Compiling MQX Libraries and Demos for the KDS Toolchain.

Introduction

Official support for MQX in KDS is slated for August of this year (2014), but what if you want to start building up your libraries now?

Through Mingw and the files included in MQX 4.1, we can compile libraries and programs using KDS' gcc_arm compiler.

This is essentially a modification of the process used to compile MQX on a generic gcc_arm compiler. Original documentation can be found in:

<MQX_ROOT_DIR> \doc\tools\gnu\MQX_GNU_Getting_Started.pdf

I will try to explain a bit more of the setup process and the file changes that must be made to correctly compile, along with debugging your executable in processor expert.

This guide is written for Windows.

Setup

You will need:

- Mingw installed, along with mingsys-SED. This can be found at http://sourceforge.net/projects/mingw/
    - Make sure 'c:\MinGW\bin' has been added to system 'PATH' variable
    - SED is part of a cmd improvement suite called mingsys. You can install the whole suite, but we only need SED
- KDS installed
- A suitable text editor that allows find and replace of text in all files of a selected directory. I recommend Notepad++ and it is what I will use in the tutorial.

MQX file changes

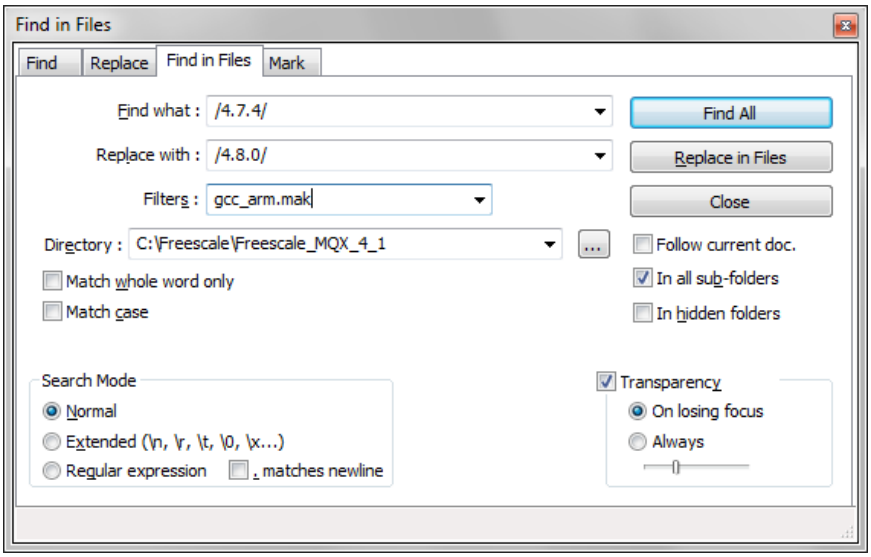
The original makefiles and batch scripts were created for gcc_arm v4.7.4, but KDS 1.0.1 is using 4.8.0. This means some directory paths have changed and these changes must be updated within MQX's files.

First, let us define the toolchain for the builder to use, in <MQX_ROOT_DIR> \build\common\make\global.mak, modify the following definition:

```
ifeq ($(TOOL),gcc_arm)
     TOOLCHAIN_ROOTDIR = <KDS_INSTALLION_DIR>\toolchain
Endif
```

Next we need to update the directory listings for gcc_arm components within its root directory. This comes down to replacing all instances of "/4.7.4/" with "/4.8.0/" in all 'gcc_arm.mak' files (you could choose to only edit the .mak files of boards you intend to use, but it's just as easy to replace all instances).

In notepad++, this is simple process of running a "Find in Files".



The files themselves will be located in directories associated with building mqx for each platform:

```
<MQX_ROOT_DIR> \mqx\build\make\<SPECIFIC_PLATFORM>\tools\gcc_arm.mak
```

Along with some entries within application builds:

```
<MQX_ROOT_DIR>
\mqx\examples\<project>\build\make\<project>_<platform>\tools\gcc_arm.mak
```

Lastly, specifically for demo application builds, there is one more directory change and one exclusion to make: remove 'armv7e-m/' from directory paths and comment out 'libnosys.a' (I must clarify I don't understand the importance of libnosys and couldn't find a suitable replacement with KDS' gcc_arm toolchain. I have build hello2 successfully without it, but maybe other projects will require it)

Again, this is a simple 'Find in Files', replacing "armv7e-m/" with no characters, then replacing the line:

```
RT_LIBRARIES += $(TOOLCHAIN_ROOTDIR)/arm-none-eabi/lib/armv7e-m/softfp/libnosys.a
```

With

```
#RT_LIBRARIES += $(TOOLCHAIN_ROOTDIR)/arm-none-eabi/lib/armv7e-m/softfp/libnosys.a
```

to comment it out.

Building Libraries

The build process is now incredible easy, as we can use the batch scripts provided in MQX to build our libraries.

Either use "<MQX_ROOT_DIR> \build\<BOARD>\make\ build_gcc_arm.bat" to create the full set of libraries for the chosen board (bsp, psp, usb, shell, mfs,rtcs), or use any "build_gcc_arm.bat" found in the component and board area of your choice. These will automatically build the libraries and move the created libraries and code to ""<MQX_ROOT_DIR> \lib\<BOARD>.gcc_arm\"

Note: when building full libraries the system will sometimes throw up errors when calling 'mkdir', saying that the folder structure already exists. I have found if you re-run the script again it will not complain the second time and behaves fine.

