Project 1. Instruction Set Simulator for Cortex-M0 Processor

Due date: 4/30 23:59 E-mail: sckim@ics.kaist.ac.kr; ranggi.hwang@kaist.ac.kr

1. Overview

In the Project 1, you will design a *C* program that emulates the CPU of ARM Cortex-M0 processor, which is usually called an instruction set simulator. Cortex-M0 is the simplest processor in ARM processors, and is based on ARMv6-M architecture that has the Thumb instruction set. Cortex-M0+ and Cortex-M1 processors are also based on ARMv6 architecture.

ARM processors usually support two set of instructions, ARM and Thumb whose length is usually 32 bits and 16 bits, respectively. Since Cortex-M0 only supports the Thumb instruction set, it is suitable for small silicon area and low power applications. However, the processor is still a 32-bit processor as all the registers are 32-bit wide.

Cortex-M0 has two operating modes, thread mode and handler mode. ***In this project, we will assume that the processor only operates in thread mode, and exclude memory barrier, 'DMB', 'DSB', 'ISB' instructions.*** Therefore, you can concentrate on the registers related to the thread mode, and those are R0-R12, MSP (R13), LR (R14), PC (R15), and APSR. The page 41 of 'Cortex-M0 Technical Reference Manual' shows the registers that Cortex-M0 has.

2. Attached Files

(1) ARMv6-M Architecture Reference Manual
   - A. Part A describes programming model and memory model along with the instruction set
   - B. Details of the ARMv6-M Thumb instruction set are described in Chapter A6

(2) Cortex-M0 Technical Reference Manual
   - A. Specialized manual for Cortex-M0

(3) Cortex-M0 Instruction Set Encoding
   - A. Summary for the instruction set

(4) Cortex-M0
   - A. iss.h – Header file **(You will modify and submit)**
   - B. core.c – Main function to execute program in test.hex file
   - C. memory.c – Functions for load/store instructions
   - D. inst.c – Functions related to PC register and bit extractions
   - E. thumb.c – Functions of instructions **(You will modify and submit)**
   - F. run.sh – Commands to compile source files, and execute ISS
   - G. test.hex – Simple program that initializes registers and increases r0 register value
   - H. test.dis – Corresponding disassembly
   - I. test_simple – Directory that contains files generating test.hex **(You can make your test program)**
     - i. test.c – *C* program to be tested
     - ii. test.s – assembly program to be tested
     - iii. compile.sh – run this file to generate test.hex and test.dis

3. Requirements

(1) Put all the relevant codes in ‘thumb.c’ and ‘iss.h’. (*Do not make other source files.)
(2) Only the annotated part of ‘your code here*’ is allowed to be modified in ‘thumb.c’ and ‘iss.h’ files.
(3) Report file, {Student_ID}_{Name}_report.pdf, which explains your codes, your test results, and reasons why you make such test program. (*10 pages or less allowed)

4. Submission

1. **Due date: 4/30 23:59**
2. Submit following 3 files on the KLMS :

   ‘iss.h’, ‘thumb.c’, ‘{Student_ID}_{Name}_report.pdf’
3. Assessment

   A. Correctness of ISS operation – Several test programs will be executed and the results will be checked

   B. Quality of the source code including annotations

   C. Quality of the report

   D. **NOTE**: If you submit past the due date, your grade will be deducted

   E. **NOTE:** If you do not satisfy the requirements, your grade will be deducted

   F. **NOTE**: If you copy other’s work, you will not receive any credit