A.G.Dean, agdean@ncsu.edu, 3/14/17

Profiling Support Tools (V1.3)

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.

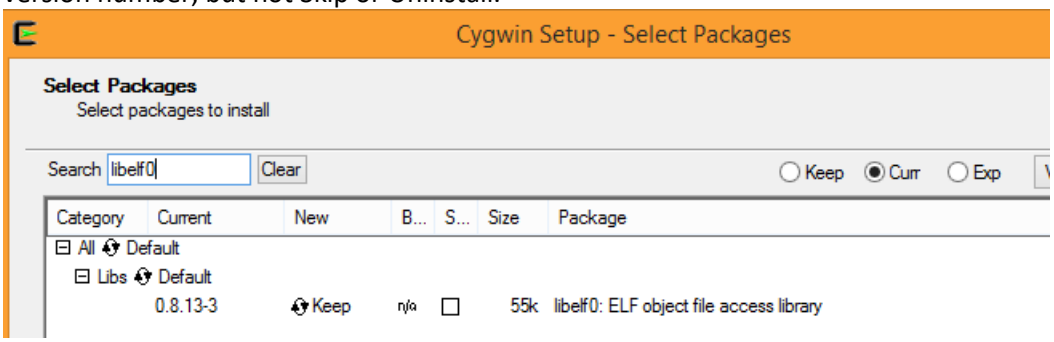
With this information it can create a C file (region.c) with profiling support:

- 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 empty table (SortedRegions) to hold sorted regions.
- 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.

Tool Installation

- Install Cygwin on your PC.
  - Download an installer from https://cygwin.com/install.html and start it up.
  - During installation, you will need to include the libelf0 package. When you reach the "Cygwin Setup – Select Packages" window, type "libelf" in the search box. Expand the Libs entry and make sure that the libelf0 entry is set to Keep, Install, Reinstall or a version number, but not Skip or Uninstall.

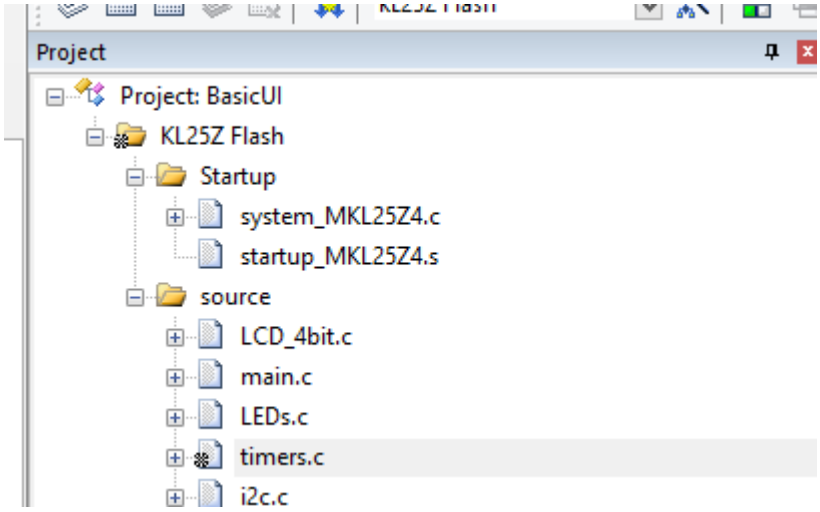


  - Select Next and continue with installation. If you do not complete this step, you may get an error regarding a missing "cyg-elf0.dll" when you try to update the regions.

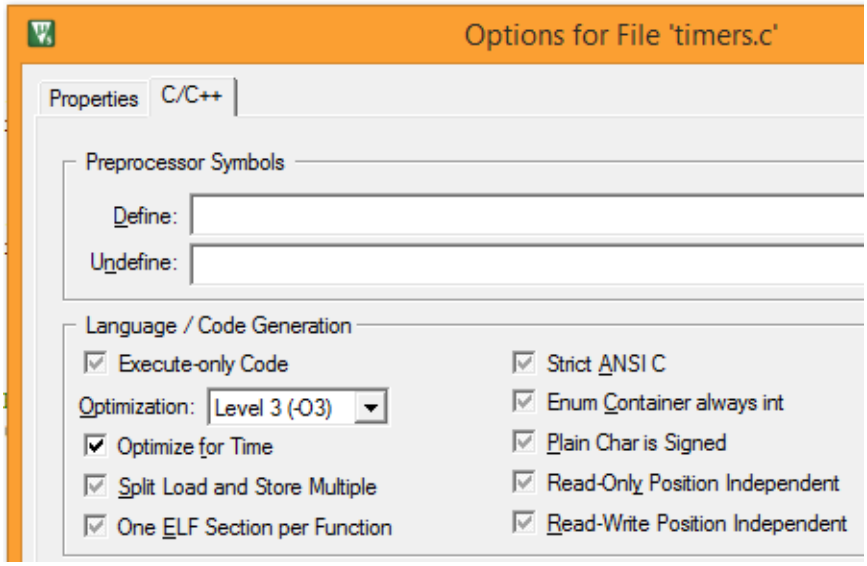
Adding Profiling Support to an MDK Project

