ARM Industrial Module
AIM 711
Software Development Environment

28th November 2003

Contents

1 Installation Guide

First of all it's importent that you install the software under an administration account!

If you've activated the AutoRun functionality of your CDROM-drive, the setup for the AIM Software will start automatically. Otherwise, you must run the file „Installer.exe" from the root of the CDROM.

Now make a choice which of the software components you wish to install. If you haven't installed the AIM Software before, it's necessary that you select all of the components (Figure 1).

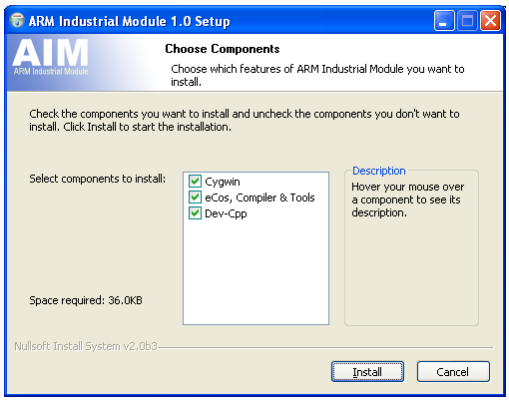

Figure 1: Choose Components

To install the Cygwin environment, you must install it from the „Local Directory" (Figure 2).

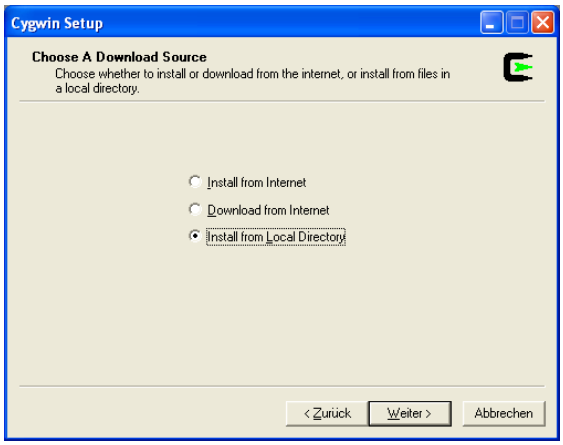

Figure 2: Install from Local Directory

For the „Root Directory" it's preferable to use the proposed directory. *And ensure that the directory name don't contain any spaces!*

In the next step you specify the „Local Package Directory". This is the directory where the files for the installation are located. In our case, they reside in the subdirectory „cygwin" on the CDROM (Figure 3).

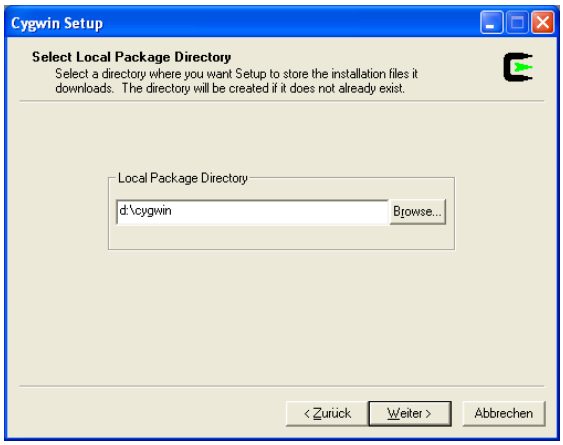

Figure 3: Local Package Directory

Now you can select packages you want to install. You only need the default- and the „AIM Support" package for the development (Figure 4).

