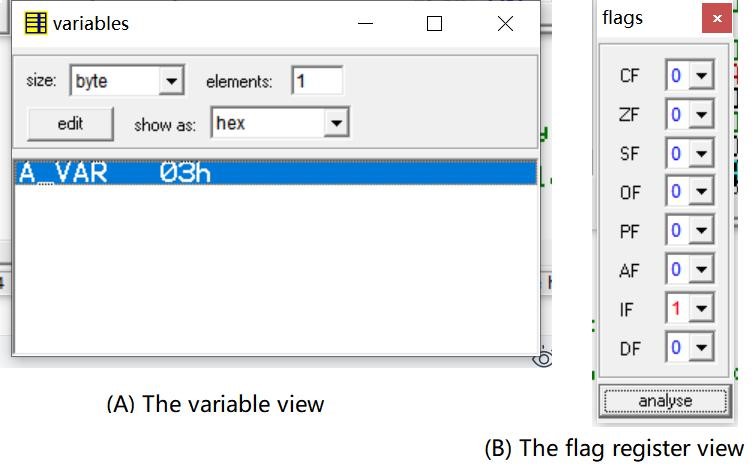
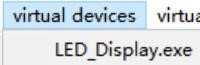


Figure1.8 The variable view and the flag register view

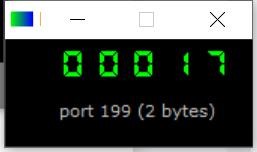
### 1.4 Peripheral devices provides with Emu8086

The emu8086 also provides some virtual devices that can be manipulated by output command words to interfacing ports. The example in figure 1.5 will use the LED displayer device.

You can open the device by using the [virtual devices] menu in the debug window.



The menu item [LED-display.exe] will open the LED displayer window below.

.

Examples to show how to use these virtual devices can be found in example sub-directory in

the emu8086 installed directory. Or you can open an example by using the [file\examples] menu group, as shown in figure 1.9.

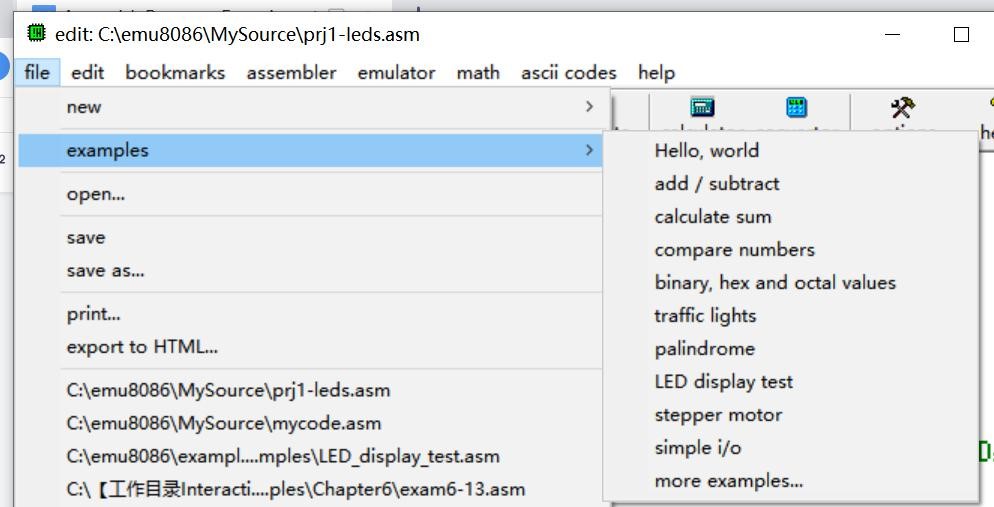


Figure1.9 Examples provided by Emu8086

## 2. VMWare and Windows XP virtual machine

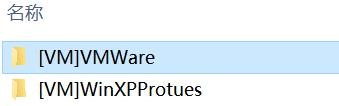
VMWare (Virtual Machine ware) is a virtual machine application software. It can emulate a virtual operation system different from your current one in your desktop computer. For example, a Windows 10 system in a Linux system, or a Windows XP over a Windows 10 system. We use VMWare to create a virtual Windows XP computer to install and run the Emu8086 and Protues software. Although both of them runs on Windows 10 platform, we insist that you to install VMWare and use the virtual machine. For the reason that we will release a all-in-one virtual machine image with all the necessary software installed. Once you mount the image in your VMWare application, your ready to write and debug your experiment program.

### 2.1 How to install the VMWare application

The installer of VMWare is provided as a self extraction package like this

.

Create a sub directory called “[VM]VMWare” in the root directory of one of you hard-disk

partition, for example “D:\[VM]VMware” . Then double click the .exe file

to open it and start to install.

The Installation guide window appears, see figure1.10. Click [next]  button to continue. In the next window, check the  to accept the licence agreement. Then, .

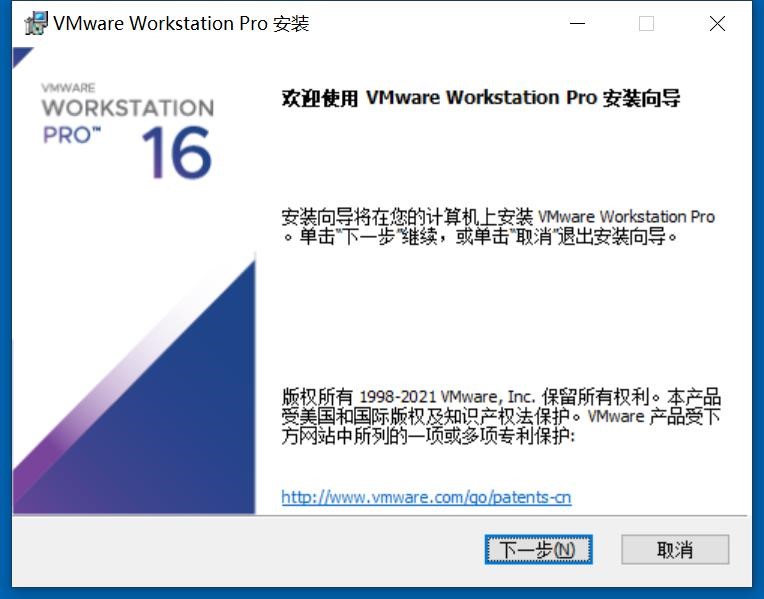


Figure1.10 VMWare installer guide window

In the third window, click  to choose your installation directory. We suggest that you choose the subdirectory you created in section 2.1, for example, see figure 1.11. And

then, click [next] .

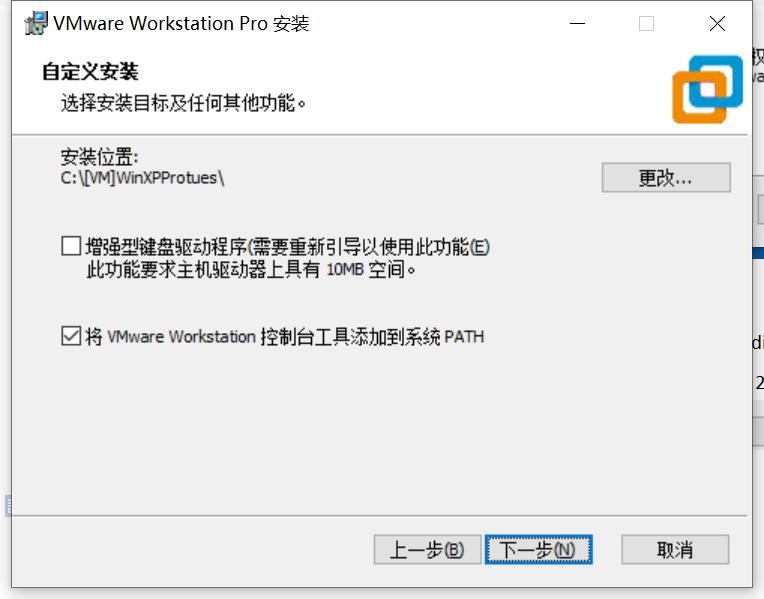


Figure1.11 Change installation directory

In the fourth and fifth window, click  . And in the sixth window, click [install]

 to start installation process (see figure1.12)

