

## Getting Started with MASM and Visual Studio 2010

Updated 8/15/2011

This tutorial shows you how to set up Visual Studio 2010 (including the Express version) to work with Microsoft MASM. If you are using Visual Studio 2008 or Visual C++ 2008 Express, [Click here](#).

If you're in a hurry to get started, you only need to read Item 1 below:

1. [Project properties settings](#)
2. [Generating a source listing file](#)
3. [Using the Visual Studio debugger](#)
4. [MASM syntax highlighting](#)
5. [Assembling, linking, and debugging with a batch file](#)
6. [Custom Build Rules](#)

The book's example programs in Chapters 1-14 have been successfully tested in Windows XP, 32-bit Vista and the 32-bit version of Windows 7. On the other hand, many programs in Chapters 15-17 will not run in any Microsoft OS later than Windows 98, because they rely on direct access to hardware and system memory. You cannot directly run 16-bit applications in any 64-bit version of Windows.

Found an error in this document? Please [email me immediately](#). Except where noted, all instructions in this document apply equally to Visual Studio 2010 and Visual C++ Express 2010.

### Required Setup for 32-bit Applications

First, you must install Visual Studio 2010 and select the C++ language option. VS 2010 and Visual C++ 2010 Express both include the current version of the Microsoft Assembler. You can verify that the Microsoft Assembler is installed by looking for the file **ml.exe** in the \vc\bin folder of your Visual Studio installation directory, such as c:\Program Files\Microsoft Visual Studio 10.x\vc\bin.

**Using Visual Studio Express? You must do the following in order to see the same menu options as the users of Visual Studio professional: from the *Tools* menu, select *Settings*, and select *Expert Settings*.**

### Next: Install the Book's Example Programs

[Click this link](#) to get the latest copy of the book's link libraries and example programs. The examples are stored in a self-extracting archive file that automatically extracts to the **c:\Irvine** folder. Unless you have some objection to using that location, do not alter the path. (Lab managers: you can designate c:\Irvine directory as read-only.) If you plan to change the installation location, read our instructions relating to [changing project properties](#).

The following files will be copied into the c:\Irvine directory:

Filename	Description
cmd.exe	Shortcut to the Windows command-line interpreter (named cmd.exe)
GraphWin.inc	Include file for writing Windows applications
Irvine16.inc	Include file used with the Irvine16 link library (16-bit applications)
Irvine16.lib	16-bit link function library used with this book
Irvine32.inc	Include file used with the Irvine32 link library (32-bit applications)
Link16.exe	16-bit linker
Irvine32.lib	32-bit link function library used with this book
User32.lib	Basic I/O link library
Macros.inc	Include file containing macros (explained in Chapter 10)
SmallWin.inc	Small-sized include file, used by Irvine32.inc
make16.bat	Batch file for building 16-bit applications
VirtualKeys.inc	Keyboard code definitions file, used by Irvine32.inc

A subdirectory named **Examples** will contain all the example programs shown in the book, as well as all the source

code for the book's 16- and 32-bit link libraries.

## Setting up Visual Studio

You will only have to do these steps the first time you use Visual Studio.

### Add the *Start Without Debugging* command to the Debug menu

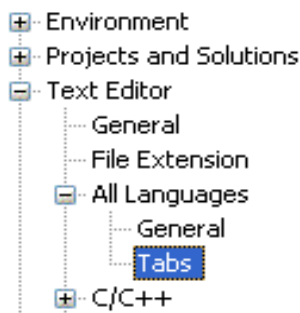
It's very useful to run programs without having to debug them. To do that, you will want to add a new command to the Debug menu: *Start Without Debugging*. Here's how to do it:

1. From the Tools, menu, select *Customize*.
2. Select the *Commands* tab.
3. Select Menu bar (radio button).
4. Click the *Add Command* button.
5. Select *Debug* from the Categories list.
6. Select *Start Without Debugging* in the right-hand list box.
7. Click the OK button.
8. Click the Close button.

In fact, you can use the same sequence to customize any of the menus and toolbars in Visual Studio.

### Set the Tab Size to 5

Start Visual Studio, and select **Options** from the **Tools** menu. Select **Text Editor**, Select **All Languages**, and select **Tabs**:



Set the Tab Size and Indent Size to 5.

## Building a Sample Assembly Language Program

Now you're ready to open and build your first project.

