Lab-1: Introduction to 8051 Instructions

This set of experiments has the following objectives:

- Familiarization with the instruction set of 8051 family.
- Familiarization with the development environment for 89C5131A.

You should browse through the manual for Keil software and the data sheet for 89C5131A uploaded on moodle site. Specifically, refer to these to understand the memory-map.

- Data sheet: Section 12 (On-chip Expanded RAM) to understand the memory-map of 89C5131A.
- Reference slides or text relevant to internal RAM of 8051, addressing modes, stack pointer etc.

89C5131A has 256 bytes of RAM, of which the top 128 bytes can only be accessed using indirect addressing. Since the stack is always accessed through indirect addressing using the stack pointer, we shall locate the stack in this part of the memory. In our assembly programs, we shall initialize the stack pointer to 0CFH, which provides 32 bytes for the stack. It is common to place arrays also in this part of this memory, because arrays are also easily accessed using a pointer. The address range from 80H to 0CFH is available for this (since memory beyond this is reserved for the stack).

Direct addressing in the address range 80H to 0FFH accesses special function registers, which control IO ports, timers, interrupts etc. In particular, the micro-controller board that we shall use has sliding switches and LEDs connected to a port. The switches and the LEDs will be our primary medium for input/output during this set of experiments. Since we do not have a monitor program running on the board, most of the debugging should be carried out using simulation at this stage.

1 Homework

1. Go through and understand the sample (`sample_led.asm`) program provided (in `lab1.zip` file) before getting started on these exercises.
2. Write a subroutine that adds two 8-bit numbers in memory locations 50H and 60H and writes the result to memory location 70H.
3. Write an assembly language program to find the sum of N numbers starting from 1 and stores the partial sum values in N consecutive memory locations. The value of N (less than 20) is given in memory location 50H. The resultant sum is to be stored in memory locations starting at 51H. As an example, if $3H$ is in 50H. Your program should result in having 1, 3, and 6 in memory locations 51H, 52H, and 53H.

You should assemble and debug these programs using Keil software on a PC or laptop. You should check that the program is operating correctly using single stepping and break points provided by Keil Software.

2 Optional: Random number generation

Write an assembly language program to generate random numbers using the pseudo random number generator function

$$(aX + b) \mod n.$$

Initial seed X, a, b, n are to be taken from consecutive memory locations starting from 50H. Generate 15 random numbers (including the initial seed) using this generator and store them in memory locations starting from 60H.

3 Lab work

The lab work involves two exercises. To perform these exercises, make use of the template code `template_adder.asm` provided in the `lab1.zip` file.

3.1 Addition of two 16-bit numbers

Use the subroutine developed as homework in this exercise. Write an assembly language program to perform addition of two 16-bit numbers given in 2's complement form. The two numbers are stored in memory locations 60H and 70H. Storage is big-endian i.e., most significant byte is in the smallest address. The result can be 17 bits long, so it needs three bytes for storage. The result is to be stored in memory locations starting from 62H.

3.2 Subtraction of two 16-bit numbers

Repeat the previous question, but now do subtraction of two 16-bit numbers. Write an assembly language program to perform subtraction of two 16-bit numbers given in 2's complement form. The two numbers are stored in memory locations 60H and 70H. Storage is big-endian i.e., most significant byte is in the smallest address. The result can be 17 bits long, so it needs three bytes for storage. The result is to be stored in memory locations starting from 62H.