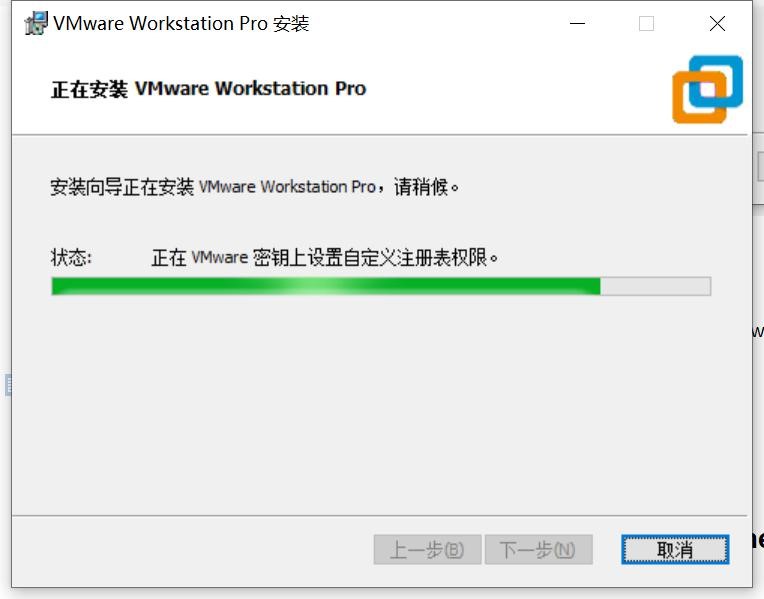


Figure1.12 VMWare installing

When the file copying process finished, the eighth window appears to hint you to provide the authorization key (figure 1.13). Click [Authorization key]  button to open the

input window (figure1.14). Input a authorization key, and press the [continue] 

button.

In

the

next

window,

click

[

over

]



button.

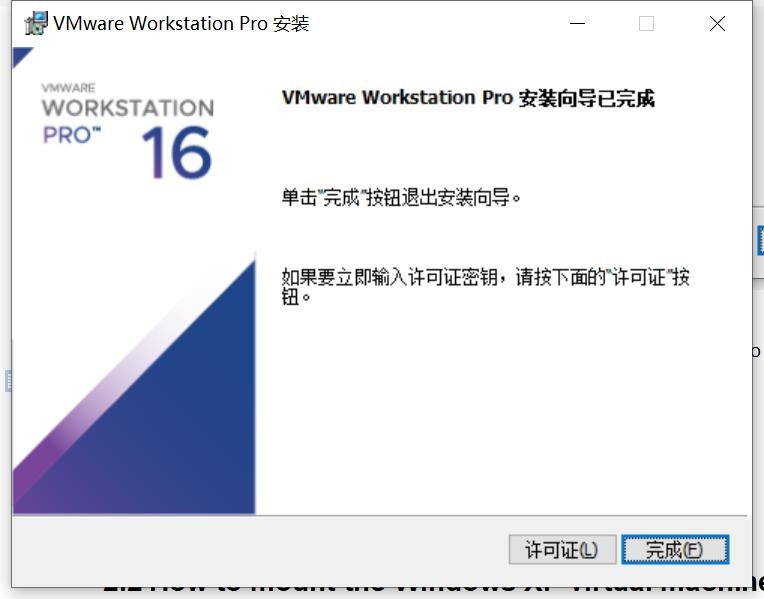


Figure1.13 Installation Hint window



Figure1.14 input authorization key

Now, the VMWare application has already installed. Follow the steps in section 2.2 to mount the all-in-one vm image.

### 2.2 How to mount the Windows XP virtual machine image

Start the VMWare application from your starting menu. The vmware home page appears,

see figure 1.15. Click the [open a virtual machine] button to mount the vm image.

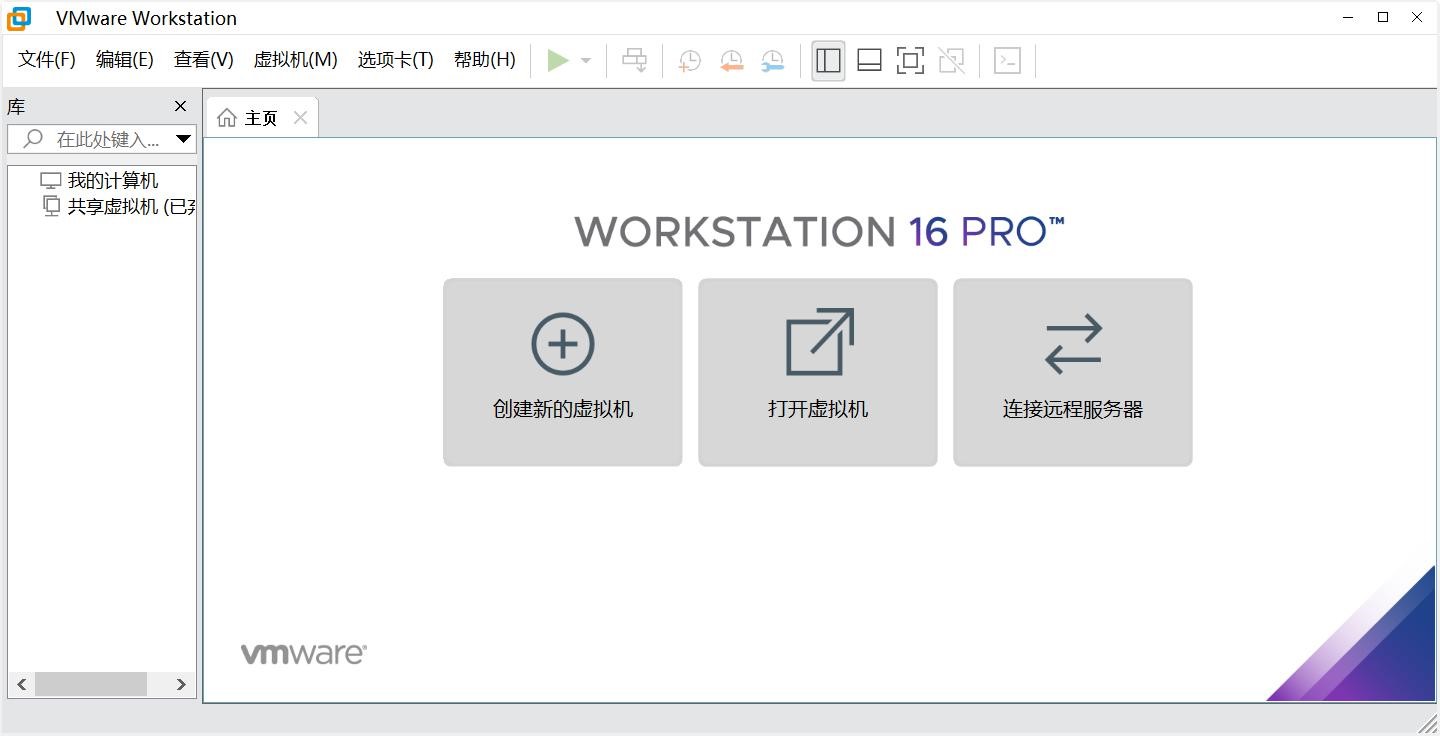


Figure1.15 VMWare home page

Suppose that you have unpacked the image we provide you in a directory called “C:\[VM]WinXPProteus”, and in the windows open file dialog, open this directory, and choose the .vmx file as shown in figure 1.16.

