# Getting Started with MASM and Visual Studio 2013

*Updated 5/6/2015*

This tutorial assumes that you are using the Seventh Edition of *Assembly Language for x86 Processors*. We show you how to set up Visual Studio 2013 (including Visual Studio 2013 Express for Windows Desktop, and Visual Studio Community 2013 edition) to work with the Microsoft assembler. Visual Studio 2013 Express and Visual Studio Community 2013 may be downloaded from Microsoft.com. *(Note: the directions shown here are **not** designed for use with "Visual Studio Express 2013 for Windows" That product is designed for creating Windows Store apps in Visual Basic and C#.)*

Topics:



The book's example programs in Chapters 1-13 have been successfully tested in both 32/64-bit Windows 7 and Windows 8. On the other hand, many programs in Chapters 14-17 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 the author. Except where noted, all instructions in this document apply equally to *Visual Studio* and *Visual Studio Express*.

### Required Setup for 32-bit Applications

First, you must install Visual Studio and select the C++ language configuration option the first time you run it. All versions of Visual Studio include the Microsoft Assembler (MASM) version 12.0. 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 12.0\vc\bin.

### Installing 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 Microsoft Install (.MSI) file that installs into the **c:\Irvine** folder. Unless you have some objection to using that location, do not alter the path. (Note to lab administrators: you can designate c:\Irvine directory as read-only.) If you plan to change the installation location, read our instructions relating to Creating a Project from Scratch.

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

| Filename | Description |
|---|---|
| b16.asm, b32.asm | Blank templates for 16-bit and 32-bit assembly language source files |
| 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) |
| Irvine32.lib | Irvine's 32-bit link library |
| Irvine64.obj | Irvine's 64-bit library |
| Kernel32.lib | 32-bit link library for Windows API |
| Link16.exe | 16-bit Microsoft linker |
| Macros.inc | Irvine's macro include file (see Chapter 10) |
| make16_vs2012.bat | Visual Studio 2012 batch file for building 16-bit applications |
| make16_vs2013.bat | Visual Studio 2013 batch file for building 16-bit applications |
| SmallWin.inc | Small-sized include file containing MS-Windows definitions, used by Irvine32.inc |

| | |
|---|---|
| User32.lib | MS-Windows basic I/O link library |
| VirtualKeys.inc | Keyboard code definitions file, used by Irvine32.inc |

A subdirectory named **Examples** will contain all the example programs shown in the book, source code for the book's 16-, 32-, and 64-bit libraries, and two sample Visual Studio projects.

## 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.

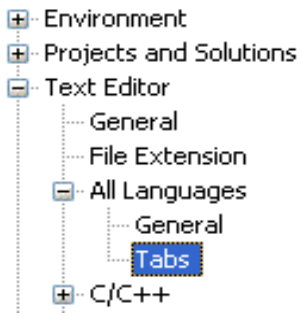
#### Select the C++ Configuration

(Skip this topic if you installed Visual Studio Express.) Visual Studio Professional, Ultimate, and Premium editions support multiple programming languages and application types. The C++ programming language configuration most closely matches that of assembly language programming, so we suggest the following steps:

1. Select Tools | Import and Export Settings from the menu
2. Select the "Import selected environment settings" radio button
3. Select the "No, just import..." radio button
4. Select "Visual C++" from the Default Settings List and click the Next button
5. Click the Finish button, then click the Close button
6. Notice the tabs on the left and right sides of the Visual Studio workspace. Close the Server Explorer, Toolbox, and Properties tabs. Use the mouse to drag the Solution Explorer tool window to the right side of the workspace. You can also select other tabs at the bottom of this window, such as "Class View", "Property Manager", and "Team Explorer", and close them. They will not be used in the future. If you need to bring back the Solution Explorer window at any time in the future, select View from the menu, and locate Solution Explorer in the list of views.

#### Set the Tab Size to 5

*(This is an optional step.)* Start Visual Studio, and select **Options** from the **Tools** menu. Select **Text Editor**, Select **All Languages**, and select **Tabs**. Optionally, you may want to select the **Insert spaces** radio button:




Set the Tab Size and Indent Size to 5.

