오픈소스 기반의 실습용 SIC/XE 컴퓨터 시뮬레이터의 구현*

김주현, 김현아, 문봉교
동국대학교 컴퓨터공학과
e-mail: juhyeonk1013@naver.com, khyun9764@naver.com, bkmoon@dgu.edu

Implementation of Open Source-based SIC/XE Computer Simulator for Educational Laboratory

Ju Hyun Kim, Hyun Ah Kim, Bongkyo Moon
Dept. of Computer Science and Engineering, Dongguk University-Seoul

요 약

기존의 어셈블러는 시각적으로 불편하고 사용자 편의를 위한 기능을 제공하지 않으며 최신의 컴퓨터와의 호환성 문제가 있었다. 이러한 문제점들의 해결책으로 나온 SIC/XE어셈블러 시뮬레이터 오픈 소스를 GitHub에서 클론하여 분석하고 테스트하였다. 본 논문에서는 오픈 소스 SIC/XE어셈블러 시뮬레이터의 다양한 오류를 분석하고 이를 수정하였다. 또한 리터럴 테이블, 심볼 테이블, 목적코드 및 오류 메시지의 시각화를 통해 기존의 SIC/XE어셈블러 시뮬레이터를 개선시켜 사용자 편의를 높인 학습용 SIC/XE어셈블러 시뮬레이터를 구현하였다.

1. 서론

컴퓨터과학의 교과목들 중에서 많은 코스가 실시간 시스템, 어셈블, 링킹, 로딩, 시스템 가상화 같은 내용을 다루고 있다. 이러한 개념들을 이해하는 것은 효율적인 시스템 소프트웨어를 설계하고 구현하는데 매우 중요하다. 컴퓨터구조, 운영체제를 비롯하여 프로세스에 영향을 주는 실행환경에는 많은 요소들이 있지만, 그 핵심개념은 동일하다.

하지만 교육현장에서 실제와 동일한 환경을 제공하는 것은 종종 재정적인 이유나 다른 문제들로 인해 불가능하거나 아주 어렵다. 이런 경우 주로 시뮬레이션을 이용하게 된다. 시뮬레이션이 전체로서는 실제 환경을 반영하지는 못하지만 학습과정을 향상시켜주는 매우 유용한 교육학적인 도구가 된다.

이러한 이유로 많은 가상의 컴퓨터들이 설계되어왔고, 다양한 코스에서 사용되어왔다. 본 논문에서 우리는 표준모델(standard model)과 확장모델(extended model)의 2가지 버전을 가진 SIC(Simplified Instructional Computer)라는 가상의 컴퓨터에 초점을 맞춘다[2].

기존의 SIC/XE 어셈블러는 콘솔 cmd 창과 유사한 검은 화면에 레지스터, 메모리 등의 내용이 한 번에 일렬로 출력되는 형식으로 표현되어 시각적으로 불편하고, 사용자들이 명령어를 직접 입력해야만 작동시켜야하는 어려움이 있었다. 그리고 32비트 컴퓨터에서만 돌아가 최근의 64비트 컴퓨터에서는 사용을 할 수 없는 등 사용 용이성이 미흡하다는 단점이 있었다. 이러한 불편함을 완화시키기 위해 자바 GUI로 SIC/XE 어셈블러 시뮬레이터가 나오게 되었다[1][3]. 이 시뮬레이터는 메모리의 상태와 레지스터 내용의 실시간 확인, 단계별 코드 진행 등의 기능이 포함되어 있다. 하지만 이 SIC/XE 어셈블러 시뮬레이터를 직접 실행시켜 테스트한 결과, 예외 처리 오류, 실행 종료 오류 등의 여러 오류로 인해 시뮬레이터의 실행에 문제가 발생하였다[1].

본 논문은 위의 문제를 해결하고 발전시키고자 시뮬레이터에서 발생한 오류를 분석 및 일부 수정하고 사용자 편의성을 높인 개선된 학습용 SIC/XE 어셈블러 시뮬레이터를 제안하고자 한다.

본 논문에서 제안한 시뮬레이터 개발을 위해 기존 오픈 소스의 코드를 분석하여 시뮬레이터에서 발생하는 다양한 오류의 발생 위치를 찾아 그 원인을 밝혔다. 이를 통해 예외 처리 오류 및 실행 오류 등 일부 오류를 수정한 후 시뮬레이터의 테스트를 진행하였다.

또한 기존 시뮬레이터에서 지원하지 않았던 리터럴 및 심볼 테이블을 시뮬레이터에 출력하는 기능을 추가하여 어셈블리어 코드에 포함된 리터럴 및 심볼을 시각적으로 나타내었다. 목적코드 및 오류 메시지를 출력하여 시뮬레이터 실행 시 발생하는 목적 파일의 내용을 바로 시뮬레이터 안에서 확인할 수 있고, 실행 시 오류가 발생하였을 경우 오류를 즉시 확인할 수 있다.