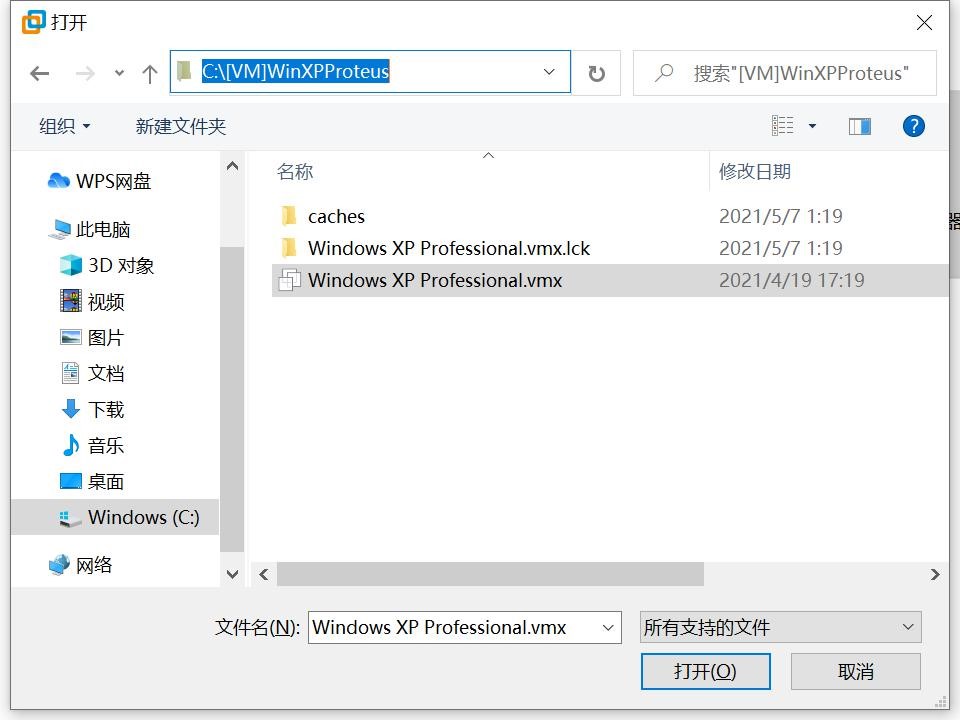


Figure1.16 Choose the vm image file

If everything is right, the virtual machine image will be mounted, figure 1.17. Click [start this virtual machine] menu item to boot up this virtual machine.



Figure1.17 WinXP vm image mounted

Maybe your vmware application will notice you that this vm has been moved or duplicated, just click [I have already duplicated the vm image]  button as directed. The

Windows XP virtual machine will boot up. And the Windows XP desktop appears (see figure

1.19).

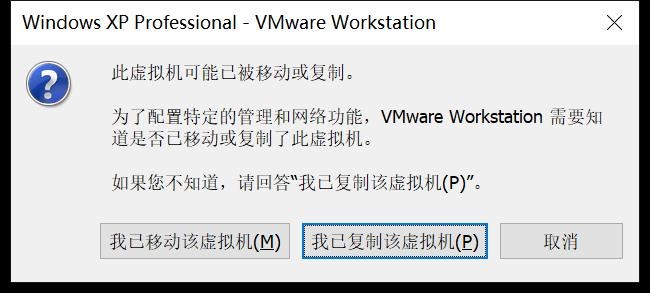


Figure1.18 Notification dialog



Figure1.19 Window XP virtual machine

### 2.3 How to import and export files between vm and its host

The virtual machine and the host can not access files between each other. Files have to be import to the vm or export to the host before open. In this section, we will setup a network file sharing directory to do import and export between the vm and the host.

1. Start vmware and boot up the WinXPProteus virtual machine, by choosing [boot up this virtual machine] menu item in the vm manager window;



Figure1.20 boot up the virtual machine

1. Then, in the VMWare application menu choose [virtual machine\setup...] as shown in figure 1.21;



Figure1.21 setup the virtual machine

1. In the vm setup window popped out, choose page [options], then [Shared Folder], and [always enable] in the right side window. See figure 1.22.

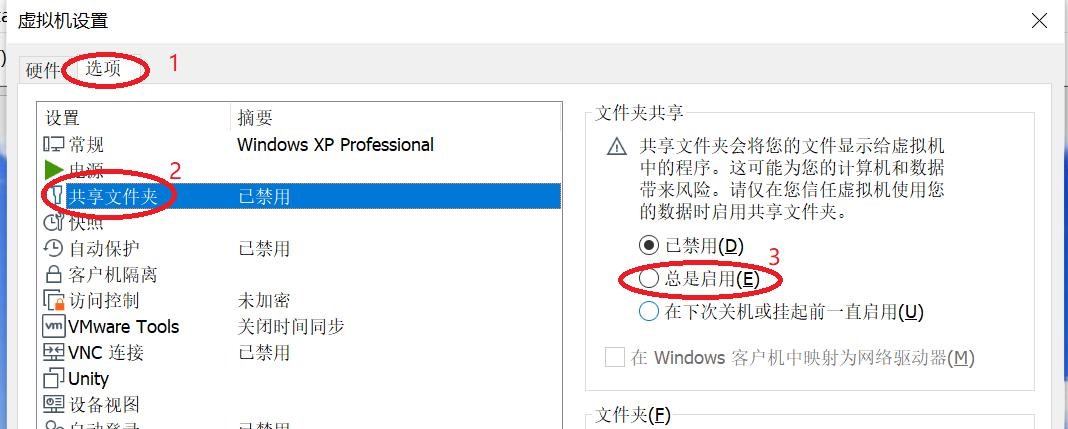


Figure 1.22 vm setup window -- enable file folder sharing

1. Then, in the file list below, insert a record of the directory path, which will be used to preserve the transmitted files. Press the [append...] button below the list

to open the guide dialog, as shown in figure 1.23.

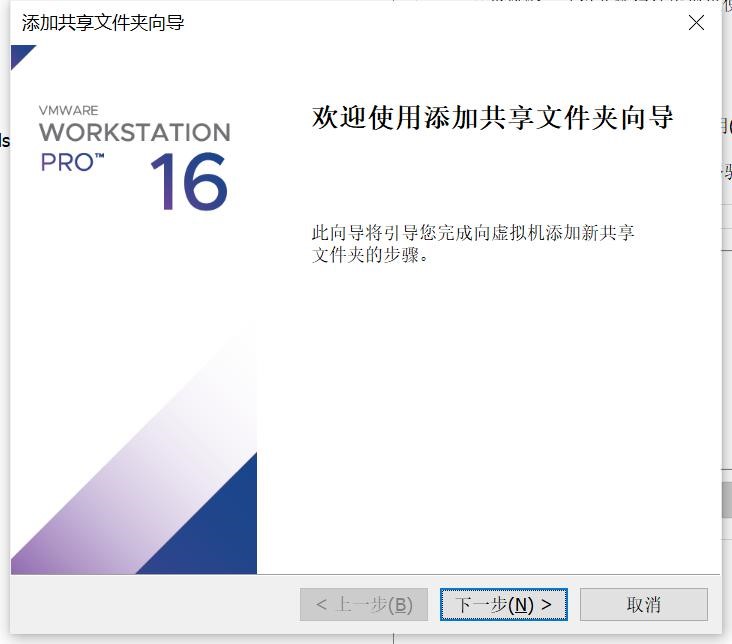


Figure1.23 shared folder append guide dialog

1. Press [next]  to continue. In the following window, press [view...]

to input the folder path. And then, give it a name appears

in the virtual machine operation system. In the example in figure 1.24, I choose folder

“[VM]FileExchange” to be the shard folder. Press [confirm]  to confirm

your selection, and return back to the second window. Give the file folder a name to be appeared in the virtual machine operating system, for example “[host]FileExchange”, as

shown in figure1.25. Then, press [next] .

