“HOW TO GET THE CODE SIZE OF MQX PROJECT”

By Oralia Soledad Godínez Vega March, 2014

Freescale

The purpose of this document is to indicate step by step how to get the memory footprint of a MQX project.

The Freescale MQX RTOS includes the tool Codesize.exe, this document is a guide for people who want to use this tool in order to know the memory RAM/ROM utilization of a MQX project.

INTRODUCTION

The Freescale MQX Real-Time Operating System (RTOS) provides real-time performance within a small, configurable footprint. This RTOS is designed to allow you to configure and balance code size with performance requirements.

The Freescale MQX RTOS was designed for speed and size efficiency in embedded systems. The RTOS delivers true real-time performance, with context switch and low-level interrupt routines hand-optimized in assembly, and can be configured to take as little as 12 KB of ROM and 2.5K RAM on ColdFireV2, including kernel, 2 task applications,1 LW Semaphore, interrupt stack, queues, and memory manager.

MQX Example Applications

MQX provides demo applications that you can find in the folder demo (<install_path>\Freescale MQX 4.X\demo).
The demo applications were written to demonstrate the most frequently used features of the Freescale MQX™ RTOS.

In addition to these demo applications, there are simpler example applications available in MQX, RTCS, MFS and USB directories that you can use like a guide to develop your own application. The examples were compiled and tested using the evaluation boards and development tools that MQX is supported.

You can find the simple example projects in the following directories:

```
<install_path>\Freescale MQX 4.X\mqx\examples
<install_path>\Freescale MQX 4.X\rtcs\examples
<install_path>\Freescale MQX 4.X\ mfs\examples
```

<install_path>\Freescale MQX 4.X\usb\device\examples
<install_path>\Freescale MQX 4.X\usb\host\examples
<install_path>\Freescale MQX 4.X\usb\common

BENCHMRK FOLDER

Freescale offers a wide range of microcontrollers and microprocessors for all user's requirements. For a new project it is imperative to consider the application in order to select the device and the operating system that best fits your needs.

The memory size limitation is an essential factor that you need to take into account when you want to develop your own application. This is the reason that Freescale MQX™ RTOS provides a benchmark report.

Freescale MQX™ RTOS includes a project that helps to determinate the timing and code sizing of any project. You can find this in the path

<install_path>\mqx\examples\benchmrk\codesize\reports

The code sizing depends on the kind of functions used in the final application. The benchmark project includes almost all the main components in MQX. This will provide the code size of these components.

Once a user has the final project compiled it is possible to get the final code sizing by using the Codesize tool. It is located in the path <install_path>\tools\codesize.exe. This tool reads the map file generated by the CodeWarrior or IAR linker and calculates the memory usage. Customers can use this as reference of how much memory is needed for each MQX component.

Remember that it doesn't mean that all of them will be used in custom project. Only if you add and use the components to your own project then will be added to the code sizing.

The html report does not show the RAM usage by default, it only displays the ROM usage. The tool is also able to calculate RAM used by static and global variables but this is not so useful information. Most of the RAM is allocated dynamically, so you need to use a MQX memory dump screens in Task-aware Debugger within CodeWarrior or IAR IDEs to analyze the memory usage in a running application.

MQX CODESIZE SCRIPT

Codesize.exe

The script analyzes the generated xMAP file produced by CodeWarrior or IAR Embedded Workbench build tools and creates codesize report in an HTML format. The source code of the Codesize script is written in PERL language and is also available in the codesize example folder.

