Getting Started with ARMSim#

Downloading and Installing

ARMSim# is a free ARM simulator (with assembler and linker) developed by the Department of Computer Science at the University of Victoria. The main website for ARMSim# is hosted by the University of Victoria:

https://connex.csc.uvic.ca/access/content/group/ARMSim/SIMWeb/index.html

To download ARMSim#, you can navigate from ARMSim#'s home page (given above), or you can follow the link below, which takes you to the ARMSim# download web page:

https://connex.csc.uvic.ca/access/content/group/ARMSim/SIMWeb/DownloadARMSimSharp.html

Installation is simple – download the .zip file using the hyperlink from the above page, extract the contents, and run the .msi file (for Windows) or .exe file in Mono (for Linux or Mac). Notice that Windows users must have the .NET 3.0 framework installed. This is likely present on most Windows installations anyway, but if not, the Downloads page for ARMSim# gives a link to download and install it from Microsoft. Linux or Mac users must have Mono installed, which allows .NET applications to be run in non-Windows environments.

Because of the wide variety of personal computers, we cannot provide individual technical support for installing ARMSim#. ARMSim# is installed on the lab computers and is also accessible through remote access with DIVMS. Remote access is done through VMWare View client, which can be downloaded for free from the DIVMS download page:

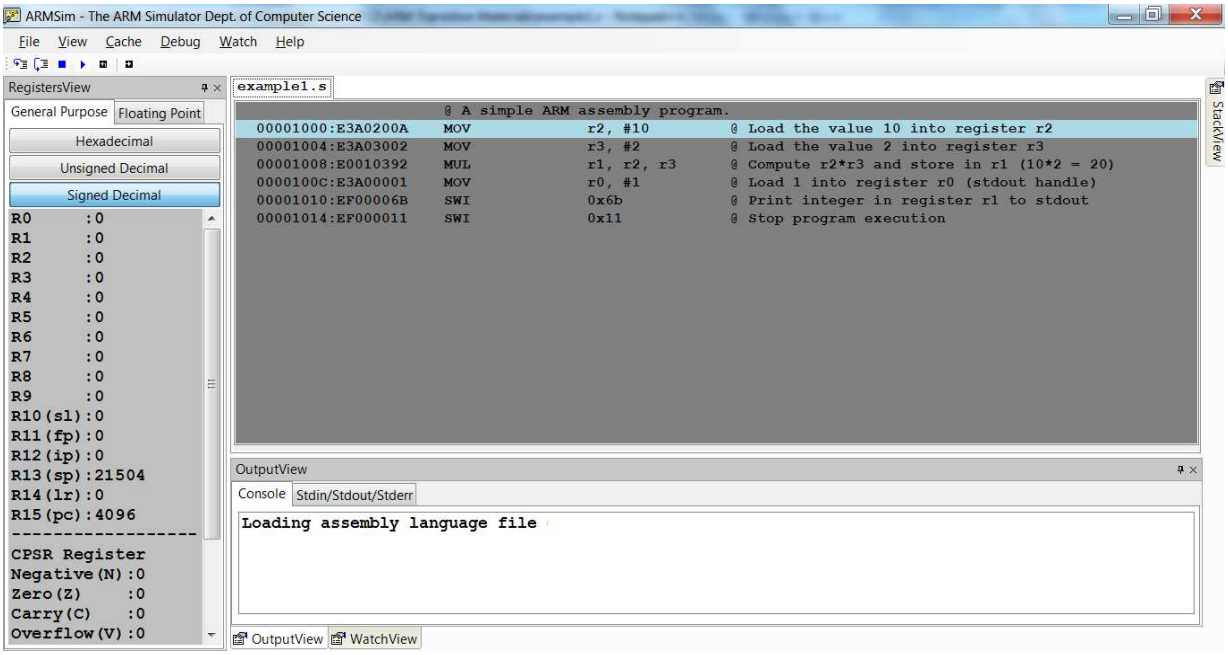
http://www.divms.uiowa.edu/csg/download.html

Creating and Running a Program

ARMSim# is *not* an editor, so you will need to create and edit your ARM assembly program with a separate text editor. Open your favorite text editor and type in the following simple program (or another simple ARM program of your choice):