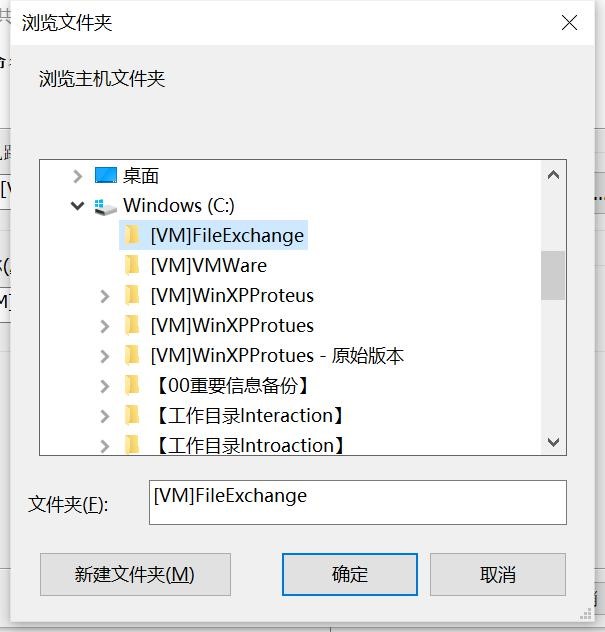


Figure1.24 select a file folder

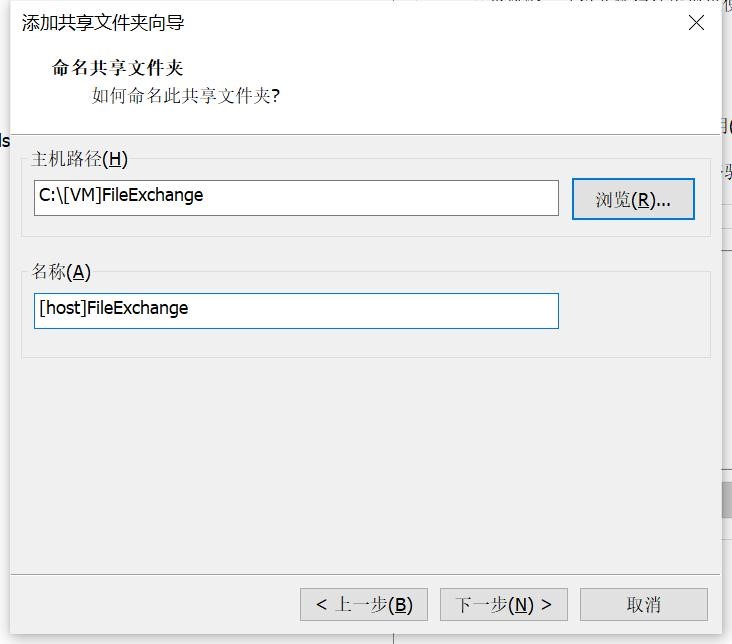


Figure1.25 name the file exchange folder

1. In the third window, make sure that you have checked the selection [start sharing]

, then press [complete] . The shared folder append

guide dialog will close, and in the vm setup window press [confirm]  to

complete setup process.

1. Now, you can open the windows resource manager, as shown in figure 1.26. The example is a windowXP system which is the vm we provided. In file path [network neighborhood\VMWare shared folders\vmware-host\Shared Folders”, you can find the host file folder, which is named as “[host]FileExchange” . File copy into this folder, can be accessed by both host and virtual machine system.

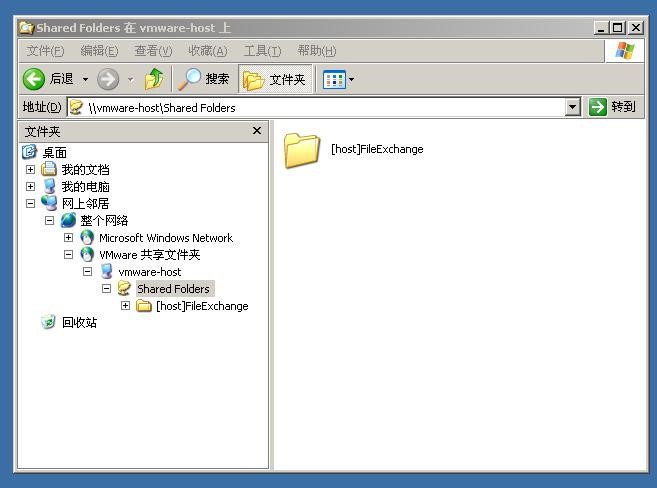


Figure1.26 vm windowsXP resource manager -- find the shared file folder

## 3. EV-Capture(EV-录屏)

EV-Capture is small utilization application. It can make a video record of all the thing happen on your desktop screen. We will use it to make the video of experiment process. And the process will be handed in as a part of the experiment report.

You can use other application software to make the record if you are quite familiar with it.

However, it must be mpeg4 compatible, which means, not a .mov file.

### 3.1 How to Install EV-Capture

The EV-Capture installer is provided as a zip file . Unzip it, and you will

find

the

installer



.

Double

click

the

installer

to

start

installation,

figure1.27.

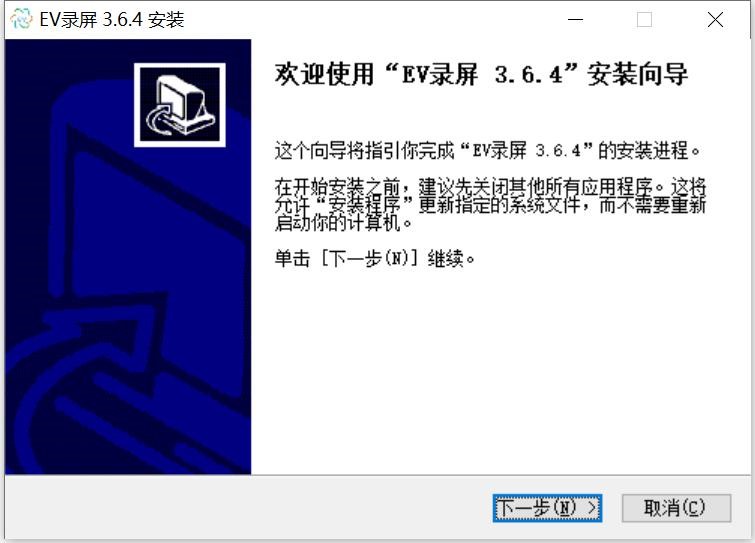


Figure1.27 EV-Capture install guide

Click the [next]  button to continue. In the next window, choose [I

accept]. And in the third window (figure1.28), it will let you choose the install target directory. We suggest that take the default settings. So , click [install]  button to

start installation.