The below tools are supported:

- CodeWarrior for MCU/ColdFire 6.3 (use -c cwcf6)
- CodeWarrior for ColdFire 7.2 (use -c cwcf7)
- CodeWarrior for mobileGT 9.2 (use -c cwmpc9)
- CodeWarrior version 10 for ColdFire platforms (use -c cwcf10)
- CodeWarrior version 10 for Kinetis platforms (use -c cwarm10)
- IAR Embedded Workbench for ARM v6.10 or later (use -c iararm6)
- ARM-MDK Keil uVision v4.23 or later (use -c uv4)

In order to create a new report it is necessary to considerate the following options:

- -M ... Using this option, it is posible to print detailed MAP file analysis report.
- -t ... With this option, it is possible to dump text output to console
- -W ... This option enables warning messages
- -o <FILE> ... Using this option customer can be able to specify report output file name
- -n <NAME> ... Customer can use this option to have an alternative MAP file name displayed in the report
- -c <FORMAT> .. With this, it is posible to select the MAP file format specifier (this option should be used before the MAP file name) supported formats: cwcf7, cwmcu6, cwmpc9, cwcf10, cwarm10, iararm6, uv4. Note that you can analyze several MAP files at once by specifying them all along with the –c option on the command line.

  NOTE: The current version of the script does NOT support GCC toochain yet.
- +PSP or -PSP .. force or suppress PSP details (default +)
- +BSP or -BSP .. force or suppress BSP details (default +)
- +MFS or -MFS .. force or suppress MFS File System details (default -)
- +RTCS or -RTCS .. force or suppress RTCS TCP/IP stack details (default -)

- +USBH or -USBH .. force or suppress USB Host stack details (default -)
- +USBL or -USBL .. force or suppress USB Dev stack details (default -)
- +SUMM or -SUMM .. force or suppress summary report (default +)
- +RAM or –RAM .. force or suppress the report on RAM utilization by static and global variables (default -)

Some usage examples are described below:

codesize.exe -c cwcf7 <MAP_file>

codesize.exe -M -c cwcf10 <MAP_file>

codesize.exe +MFS -PSP -BSP -c cwcf10 <MAP_file>

codesize.exe -c cwcf10 <MAP_file1> -c cwmpc9 <MAP_file2> ....

How to use the CodeSize script

In order to use the CodeSize tool, it is necessary to follow the below steps.

1. Build all the MQX libraries (BSP, PSP, SHELL, etc) and the custom project application. For this example it is used CodeWarrior 10.5, MQX 4.1 and the web_hvac demo for TWR-MCF52259. Figure 1, shows the demo and MQX libraries for the TWR-MCF52259.

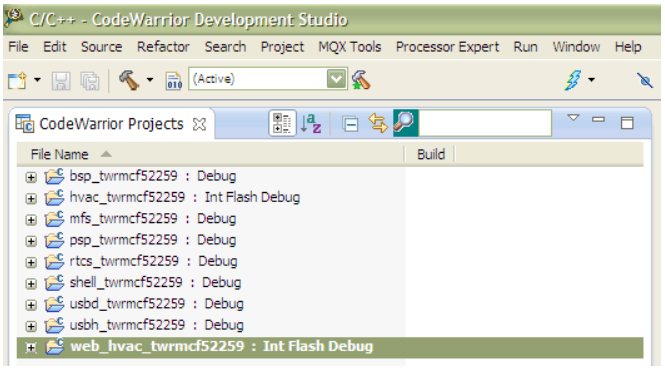

Figure 1.

2. After build the project, it is generated the xMAP file. This is located at NAME_project -> Int Flash Debug -> NAME_file.xMAP Please check Figure 2.

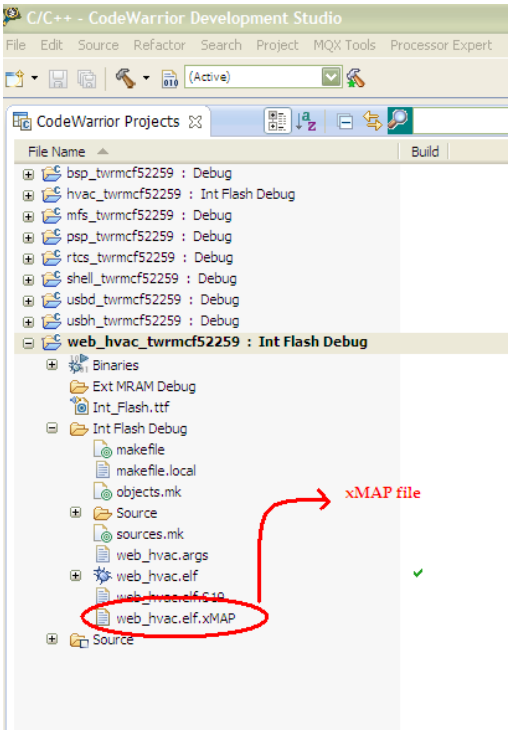

Figure 2.