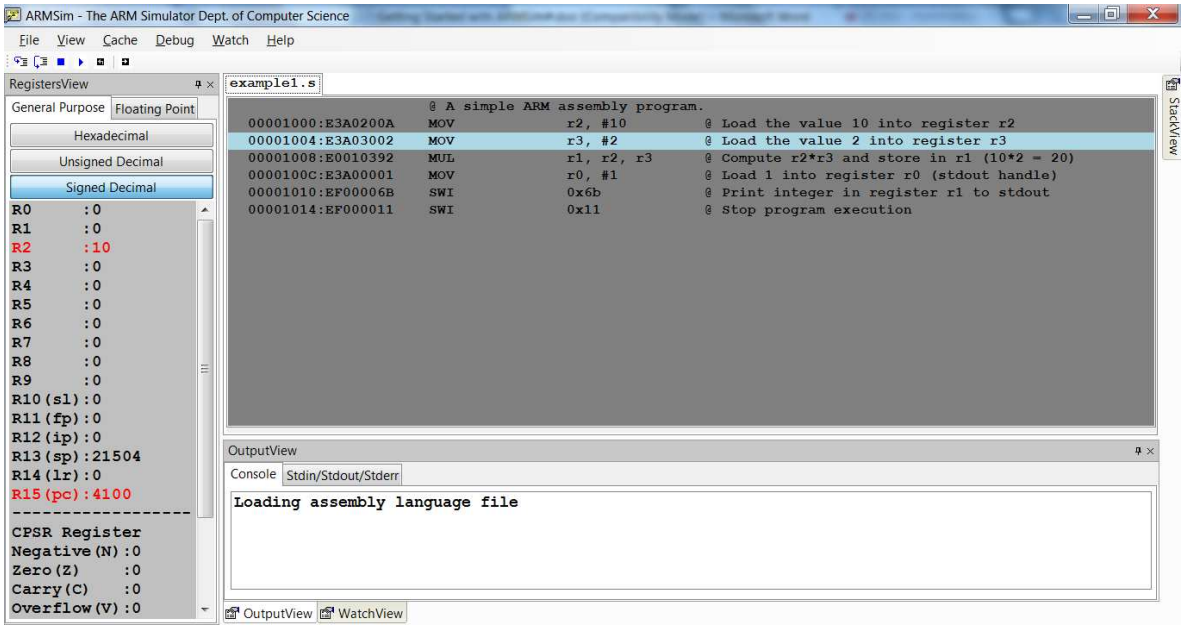
```
@ A simple ARM assembly program.
MOV        r2, #10    @ Load the value 10 into register r2
MOV        r3, #2     @ Load the value 2 into register r3
MUL        r1, r2, r3 @ Compute r2*r3 and store in r1 (10*2 = 20)
MOV        r0, #1     @ Load 1 into register r0 (stdout handle)
SWI        0x6b       @ Print integer in register r1 to stdout
SWI        0x11       @ Stop program execution
```

Do not worry if some of these instructions are unfamiliar at the moment, they will be covered in time. Save your program as example1.s – note the '.s' file extension is required in order for ARMSim# to read your file. Once you have saved your ARM program, you need to load it into ARMSim#. Open the ARMSim# program, select 'File -> Load' from the menu, and select your example1.s file. It should open your program and make sure it is without syntactic errors. Your screen will look like the one below.



On the left side of the screen, the registers for the simulated ARM processor are displayed. The source code window is in the center, while the console window is at the bottom.

There are three options for running (simulating) your assembly language program – *run*, *step into*, and *step over*. The two that will likely be of most use (and their toolbar icons) are (i) *run* (▸) and (ii) *step into* (). *Run* (keyboard shortcut F5) will run your assembly language program from start to finish, while *step into* (keyboard shortcut F11) will execute one instruction at a time, allowing you to see how each instruction is modifying the state. Press "Step Into", and you should see a screen like the one below.



On the left hand side, you will see that the registers that have been changed by the executed instruction are red. In the source code window, you will notice that the highlighted blue line is now the next instruction – is the instruction pointed to by the program counter, and will be executed next. Try stepping through the remainder of the simple program, verifying that the registers are changing as expected.