Profiling Support Tools

The time profiling tool relies on a C program called getregions.exe which reads the program’s executable image (axf) file. Getregions extracts each function’s name, starting code address, and code size. With this information it can create a C file (region.c) with profiling support:

* A constant table (RegionTable) containing name, starting and ending addresses for each function’s code.
* A table (RegionCount) to track the access count for each region.
* An initialized constant variable NumProfileRegions.

Getregions can instead create a list of all functions and their size in bytes, allowing the user to determine which functions use the most code space as a first step in code size optimization.

# Installation

* Install Cygwin on your PC, or install the cygwin1.dll (e.g. at C:\cygwin64\bin).
* Copy the Scripts folder into your project folder.

# Time Profiling Procedure

Use the following procedure:

* Build your program in Vision.
* Run **update\_regions\_manually.bat** (in the Scripts folder) to create a new region.c file (which is placed in the src folder). This file will have the correct number of entries but the addresses may be wrong (this is ok – a later pass will fix this).
* Rebuild your program in Vision to generate an executable with the correct size region table. **Note that this table still uses the old function addresses, which will be wrong if the previous region table had a different number of entries.**
* Run **update\_regions\_manually.bat** to create a new region.c file with the correct function addresses and the correct number of entries.
* Rebuild your program in Vision to generate an executable with the correct region addresses.
* Download and execute your program on the target hardware. This will populate the RegionCount table to indicate the execution time profile.

# Warnings and Troubleshooting

* You must regenerate region.c and rebuild your executable every time you change your program in a way that might change functions sizes or locations.
* There is a chicken-and-egg problem: in order to create region.c, we need the map file. But the linker will not create the map file if region.c does not exist. So the solution is to copy a dummy region file (dummyregion.c) to region.c. Then the linker will be able to build the executable, which getregions can analyze to create the correct region.c.
* If Vision complains about not being able to reload region.c, try closing and reopening the region.c window.

# Code Space Profiling Procedure

To find the functions which use the largest amount of code space, use this process:

* Build your program in Vision.
* Run **find\_sizes.bat** (in the Scripts folder) which will create **sorted\_function\_sizes.txt** in the obj folder. This is a list of functions sorted by decreasing size.
* You can automate this step by adding it to the project build user options in Vision.

# Rebuilding getregions.exe

If you need to modify and recompile getregions.exe, follow these instructions.

* This program is compiled with gcc under Cygwin because it relies on the libelf library, which handles the ELF-format AXF file. Compile with gcc:

$ gcc -w -o getregions getregions.c -g -lelf

* Copy the new executable (getregions.exe) to the Scripts directory in your project.
* The batch file has a command which adds the path to the cygwin1.dll to your Windows system path (C:\cygwin64\bin) before trying to run getregions:

set PATH=%PATH%;C:\cygwin64\bin