### Opening a Project

Visual Studio and Visual Studio Express require assembly language source files to belong to a *project*, which is a kind of container. A project holds configuration information such as the locations of the assembler, linker, and required libraries. A project has its own folder, and it holds the names and locations of all files belonging to it. We have created a sample project folder in the `c:\Irvine\examples\ch03` directory, and its name is *Project*.

Do the following steps, in order:

1. Start Visual Studio.
2. First you will open an existing Visual Studio project file. If you're using Visual Studio, select **Open Project** from the File menu. Or, if you're using Visual Studio Express, select **Open**, and select **Project/Solution**.
3. Navigate to the `c:\Irvine\Examples\ch03` folder and open the file named **Project.sln**.
4. In the *Solution Explorer* window, you will see the word Project. This is the name of a Visual Studio project.
5. Next, you need to add a source code file named *main.asm* to the project. To do that, right-click on **Project**, select **Add**, select **Existing Item**, select **main.asm**, and click the **Add** button to close the dialog window. (You can use this sequence of commands in the future to add any asm file into a project.)
6. Next, you will open the main.asm file for editing. Double-click the file named **main.asm** to open it in the editing window. (Visual Studio users may see a popup dialog asking for the encoding method used in the asm file. just click the OK button to continue.)

**Tip:** If the *Solution Explorer* window is not visible, select *Solution Explorer* from the *View* menu. Also, if you do not see *main.asm* in the *Solution Explorer* window, it might be hidden behind another window. To bring it to the front, click the *Solution Explorer* tab from the tabs shown along the bottom of the window.

You should see the following program in the editor window:

```
TITLE MASM Template                                (main.asm)

; Description:
;
; Revision date:

INCLUDE Irvine32.inc

.data
myMessage BYTE "MASM program example",0dh,0ah,0

.code
main PROC
    call Clrscr

    mov  edx,OFFSET myMessage
    call WriteString

    exit
main ENDP

END main
```

Later, we'll show you how to copy this program and use it as a starting point to write your own programs.

### Build the Program

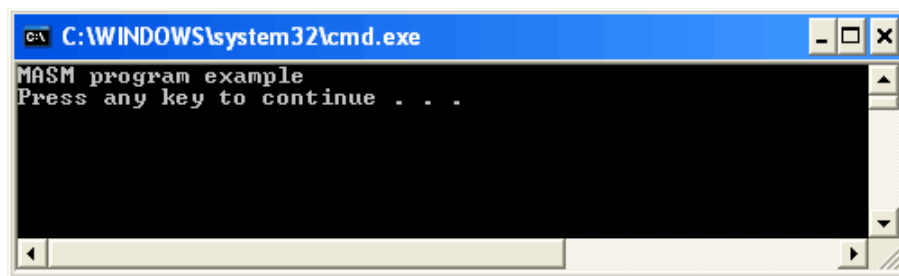
Next, you will build (assemble and link) the sample program. Select **Build Project** from the Build menu. In the Output window for Visual Studio at the bottom of the screen, you should see messages similar to the following, indicating the build progress:

```
1>----- Build started: Project: Project, Configuration: Debug Win32 -----
1>Assembling...
1>Assembling: .\main.asm
1>Linking...
1>Embedding manifest...
1>Build log was saved at "file://g:\masm\Project_sample\Debug\BuildLog.htm"
1>Project - 0 error(s), 0 warning(s)
===== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped =====
```

If you do not see these messages, the project has probably not been modified since it was last built. No problem--just select **Rebuild Project** from the Build menu.

### Run the Program

Select **Start without Debugging** from the Debug menu. The following console window should appear, although your window will be larger than the one shown here:



The "Press any key to continue..." message is automatically generated by Visual Studio.

Congratulations, you have just run your first Assembly Language program.

Press any key to close the Console window.

When you assembled and linked the project, a file named **Project.exe** was created inside the project's \Debug folder. This is the file that executes when you run the project. You can execute Project.exe by double-clicking its name inside Windows Explorer, but it will just flash on the screen and disappear. That is because Windows Explorer does not pause the display before closing the command window.