Building Examples

So far only tested 'hello2'on twrk60n512, but principles should apply to all boards and projects

Again, we use the batch scripts provided in MQX, found in locations such as:

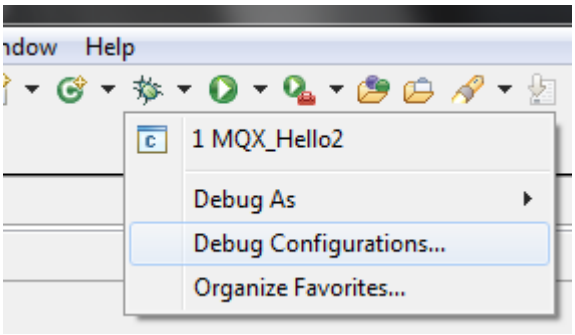
<MQX_ROOT_DIR>\mqx\examples\<PROJECT>\build\make\<BOARD>\build_gcc_arm.bat

Running these batch scripts will create another folder within the directory, called "gcc_arm", in which you can find "initflash_debug" and "initflash_release". Inside each of these folders is your compiled application.

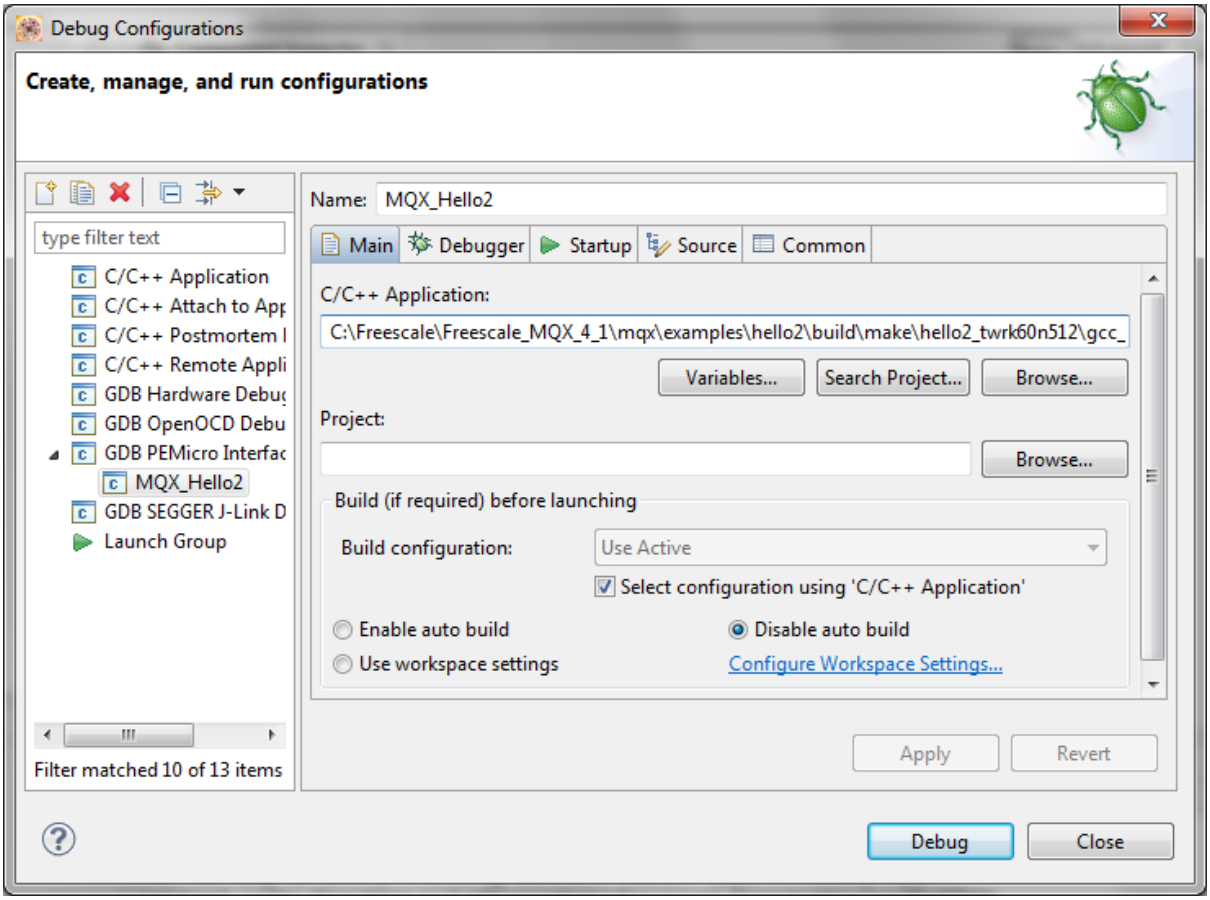
Debugging your examples

Now let's actually do something in KDS: debugging the examples and showing that they work.

For this you need to have a workspace (otherwise KDS won't run), but you can close any open projects if you wish. Then Click on the debug menu and select "Debug Configurations..."



Double click on your chosen debug interface and under the option for "C/C++ Application", choose browse and select the '.elf' file from the 'initFlashdebug' folder of your application.



I hope this helps some people who need to get MQX running through KDS before the official support becomes available. For information on when MQX will be supported on KDS, see https://community.freescale.com/docs/DOC-95477