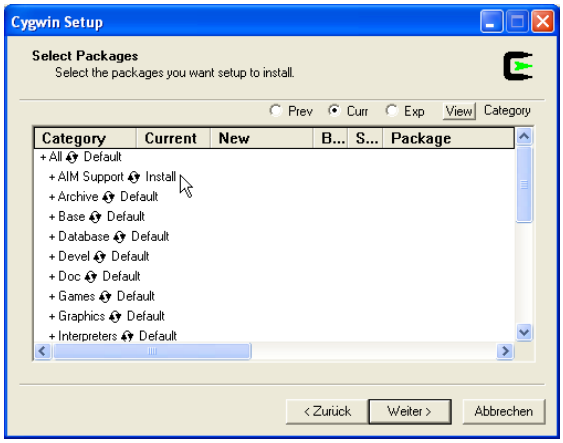

Figure 4: Select Packages

Until the installer finished the installation of cygwin, the second component „eCos, Compiler and Tools" will be prepared for installation. First of all, you choose a destination directory for the components. The default proposal is always a good choice (Figure 5). *And again, make sure that the directory name don't contain any spaces!*

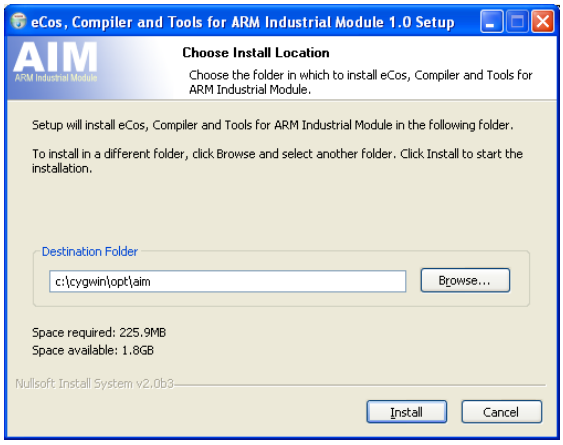

Figure 5: Choose Install Location

Finally, the installation of „Dev-C++" begins. You have the possibility to choose what components and options you want to install from the distribution (Figure 6). After the installation of „Dev-C++" is finished, the program starts automatically for the first time. *Be sure, that you close „Dev-C++" before you finally close the „Installer"!*

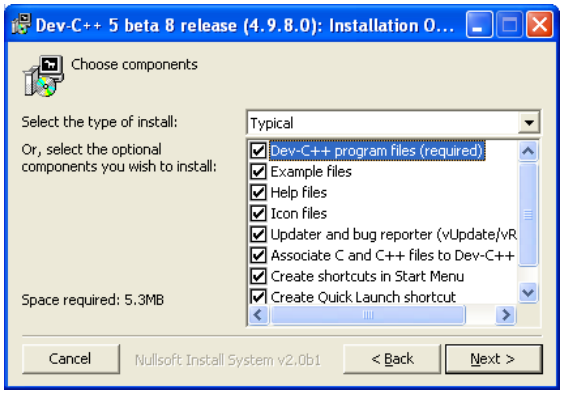

Figure 6: Dev-C++ Installation

After you've installed the „AIM Software Components", you're ready for the „ARM Industrial Module" developing.

Good luck :)

2 Overview of the installed Software

2.1 Operating System eCos

eCos[1] is an open source, configurable, portable, and royalty-free embedded real-time operating system.

One of the key technological innovations in eCos is the configuration system (Figure 7). The configuration system allows the application writer to impose their requirements on the run-time components, both in terms of their functionality and implementation, whereas traditionally the operating system has constrained the application's own implementation. Configuration also ensures that the resource footprint of eCos is minimized as all unnecessary functionality and features are removed.

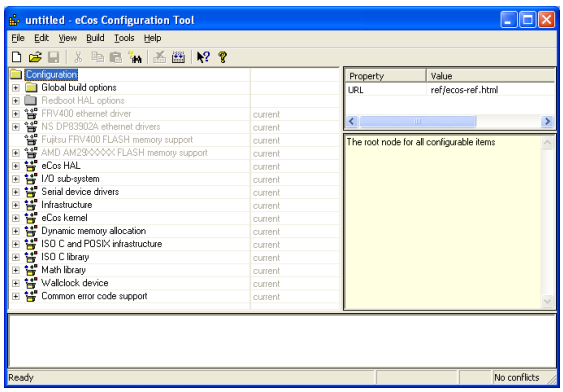

Figure 7: eCos Configuration Tool

eCos has been designed to support applications with real-time requirements, providing features such as full preemptability, minimal interrupt latencies, and all the necessary synchronization primitives, scheduling policies, and interrupt handling mechanisms needed for these type of applications.