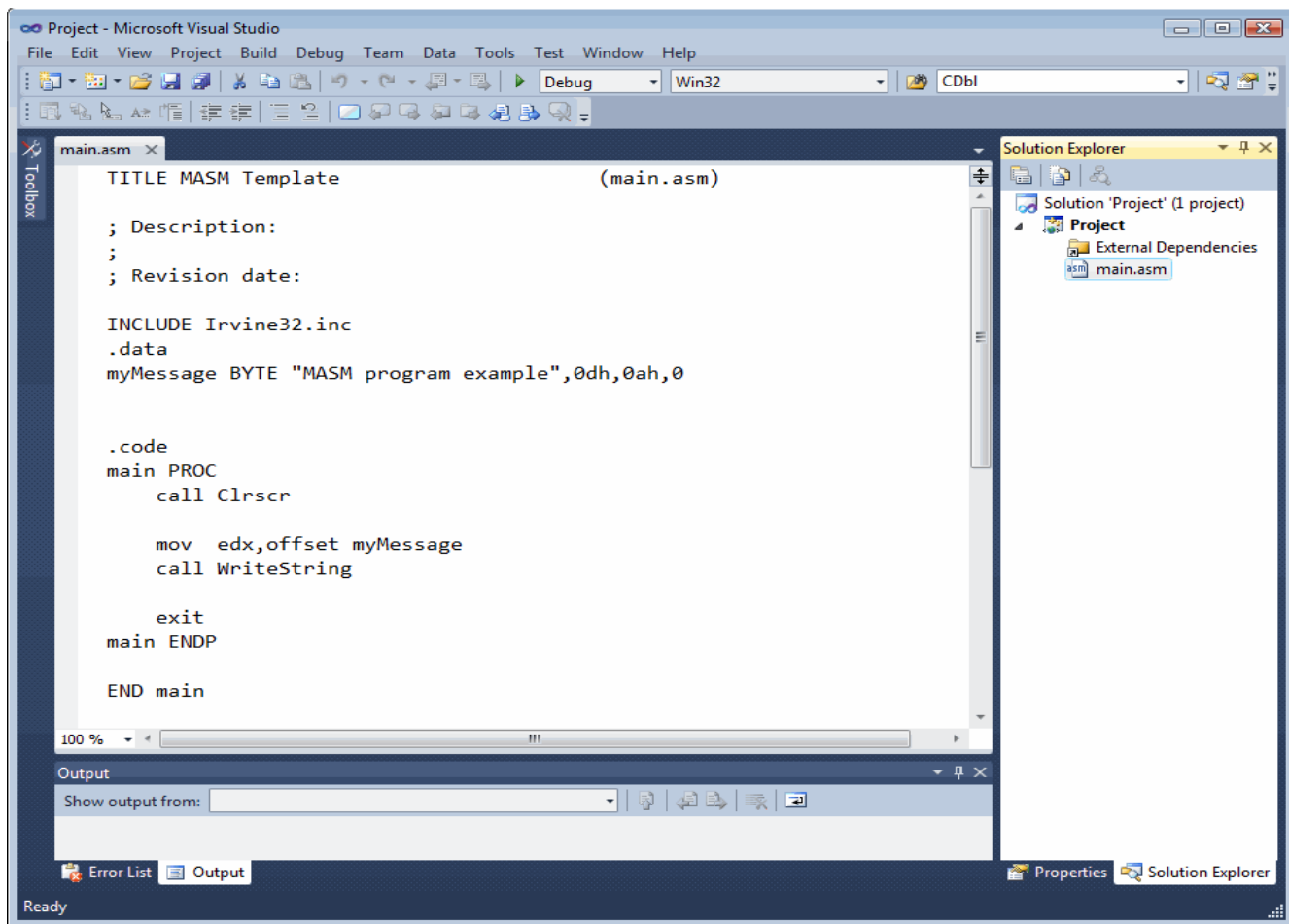
### Creating New Projects of Your Own

Before long, you will want to create your own projects. The easiest way to do this is to copy the entire **c:\Irvine\Examples\Project\_Sample** folder to a new location. Copy it to a folder in which you have read/write permissions. (If you're working in a college computer lab, a useful location is a portable USB drive. Then you can modify the program, build, and run it again.)

### Step 5: Running the Sample Program in Debug Mode

In this step, you will set a breakpoint inside the sample program. Then you will use the Visual C++ debugger to step through the program's execution one statement at a time.

