Core Toolchain Setup

```
for MIOS application development on Windows
```

Don't forget The QuickStart Guide! It's the quick and easy setup. It will work for nearly everyone.

The technical jargon:
This is the *Core Tools* section in a walkthrough on the process of developing MIOS Applications, on Microsoft Windows XP, primarily in C, utilising the MIOS C Wrapper but also in ASM. The IDE platform used will be Code::Blocks IDE or NotePad++ and the applications will be built for the MIDIBox Hardware Platform (MBHP) stuffed with a Microchip PIC18F. C code will be compiled with SDCC, and assembled with GPUtils, using MSYS to generate and run DOS-Console-based makefile scripts, and MIOS Studio will be used for debugging on MBHP. For PC emulation for debugging purposes, you may also use AC-Sim (AudioCommander's CSimulator) compiled with GCC from MinGW.

Enough of that.
Please don't be put off by all that mumbo jumbo. If you're totally newbie then this document is also for you :) I've included everything you need to do, step-by-step, for everything except for actually coding the app. I hope this will not only make it easier for newbies to get started, but for ASM coders like myself to transfer their skills into C development (which seems to be the future of MIOS, and is an excellent rapid prototyping method) or as a reference point for the experienced. In order to do this, I've included as much information as possible, but I have created links to jump through the document for those who don't need all that detail.

I hope you find this document helpful, and that you will feel free to make suggestions or criticism or corrections or any kind of modifications as you see fit :)

Preparation

It is important that you start with a clean installation. We all hate to do this, but you should start by uninstalling any of the following applications first, and then rebooting your PC.

Windows Utilities

Sun Java J2SE

Chances are you already have the Java runtime... I don't recommend upgrading the existing version if you do.

- Download J2SE
- Install with defaults or whatever meets your requirements. I recommend disabling automatic updates.

7zip

This tool is needed to extract compressed files. If you already have winzip or winrar then you won't need this. I recommend the latest version MSI installer for ease of use.

- Download 7zip
- Install the complete application using the defaults

Optional Tools

Small Device C Compiler (SDCC)

This tool is required for all applications coded in the C language. If you're not sure, I'd install it - it won't cause any problems either way.

- Download SDCC. 2.80 is officially supported or you can get the latest snapshot if you're more experienced.

This is a point of some contention. If you are new, I would recommend 2.80. Some PIC compiler routines have been improved in later snapshot builds but they can also cause perfectly good code to fail. If you get the snapshot, search for “i586-mingw32msvc-setup” and take the file at the top of the list.

- Either way, I recommend you download the relevant documentation for SDCC - it will come in handy later! If you are getting v2.80, you can get the manualhere, but it should be installed with SDCC. If you are getting the snapshot build, search for “docs” and take the .zip file at the top of the list.
- Install using defaults (Note that this installs the PDF doc as well which can be very useful)

Required Tools

MSYS

MSYS provides us with a POSIX (UNIX style) environment for our Windows PC. This packages includes tools such as make and sed and sh, which are required to correctly run gputils later on. I recommend the latest version, but v1.0.10 has tested OK. If you are going to use MinGW for AC-Sim, ensure that you have installed that completely before performing the MSYS install. See the above step in this document.

- Download MSYS (more info here)
- Install using the defaults.
- When the install is almost complete, you will see a DOS console window open, and you will be prompted: “Do you wish to continue with the post install? [yn]”. Type 'y' (without the quotes) and hit Enter.
- Next you will be asked: “Do you have MinGW installed? [yn]” Type 'n' without the quotes and hit enter.
- Press Enter, to exit the postinstall script, and click 'Finish'.

GNU PIC Utilities (GPUtils)

- Download gputils-win32. Grab the latest. v0.13.5 is tested OK.
- Install the complete application using the 'Full' option.
- When the installer asks you, click 'Yes' to add the installation directory to the PATH environment variable.

PATH Environment Variables

PATH Environment variables take a little bit of attention.