* 본 연구는 동국대학교 컴퓨터공학과의 지원을 받아 수행된 개별연구임

2. 관련 연구

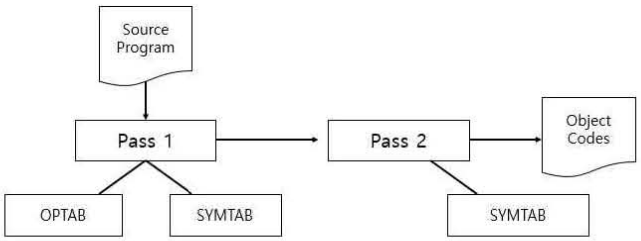
2.1 이중 패스 어셈블러

어셈블러란 어셈블리어로 된 코드를 기계어 형태의 목적 코드로 해석하는 컴퓨터 언어 번역 프로그램이다.

첫 번째 패스에서는 원시 프로그램에서의 심볼을 읽어 각 명령어나 데이터 크기 등을 파악하여 심볼의 상대주소를 정하여 심볼 테이블에 각 심볼의 값을 기록한다. 이때에 재배치해야 할 심볼들을 판별하여 재배치의 여부를 표시한다.

두 번째 패스에서는 원시 프로그램의 명령어의 2진 코드를 찾아 기계어 코드로 만들고, 첫 번째 패스에서 만들어진 심볼 테이블을 바탕으로 심볼의 forward reference가 있을 경우 심볼 테이블에서 심볼 대신 심볼의 값을 기입한다. 이러한 과정을 통해서 두 번째 패스에서는 목적 프로그램을 만든다.

(그림1)은 이중 패스 어셈블러의 동작이다[3].



(그림1) 이중 패스 어셈블러 동작

3. 개선된 SIC/XE어셈블러 시뮬레이터

오픈 소스 SIC/XE어셈블러 시뮬레이터는 여러 오류를 포함하고 있었으며, 다양한 기능을 제공하였지만 학습용으로 사용하기에는 기능이 부족하였다. 따라서 본 논문에서는 오픈 소스 SIC/XE어셈블러 시뮬레이터의 오류를 분석하고 일부 수정하였으며, 사용자의 학습을 위하여 리터럴 테이블, 심볼 테이블, 목적 코드, 오류 메시지 출력 기능을 구현하였다.

3.1 오픈소스 SIC/XE 시뮬레이터 오류분석 및 수정

오픈 소스 SIC/XE어셈블러 시뮬레이터는 SIC 명령어만 포함되어 있는 코드에서는 문제없이 작동하였으나, 일부 SIC/XE명령어 및 외부 심볼이 포함된 코드에서는 비정상적으로 작동되거나 실행이 중단되는 경우가 발생하였다. 발생하는 오류의 종류로는 예외 처리 오류, 중복 오류가 있었다. 예외 처리 오류가 발생하는 부분은 매개변수 또는 입력 값 등에 대해 적절한 예외 처리를 하지 않아 발생한 오류로, 해당하는 메소드에 NULL Pointer Exception 또는 IO Exception에 관한 예외 처리하여 오류를 수정하였다.

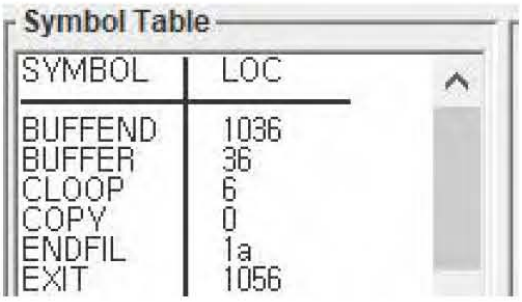
그러나 외부 심볼에 대하여 중복 오류 및 심볼을 찾을 수 없는 오류는 계속해서 발생하였다. 외부 심볼 테이블, 심볼 테이블 등이 구현되는 함수 안에서 각 테이블의 위치 및 이름 등의 정보를 출력해봄으로써 테이블이 정상적으로 구현되는지 테스트하였다. 그 결과 외부 심볼이 정상적으로 외부 심볼 테이블에 들어가지 않는 것을 발견하였다. 모든 외부 심볼이 외부 심볼 테이블에 포함되어야 하나 첫 번째 또는 두 번째 외부 참조 심볼이 포함되지 않아 코드가 진행되면서 외부 참조 심볼을 사용해야 할 때 찾을 수 없는 오류가 발생하게 되었다. 또한 외부 심볼을 사용하면서 몇 개의 외부 정의 심볼들이 중복되게 심볼 테이블에 들어감으로써 다른 섹션에서 해당 심볼을 사용하게 될 때 충돌이 발생하였다. 이러한 오류들은 사용된 변수의 중복 사용 또는 매개변수의 잘못된 전달로 인해서 발생한 오류라 판단하였다. 이를 수정하고자 새로운 변수와 배열 등을 이용하여 변수를 따로 저장하는 등의 방법을 통해 일부 변수의 중복은 제거하였으나 완전히 수정하지는 못하였다.

