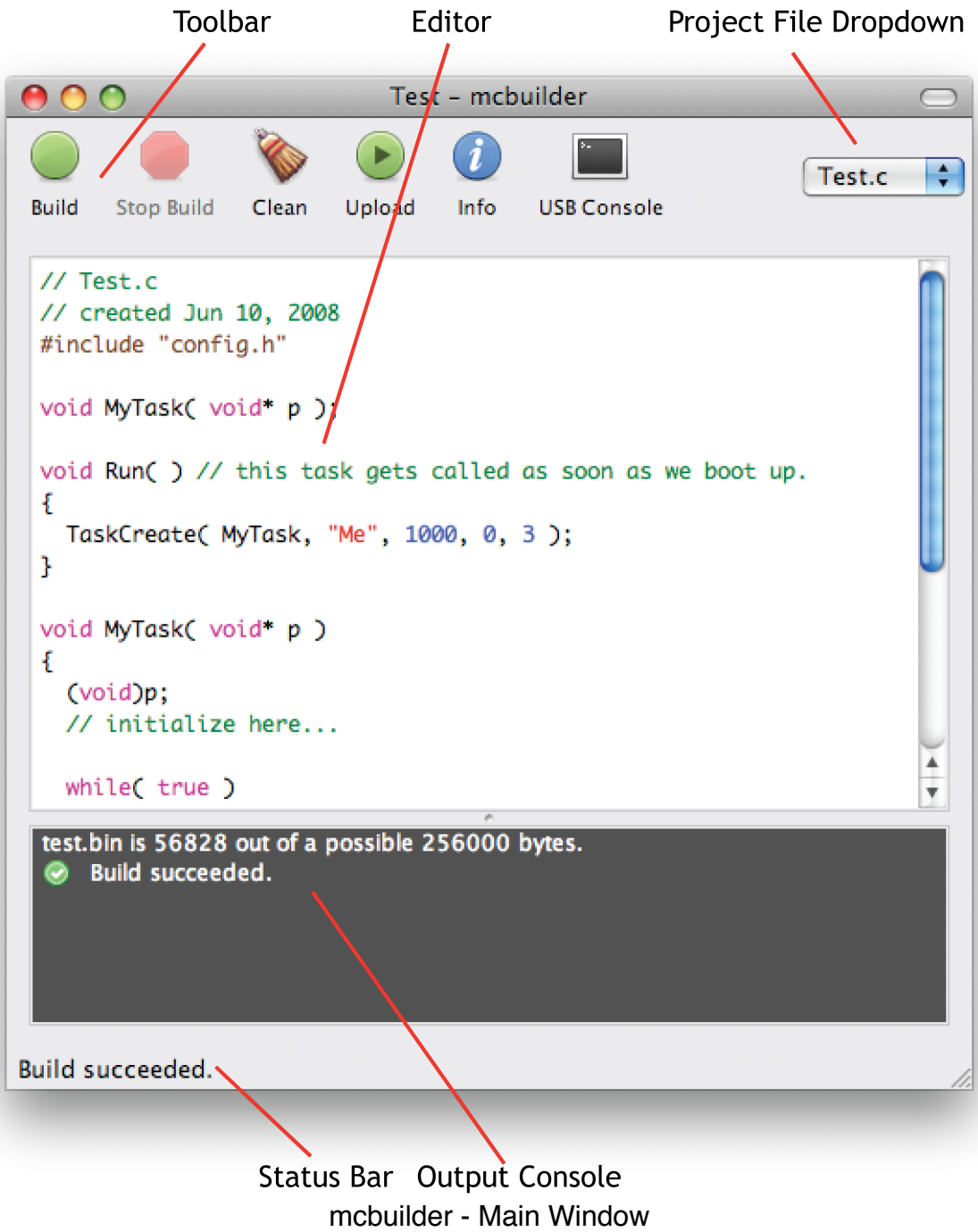
mcbuilder User Manual

Overview

mcbuilder is an application that makes it simple to build programs for the Make Controller. It provides a very simple IDE (integrated development environment) - including firmware uploader, Makefile and configuration file generators - and project management. Below is a screenshot of the main window, with pointers to the main features.



mcbuilder - Main Window

Start Building

Now, on to creating your first project.