First, a bit about what it is: The environment variable 'PATH' is a 'search path' that is used whenever you run a command in your DOS console. If the program you are trying to run (like GPASM or SDCC) is not in the current working directory, then the system will look for that program in each of the directories specified in the PATH.
There are actually two PATH variables, and the User PATH Variable is appended to the System PATH Variable, and each directory in those variables will be searched in order.
For our purposes, it is necessary to ensure that the above tools have entries in the PATH, and that they are in the correct order. Usually, there is no need to have user-specific PATH variables, so the following procedure will do away with them, and we will just work with the System PATH for the sake of simplicity:

- Minimise any open windows so that you can see the desktop.

- Right-click on 'My Computer', select 'Properties'.
- Click on the 'Advanced' tab, then click 'Environment Variables'.
- Under System Variables, select the variable 'PATH' and click 'Edit'.
- Highlight all of the text in there, and hit CTRL+X to cut it out. Click OK.
- Click the 'Start' button, and select 'Run', and type “notepad.exe” and click OK.
- Hit CTRL+V to paste the System PATH into the empty text file. This is for ease of editing.
- Hit ALT+TAB to switch back to the Environment Variables dialog. Under User variables, select the variable 'PATH' and click 'Edit'.
- Highlight all of the text in there, and hit CTRL+X to cut it out. Click OK.
- Hit ALT+TAB to switch back to notepad.
- Hit Enter to go to a new line, then hit CTRL+V to paste the user path.

Now we can work with just the one PATH variable and continue to ensure that the required directories exist in the path in the right order.

- Add these entries if they don't exist (Entries should be separated by semicolons):
  - ;C:\MSYS\1.0\bin
  - ;C:\Program Files\gputils\bin
  - ;C:\Program Files\SDCC\bin

It is STRONGLY recommended to put these directories at the BEGINNING of the PATH Variable! I will show my PATH variables as a demonstration. Your paths may be different, as they will reflect the software installed on your PC - but these will hopefully serve as a good illustration. If you are using an international variant of Windows, then your Program Files directory may have a different name. If so, you can simply replace “C:\Program Files” with “%programfiles%”.

System PATH Variable at the start of this step:

```
C:\Program Files\Windows Resource
Kits\Tools\;%SystemRoot%\system32;%SystemRoot%;%SystemRoot%\System32\Wbem;C:\
Program Files\ATI Technologies\ATI Control Panel;C:\Program
Files\Symantec\pcAnywhere\;C:\Program Files\ActivCard\ActivCard
Gold\resources;c:\program files\MSSQL7\BINN;c:\Program Files\Microsoft
SQLServer\90\Tools\binn\;C:\Program Files\MKVtoolnix;
```

User PATH Variable at the Beginning of this step: (Note that only two of the above installations have added their directories to the path)

```
C:\Program Files\SDCC\bin;C:\Program Files\gputils\bin
```

Combined PATHs pasted into Notepad: (note the System PATH goes first)

```
C:\Program Files\Windows Resource
Kits\Tools\;%SystemRoot%\system32;%SystemRoot%;%SystemRoot%\System32\Wbem;C:\
Program Files\ATI Technologies\ATI Control Panel;C:\Program
Files\Symantec\pcAnywhere\;C:\Program Files\ActivCard\ActivCard
Gold\resources;c:\program files\MSSQL7\BINN;c:\Program Files\Microsoft
SQLServer\90\Tools\binn\;C:\Program Files\MKVtoolnix;C:\Program
Files\SDCC\bin;C:\Program Files\gputils\bin
```

Final PATH Variable after adding necessary directories, and moving them to the beginning of the path: (Note that the directories at the end of the above, have been *moved* to the beginning of this; also

there are semicolons between every entry, there are no duplicated entries, and the whole thing is one big long line of text!)

```
C:\MSYS\1.0\bin;C:\Program Files\gputils\bin;C:\Program
Files\SDCC\bin;C:\MinGW\bin;C:\Program Files\Windows Resource
Kits\Tools\;%SystemRoot%\system32;%SystemRoot%;%SystemRoot%\System32\Wbem;C:\
Program Files\ATI Technologies\ATI Control Panel;C:\Program
Files\Symantec\pcAnywhere\;C:\Program Files\ActivCard\ActivCard
Gold\resources;c:\program files\MSSQL7\BINN;c:\Program Files\Microsoft
SQLServer\90\Tools\binn\;C:\Program Files\MKVtoolnix;
```

Once you have the path edited correctly:

- Hit CTRL+A to select the whole line, hit CTRL+C to copy it, then hit ALT+TAB to switch to the Environment Variables dialog again.
- Under System Variables, select the variable 'PATH' and click 'Edit'.
- This should be empty now (remember you cut it out before?). Hit CTRL+V to paste the nicely edited PATH in there.
- Click OK. You're done!

