ECE-251 Computer Architecture
Project 2 Report

Di Mei, Zhihao Wang
Professor Mohammed Billoo

Introduction

In this project, we implemented a program of sorting by using the ARM assembler in the Raspberry Pi. The sorting program reads a file containing at most 100 lines of 32-bit integers on each line. To sort these integers, it makes use of the algorithm of “insertion sort” and then outputs sorted integers line by line in a separate file. When running this assembly program, users are required to type the name of the input file and set the name of the output file. If the input file contains any integer larger than 32 bits or more than 100 lines of integers, the program will return error warning.

Program Architecture

File I/O

In the “.data” section, we directly declared addresses of different variables. They include input/output commands, scan/print formats, read/write modes, array of integers, length of the array, and link register (return). In the “.text” section, the program can directly load registers from addresses shown above by using “ldr register, =address”.

The I/O section consists of three parts: input file reading, output file writing, and file closing. To effectively conduct various I/O operations, we used several functions from the C library. Functions “printf” and “scanf” are used to print commands and scan names of input/output files. Functions “fopen” and “fscanf” are used to open and read the input file. The function “fprintf” helps write the sorted integers into the output file and “fclose” enables the program to close input/output files.

Insertion Sort

The insertion sort is a simple sorting algorithm that builds the final sorted array one item at a time. Compared to other sorting algorithms, the insertion sort is relatively stable because it does not change the relative order of elements with equal keys. It is also adaptive since it is efficient for data sets that are already substantially sorted. We implemented this sorting algorithm between “read” and “write” sections in our program. High-level programming language’s implementation of the insertion sort is shown below.

```
/* C/C++ Code */
null insertion(int[] a, int n)
{  for(int i=1; i<n; i++)
   {  temp = a[i];
      j = i-1;
      while(j>=0 && temp<a[j])
      {  a[j+1] = a[j];
         j = j-1;
      }
      a[j+1] = temp;
   }
}
```

Challenges & Solutions

The largest challenge we faced was to input and output files. We first tried to use SVC supervisor calls with register r7 and svc 0 commands to input and output. However, we failed by having segmentation errors. We also encountered the type error where the output file shows random symbols instead of sorted integers. Then we changed our style by calling to C style standard I/O. We used C style file commands, fopen, fclose, fprintf, and fscanf, to get input from and write output into files. The command fscanf made it easier to read line by line from the input file.