EEE 216: Microprocessor Systems Lab

Anthony Centeno

(Anthony.Centeno@xjtlu.edu.cn)

In this seesion you will use the ARM assembly programming and the Keil µVision IDE.

i. Getting started with Keil µVision

- Start Keil µVision 4 or 5 (Both are installed in the computer lab machines)
- Project → µVision project → select project directory and project file name → select target CPU (ARM → ARM7 – little endian).
- Create new source file (File → new). Assembler file must have .s extension.
- Add the source file to a source group in the project (e.g. the default one).
- Write your program and save the source file.
- Build the project (Build Targets).
    - If you have edited the source file have previously built the project you should first clean targets before doing Build again
- Switch to the debugging/running mode.
- Control the program execution and observe the registers and memory.

Sample Program

```
AREA    ArmExample,     Code
ENTRY
        MOV r1, #10
        MOV r2, #&10
SWI   &0
END
```

TASK 1.

Run the sample program. Once it's running write comments into the program to explain what it is doing. [10 marks]

Note: Semicolons mark the beginning of comments

Example:

```
        AREA        ArmExample,     CODE
; code area declaration and name
        ENTRY       ;program entry point
```

NOTE: You will submit your programs for assessment. **Every line of code should be commented.**

A. 16 Bit Transfer

Consider the code below:

```
      AREA  ArmExample, CODE

      ENTRY

Main
            LDR R1, Value       ;load the 16 bit value to be moved
            ldr R2, Res         ;load the memory address to store the
value
            STR  R1, [R2]       ;save the value to the memory address Res
            SWI  &0

Value   DCW &C123               ;DCW enables you to write 16 bits to memory
Res     DCD &FFFFFFF0           ;DCD enables you to write 32 bits to memory

      END
```

It transfers 16 bits from "Value" to "Res".

Task 2.

i. Modify the code to transfer 32 bits between "Value" and "Res". [20 marks]
ii. Write some comments at the end of the program discussing how the code varies compared to writing it for the ViSUAL Arm Emulator. In particular consider where the DCD (and DCW) commands are placed. [Hint: Try putting them after END] [20 marks]

B. Counting Iterations

Task 3.
Write an ARM assembly program which contains an infinite loop incrementing at each iteration the value of r1. [20 marks]

C. Ordered pair

Write an ARM assembly language program which assigns a 16 bit value to r1 and a 16 bit value to r2, then checks of r1 = r2. If it is not the same an eight bit word 0xFF is sent to a memory address called fault.
[30 marks]

Note: You will need to put in two 16 bit values something like this:

```
Value1   DCW  &1234

         align

Value2  DCW  &1235
```

[See what happens if you don't use align.]

Submission Requirements

Programs will need to be submitted on ICE in Week 12. All programs must contain comments to clearly explain how the program works. Working programs with no or minimal comments will receive a lower grade.