NOTE: Win98 users may edit the autoexec.bat file to get the path right. If you have followed the previous installations, you'll find a line in autoexec.bat that reads something like:

SET PATH=%PATH%;C:\PROGRA~1\SDCC\bin

Edit that line to read

SET
PATH=C:\PROGRA~1\SDCC\bin;C:\MSYS\1.0\BIN;C:\PROGRA~1\GPUTILS\BIN;C:\MINGW\BIN;%PATH%

This should set your path correct, or at least close to correct. 

make version check

Because you are reading this (or editing it) I'll assume that like many of us, you are a geek... and like many of us, you have various compilers for various languages installed. The copy of make we use, could be preceded by other rogue copies of make, and if this happens, like it has to many of us, your app won't compile.

It may be worth checking this out, it's a quick procedure:

- Click 'Start... Run'.
- Type “CMD” (without the quotes) and hit Enter.
- Paste in the following command:

```
make –version
```

- Hit Enter. You should see the following output:

```
>make --version
GNU Make 3.81
Copyright (C) 2006  Free Software Foundation, Inc.
This is free software; see the source for copying conditions.
There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A
PARTICULAR PURPOSE.

This program built for i686-pc-msys

>
```

The first and last lines must be the same as the above. If not, you need to check your PATH environment variable. make should be running from c:\MSYS\1.0\bin, and it should in fact be the version copied in from the MinGW setup, as above. Other versions of make may (will) not work. We need v3.81.

On at least one occasion, it has been seemingly impossible to bypass another make version - in this case it was the borland make - *Delphi users be aware!!* In this case, the only solution was to supply the full path to make.exe - so anywhere you would type “make”, you now type “c:\MSYS\1.0\bin\make.exe”. This should not normally be necessary, and should be avoided unless you're a big guru like this guy, and can get away with it ;)

MIOS Source Files

These files are required to compile custom applications prior to release.

There are 2 methods

MIOS Base Package

Download the MIOS Base Package here and extract somewhere nice on your hard drive

You need add 2 environment variables to your system. MIOS_PATH and MIOS_BIN_PATH

Right click 'My Computer'
Select 'Properties'
Go to the 'Advanced' tab
Click 'Environment Variables'
Under 'System variables', Click 'New'
Variable name: MIOS_PATH
variable value: <full path to the mios base package folder> eg: C:\MIDIbox\Mios_Base
Click OK
add another new system variable
Variable name: MIOS_BIN_PATH
variable value: <full path to the mios base package 'bin' folder> eg: C:\MIDIbox\Mios_Base\bin

SVN Repository

The repository stores all the latest source files. Including demos, examples and applications.

See the SVN Page for details

MIOS Studio

If you made it this far, you've actually done enough. You can code C or ASM apps in wordpad.exe, run 'make' to create your app.
Please note that notepad.exe is not compatible with GPASM. You should visit The MIOS Studio Page for instructions on how to install MIOS Studio to upload the app.

Core Tools Complete

If you'd like to have a more advanced tool set in your hands, please read on to install and configure an IDE. Currently, there is documentation for:
Code::Blocks - An open-source tool for developing C apps (Recommended for AC-Sim users)
NotePad++ - An open-source tool suitable for either ASM or C apps, or those that mix both.
You may also like to see the documents for the additional tools:
TortoiseSVN
MIOS Studio

From:
https://wiki.midibox.org/ - **MIDIbox**

Permanent link:
https://wiki.midibox.org/doku.php?id=windows_toolchain_core&rev=1282443678

Last update: **2010/08/22 02:21**