Introducing ARMSim# Version 2.1

1. What is Different?

Version 2.1 is a major re-design of ARMSim# in three main respects:

1. Instead of parsing and assembling ARM source code itself, ARMSim# now invokes the Gnu Assembler program `as` to perform the task.
2. Instead of using a set of extended SWI instructions based on the ARM RDI family to perform I/O and other system tasks, a new set known as the Angel SWI instructions has been adopted as the default set.
3. The undocumented support for scripting has been replaced by an extended set of command-line options.

Each of these main changes, and its amplications, are described below.

Some tidying up of code and numerous bug fixes have also been applied.

Use of the Gnu Assembler to Process ARM Source Code

Our own implementation of an assembler as a .NET program had its advantages (speed and a smaller footprint in memory) but also had some major disadvantages. One of these was that support for the full assembly language syntax would take too much programming effort, so there were many holes in the language coverage, and another is that ARM has introduced a new syntax for its assembly language. The new syntax known as UAL (Unified Assembly Language) is already supported by the Gnu Assembler in addition to the older Gnu syntax.

ARMSim# invokes the version of the Gnu Assembler distributed by Mentor Graphics in its Code Sourcery product. (Since the Gnu code is open source, with a GPL license, we are allowed to do that.)

Adoption of the Angel Extended SWI Instruction Set

The SWI instruction family previously used by ARMSim# was ad hoc and inconsistent because additional features were added piecemeal. This SWI family is *still supported* and we call it the **Legacy SWI** Family.

However, we encourage everyone to switch to the **Angel SWI** Family instead. The reason to do this is that it opens up the possibility of calling functions in the Standard C Library. Many functions in the C Library make calls to the operating system (typically for file and standard I/O access). The version of the C library distributed by Mentor Graphics uses the Angel SWI instruction to request the special services from an operating system.

A disadvantage of the Angel SWI is that the operations are lower level than those provided in the Legacy SWI set. For example, the Legacy SWI provided the ability to input or output decimal numbers, whereas the Angel SWI supports input and ouput of single characters only. As partial compensation, a file containing code to perform some common operations including I/O of numbers with the Angel SWI has been provided. Alternatively, functions such as printf and scanf in the C Library can be invoked.

Command-Line Usage

If an instructor wishes to compile and run a set of student programs, the ability to invoke ARMSim# from the command line or from a batch script should prove useful. The command line supports options to stop a program after it has executed a specified number of instructions, and it supports optkons to dump the contents of memory regions after execution has finished.

2. Current Distribution Status

ARMSIm# version 2..1 is available for Windows. It has been tested on Windows 8.1.

It has been tested on Ubuntu Linux under Mono. The docking windows feature available on Windows does not work on Linux (due to differences in its support for .NET Forms).

It does not yet work on Mac OS X, apparently due to a difference in the way that scrolling text wirdows are implemented in Mono on a Mac OS X system.