# A Peer-Assessment Based Approach for Teaching Microprogramming

**Kanika[1], Shampa Chakraverty[2], Pinaki Chakraborty[3]**

[1,2,3] Department of Computer Science and Engineering, Netaji Subhas University of Technology, New Delhi 110078, India
[3] pinaki_chakraborty_163@yahoo.com

**Abstract:** The course on microprocessors introduces undergraduate computer science students to hardware-level programming. The course was taught by the authors to 130 students in context of the 8085 and 8086 microprocessors in the Spring semester of 2019. The students executed their programs on hardware kits, and participated in a double-blind peer-assessment exercise in which they assessed and rated programs written by their peers and also advised them on improving the efficiency and readability of their programs. It was found that the peer-assessment exercise helped the students to learn better and score 6.97% higher marks in examination (Z=2.68, P<0.05). There was a moderately strong correlation (r=0.43, P<0.05) between the ratings received by the students from their peers and their score in the post-intervention test. The students felt that they benefitted from the advice they received from their peers and 78% of the students agreed with the ratings they received. The students felt that assessing programs written by others also helped them in improving their own programming skills.

**Pinaki Chakraborty**

Department of Computer Science and Engineering, Netaji Subhas University of Technology, New Delhi 110078, India
pinaki_chakraborty_163@yahoo.com

**Keywords:** Computer science education, microprocessor, microprogramming, peer-assessment.

## 1. Introduction

Contemporary undergraduate curricula on computer science emphasize on teaching programming using high-level languages. However, a course on microprocessors is included in the curriculum at most universities. This course provides students a rare opportunity to learn hardware-level programming. Unlike the courses on microprocessors offered to students of other engineering disciplines, the focus of the course offered to computer science students is to demonstrate to them how the hardware of a computer xecutes programs.

The course on microprocessors is considered to be difficult by students as well as instructors. As a result, instructors have been using specialized tools and techniques to teach the course. Four techniques that are commonly used for teaching the course on microprocessors are as follows.

- Computer-based emulation. Some instructors interface the microprocessor they are teaching about and other devices with a computer, and recommend students to execute their programs on the microprocessor. For example, Vallejo et al. (1992) developed a system that allowed students to perform hardware and software experiments on the

68010 microprocessor. Recently, Qaralleh and Darabkh (2015) taught a course on microprocessors using computer-based emulation of the 8086 microprocessor. Cadenas et al. (2015) also followed a similar approach.

- Simulation software. Some instructors use software tools to simulate the working of microprocessors. The programs are interpreted by the simulation software and not actually executed on the microprocessor of interest. For example, Diab and Demashkieh (1991) used a software package consisting of an assembler and a simulator to teach microprogramming. The tool used animation to explain the internal working of the Z80 microprocessor and execution of programs by the same.

- Project-based learning. Instructors have also taught the course on microprocessors using project-based approaches. For example, Hamrita and McClendon (1997) and Larruscain et al. (2018) emphasized on designing microprocessor-based solutions to real-world problems. Alternatively, Jeon (2000) taught a course on microprocessors to electrical and computer engineering students in which the students had to develop a simple microcomputer of their own. Project-based approaches were also used by Al-Dhaher (2001), Joseph et al. (2016) and Rodriguez-Sanchez et al. (2020).

- Online assessment systems. Instructors also take help of online assessment systems to evaluate and grade students. For example, Merat and Chung (1997) used Web-based system quizzes for teaching the course on microprocessors. The system automatically evaluated and graded the answers submitted by the students. The system could provide timely feedback to the students irrespective of the class size.

Providing constructive and timely feedback to students on their work is crucial in any educational system (Trengove, 2017). However, it becomes difficult for instructors to assess the work of students and provide them feedback if the class size is large. The problem becomes acute in programming-related courses where instructors require more time for evaluating assignments (Kanika et al., 2020). Consequently, some instructors recommend the students attending their courses to assess the work done by one another. Peer-assessment is a meta-cognitive learning technique that allows learners to evaluate the quality of the work done by their peers and give them suggestions for improvement (Smith et al., 2002). In peer-assessment, students play two roles at the same time and gain differently as assessors and assessees (Cevik, 2015). Peer-assessment helps in motivating students to learn programming (Sitthiworachart and Joy, 2003), and improving their interpersonal skills (Sluijsmans et al., 2002), understanding of the subject (Reinholz, 2016) and critical reflection skills (Wanner and Palmer, 2018). This paper presents a study on the utility of peer-assessment in teaching microprogramming to undergraduate computer science students attending a course on microprocessors.

