Extension of LMS to Practice by Simulator

Kazuhiro MURAMATSU[a*]
[a]*College of Science and Technology, Royal University of Bhutan, Bhutan*
*muramatu@cst.edu.bt

Abstract: Focused on practice of an 8085 microprocessor, in this paper, the function of a LMS (Learning Management System) is extended to practices. Firstly, an 8085 microprocessor simulator which simulates the 8085 microprocessor trainer is developed in PHP (Hypertext Preprocessor) language, because the Moodle-based LMS of our college has been built on PHP. Next, the 8085 microprocessor simulator is implemented on the LMS. Finally, the microprocessor simulator is used for practices of the microprocessor-related subjects instead of the microprocessor trainer.

Keywords: LMS, Moodle, 8085 Microprocessor, Simulator, PHP

1. Introduction

At present, an open-source LMS [Learning Management System] is becoming popular in institutions of higher education. For example, Sakai CLE [Collaboration and Learning Environment] is used by over 350 organizations. Also, Moodle [Modular Object-Oriented Dynamic Learning Environment] has been used on over 68,000 registered sites and the number of users is over sixty million. These LMSs are used for lectures, tutorials, and assignments.

On the other hand, practices and experiments are considered as important at science and technological institutions, and students promote a better understanding of lectures and master practical skills from them. However, practical devices or experimental equipment are usable only for practice or experiment hours. Namely, it is impossible for students to use them to prepare or review practices. Therefore, we have started to develop simulators which simulate the above devices or equipment, and these simulators are implemented on a LMS. This makes it possible for students to prepare or review practice by the use of the simulator on the LMS.

However, there is a variety of equipment for practice or experiment. In this paper, we focus on the practice of an 8085 microprocessor. The 8085 microprocessor has a simple architecture and an adequate instruction set (Gaonkar, 2002). Therefore, the 8085 microprocessor is an excellent teaching material to teach basic concepts and programming concepts of a microprocessor, and it is one of the most widely used microprocessor for the subject of microprocessor in higher education. Really 8085 microprocessor trainers are used for the practices of the microprocessor-related subjects at our college.

Firstly, we develop the 8085 microprocessor simulator on PHP: Hypertext Preprocessor, because the Moodle-based LMS of our college has been built on PHP. Secondly, the simulator is implemented on the LMS, and it is used on the LMS instead of the microprocessor trainer.

2. Development of the 8085 Microprocessor Simulator

The 8085 Microprocessor Trainer

At present, 8085 microprocessor trainer kits made by Minmax Electronics in India are used as practical devices in the practices of the microprocessor-related subjects at our college. The 8085 microprocessor has only 246 instruction sets. The students translate these instructions to Hex machine codes by the use of a manual, and input the each code to the trainer kits. Then, a set of machine codes is executed and the result from the trainer kits is confirmed.

Development of the 8085 Microprocessor Simulator in PHP

As mentioned above, the 8085 microprocessor is an excellent teaching material. Therefore, many 8085 microprocessor simulators such as 8085 SimuKit, Gsim85 and GNUSim8085 have been developed (Chehab, et al., 2004; Dahiya & Kumar, 2009; Sridhar, 2003). However, these simulators have been built on Windows or Linux, not on Web development languages. Therefore, they are not suitable to a web-based LMS.
Thus, we develop the 8085 microprocessor simulator on PHP, which simulates the 8085 microprocessor trainer, because the Moodle-based LMS of our college has been built on PHP. We adopt the following three policies on the development of the simulator:

1) Users input assembler programs on the simulator, instead of inputting machine programs on the trainer.
2) In order to decrease input mistakes, the simulator makes it impossible to input grammatically improper components of the assembler program.
3) Taking account of a tablet computer as a client, users can input the assembly program only using a mouse, not a keyboard.

Our microprocessor simulator is composed of four modules: Main module, Input module, Execution module, and GUI [Graphical User Interface] module. Each module has following functions.

Main Module

Main module translates an assembly language program sent from Input module to Hex machine codes. A set of the machine codes is stored in a cookie on a client side. Also, the module sends the machine codes to Execution module when the program is executed.

Input Module

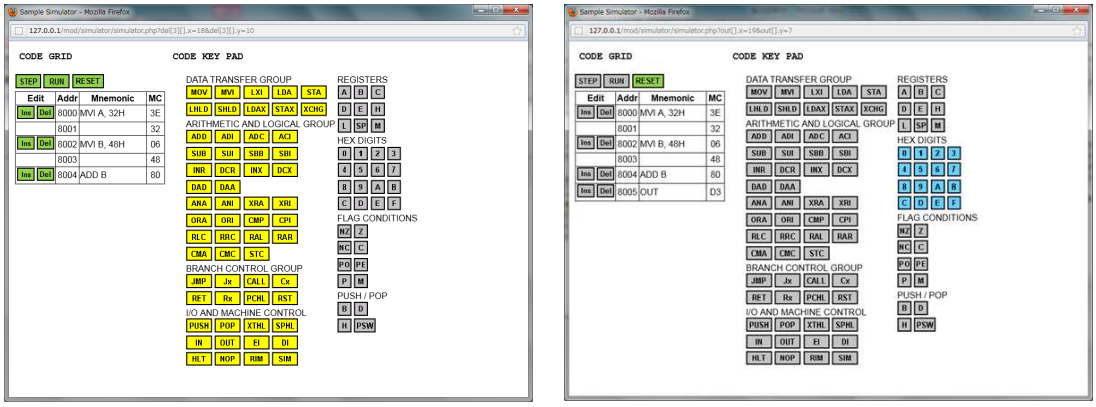
Input module transfers the assembly language program from GUI module to Main module. Also, the module sends information which instructions of the assembly program are possible to be input.