1. Make sure the ASM source code file is open in the editor window.
2. To begin stepping through your program in Debug mode, press the F10 key.
3. A yellow arrow will appear next to the first program statement (*call C!rscr*). The arrow indicates that the statement is next to be executed.
4. Press the F10 key (called *Step Over*) to execute the current statement. Continue pressing F10 until the program is about to execute the **exit** statement.
5. A small black window icon should appear on your Windows status bar. Open it and look at the contents of the Command window. You should see the words "MASM program example" in the window.
6. Press F10 one more time to end the program.



## Registers

If you want to display the CPU registers, do the following: Start debugging the program, then select *Windows* from the *Debug* menu. Select *Registers* from the drop-down list. The bottom window will display the register contents. Right click this window and check the item *Flags* to enable the display of conditional flags.

You can interrupt a debugging session at any time by selecting *Stop Debugging* from the *Debug* menu. You can do the same by clicking the blue square button on the toolbar. To remove a breakpoint from the program, click on the red dot so that it disappears.

## Setting a BreakPoint

If you set a breakpoint in a program, you can use the debugger to execute the program a full speed (more or less) until it reaches the breakpoint. At that point, the debugger drops into single-step mode.

1. Click the mouse along the border to the left of the call `WriteString` statement. A large red dot should appear in the margin.
2. Select *Start Debugging* from the *Debug* menu. The program should run, and pause on the line with the breakpoint, showing the same Yellow arrow as before.
3. Press F10 until the program finishes.

You can remove a breakpoint by clicking its red dot with the mouse. Take a few minutes to experiment with the *Debug* menu commands. Set more breakpoints and run the program again. For the time being, you can use the F11 key to step through the program in the same way the F10 key did.

## Building and Running Other Programs

Suppose you want to run another example program, or possibly create your own program. You can either edit and modify `main.asm`, or you can remove `main.asm` from the project and insert some other `.asm` file into the project.

- To remove a program from a project without deleting the file, right-click its name in the *Solution Explorer* window. In the context menu, select **Exclude from Project**. If you change your mind and decide to add it

back to the project, right-click in the same window, select **Add**, select **Existing item**, and select the file you want to add.

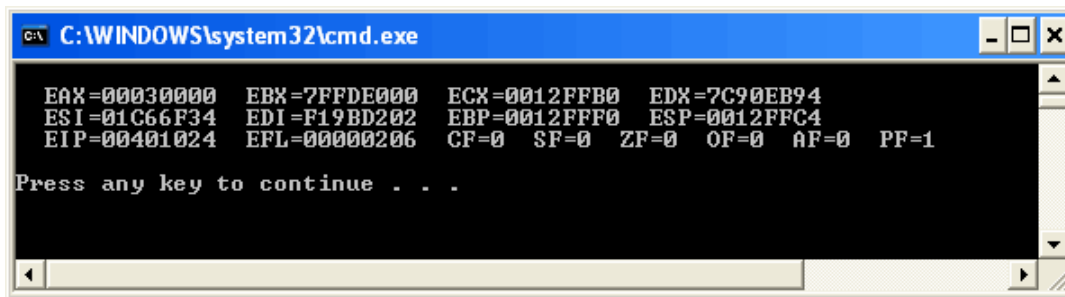
- To remove a program from a project and delete the source code file, select the file with the mouse and press the **Del** key. Or, you can right-click the file name and select **Remove**.

### Adding a File to a Project

The easiest way to add an assembly language source file to an open project is to drag its filename with the mouse from a Windows Explorer window onto the name of your project in the Solution Explorer window. A reference to the file (not a copy) will be inserted in your project's directory. Try this now:

1. Remove the main.asm file from your project.
2. Add a reference to the file c:\Irvine\Examples\ch03\AddSub.asm to the project.
3. Build and run the project.

Here is what you should see in the Console window, except that only your EAX register will have the same value as ours:



```
C:\WINDOWS\system32\cmd.exe
EAX=00030000  EBX=7FFDE000  ECX=0012FFB0  EDX=7C90EB94
ESI=01C66F34  EDI=F19BD202  EBP=0012FFF0  ESP=0012FFC4
EIP=00401024  EFL=00000206  CF=0  SF=0  ZF=0  OF=0  AF=0  PF=1
Press any key to continue . . .
```

When you press a key, the console window will close.

### Copying a source file

If you want to make a copy of an existing file, use Windows Explorer to copy the file into your project directory. Then, right-click the project name in Solution Explorer, select Add, select Existing Item, and select the filename.

[Return to top](#) or read about [Project Properties settings](#).