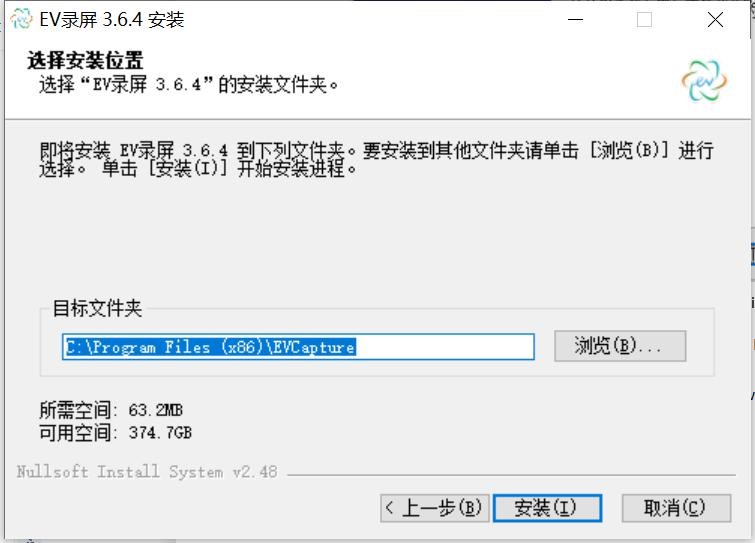


Figure1.28 Choose install target directory

### 3.2 How to use EV-Capture

Open the EV-Capture application, a update dialog may appears. Choose [cancel]

button to cancel update, for we do not need to update.

The main window of EV-Capture looks like figure 1.29.



Figure1.29 The main window of EV-Capture

Now click the [start record]  button in the bottom-left corner. After 3 down counting, the main window minimized, and the record begin. From now on, everything happens on the screen will be captured and recorded. And now you can carry on the experiment.

When you complete the experiment, double click the EV-Capture icon in the hot pan, so that to recall the main window of EV-Capture. You can find how long have you made the video record:

. Now, press the [stop]  button to terminate the

capturing process. And the main window will shift to the file list window in figure 1.30. The newly made record file is been selected and wait for you to give it a more meaningful name. Press the [return] key on your keyboard after you type in the new file name.



Figure1.30 Record file list window

If you want to hand in the video, use the [open file directory] button. It will open your window resource manager, and shift to the sub-directory where the video files being saved.

Choose the file you want, and upload it.

## 4. Prepare your experiment report

Experiment report is the summary of your experiment work and process. For we will conduct our experiment class online, the report will be different from the traditional type. It will be constructed by three files:

1. A .doc file of the experiment assignment problems. You can make a copy of the assignment description from this lecture notes.
2. The program source code file. And at the very beginning of the program file, there should be a comments to indicate who the file belongs to. Like the example below:

;=========================================================

;Description: this is the program of assignment 1 experiment 1

;Author: Yin LU, 2018xxxxxxxx

;Date: 2021-4-22

;=========================================================

1. A video of the experiment process. In the beginning of the video, there should be a shot to show the date of video recording, that is, the date of experiment completion. If you are making the video with the EV-Capture program, you can make the shot of date

by showing system date and time: 

# Chapter 2 Experiment 1 Brunch and Loop structure

In Chapter 2, it is our first experiment. And in this experiment, we will practice to create and debug x86 assembly language program with the Emu8086 application software.

There are all together 3 assignments in this experiment. Please program to solve the problem, and make a video of how you debug the program, and the result of your program execution. In order to see the result, please open the VAR window.

## 1. Assignments

1. Addition of multi-bytes BCD numbers. There are two large BCD numbers in the data segment, DATA1 and DATA2. Each number takes 5 bytes, and it is stored in little endian format (lower digit goes to lower address). Please adds DATA1 and DATA2 together, and save the result in the third variable DATA3. The Template of the program is provided below, and all the variables are predefined in the data segment.

|  |
| --- |
| ;========================================================  ;Description: Program of Assignment 1 Experiment1  ;Author:[name][student ID]  ;Date:[Date]  ;========================================================  .MODEL SMALL  .STACK 32  .DATA  DATA1 DB 10H,34H,56H,78H,00H  DATA2 DB 90H,88H,77H,66H,00H  DATA3 DB 00H,00H,00H,00H  .CODE  MAIN PROC FAR  ; INITIALIZE DATA SEGMENT  MOV AX, @DATA  MOV DS, AX  ;here is the program body  ; RETURN TO DOS  MOV AX, 4C00H  INT 21H  MAIN ENDP  END MAIN |

Hint: A similar multi-bytes data addition program can be found in example 3-4 chapter 3. And BCD addition algorithm can be found in section3.4.

1. Sorting of 10 Defined Byte data. There are 10 Defined Byte data in the variable called DATAS, and the count of 10 in the variable DATANUM. Please sort the 10 data in DATAS in the order

of from small to large. The Template of the program is provided below, and all the variables are predefined in the data segment.

|  |
| --- |
| ;========================================================  ;Description: Program of Assignment 2 Experiment1  ;Author:[name][student ID]  ;Date:[Date]  ;========================================================  .MODEL SMALL  .STACK 32  .DATA  DATANUM DB 10  DATAS DB 21H,13H,4H,5H,7H, 6H,8H,20H,9H,11H  .CODE  MAIN PROC FAR  ; INITIALIZE DATA SEGMENT  MOV AX, @DATA  MOV DS, AX  ;here is the program body  ; RETURN TO DOS  MOV AX, 4C00H  INT 21H  MAIN ENDP  END MAIN |

## 2. Experiment Preparation

All the things below should be prepared for experiment1:

1. The programs for assignment 1 and 2;
2. Emu8086 application. You can install the Emu8086 in your own computer system. Or you can use the Emu8086 in the virtual machine image we provided.

## 3. Experiment Circuit Scheme

We don’t have hardware device involved in experiment 1.

## 4. Experiment Process

1. Starting EMU8086, and use [NEW] to create a new empty workspace:

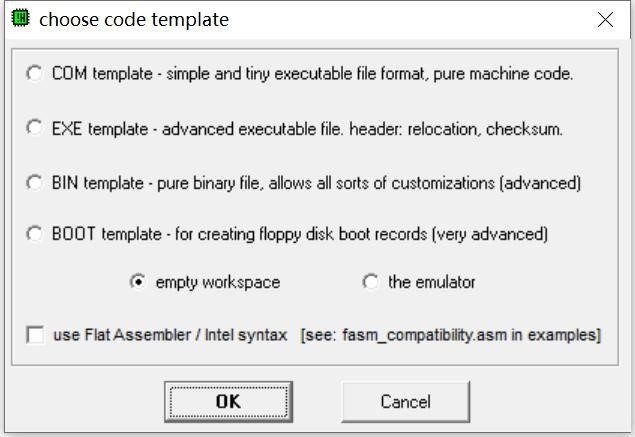


Figure2.1 create a assemble program project

1. Input your program, and save it as a .asm file;
2. Use  button to assemble your program. If there is errors, a hint dialog will appear. Fix these bugs.

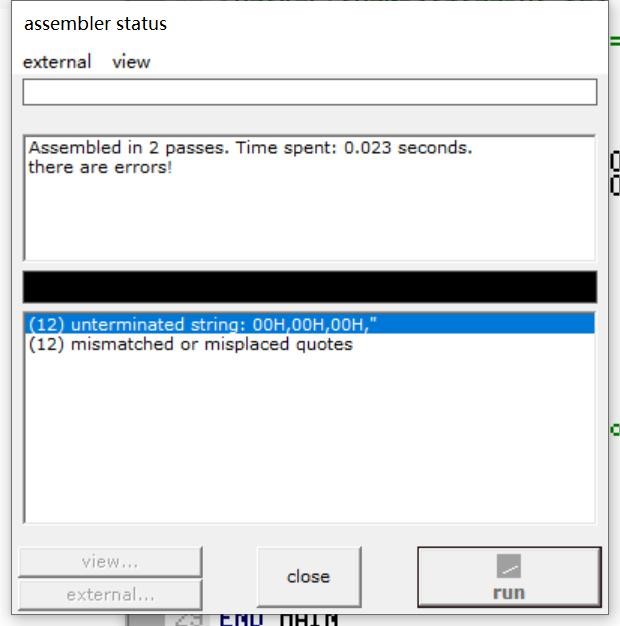


Figure2.2 try to assemble the source code

1. Use the  button to run and debug you program.
2. to see the result.

Use

the

function

dialog

of

[

aux/memory

]

or

[

vars

]



Figure2.3 check the variables

# Chapter 3 Experiment 2 Simple IO and Lantern Control

In Chapter 3 we will practicing how to manipulate peripheral devices, which means, try to use input and output instruction to access device ports.