3.2 오픈소스 SIC/XE 시뮬레이터의 기능 개선

기존 오픈 소스 SIC/XE 어셈블러 시뮬레이터는 CPU 레지스터에 저장된 값을 나타내며 디어셈블리와 메모리 입력 값을 시각화하여 보여준다. 이 기존 시각화된 화면 아래에 목적코드, 심볼 테이블, 리터럴 테이블과 그 아래 오류 메시지 화면을 볼 수 있다.

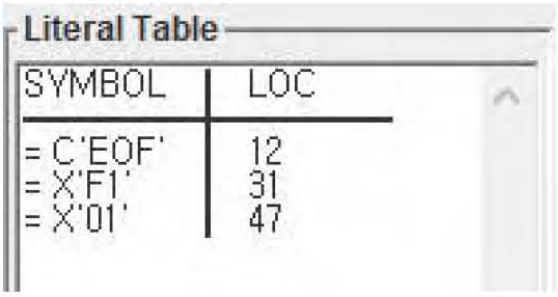
(그림2) 실행된 목적코드 화면

목적 코드는 어셈블러에 의해 만들어진 코드로 어셈블리어가 실행될 때 생성되는 데이터이다. 목적 코드는 해당 코드의 이름으로 시작되며 (그림2)와 같이 명령어와 명령어의 실행 주소, 데이터의 주소 등이 명시되어 있는 것을 확인할 수 있다.



(그림3) 실행된 심볼 테이블 화면

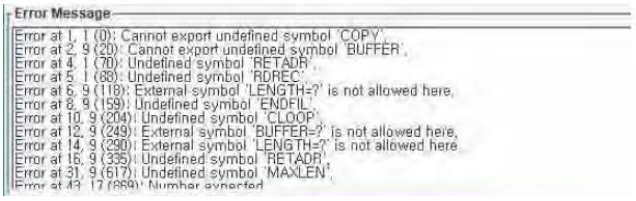
심볼 테이블은 (그림3)과 같이 실행된다. 어셈블리 코드에서 명시되어 있는 심볼들이 첫 번째 패스 동안에 생성되는 심볼 테이블에 저장되는데 심볼의 이름과 그 심볼의 위치를 저장한다. (그림3)에서의 'COPY' 심볼은 해당 어셈블리 코드의 이름에 해당하는 부분으로 시작주소인 0을 갖는 것을 확인할 수 있으며 'CLOOP' 라는 심볼은 6인 주소를 갖는다는 것을 시각적으로 보여준다.



(그림4) 실행된 리터럴 테이블 화면

리터럴 테이블은 '=' 형태로 식별되는 피연산자인 리터럴을 저장하는 테이블이다. 리터럴 테이블은 리터럴의 이름과 리터럴이 저장된 위치의 주소를 시각화한다.

(그림5) 성공한 오류 메시지 화면

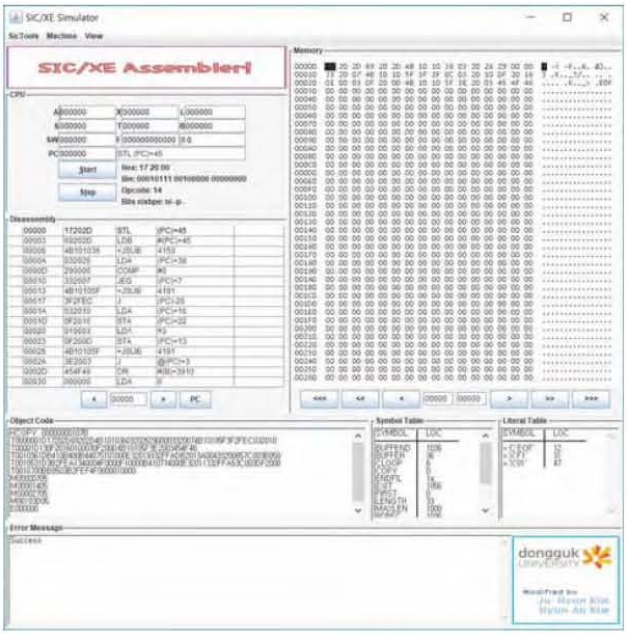


(그림6) 오류가 나타난 오류 메시지 화면

다음으로는 시뮬레이터의 맨 아래 레이아웃에 위치한 오류 메시지 화면이다. 정상적인 실행 코드를 활용하여 시뮬레이션을 성공하면 (그림5)와 같은 'success' 한 오류 메시지 화면을 확인할 수 있으며, 만약 잘못된 실행 코드를 활용하여 시뮬레이션에 실패한다면 (그림6)과 같이 오류의 원인과 그 위치를 시각적으로 제공한다.

