Core Toolchain Setup

```
for MIOS32 application development on Windows
```

The QuickStart Guide still needs to be done.

The technical jargon:
This is the *Core Tools* section in a walkthrough on the process of developing MIOS32 Applications, on Microsoft Windows XP, primarily in C, utilising the MIOS32 Operating System. The IDE platform used will be NotePad++ and the applications will be built for the MIDIBox Hardware Platform (MBHP) stuffed with a STM32. C code will be compiled with CodeSourcery G++ Lite, using MSYS to generate and run DOS-Console-based makefile scripts, and MIOS Studio will be used for debugging on MBHP.

Enough of that.
This file is for newbies too, since once the environment is correctly set up, it is very easy to use.

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

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 v1.0.11 has tested OK.

- 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'.

CodeSourcery G++ Lite

This is the modified GNU Compiler Collection, ready for use with the ARM Cortex M3 platform. Since the STM32 is a flavour of Cortex M3, this is what we want to have. Later, once Cortex support makes it into the regular gcc distribution, we can use that instead, but for now, we need the CodeSourcery distribution.

- Download here

- Install using the defaults.
- Answer the question for adding to the PATH variable with “yes” or activate the check box that enables this.

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' (German: 'Arbeitsplatz'), select 'Properties' (German: 'Eigenschaften').
- 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

If you missed adding the path to the CodeSourcery compiler upon installation, add this one too

- ;C:\Program Files\CodeSourcery\Sourcery G++ Lite\bin

It is STRONGLY recommended to put these directories at the BEGINNING of the PATH Variable! For users with different Windows language versions, the “Program Files” directory may be named differently and needs to be adapted.

Merge everything in notepad.

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!

Additional Environment Variables

MIOS32 requires some additional environment variables to be set. We will cover only the settings for using the CORE32 board here:

Before compiling code, set the MIOS32 environment variables:

DOS: (we assume, that the repository has been downloaded to D:\)

```
set MIOS32_PATH=D:\mios32\trunk
set MIOS32_BIN_PATH=D:\mios32\trunk\bin
```

Additional environment variables to configure the environment: (they have to be set, otherwise compiler will fail)

MBHP_CORE_STM32:

```
set MIOS32_GCC_PREFIX=arm-none-eabi
set MIOS32_FAMILY=STM32F10x
set MIOS32_PROCESSOR=STM32F103RE
set MIOS32_BOARD=MBHP_CORE_STM32
set MIOS32_LCD=clcd
```

You can set these in the Windows preferences (just like the PATH variable) by clicking “Add New” and entering the respective pairs. You can do that either in user variables or system variables; I prefer having these in the user environment variables, since that way I can set up settings for other targets in another user account.

Alternatively, you can copy all those set instructions into a batch file and run that before compiling any MIOS32 apps.

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. 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 ;)

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 apps in wordpad.exe, run 'make' to create your app.
I'd recommend taking a look at Notepad++, which is a real nice programming editor.
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:
NotePad++ - An open-source tool suitable for C apps.
You may also like to see the documents for the additional tools:
TortoiseSVN
MIOS Studio

From:
https://www.midibox.org/dokuwiki/ - **MIDIbox**

Permanent link:
https://www.midibox.org/dokuwiki/doku.php?id=windows_mios32_toolchain_core&rev=1282443715

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