The relevant files reside in the subdirectory „ecos" of the „eCos, Compiler and Tools" installation path. There you will find examples and all the source code for eCos. But normally you must not care about the source, because everything related to the AIM is already implemented (i.e. serial, flash and I^2C drivers). You will use the top level API for these components in your application instead.

[1] „embedded Configurable operating system"

2.2 Cross Compiler and Debugger

The gcc („GNU Compiler Collection") includes the cross compiler „arm-elf-gcc.exe" and other compiler tools like:

- arm-elf-objcopy.exe - allows transformation between different file types (i.e. ELF[2] to binary)
- arm-elf-readelf.exe - displays extended information on ELF files (i.e. section sizes or symbol tables)
- arm-elf-strings.exe - displays all strings in files
- arm-elf-strip.exe - removes symbols and sections from files

You find these tools in the subdirectory „arm-elf\bin" of your „eCos, Compiler and Tools" installation path. There you will also find the cross debugger „Insight" (Figure 8).

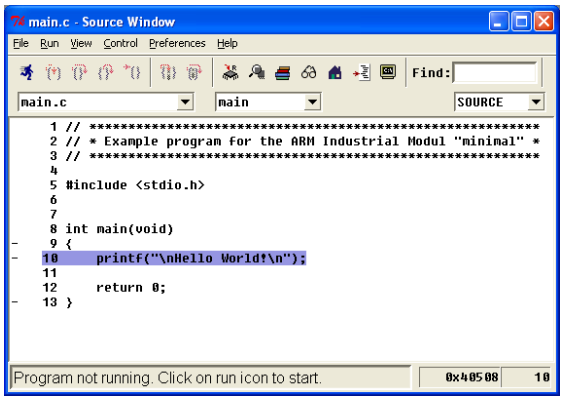

Figure 8: Insight Debugger

„Insight" is a version of GDB[3] that uses Tcl/Tk to implement a graphical user interface. The interface consists of several separate windows, which use standard elements like buttons, scrollbars, entry boxes and such to create a fairly easy to use interface. The windows contain things like the current source file, a disassembly of the current function, text commands (for things that aren't accessible via a button), and so forth.

[2] today ELF is the default binary format on operating systems such as Linux, Solaris 2.x, and SVR4
[3] GNU Project debugger, allows you to see what is going on 'inside' another program while it executes

2.3 Dev-C++

Dev-C++ (Figure 9) is a full-featured and free „Integrated Development Environment" (IDE) for the C/C++ programming language.

These are the minimum requirements of Dev-C++:

- Microsoft Windows 95, 98, NT 4, 2000, XP
- 100 Mhz Intel compatible CPU
- 30 MB free disk space

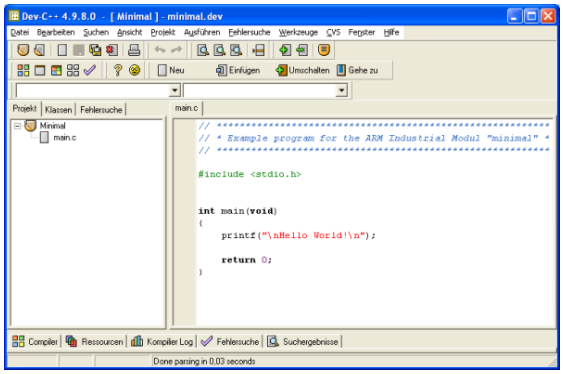

Figure 9: Dev-C++

Dev-C++ features are:

- Support GCC based compilers
- Integrated debugging (with GDB)
- Class Browser
- Debug variable Browser
- Code Completion
- Function Listing
- Project Manager

2.4 AIM-Tools

The main important tool is the „AIM Manager" (Figure 10) which allows you to configure the bootloader „RedBoot" in a comfortable way. After you're connected to the destination set the values of your choice and save them. Programs for the AIM can also be uploaded - you'll see all the images which exists on the module in the listbox. There you can delete one of them or start any. If you do so, the debugging output is shown in the console window. This window also allows you to type any „RedBoot" command directly and see the results.