- Copy the Scripts folder into your project folder. The scripts are written expecting the .axf file to be located in a folder called Objects\, which is at the same level as the Scripts folder. You may need to modify the scripts to match your folder structure and file locations.
- Copy these files to the source code directory in your project
  - profile.c
  - region.c
  - timers.c

- Copy these files to the include file directory in your project
  - profile.h
  - region.h
  - timers.h
- Add these existing files to your project in MDK:
  - profile.c – Note that you can change the sampling frequency from the default of 1 kHz to a different value by modifying the argument of the function call Init_PIT in the function Init_Profiling.
  - region.c
  - timers.c – Note that you need to override default compiler optimization settings for this file so it will always be compiled with -O3 optimization for time. Do this by right clicking on the timers.c entry in the Project window of MDK and selecting "Options for File 'timers.c'".



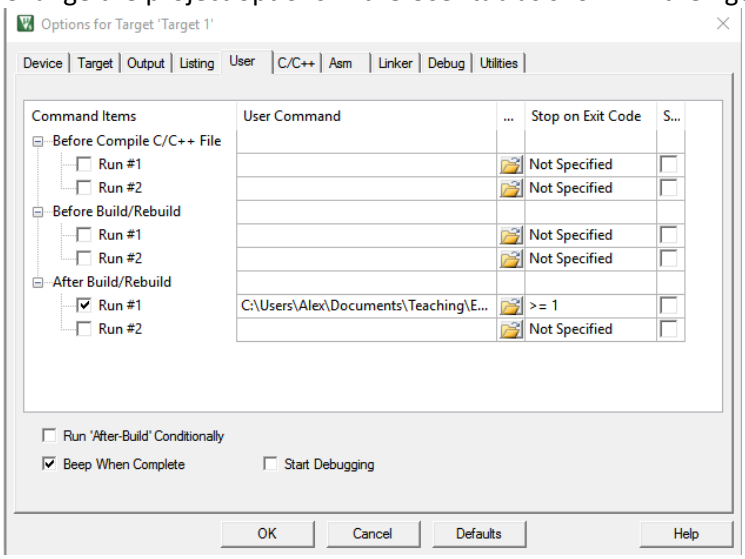
Then select the C/C++ tab, set it as shown below, and select OK.



Time Profiling Procedure

Use one of the two following procedures:

- Manual procedure
  - Build your program in µVision.
  - 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 µVision 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 µVision to generate an executable with the correct region addresses.
- Automated procedure
  - Change the project options in the User tab as shown in the figure.



    - Under "Run User Programs After Build/Rebuild"
      - Check box "Run #1"
      - Add the command **Scripts\update_regions_MDK.bat** to the text box.
      - Set "Stop on Exit Code" to **>= 1** so the build will fail if update_regions_MDK.bat fails.
    - Click OK to save these changes.
  - Now every time you build your program the regions will be updated.
  - To ensure the region information is correct, you may need to build the project up to three times if you have changed the source code significantly (e.g. both the number and size of the functions).
- Now you can 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 μVision complains about not being able to reload region.c, try closing and reopening the region.c window.
- If you doubt the profiling results, consider the following
    - Did you override the project settings for timers.c to set optimization to **–O3** for **time**?
    - Is the region table up to date (i.e. did the script generate a new region.c)?
        - Check the file's time stamp
    - Does the region table look reasonable?

Code Space Profiling Procedure

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

- Build your program in μVision.
- Run **find_sizes.bat** (in the Scripts folder) which will create **sorted_function_sizes.txt** in the object code 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 μVision, but you will need to change the find_sizes batch file since MDK runs it from the project directory, not the Scripts directory.

Rebuilding getregions.exe

If you need to modify and recompile getregions.exe, follow these instructions. Source code and documentation are included in the GetRegions directory.

- 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
  ```