Using AVRX with the AVRFreaks GCC distribution

The AVRX RTOS(Real Time Operating System) works well with the free development system Atmel's AVR Studio 3 and the Gnu GCC compiler. The folks at www.avrfreaks.net have a Windows installable package that sets up the GCC compiler for you. But, they only release this software about once a year, and problems and bugs do not get corrected. So, this document should help you get the IDE, compiler, and AVRX installed and working for your projects.

Contents:

1. Install the AVRFreaks.Net GCC distribution

This document will not attempt to instruct you on how to do this installation. The following links will guide you through the process. Be sure to test the GCC installation by creating your own project with something simple, like blinking LEDs or something, so you know the GCC installation is working properly.

Installing and using AVRGCC with AVR Studio:

http://www.avrfreaks.net/Tools/ToolFiles/228/avrgcc_studio.pdf

The document will take you through the installation process and setting up projects in AVR Studio to work with GCC. Again, do a test project to ensure it all works correctly and you understand how it works.

2. Install the most recent AVRX distribution.

Download the latest version of AVRX

Yahoo groups files section: http://groups.yahoo.com/group/avrx/files/AvrX/

This document was written for installing AVRX 2.6e. I installed it in the directory C:\AVRX

3. Changes required to compile the AVRX library

Edit the file C:\AVRGCC\AVR\INCLUDE\ sfr_defs.h

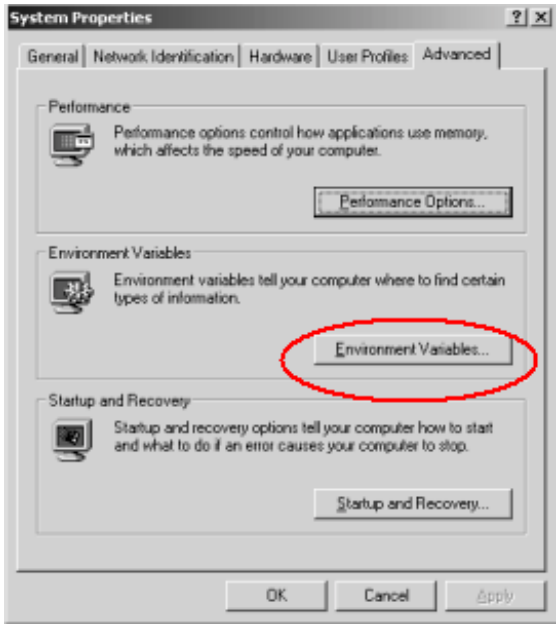
(Note: I've found that using Windows Notepad works well for this, some more advanced editors can tend to change file format slightly and you will get "missing separator" errors when you make the library)

Cut out the following 2 lines(they are located near the bottom of the file and not together)

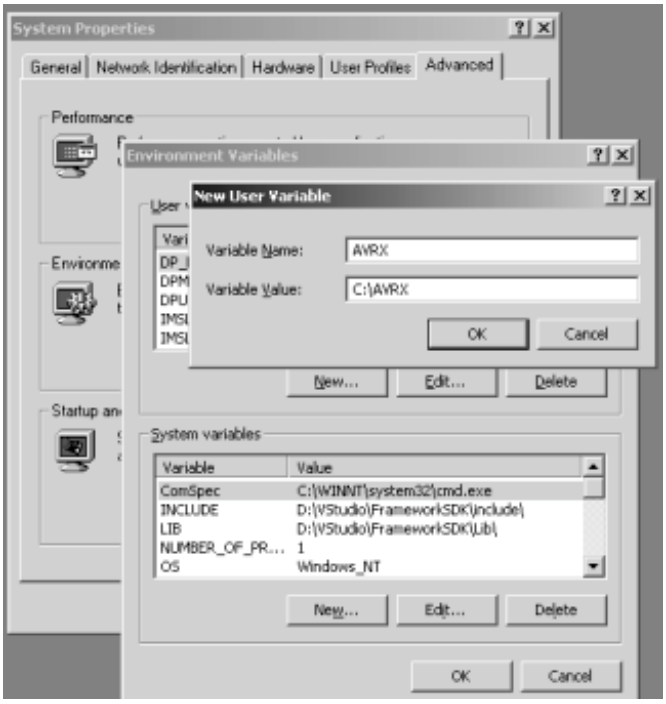
```
#define _VECTOR(N) __vector_ ## N
#define BV(bit) _BV(bit)
```

Now paste these 2 lines at the very top of the file and save. These need to be moved above some of the #ifdef statements for compatibility reasons.

Add a new system environment variable called AVRX. With Windows NT or 2000 you need to go to the System icon in your Control Panel under the Advanced tab:



Then add a new User Variable called AVRX as shown here:



If you have Windows 95/98 just edit your C:\AUTOEXEC.BAT file and add the following line at the end of the file:

```
Set AVRX=C:\AVRX
```

In either case, you should reboot to ensure these new setting are propagated throughout your system. Yes, even on Win-2000. (Windows does, in fact, suck[1])

4. Compile the AVRX library

Now open up a command prompt (DOS) window. Change directories to C:\AVRX\AVRX and type ***make gcc***

If you have followed the instructions, added the new PATH and AVRX environment variables correctly, and rebooted[1] this should now create the file ***avrx.a***

If your make sends you some strange errors, type ***echo %PATH%*** in the DOS window and make sure C:\AVRGCC\BIN is **first** in your path or you may be executing a different make.