There is also the possibility to start the tool with a macro file, like „AIMManager.exe macro.rbm". You will find such a file in the „aimm" directory under your AIM installation path. All commands are described in this file, so take a look.

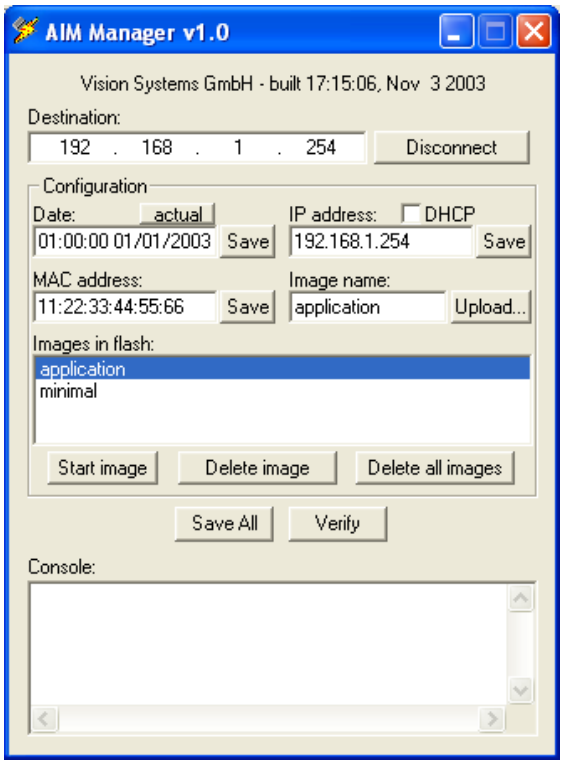

Figure 10: AIM Manager

The second tool is the „AIM Project Templates". It allows you to install preconfigured projects into a desired directory. It's an easy to use and self explaining tool.

3 Environment of the AIM

3.1 Overview of RedBoot

RedBoot is a complete bootstrap environment for embedded systems. Based on eCos, RedBoot inherits the qualities of reliability, compactness, configurability, and portability.

RedBoot allows download and execution of embedded applications via serial or ethernet. It can be used for both product development (debug support) and in deployed products in the field (flash update and network booting).

Ethernet download and debug support is included, allowing RedBoot to retrieve its IP parameters via BOOTP or DHCP, and program images to be downloaded using TFTP. Images can also be downloaded over serial, using X- or Y-modem.
*The standard configuration of the module's RedBoot is the static IP „**192.168.1.254**"!*

RedBoot can be used to communicate with GDB to debug applications via serial or ethernet, including the ability to interrupt a running application started by GDB.

An interactive command-line interface is provided to allow management of the Flash images, image download, RedBoot configuration, etc., accessible via serial or ethernet. For unattended or automated startup, boot scripts can be stored in Flash allowing for example loading of images from Flash or a TFTP server.

3.2 JTAG work mode

With the JTAG support built into a target CPU, you can use this interface to download code, execute it, and examine register and memory values. These functions cover the majority of the low-level functionality of a typical debugger. An inexpensive remote debugger (i.e. „OCD Commander" 5.4) can be run on a workstation to assist with software debug.

For this type of debugging, you need an external JTAG-adapter like the „Wiggler" from Macraigor Systems 5.4.