We will start from a virtual device provided by the emu8086. Then move to the Proteus and try to run some program on the virtual hardware circuit.

## 1. Assignments

(1) Out put a data to a typical device port.

The emu8086 provides a virtual led display device, which is emulated by a program called “led\_display.exe”. The virtual device can display decimal number up to 5 digits, as shown in figur 3.1.

By output a word type data to port 199, which is a word sized IO port addres, you can change the display to the number you output. Now write a program to display numbers from 0 to 65535 in a loop.

Each time you write a number to the port, remember to call a sub program called “delay”, so that to wait for the display to be stable. The template of the program is provided in the list below.

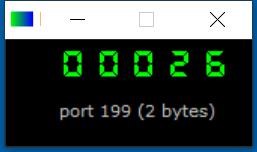


Figure3.1 The virtual led display device

List3.1 program template of assignment 1

|  |
| --- |
| ;========================================================  ;Description: Program of Assignment 1 Experiment1  ;Author:[name][student ID]  ;Date:[Date]  ;========================================================  ;This is the program for experiment2 assignment 1  ;In this program, we try to display numbers with  ; a virtual led display device provided by emu8086.  ;The port to setup the display is 199  ;======================================================== ;start the virtual LED display device with the command below: #start=led\_display.exe#  .MODEL SMALL  .STACK 64  .DATA  PORT\_LED EQU 199  .CODE  MAIN PROC FAR;this is the program entry point  MOV AX, @DATA ;load the data segment address  MOV DS, AX ;assign value to data segment register  ;TODO1: display 8888 to test the device  MOV AX, 8888  MOV DX, PORT\_LED  OUT DX, AX  CALL DELAY ;call delay sub procedure  ;TODO2: start to display numbers  ;(put your program to do the display of numbers here)  MOV AH, 4CH ;set up to  INT 21H ;return to DOS  MAIN ENDP  ;===========================================================  DELAY PROC NEAR  PUSH BX;  PUSH CX;  MOV BX,0Ah loop\_OUT: MOV CX, 03h  loop\_inner: LOOP loop\_inner  DEC BX  JNZ loop\_OUT  POP CX;  POP BX;  RET  DELAY ENDP  END MAIN ;this is the program exit point |

(2) Do output with 8255 IO controller.

Now let us move to real hardware devices, although it is still an emulated one.

In this experiment, 8 LED lights are connected to the data bus the 8086 processor through

port A of a piece of 8255, and a piece of 74LS138 is used to do address decoding. As is shown in figure3.2. The Y0 output of 74LS138 is assigned to 8255. And Address BUS A3 to A5, together A6 and A7 are used to be input of address decoder, as is shown in figure3.3. The port address of 8255 can be calculated based on figure3.3.