5. Changes required for AVRStudio and AVRX projects

Now we have to prepare our projects to include the AVRX headers and library for use in the AVR Studio projects. If you have done the test project in AVR Studio, you already know there is a special ***makefile*** that is required in each project. To add AVRX to your project, you will need to include the headers and the library. At first glance this would seem easy, just add the following to your make file.

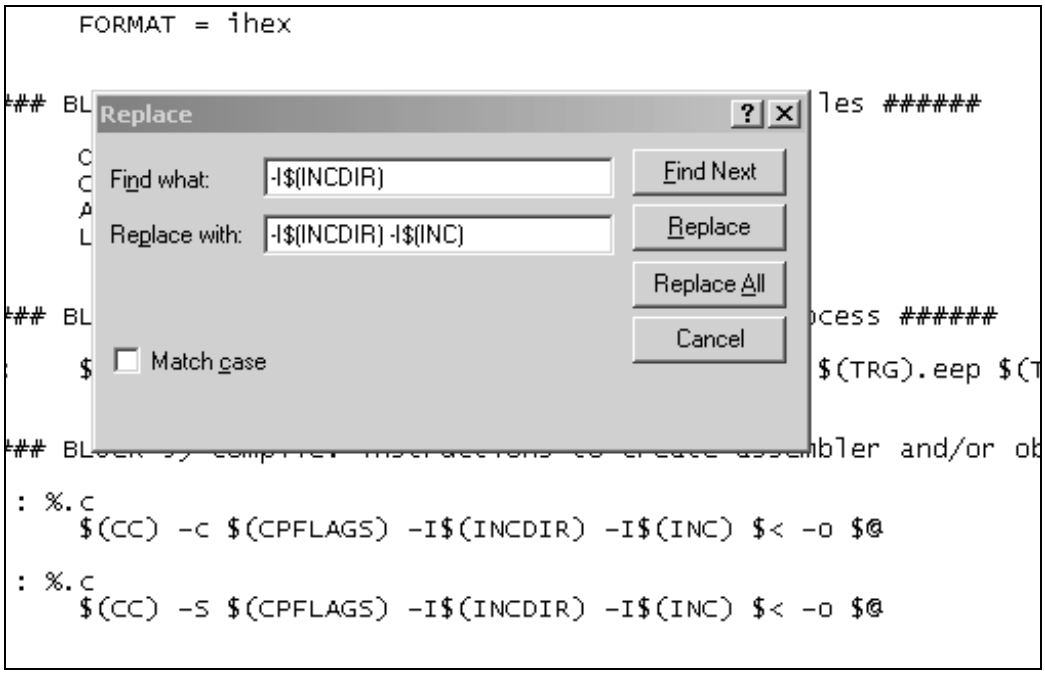
```
#additional libraries and object files to link
     LIB  = $(AVRX)\avrx\avrx.a

#additional includes to compile
     INC  = $(AVRX)\avrx
```

As you can see, the AVRFreaks make file already comes set up to allow adding other headers and libraries. This would be all there is to it… if only it actually worked.

It turns out, that the file ***avr_make*** that is read at the bottom of this make file does not agree that it should include all of these additional items. It does not pay any attention to your INC setting.

So, edit C:\AVRGCC\AVRFREAKS\AVR_MAKE . In Notepad use the ***Edit->Replace*** function as shown here:



6. Other AVR Devices

You will see in the C:\AVRX\AVRX\MAKEFILE that it is building the AVRX kernel for the 8535. If you have done everything else here and it all works, switching devices is quite easy. The most important file to look at is C:\AVRX\AVRX\IOAVR.H (This may not even be necessary in the GCC version of AVRX, but just in case…). If your preferred device is not here, copy a pair of lines for a predefined device like:

```
#elif defined(__AT90Mega161__)
#include "iom161.h"
```

Then, check in the C:\AVRGCC\AVR\INCLUDE directory for the header file for your preferred device and insert the 2 lines as such:

```
#elif defined(__AT90Mega323__)
#include "iom323.h"
```

Now I can compile for my ATMega323!(Somehow I have a ton of these)

Theoretically, you could compile avrx.a for several devices, and save it as avrx323.a and avrx8535.a , etc. Then change the make file to use the correct library in AVR Studio.(Note: I have not tested this thoroughly, someone please confirm or debunk this)

If your preferred device is not found in the AVRGCC include directory it gets much more difficult, and I would suggest you wait until the new version of the compiler is released. Adding new devices is for the hard-core folks and way out of the scope of this document.

NOTE:
At this point the ELFCOF utility still is not fully functional and I cannot get debugging to work in AVR Studio at all. I will update this with anything I learn to help make it work better. But, there appears to be a problem with ELFCOF and external lib files.

PLEASE:
Please, feel free to e-mail me comments and suggestions on this document. I'm sure I've made some mistakes or misunderstood something. Please, let me know so I can fix it. I can be reached at joe@cygan.com

Footnotes:
1. Windows does , in fact, suck. http://www.linuxmafia.com/cabal/os-suck.html