Explore the examples

The easiest way to get started is with the example projects. Under the **Projects** menu item, select **Examples** to browse the list of the available examples, grouped by system. This will give you a good idea of what's involved in creating a simple project, and also provides plenty of sample code to borrow in starting your own projects. Try building and uploading some of the examples to get a feel for the workflow.

Create a new project

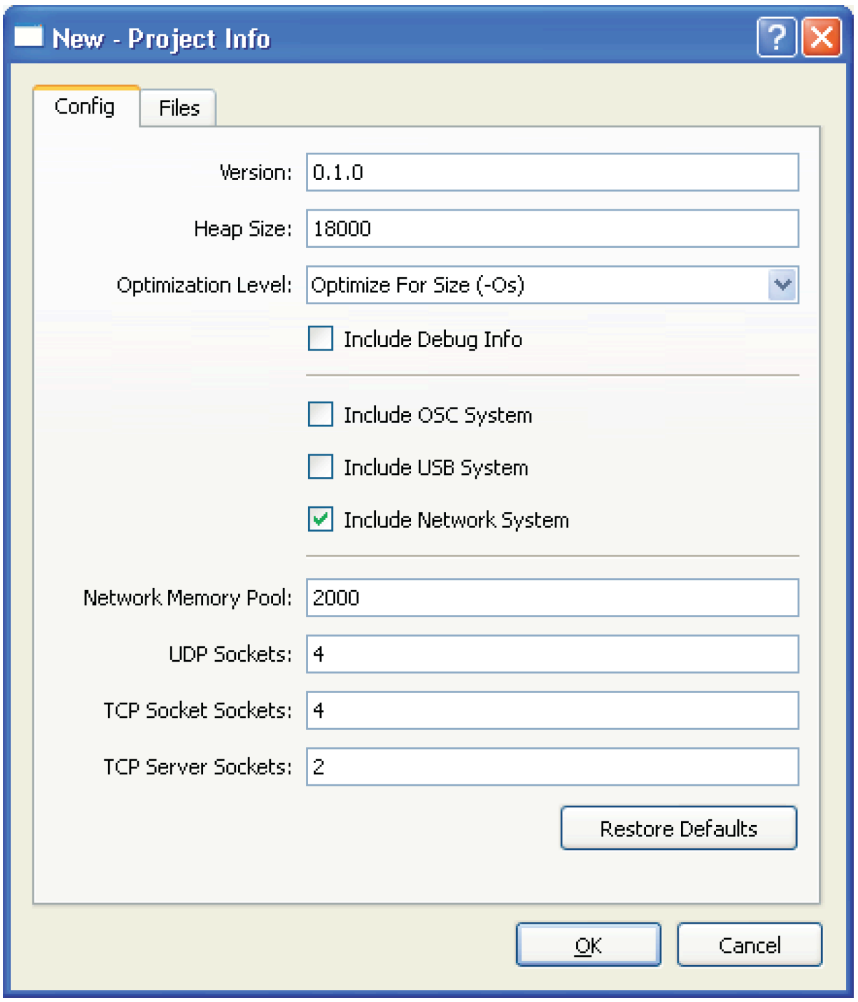
To create your own project you can either save a new copy of an existing project, or create a new one from scratch. To save a copy of an existing project, open it and select **Save Project As…** from the **File** menu. A dialog will pop up allowing you to rename the project. This will rename the project file, main

To create a new project from scratch, select **New Project…** from the **File** menu. A dialog will give you the option to name your project and it will be created for you with the default settings, and some stubbed out source code to get you started.

Build your project

Building a project is very simple - just click the "build" icon, or select **Build** from the **Project** menu. This will start compiling the code for your project. If there are any errors or warnings, you'll see them printed out to the output console of the main window. When your build has succeeded, you can then upload it to your board to run your new program.

Configure your project
Each project can be configured to include different systems, libraries, and allocate resources differently. These per-project settings can be accessed in the **Project Info...** dialog, available under the **Project** menu, or by clicking the Info button in the main window.