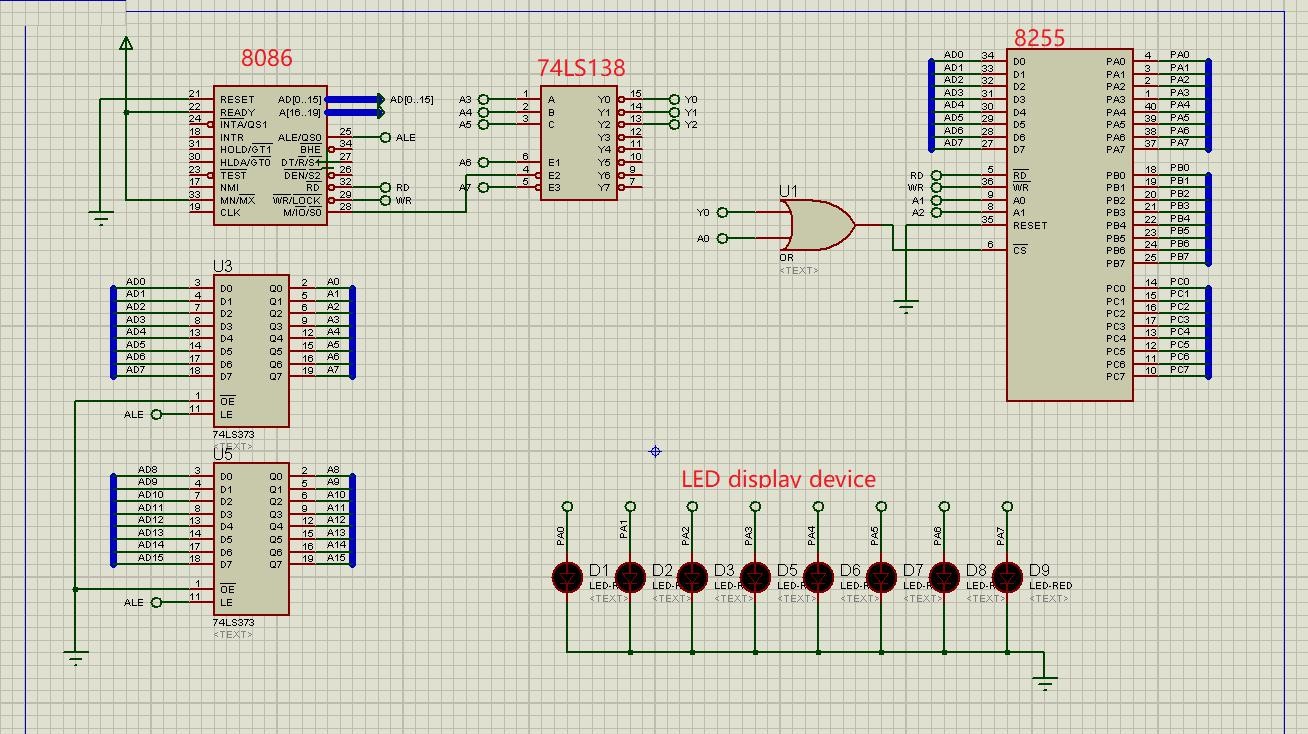


Figure3.2 Circuit schema of assignement2

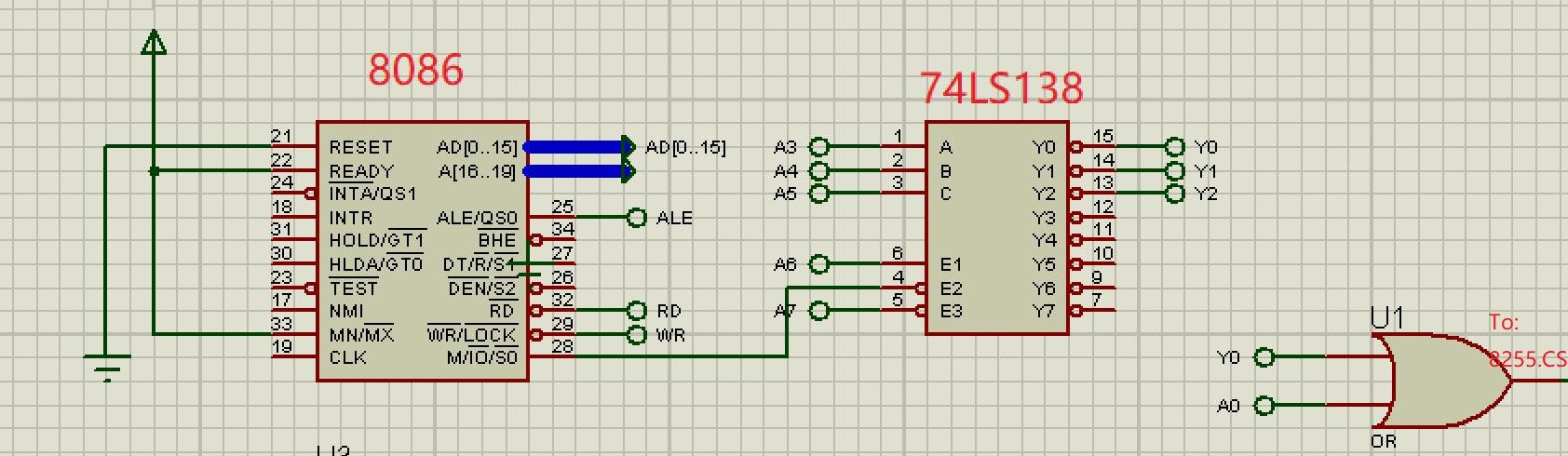


Figure3.3 Address decoding of assignement2

You are required to write a program to light the led one by one. You may start with D9, then light D8 and turn off D9, then move to the next D7, etc. When you reaches D1, please roll back to D2, and so on, till you turn on D9 again. And please do it repeatedly.

You may use logic shift instructions like SHL and SHR to generate the pattern code, and output pattern code through 8255.PortA to light the LEDs. And make use of a flag variable in your program to indicate on which direction should you shift the light.

(3) Input and out put with 8255.

Now we try to do input and output together.

In assignment 3, we change the led lights into a nixie tube, and 4 pin switches. The switches are connected with PortA.PA0 to PA3, and are used to input a single digit hex decimal number. This number should be displayed by the nixie tube. The nixie tube is connect with PortB of 8255.

You are required to write a program, input the number setup by the switches, then convert the number into a pattern code. And then, use the pattern code to light the nixie tube and display the input hex decimal number.

You can make use of the XLAT instruction to do the convert from hex into pattern code.

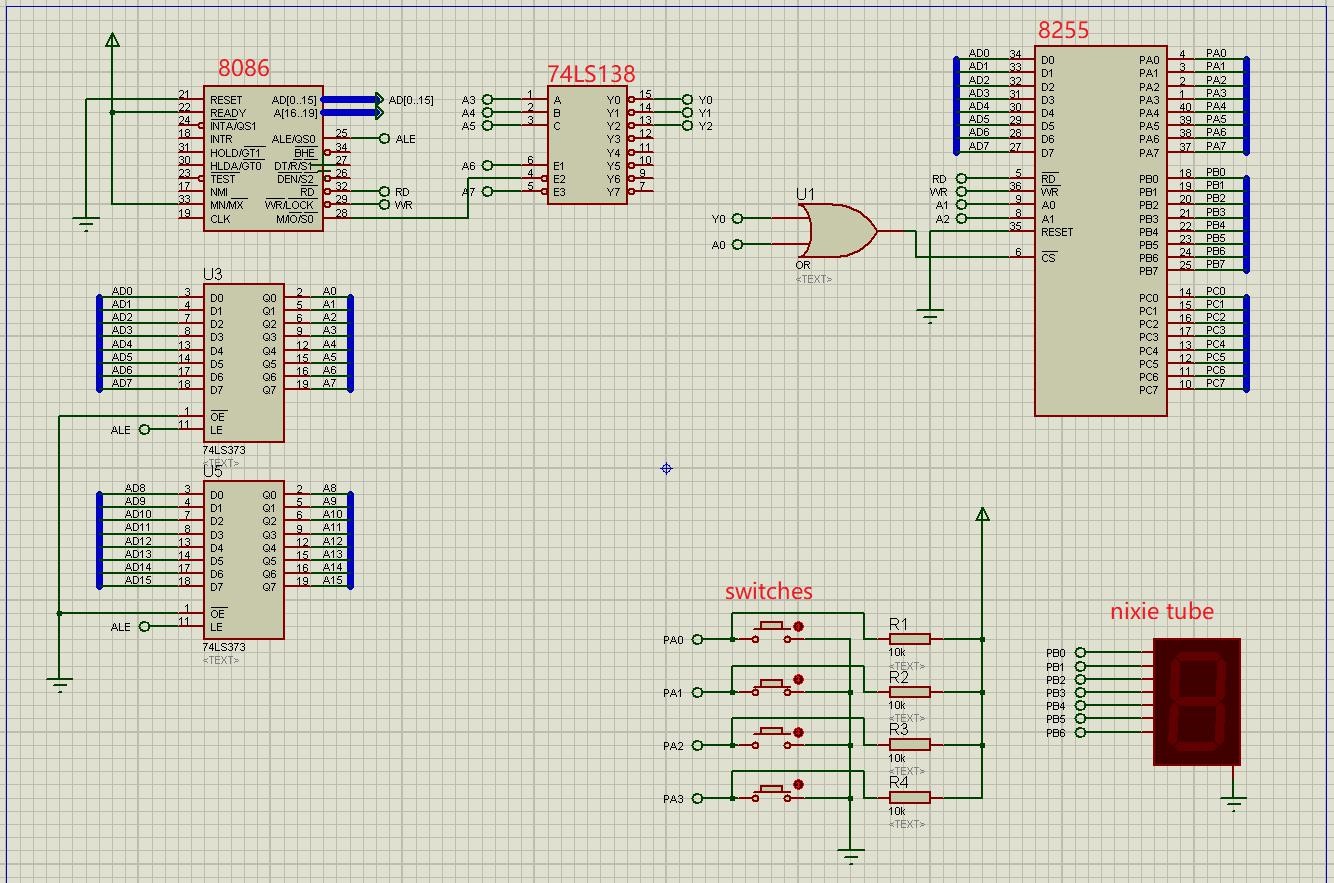


Figure3.4 Circuit schema of assignement3

## 2. Experiment Preparation

All the thing below should be prepared for experiment1:

1. The programs for assignment 1, 2 and 3;
2. The VMWare Virtual Machine WindowsXP with Emu8086 and Proteus installed.

## 3. Experiment Circuit Scheme

See figure 3.2 and figure 3.4.

## 4. Experiment Process

Start the the virtual machine WindowsXP, and import all the three programs of the assignments.

Assignment 1: Out put a data to a typical device port.

1. Start Emu8086, and open the program of assignment1;
2. Emulate  run the program. The virtual led display device will appear automatically;
3. Run  the program, and debug it till you see correct display.

Assignment2:Do output with 8255 IO controller.