3. Copy the xMAP file and paste it where is located the CodeSize tool. Figure 3 shows and example using TWR-MCF52259 and the web_hvac demo.

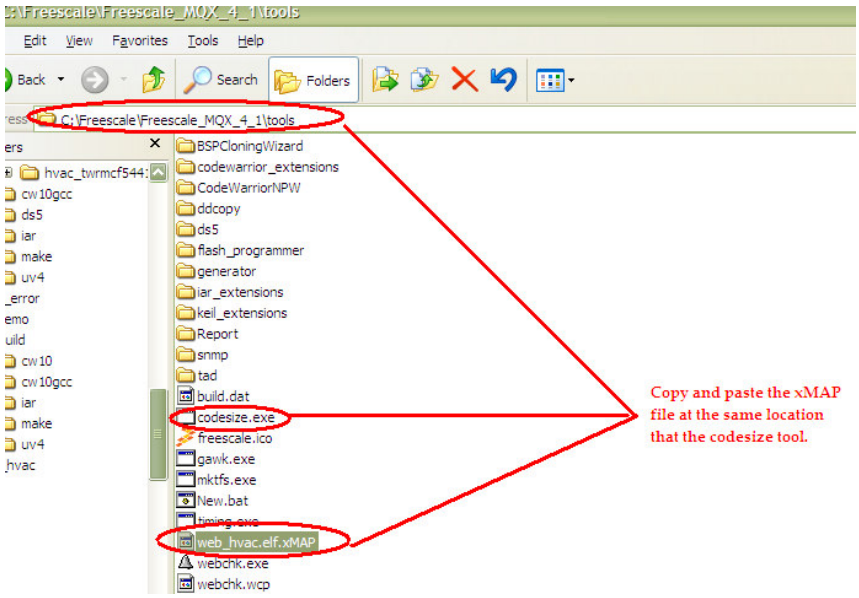

Figure 3.

4. Create a bat file. This file should be located at same path that the CodeSize tool and the xMAP file. This file uses the options previously mentioned.

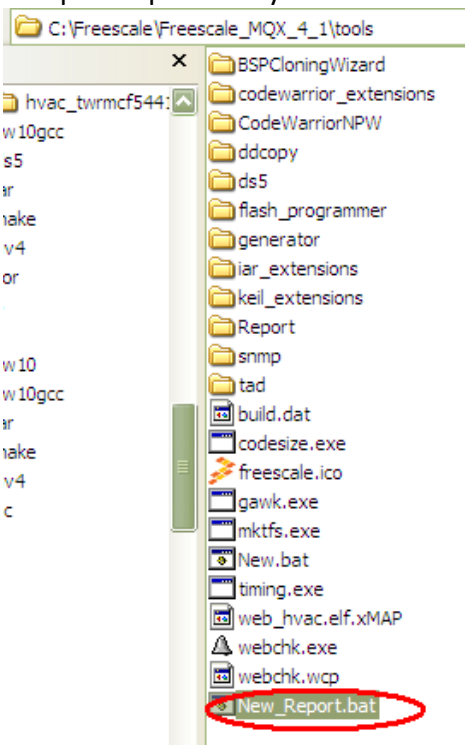

Figure 4.

Must contain at least: a call for the CodeSize tool, the name of the new report generated and the name of the xMAP file that will be analyzed.