# Tutorial: Building a 32-Bit Assembly Language Program

Now you're ready to open and build your first 32-bit project.

**Opening a Project**

Visual Studio requires 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* directory, and its name is *Project32*.

Do the following steps, in order:

1. Copy the c:\Irvine\Examples folder to a location on your hard drive that permits you to read and write files. You can use a USB drive, although Visual Studio may run a little more slowly when it creates temporary files during the build process.
2. Start Visual Studio.
3. To begin, open our sample Visual Studio project file by selecting **File/Open/Project** from the Visual Studio menu.
4. Navigate to the **c:\Irvine\Examples\Project32** folder and select the file named **Project.sln**.
5. Once the project has been opened, you will see the project name in the Solution Explorer window. If there are any files with .asm file extensions in the Solution Explorer window, select and delete them now.
6. Next, you will add an existing source code file to the project: In the Solution Explorer window, right-click on **Project**, select **Add**, select **Existing Item**, navigate to the "Examples\ch03" folder, select **AddTwo.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 to a project.) You should now see the AddTwo.asm file in the Solution Explorer window.
7. Next, open the AddTwo.asm for editing by double-clicking its filename in the Solution Explorer window.

You should see the following program in the editor window:

```
; AddTwo.asm - adds two 32-bit integers.
; Chapter 3 example

.386
.model flat,stdcall
.stack 4096
ExitProcess proto,dwExitCode:dword

.code
main proc
        mov     eax,5
        add     eax,6

        invoke ExitProcess,0
main endp
end main
```

In the future, you can use this file as a starting point to create new programs by copying it and renaming the copy in the Solution Explorer window.

**Build the Program**

Now 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:

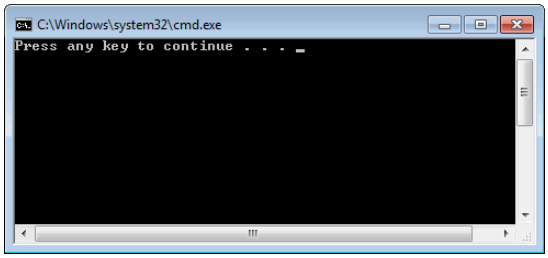
```
1>------ Build started: Project: Project, Configuration: Debug Win32 ------
1>  Assembling ..\ch03\AddTwo.asm...
1>  Project.vcxproj -> G:\ASM Book Examples\Project32_VS2013\Debug\Project.exe
========== Rebuild All: 1 succeeded, 0 failed, 0 skipped ==========
```

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.