The Project Info dialog

Upload your project
To upload your project to your board, you must first ensure your Make Controller has been erased. To do this, you must short together the two pins of the jumper labeled **ERASE** on the Application Board. This can be done with any piece of metal - a screwdriver, or paperclip, or whatever you have handy. Then unplug and replug the board to complete the erase process.

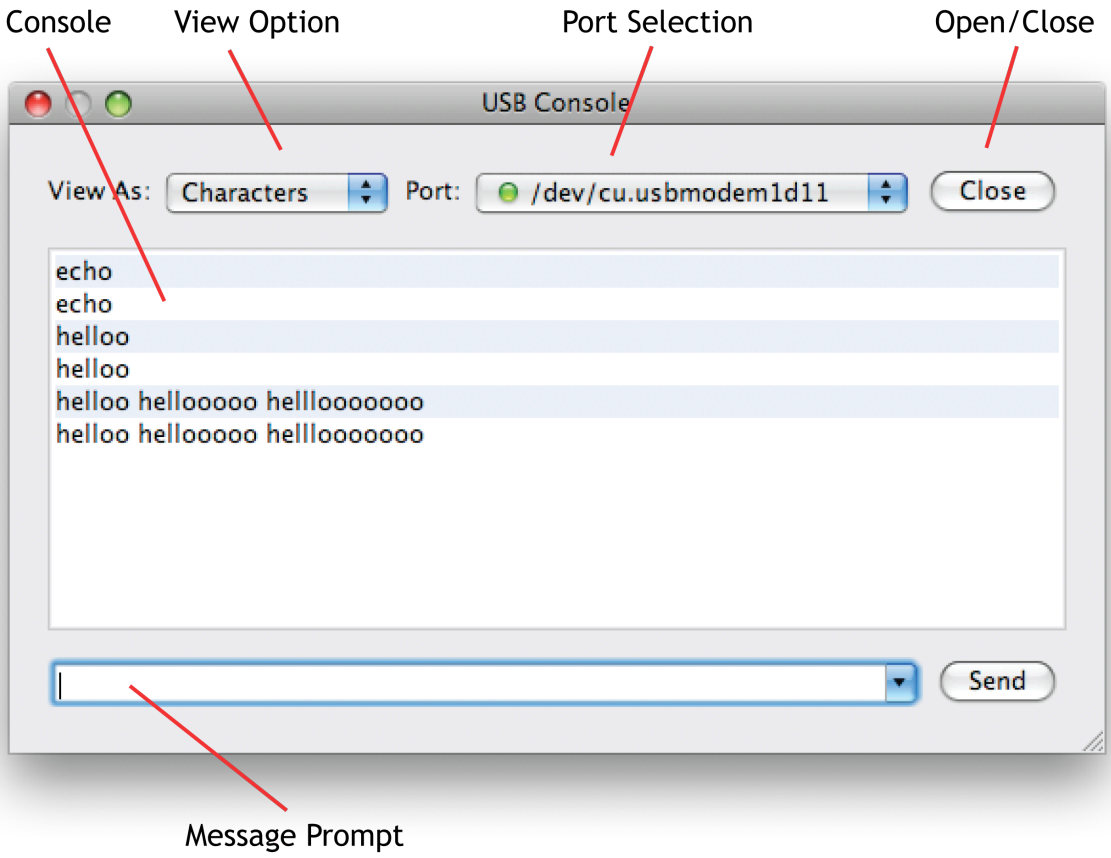
Once this is done, click on the “upload” button or select **Upload Project to Board** from the **Project** menu. mcbuilder will build the current project before trying to upload it - if there are errors, it will prompt you to fix them before continuing with the upload.

Alternatively, if you just want to upload a .bin file to your board (not necessarily the current project), select the **Upload File to Board…** option in the **Project** menu and navigate to the .bin file you'd like to upload.

USB Console

The USB Console is a dialog that makes it easy to experiment with and debug your firmware over the USB port. It can print out USB data sent back from the board, and you can send arbitrary data to the board.