For example:

```
codesize.exe -o NewReport.html +MFS +PSP +BSP +USBH +RAM +RTCS +SUMM -c
cwcf10 web_hvac.elf.xMAP
```

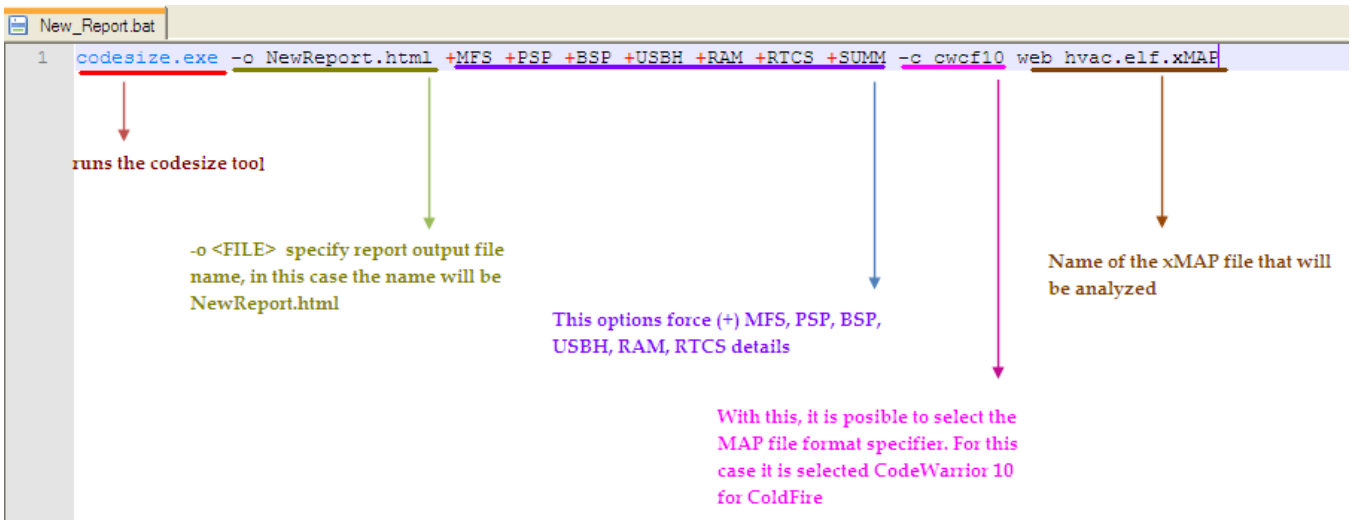

Figure 5.

5. Double click to the bat file in order to run it.
6. At this point the new report is created.

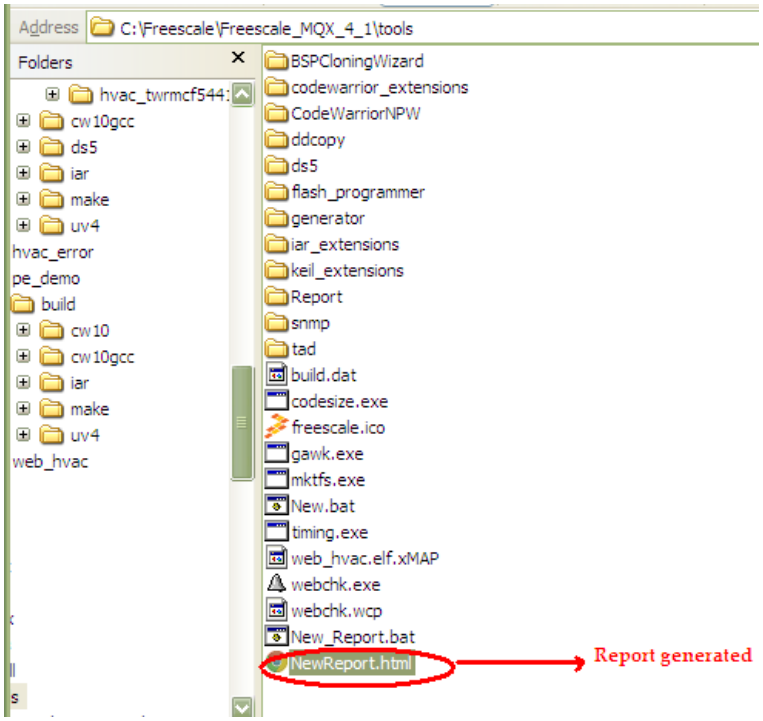

Figure 6.