1. Start Emu8086, and open the program of assignment2;
2. Compile the program to generate a executable file for windows. Save the .exe file in your file folder;

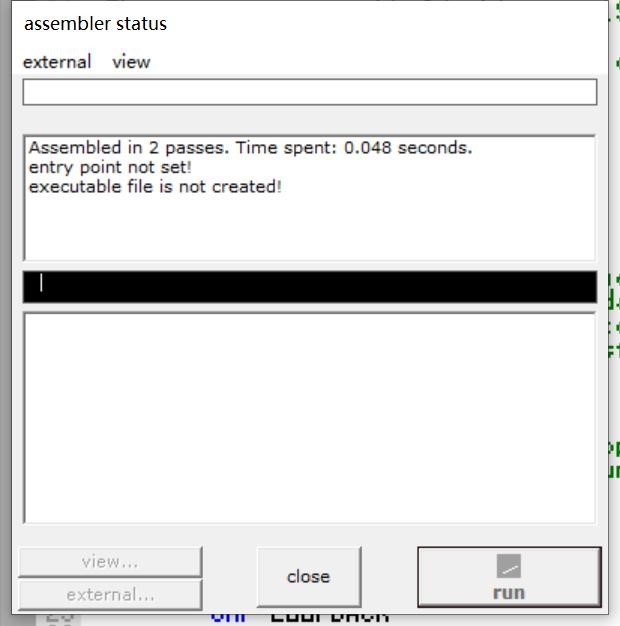


Figure3.5 compile the assembly program

1. Start Proteus application, load in the hardware circuit schema 8255LED.DSN file Into proteus.

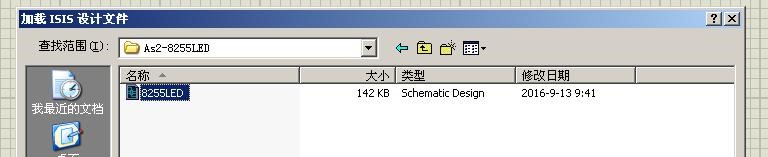


Figure3.6 load 8255LED circuit schema

1. Double click the 8086 processor to open a property editor dialog. And in the Program File: input box, select the executable file you produced in step (2);
2. Use the [run]  button to run your program with this hardware circuit emulation program.
3. If the program contain bugs, use the [debug] menu and

[start to debug...] menu item to start single step debugging process of your program. As shown in figure 3.8.

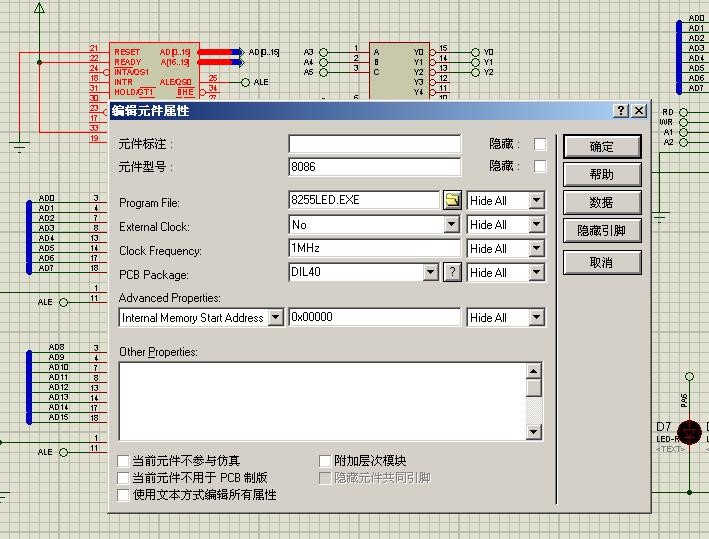


Figure3.7 load the executable file into 8086 processor

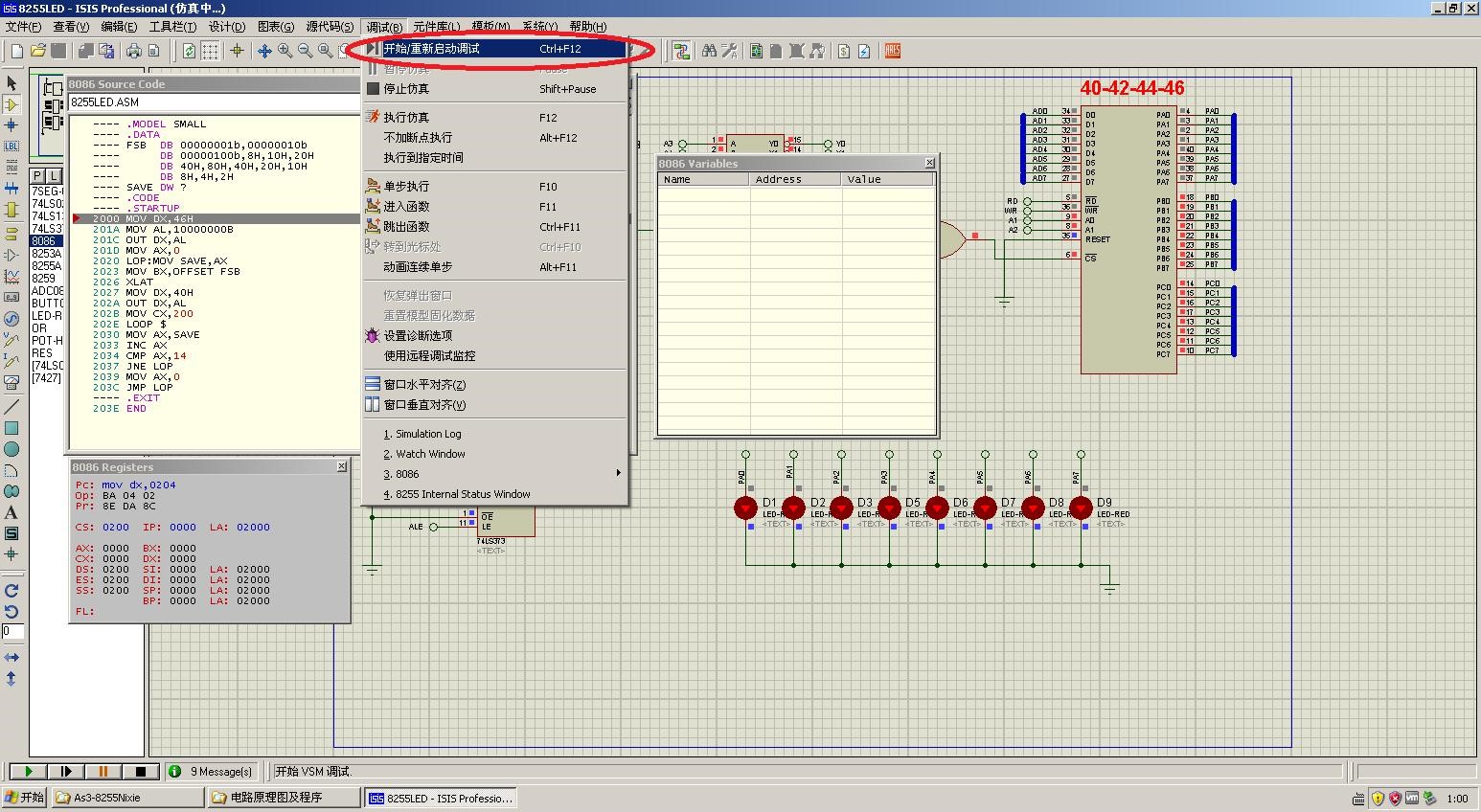


Figure3.8 Debug the program

Assignment3:Input and out put with 8255.

1. Start Emu8086, and open the program of assignment2;
2. Compile the program to generate a executable file for windows. Save the .exe file in your file folder;
3. Start Proteus application, load in the hardware circuit schema 8255NUM.DSN file Into proteus.
4. Double click the 8086 processor to open a property editor dialog. And in the Program File: input box, select the executable file you produced in step (2);
5. Use the [run] button to run your program with this hardware circuit emulation program.
6. If the program contain bugs, use the [debug] menu and [start to debug...] menu item to start single step debugging process of your program.



(this is the last page)

Out line

1. Knowing the Applications
   1. Emu8086 (what is, what for, how to use)
   2. Vmware (what is, what for, how to use)
   3. Protues (what is, what for, how to use)
   4. Install and set up the experiment toolkit (how to setup vmware, and install the virtual machine. All the MASM toolkit has all ready been installed in the virtula machine. )
2. Experiment I: Starting debug your first program
3. Experiment II: Knowing Protues from the simplest IO circuit
4. Experiment III: Input and Out
5. Experiment IV: A complicate AD Sampling circuit