EE2016 Microprocessor Lab & Theory July-Nov 2022

EE Department, IIT, Madras.

Experiment 6: ARM Assembly 2 - Computations

1 Aim

To (a) learn advanced ARM instructions, conditional execution etc (b) go through example programs in Welsh and (c) write assembly language programs for the given set of problems at the end of this document.

2 Equipments, Hardware Required

The list of equipments, components required are:

1. A PC with Window OS
2. KEIL Microvision V5 IDE for ARM

3 Background Material

3.1 Advanced ARM Instructions

Though the last week's experiment was also on ARM assembly, the assembly program which we would learn in this week (experiment 6) would be a bit advanced. For example we would be learning about the conditional execution of instructions, instructions which automatically sets the flags etc. In terms of tasks you are asked to implement, would also be more practical and involved. The last of the tasks is about a link level algorithm running in ubiquitous digital communication links. It is HDLC framing protocol. ClickHereForHDLC

3.2 ARM Assembly Programming

Refer Welsh Chapter 7. Also browse through earlier chapters.

3.3 ARM Demo Program

The following program adds a series of 16 bit numbers. Take this as an exercise to make yourselves familiar with the ARM assembly programs. This is optional. Note that you must include the programming details for implementing the tasks listed in next section.

Program 7.1a: `Ch5Ex1.s` — *Add a series of 16 bit numbers by using a table address*

```
1   *       Add a series of 16 bit numbers by using a table address look-up
2
3           TTL     Ch5Ex1
4           AREA    Program, CODE, READONLY
5           ENTRY
6
7   Main
8           LDR     R0, =Data1              ;load the address of the lookup table
9           EOR     R1, R1, R1              ;clear R1 to store sum
10          LDR     R2, Length              ;init element count
11  Loop
12          LDR     R3, [R0]                ;get the data
13          ADD     R1, R1, R3              ;add it to r1
14          ADD     R0, R0, #+4             ;increment pointer
15          SUBS    R2, R2, #0x1            ;decrement count with zero set
16          BNE     Loop                    ;if zero flag is not set, loop
17          STR     R1, Result              ;otherwise done - store result
18          SWI     &11
19
20          AREA    Data1, DATA
21
22  Table   DCW     &2040                   ;table of values to be added
23          ALIGN                           ;32 bit aligned
24          DCW     &1C22
25          ALIGN
26          DCW     &0242
27          ALIGN
28  TablEnd DCD     0
29
30          AREA    Data2, DATA
31  Length  DCW     (TablEnd - Table) / 4   ;because we're having to align
32          ALIGN                           ;gives the loop count
33  Result  DCW     0                       ;storage for result
34
35          END
```

4 Tasks: Engineering Problem

Solve the following engineering problems using ARM through assembly programs:

1. Given a 32-bit number, generate even parity bit for that (32-bit) word.
2. Determine the length of an ASCII message. All characters are 7-bit ASCII with MSB = 0. The string of characters in which the message is embedded has a starting address which is contained in the START variable. The message itself starts with an ASCII STX (Start of Text) character (0x02) and ends with ETX (End of Text) character (0x03). Save the length of the message, the number of characters between the STX and the ETX markers (but not including the markers) in the LENGTH variable
3. Given a sequence of 32-bit words (sequentially arranged) of length 8 (32 bytes or 256 bits), identify and track special bit patterns of 01111110 in the sequence (if at all appears in the sequence). [This special bit sequence is called "framing bits", which corresponds to HDLC protocol]. Note that this special bit pattern may start at any bit, not neccessarily at byte boundaries. Framing bits, allow the digital receiver to identify the start of the frame (from the stream of bits received).

5 Procedure

Since it is a simulation experiment, we dont need hardware. It is enough if we have a PC loaded with Keil software. For snapshots of KEIL software, see the previous experiments writeup.

1. Go through Welsh thoroughly. Do all the home work - meaning browse ARM architecture, go on till example program 7.1(a). Demo the example program in KEIL for yourselves.
2. Write the assembly programs for the above problems (one at a time).
3. Enter the above program in KEIL software, edit and compile / assemble.
4. Run it in the '*debug*' mode to see whats happening to the registers.
5. Finally, demonstrate its working, before your TA

6 Results

1. Run the program and ask the TA to see the output
2. Take a snapshot using your mobile and make a report