4. 개선된 학습용 SIC/XE어셈블러 시뮬레이터 테스트 및 구현 결과

(그림7)은 본 논문의 SIC/XE 시뮬레이터를 특정 어셈블리 코드로 실행시킨 동작이다. 좌측 상단의 CPU 화면에서는 각 레지스터에 저장되는 정보가 출력되며 'Start'와 'Stop' 버튼을 활용하여 특정 시점에서 각 레지스터에 저장되어 있는 정보를 확인할 수 있어 어셈블리 코드의 흐름을 파악하기에 용이하다. asm 코드를 로드하면 CPU 화면 아래 'Disassembly' 화면에 코드 내용이 적재된다. 코드 각 줄의 명령어와 위치, 목적코드가 시각화되며 이 표를 이용하여 그 아래의 'Object Code' 의 결과와 비교하여 내용을 학습할 수 있다. 우 측 상단의 'Memory' 화면은 실제 메모리에 저장되는 내용을 확인할 수 있으며 'Disassembly'의 첫 목적 코드인 '17202D'부터 메모리에 적재되는 것을 확인할 수 있을 것이다. 메모리 아래에 위치한 두 개의 테이블은 심볼 테이블과 리터럴 테이블이다.



(그림7) 전체 실행 화면

어셈블러의 첫 번째 패스 동안 생성된 내용이 명시되어 있다. 각각의 심볼이 명시된 위치를 테이블을 통해 확

인하며 코드를 따라 분석하는 과정이 용이해질 수 있다. 또한 '='로 식별되는 리터럴을 따로 추출하여 작성한 테이블인 리터럴 테이블은 피연산자가 저장된 위치를 가지고 있으며, 해당 위치로 이동할 수 있는 지표가 되는 테이블이다. 마지막으로 하단에 위치한 'Error Message' 화면은 이러한 실행 과정이 성공하면 오직 'Success'를, 오류가 발생하면 오류의 원인을 규명하고 오류가 발생한 위치를 출력하여 실행 코드의 문제점을 분석할 수 있다. 예를 들어 "Error at 10, 9 (204): Undefined symbol 'CLOOP'" 와 같은 오류가 'Error Message' 화면에 출력된다면, 'CLOOP'라는 심볼이 심볼 테이블에 명시되어 있지 않은 확인할 수 없는 심볼이라는 것을 확인할 수 있다. 따라서 사용자는 이러한 오류를 수정하여 다시 코드를 실행할 수 있다.

5. 결론

본 논문이 제안한 학습용 SIC/XE어셈블러 시뮬레이터는 기존의 SIC/XE 어셈블러 시뮬레이터에 포함된 오류를 분석하고, 일부 오류를 수정함으로써 더 원활한 실행이 가능하게 하였다. 또한 기존 SIC/XE어셈블러 시뮬레이터에 테이블들과 목적코드, 오류 메시지 등의 추가함으로써 시각적 편리성을 높여 학습용에 적합하도록 디자인하였다. 기존 시뮬레이터는 CPU에 적재된 내용과 디어셈블리, 메모리 화면만이 포함되어 있어 어셈블러에 대한 어느 정도의 지식을 가지고 있는 사용자에게 적합하였지만, 어셈블러에 대해 초보자는 해당 시뮬레이터를 이용하기에 어려움이 많다. 따라서 본 논문에서는 SIC/XE에 입문한 사용자가 실행 코드와 실행 과정을 비교해가며 학습할 수 있도록 시뮬레이터가 진행되기 위해 사용되는 가공되지 않은 데이터들을 시각화함에 중점을 두었다. 따라서 사용자들은 실행하고자 하는 asm 파일을 로드하면, 'Disassembly' 화면에 코드가 나타날 것이고 이를 실행하면 CPU화면의 각 레지스터에 적재되는 내용과 메모리에 저장되는 내용을 확인함과 동시에 목적 코드의 내용을 비교하며 심볼, 리터럴 테이블을 응용할 수 있다.

참고 문헌

[1] http://jurem.github.io/SicTools/

[2] Leland. L. Beck, "System Software: An Introduction to System Programming, 3rd Ed.", Addison Wesley Longman, 1997.

[3] Jurij Mihelic, Dr.Tomaz Dobravec, "SICSIM : A Simulator of the Educational SIC/XE Computer for a System Software Course", pp.137 - 146, Vol.23, No.1, Jan. 2015.