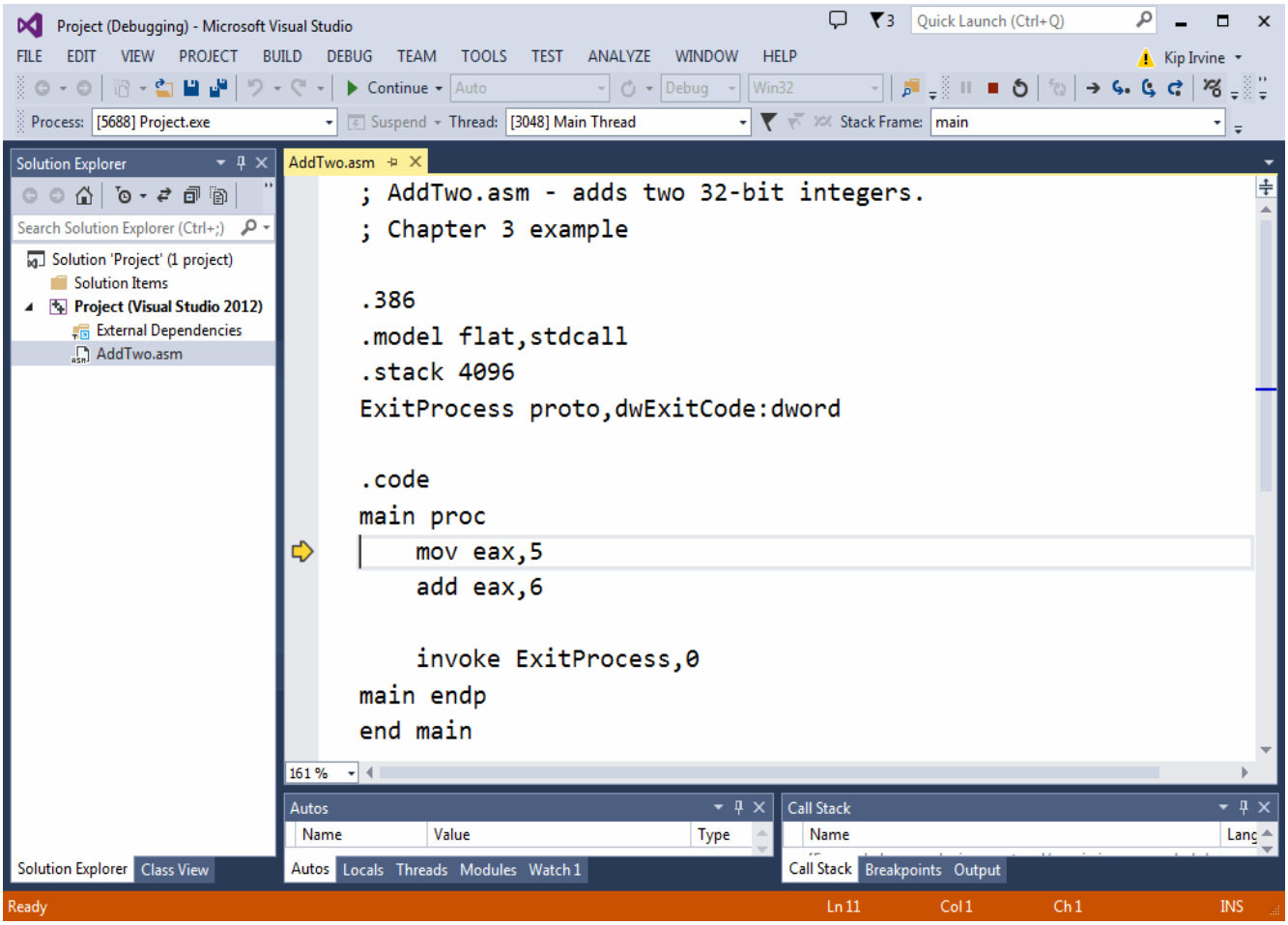
**TIP:** When you assembled and linked the project, a file named **Project.exe** was created inside the project's \Debug folder. This file executes when you run the project. You can execute Project.exe by double-clicking its name inside Windows Explorer, but it will just flash on the screen and disappear. That is because Windows Explorer does not pause the display before closing the command window. On the other hand, you can open a Command prompt window, move to the Debug directory, and run Project.exe by typing "Project" (without the quotes). You will need to do some reading on Windows shell commands if you plan to use the command line.

Any time you want to remove a source file from the Visual Studio window, right-click its filename and select **Remove**. The file will not be deleted from the file system. On the other hand, if you want to delete the file, select it and press the Del key.

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

In this step, you set a breakpoint inside the sample program. Then you use the Visual Studio 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 should appear next to the first program statement.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 **invoke** 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. The window should be blank because this program does not display any output.
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 Registers window may appear at the bottom of the workspace, as a tab highlighted in yellow. Use the mouse to drag the window to the right side of the work area. Right click inside the Registers window and check the item *Flags* to enable the display of CPU status 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 maroon-colored square button on the toolbar. To remove a breakpoint from the program, click on its red dot to make it disappear.

### 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. In our sample program, click the mouse along the border to the left of the **mov eax,5** 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 remove the existing assembly language file from the Solution Explorer window and insert a new .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 **Remove**. 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.

### Adding a File to a Project

An easy 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. The physical file will not be copied--the project only holds a reference to the file's location. Try this now:

1. Remove the AddTwo.asm file from your project.
2. Add a reference to the file Examples\ch03\AddTwoSum.asm to the project.
3. Build and run the project.

### Copying a Source File

One way to make a copy of an existing source code file is to 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.