3.3 AIM Service Adapter

The AIM Service Adapter gives you the possibility to watch the (debug-) output from RedBoot and the application, over the ARM's first internal serial port. Therefor, make a connection to the AIM, over a terminal program like HyperTerminal, with a nullmodem cable.
*The connection parameters are **38400** baud, **no parity** and **one stop bit!***

4 AIM Development by way of an Example

4.1 eCos Kernel build

First of all, you must start the „eCos Configuration Tool" from the „Programs\AIM"-folder or from the Dev-C++ menu „Tools". Then you can start by opening the templates window via „Build->Templates" (Figure 11). Choose the desired target (i.e. ARM Industrial Module AIM 711). The listbox „Packages" gives you the possibility to select a predifined package. For this example application, the package „default" is the right choice. Whenever you need other packages (i.e. for network support), you can take another predifined package or add one by selecting it from „Build->Packages".

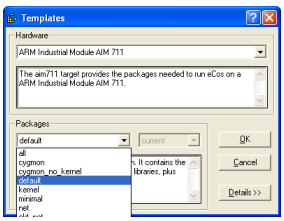

Figure 11: Templates

Now you can save your configuration over „File->Save As..." in a desired project path. Create the build trees over „Build->Generate Build Trees". Then comes the most important step, the compilation of your eCos kernel, over „Build->Library". After the compilation is done, you will find the needed files in the created subdirectories of the project path. The only relevant directory is „...._install". There resides the library, linker script and include files.

The description above is for a better understanding to „What can I do if I need a special solution?". Normally, you use one of the precompiled kernels from the „AIM Project Templates". Install these templates with the template tool (Figure 12), which you find in the „Programs\AIM"-folder and the Dev-C++ menu „Tools". These templates also includes the configured project file for Dev-C++. But for better understanding, we create one - step by step - in the next subsection.

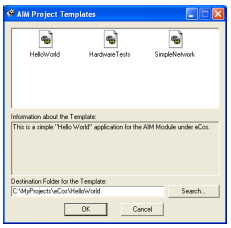

Figure 12: AIM Project Templates

4.2 Creation and Configuration of the Dev-C++ Project

You create a Dev-C++ project over „File->New->Project...". The type of the application is „Console" and will be programmed in „C". After you've found a name for your project, save it in the directory where you've created the eCos kernel before (Figure 13).

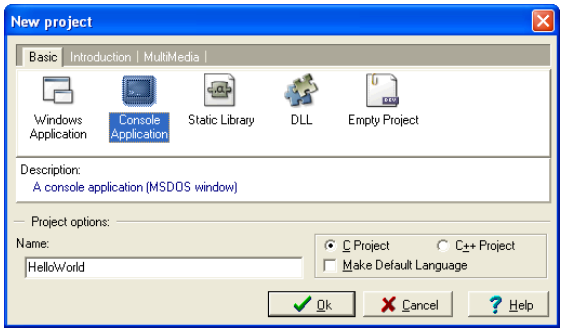

Figure 13: New project

Then configure some project options over „Project->Project Options". Choose „arm-elf" for the compiler set and turn the linker option, which is for not using the standard startup files and libraries on (Figure 14).

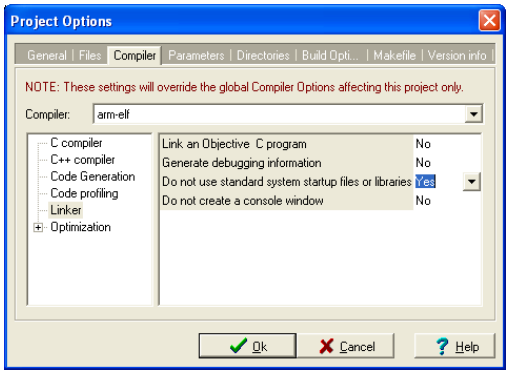

Figure 14: Project Options

After that, you must add some additional command-line options under the property page „Parameters". For „Compiler" add „-mcpu=arm7tdmi -mbig-endian -mno-short-load-words -g" and for „Linker" add „-mcpu=arm7tdmi -mbig-endian -nostartfiles -Wl,–gc-sections -Ttarget.ld -nostdlib". Now it's time to insert the library- and include directories under „Directories". For the libraries you must add the subdirectory „..._install\lib" of your new project path. And for the include directories please insert the „..._install\include" path of the project directory.