Execution Module

Execution module executes a set of Hex machine codes received from Main module, and transfers the results to GUI module. Execution module has two execution modes: RUN mode and STEP mode. The RUN mode batch processes the machine code and transfers the results to GUI module at a time. On the other hand, the STEP mode executes the machine code and transfers each result to GUI module one step at a time.

GUI Module

Users input the assembly program by GUI module, and GUI module sends the input program to Input module. Figure 1 shows input windows of the assembly program. Each instruction of the program consists of an opcode which is the task to be performed and an operand which is the data to be operated on. The operand input and the opcode input are shown in Fig. 1 (a) and (b), respectively.



(a) Opcode Input (b) Operand Input

Figure 1 Input Windows of an Assembly Program

Also, GUI module displays the results of Hex machine codes transferred from Execution module. Figure 2 shows an output window of the machine codes.

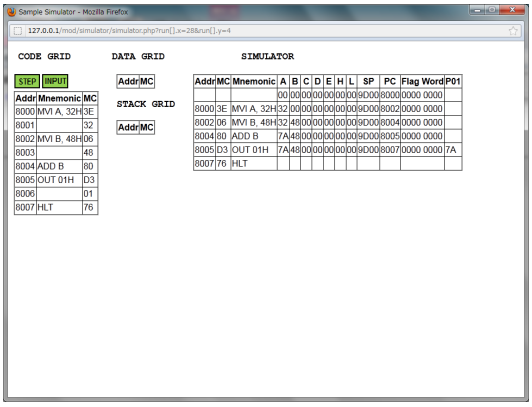


Figure 2 Output Window of Hex Machine Codes

3. Implementation of the 8085 Microprocessor Simulator on LMS

After the development of the simulator, it is implemented on the Moodle-based LMS of our college as an activity module. Generally, a new activity module of the LMS is created by modifying the NEWMODULE template which can be downloaded from the Moodle website (Moore, et al., 2010). The NEWMODULE template is composed of four folders and fifteen files. Main modifications from the template are described as follows.

Input Form

The file 'mod_form.php' provides the input form when teachers register the simulator as an activity module on their course. The file is modified to be necessary to input the simulator name and its instructions. Also teachers can define a period that the simulator is available.

View Form

The simulator is available for students after the registration by teachers. The file 'view.php' is responsible for displaying the simulator as an activity module when students access it. We modify it to display the sentence 'Click here to run 8085 Simulator' on the window. If the sentence is clicked, the 8085 simulator starts on a pop-up window. The following line is added to the 'view.php' file.

link_to_popup_window
("/mod/simulator/simulator.php","simulator$course->id$simulator->id$groupparam",
"$strentersimulator", 650, 950, get_string('modulename', 'simulator'));
Here, 'simulator.php' is correspondent to Main module in Section 2.

Local Language Form

The file 'lang/en_utf8/simulator.php' is called a local language file, and customizes localization strings with a hash array called $string. Here, we define several values for the $string hash as follows.

```
$string['simulatorname'] = '8085 Simulator';
$string['instructions'] = 'Instructions';
$string['entersimulator'] = 'Click here to run 8085 Simulator';
```

After finishing the above implementation, the 8085 microprocessor simulator is being used for the practices of the microprocessor-related subjects at our college.

4. Conclusion

We have developed the 8085 microprocessor simulator on PHP, and implemented it on the LMS of our college. At present the microprocessor simulator is used four times and the microprocessor trainer is used six times in ten practices of the microprocessor-related subjects at our college. After implementing additional functions such as saving function of an assembly program and calculating function of total T-states and total machine cycles, all practices will be done by the microprocessor simulator.

Acknowledgements

I would like to thank Mr. Yeshi Wangchuk for the implementation of the 8085 microprocessor simulator on the Moodle-based LMS of our college.

References

[1] Gaonkar, S.R. (2002). *Microprocessor Architecture, Programming, and Applications with the 8085, 5th edition*. Upper Saddle River, New Jersey: Prentice Hall.
[2] Chehab, A., Hanna, S., Kabalan, Y.K. & El-Hajj, A. (2004). 8085 microprocessor simulation tool "8085 SimuKit". *Computer Applications in Engineering Education*, *12*(4), 249-256.
[3] Dahiya, V.K. & Kumar, H. (2009). GSim85 - An 8085 Microprocessor Simulator. Retrieved from http://gsim85.sourceforge.net
[4] Sridhar, R. (2003). GNUSim8085 - 8085 Simulator for Linux and Windows. Retrieved from http://gnusim8085.org/index.php
[5] Moore, J. & Churchward, M. (2010). *Moodle 1.9 Extension Development*. Birmingham, UK: Packt Publishing.