To open the USB Monitor, click **USB Console…** under the **View** menu item, or click the console button in the main window. A list of available Make Controllers are listed in the 'ports' dropdown at the upper right. The red or green light indicates whether the connection to the device is open or closed. Messages that you send have a blue background, and responses from the board have a white background. You can view the data exchanged with the board either as characters, or as hex values - select the desired option from the **View As** dropdown.



The USB Console

Tools Configuration

On OS X and Windows, mcbuilder comes with all the tools you need to build firmware for the Make Controller, so you'll probably never need to deal with this section. On

other Unix variants however, we do not ship the tools with mcbuilder since we don't know the specifications of your system in order to provide pre-built binaries. In this case, you can install and build the appropriate tools and tell mcbuilder where they are.

Setting custom paths for tools

There are 3 sets of tools mcbuilder needs to know about for its build process - the **make** utility, the **arm-elf-gcc** toolchain, and **sam7utils**. We'll assume you have make installed on your system.

The arm-elf-gcc toolchain can be downloaded from a variety of places, but sites that are known to have up to date sources are:

- http://www.gnuarm.com
- http://www.mikrocontroller.net/articles/ARM_GCC_toolchain_for_Linux_and_Mac_OS_X

sam7utils is a utility that uploads new firmware to the Make Controller, and it can be found at http://oss.tekno.us/sam7utils/download.php. It's recommended to use the latest version.

Once you have installed these tools, open up the **Preferences** dialog in mcbuilder and enter the path to these packages in the appropriate fields. You can always test by trying to build and upload an example project.

Libraries

The firmware source code for the Make Controller is divided into the core and a variety of libraries that provide additional functionality. To browse the list of available libraries, select **Libraries** under the **Project** menu - for each library you should have the option to import it to the current project, or view documentation for it.

When mcbuilder compiles your project, it scans your project files for any declarations in the form of **#include "somelibrary.h"**. If **somelibrary** matches any of the available libraries, it's pulled into your build. To exclude it, simply remove the #include directive in your source file(s).

Installing a New Library

A library simply takes the form of a directory. To install it, drag the directory to the **libraries** directory within your mcbuilder installation. Always consult the library's author for more details, but usually that's it!

Creating Your Own Library

If you've created some code that is generally useful, please consider packaging it as a library and sharing. This might be code that makes it easy to interface to a specific piece of hardware (a GPS device, or Bluetooth device, etc) or provide additional software functionality.

A library consists of only a few documents - your source code, and an XML library file that provides information about it. The easiest way to get started is to copy the **library_template.xml** file from within the resources/templates directory in your mcbuilder install and start to modify it. You can also check the existing libraries for examples - they're pretty simple.

Edit the XML file to list each of the .c files included in your library, specifying the path relative to your library's root directory. Provide a link to documentation for your library - this is important! The link can be either to a local or online HTML file or a PDF. The 'display_name' corresponds to how mcbuilder will display the name your library in the Libraries menu.

More Information

This section touches a bit on how mcbuilder works - not required reading, but you may be interested.

mcbuilder uses the 'make' utility to create its builds. Each time you press 'build', mcbuilder does a few things:

- makes sure there's a 'build' directory inside your project folder
- creates/updates the Makefile there
- creates/updates the config.h file in your project folder
- runs 'make' on the newly created Makefile

This means that you can use mcbuilder just to generate Makefiles if you want to ultimately continue working in another development environment. This can be handy for creating new projects which otherwise involves a lot of manual copying/pasting/modifying.

mcbuilder maintains a project file for each project that contains the following info:

- the source files to be compiled for that project
- the include directories to be specified for that project
- the configuration options as specified in the project info dialog

This is stored in an XML file that has the same name as the project directory. So a project named **MyProject** will have a **MyProject.xml** file in it. You can open up that file and edit it manually if you want to remove/include files from your project. The structure of the document is reasonably self-explanatory if you have any experience with XML.

Support

Support for mcbuilder and other MakingThings software is offered primarily through the forum on the MakingThings website, and via live chat in the MakingThings IRC channel on freenode. Come join us!

Forum: http://www.makingthings.com/forum

IRC: #makingthings on irc.freenode.net