4.3 Creation and Compilation of „Hello World”

Now insert the „Hello World” code into the main function like:

```
#include <stdio.h>

int main(int argc, char *argv[])
{
    printf("Hello World\n");
    return 0;
}
```

Listing 1: "Hello World" application

After all, the time has come to compile the example application. You start compiling over „Execute->Compile” and can watch the process in the „Compile Log” window below (Figure 15). If you're lucky, no error will be occour and you can go on with the next section.

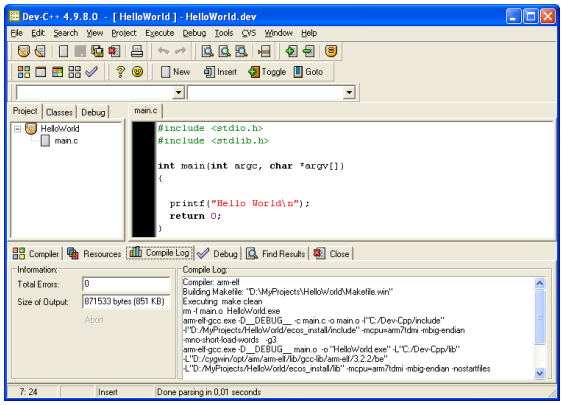

Figure 15: Dev-C++ Compile Process

4.4 Application upload and debugging

For remote debugging use the debugger „Insight" („Tools->Insight Debugger"). Before we go on with the next step, please prepare your AIM. You can be sure that everything is ok by watching RedBoot's output over the „AIM Service Adapter" 3.3.

The first step is the configuration of the remote target „File->Target Settings..." (Figure 16). Choose „Remote/TCP" for the target type, insert the IP address from your RedBoot (default: **192.168.1.254**) and the destination port (default: **9000**).

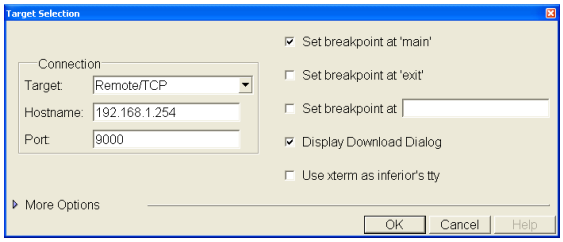

Figure 16: Target Selection

Now it's possible to run the application over „Run->Run". After the upload is finished, a standard breakpoint appears at the main function. Open the console window „View->Console" and let the program go on with the printing of „Hello World", over „Control->Next". You see the output in the console window.

That's the time that you've created, compiled and debugged your first AIM application. Now you know the basics for your further work with the AIM. If you like to get an bigger overview of developing, it's a good choice to take a look at the examples from the „AIM Project Templates".

5 Resources and further Information

5.1 VisionSystems

Here you will find all information related to the „ARM Industrial Module". Take a look at the website (http://www.visionsystems.de) for further products and software updates.

5.2 RedHat

Documentation related to RedBoot and eCos is available over RedHat's „Projects Homepage" (http://sources.redhat.com). There you find all answers to the questions that comes by the way of developing. And if you don't find anything for your problem, post it to one of the mailing-lists. Extendend links to developing tools like gcc and Insight are also placed on this site.

5.3 Bloodshed Software

Information and software updates of Dev-C++, you will find on the homepage of Bloodshed Software (http://www.blodshed.net/dev/index.html).

5.4 Macraigor Systems

Macraigor Systems (http://www.macraigor.com) is the main manufacturer of the JTAG-adapter „Wiggler". They also offer a software for JTAG debugging („OCD Commander") and other tools like a flash programming software - also over JTAG.