## 2. Materials and Methods

Fig. 1. A hardware kit used by the students to execute their programs. A course on microprocessors was offered by us at Netaji Subhas University of Technology in Spring 2019. The course was taught by the third author of this paper, while the first- and the second authors acted as consultants. The course was attended by 130 undergraduate computer science students. The students had attended courses on digital electronics and computer architecture in the previous semesters. The students were taught the fundamentals of microprocessors and microprogramming using the 8085 and 8086 microprocessors in the course. The course was allocated three hours of lecture and two hours of laboratory work per week. The students were provided with a list of 20 programming tasks. The students had to use instructions of different addressing modes, iterations, subroutines and recursion to write those programs. The list included programs for calculating the square root of a given number, list
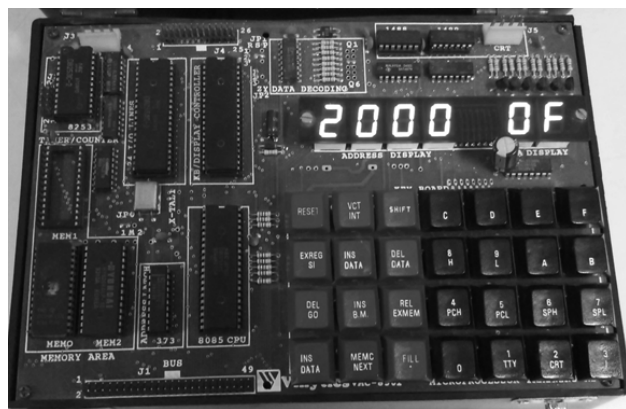


**Fig. 1: A hardware kit used by the students to execute their programs.**

processing operations like searching and block copying, implementing algorithms like bubble sort and quicksort, and using recursion to calculate the factorial of a number and list Fibonacci numbers. The students had to execute their programs on hardware kits (Fig. 1) in the laboratory and note down the programs in their laboratory notebooks. Although the correctness of the programs written by the students was checked by the instructor, the programs suffered from minor runtime errors and unsatisfactory programming styles at times. Therefore, a peer-assessment exercise in which students assessed and rated programs written by their peers was incorporated in the course.

The course was taught conventionally for four weeks and then a pre-intervention test to assess the knowledge of the students was conducted. The students were then asked to upload scanned copies of their laboratory notebooks in an online shared folder. The students were then asked to see the laboratory notebooks of three of their peers. They were asked to rate the programs written by their peers on a 5-point Likert scale. The students were also asked to advise their peers on how to improve their programs in terms of efficiency and readability. The peer-assessment exercise was double-blind, i.e. the students did not know whose programs they were assessing and who were assessing their programs. Each student received ratings and advices from three of their peers. A post-intervention test was conducted eight weeks after the beginning of the course. The two tests were of 15 marks each. The answer sheets of the two tests were evaluated by the instructor and the same were discussed with the students. A survey was conducted at the end of the exercise where the students were asked if they agreed with the ratings they received from their peers and how much they benefitted from the peer-assessment exercise. Fig. 2. Scores (Mean + SE) received by the students in the pre-intervention test and the post-intervention test.

The effect of the peer-assessment exercise was evaluated in four ways as follows. First, the effect of the exercise on the performance of the students in examination was studied. The scores that the students received in the pre-intervention test and the post-intervention test were compared. Second, the advice provided by the students to their peers for improving their programming skills was analyzed. The advices were categorized into four categories, viz. general advice, advice for improving only the efficiency of programs, advice for improving only the readability of programs, and advice for improving both efficiency and readability of programs. Third, the correlation between the ratings that the students received from their peers and the score they received in the post-intervention test was determined. The Pearson's correlation coefficient was calculated for the two variables. Fourth, the feedback provided by the students at the end of the exercise was analyzed.

## 3. Results

It was observed that the peer-assessment exercise helped the students to perform better in examination (Fig. 2). The students scored 6.97% more marks in the post-intervention test than in the pre-intervention test ($Z=2.68$, $P<0.05$). The analysis of the advice provided by the students to their peers revealed that 25% advices were related to improving readability of programs, 12% advices were related to improving efficiency of programs, 7% advices were related to improving both efficiency and readability of programs, and the remaining advices were general in nature (Fig. 3). It was observed that there existed a moderately strong correlation ($r=0.43$, $P<0.05$) between the ratings received by the students from their peers and the scores received by them in the post-intervention test. In the survey conducted at the end of the exercise, 78% students agreed with the ratings they received from their peers (Fig. 4a). It was observed that 47% students felt that the advice they received from their peers helped them significantly (high or very high rating) in improving their programming skills (Fig. 4b). Alternatively, 68% students felt that assessing programs written by their peers helped them significantly (high or very high rating) in improving their own programming skills.

