VisUAL
A highly visual ARM emulator

Cycle Models

Introduction

VisUAL supports basic customisation of the cycle counts displayed.

The cycle model in VisUAL is a lookup table that matches the base opcode of an instruction with its corresponding cycle count. The default cycle model VisUAL uses is given in the following table:

Base Opcode	Default Model Cycle Count
`MOV`	1
`MVN`	1
`ADD`	1
`ADC`	1
`SUB`	1
`SBC`	1
`RSB`	1
`RSC`	1
`AND`	1
`EOR`	1
`BIC`	1
`ORR`	1
`ORN`	1
`ASR`	1
`LSL`	1
`LSR`	1
`ROR`	1
`RRX`	1
`CMN`	1
`CMP`	1
`TST`	1
`TEQ`	1
`ADR`	1
`LDR`	2
`STR`	2
`LDRB`	2
`STRB`	2
`LDM`	2
`STM`	2
`B`	3
`BL`	3
`NOP`	1

Instruction Cycle Behaviours

With the exception of the following, the cycle count used for each instruction is the one specified directly in the model:

- `LDR Rn, =...` - Actual cycle count used is the cycle count specified for `MOV` instructions by the cycle model
- `LDR PC, [...` - Actual cycle count is incremented by `m-1`, where `m` is the cycle count specified for branch instructions `B` by the cycle model

- `LDM / STM` - Actual cycle count is `m+n`, where `m` is the cycle count specified by the model and `n` is the number of registers to load / store.
- `LDM` with `R15 / PC` in register list - Actual cycle count is incremented by `m-1`, where `m` is the cycle count specified for branch instructions `B` by the cycle model
- `MOV PC, ... / LDR PC, =...` - Actual cycle count used is the cycle count specified for branch instructions `B` by the cycle model
- Any instruction predicated false - Actual cycle count used is the cycle count specified for `NOP` by the cycle model

Using a Custom Cycle Model

A custom cycle model can be used with both the GUI and headless versions of the application. A template file for the default cycle model is provided in the following locations relative to the installation folder depending on your operation system:

Operating System	Model Template
Windows	`VisUAL\content\cycle_model.txt`
Mac OS X	`VisUAL.app/Contents/MacOS/cycle_model.txt`
Linux	`VisUAL/content/cycle_model.txt`

To create a custom cycle model, create a copy of this file and edit the number in front of each opcode. This will act as the base cycle count for each instruction. The order the opcodes are specified in does not matter, provided that all opcodes are present. Otherwise, the model will be treated as invalid, and the application will revert to the default model with a warning.

The file format is that of a comma-separated file, with each line corresponding to one instruction-cycle mapping as follows: `opcode, cycles`

You can select the custom cycle model to use in the Settings panel of the application in the GUI version, and via the `--cyclemodel` argument for the headless version.