Computer Architecture Project 2 : Building a Simple MIPS Emulator

201911106 유제이

1. Files

Assembler Program의 소스파일은 다음과 같이 2가지로 구성되어 있다.

1) emulator.cpp

: emulator.cpp는 main function이 있는 소스파일로, 프로그램 작동을 위해서는 해당 파일을 컴파일 및 실행해야 한다.

2) Decode.h

: Decode.h는 emulator.cpp가 include하는 헤더파일로, 읽어들이는 file의 정보를 Decode, Words, Instruction class의 instance로 저장한다. Instruction Decoding을 담당하는 class다.

2. Compile and Execution

해당 코드는 Window hyper-V를 이용한 가상 컴퓨터 내 Ubuntu 20.04 환경에서 g++ 9.4.0 ver을 사용하여 컴파일 후 실행하였다. 컴파일 및 실행 방법은 아래와 같다.

1) Compile

assembler.cpp, Directive.h, sample.o, sample2.o 등 실행 및 읽기에 관련된 파일이 있는 Directory에서 Open in terminal -> `g++ -o emul emulator.cpp`을 입력한다.

2) Execution

Complie 후 다음과 같은 옵션을 조합하여 실행할 수 있다. -m, -d, -n flag 이외의 flag가 입력되면 "unknown flag inserted"를 출력하고 프로그램을 종료한다.

```
./emul [-m addr1:addr2] [-d] [-n num_instruction] <input file>
```

1) **-m** : 메모리 주소 범위를 받아 프로그램이 종료될 때 범위 내 내용들을 출력한다. 4 Byte 단위로 address를 읽어들이며, 따라서 addr1과 addr2가 0x00400000, 0x10000000 + 4*m (m은 양의 정수) 꼴로 나타내질 수 있어야지만 해당 옵션에 따라 프로그램이 동작할 수 있으며, 유효한 범위가 아닐 때 message를 띄우고 프로그램을 종료한다.

예외 상황과 각각의 error message는 다음과 같다.

- addr1, addr2 is null or ":"로 범위가 나눠지지 않은 경우 : "The range of address is required with flag -m."
- addr의 입력 형태가 '"0x" hexadecimal이 아닌 경우 : "The range of address should be the form of hexadecimal like '0xffffffff'."
- addr1 > addr2 : "first address should be lower than second one."
- addr1 < 0x00400000 : "The address is out of the range. The minimum is 0x00400000."

- addr1과 addr2가 0x00400000, 0x10000000 + 4m의 꼴이 아닐 때 : "The Range of address should be the unit of word."

2) **-d** : -d 옵션이 주어질 경우 각 instruction의 실행이 끝나는 매 순간에 R0~R31 register 내 저장된 데이터를 출력한다. -d 옵션이 없을 경우는 끝날 때 한 번만 출력한다.

3) **-n** : 실행할 instruction의 개수를 지정하고 이에 따라 실행횟수를 제어한다. 예외상황과 error message는 다음과 같다. 이때, n을 float값으로 입력하게 된다면 정수로 변환하여 실행횟수를 설정한다.

- n flag 이후 값이 주어지지 않는 경우 : "The number of instruction is required with flag -n."

- n이 음수인 경우 : "The number of instruction must be positive integer number."

4) **Input file**: input file은 필수 입력 사항으로 파일 이름이 주어지지 않았을 때 "There's no file provided"를 출력하고 프로그램을 종료한다.

3. Functions and Flow

전체적인 플로우는 다음과 같다.

① 입력된 값들을 통해 출력 형식 결정 및 참조 파일 불러들이기

② 파일 내 정보를 읽어들이고 data 공간 할당 및 Instruction, word를 가상 data공간에 저장

③ 저장과 더불어 Decode.h class instance construction을 통한 Instruction Decoding.

④ PC 값을 받아 가리키는 Instruction Fetch (더불어 PC값에 4를 더해준다.), 이후 Decode 된 instruction을 실행, 상황에 따라 PC값을 업데이트 시키며 instruction이 없을 때까지 (혹은 지정한 횟수까지) 실행 이후 내용 출력 및 종료

Decode.h

Decode class는 Words, Instructions class의 Base class로 instruction 및 data의 32 bit 정보를 받아 해석하고 관련 정보를 저장한다.

Words : word data의 정보를 다루는 class로 instance 생성 시 32 bit를 각 8 bit씩 4개로 쪼개어 Big Endian을 따라 값을 해당 address에 저장한다.

Instructions : instruction 32 bit를 받아 각 instruction을 해석하고, 이에 맞는 Control signal 등을 생성한다. 각 operation, rs, rt, rd, function 및 offset, immediate 등의 정보를 갖고있어 해당 class instance의 생성 후 (Decoding 후) instruction을 실행할 수 있다.

Emulator.cpp

Emulator에서는 decoding을 제외한 모든 과정이 이루어진다. 옵션 분석 및 출력 형태 결정, 파일 데이터 읽기, Instruction Fetch, Execution 및 결과 출력 등을 담당한다. Register의 번호와 값 pair를 저장하고 있는 map R을 global variable로 가지고 있어 Execution 시의 register update를 반영, 저장한다.