## 4. Discussion

Although simulation software and online assessment systems can help in enhancing the course on microprocessors (Diab and Demashkieh, 1991; Merat and Chung, 1997), the authors of this paper strongly recommend the use of hardware experiments to teach microprogramming. Computer science students attend several courses where they perform experiments using software only. The course on microprocessors should be used to provide them an opportunity to work with hardware. However, project-based learning approaches can make the course on microprocessors too difficult for computer science students who are more interested in the programming aspect of microprocessors. As a result,
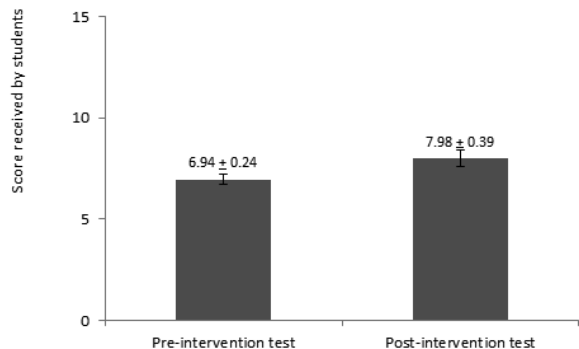
**Fig. 2:Scores (Mean + SE) received by the students in the pre-intervention test and the post-intervention test.**
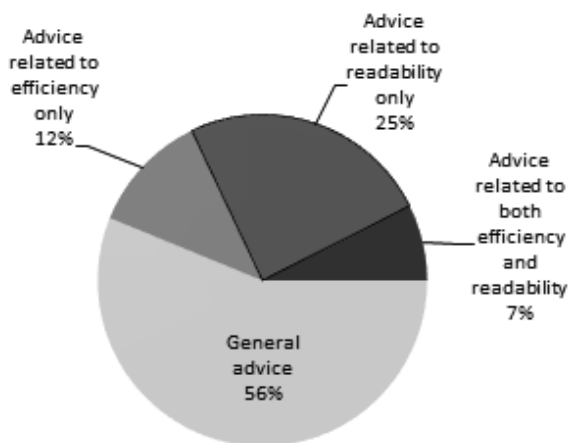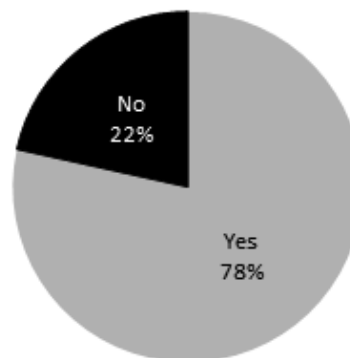


**Fig. 3 : Types of advice provided by the students to their peers (130 students participated in the peer-assessment exercise and each student advised three other students).**
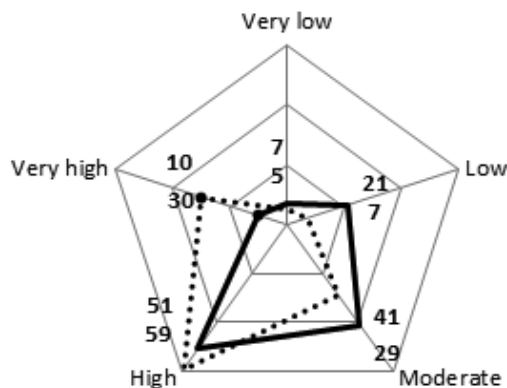


**Fig. 4 :Feedback received from the students (N=130) at the end of the peer-assessment exercise. (a) Proportion of the students who agreed with the ratings they received from their peers and (b) the factors that helped them in improving their programming skills.**

the use of simple hardware kits or computer-based emulation (Vallejo et al., 1992; Qaralleh and Darabkh, 2015) to teach the course is recommended.

The course on microprocessors was taught by us using hardware kits and incorporated a peer-assessment exercise to enhance the course. The results revealed that the peer-assessment exercise helped the students to understand the concepts as observed earlier by Reinholz (2016). The peer-assessment exercise ensured that the students receive detailed, constructive and timely feedback. It was observed that the students helped their peers to improve the efficiency and readability of their programs. Cevik (2015) has earlier claimed that students learn as both assessors and assessees in peer-assessment exercises. The results suggest that students learnt slightly better by assessing programs written by their peers than from the advices they received from their peers. It was also observed that a majority of the (a)

(b)students who attended our course agreed with the ratings they received from their peers thus showing that peer-assessment is acceptable to students as an instructional technique as observed by Trengove (2017).

## 5. Conclusion

The course on microprocessors was taught by us to undergraduate computer science students focusing on the programming of simple 8- and 16-bit microprocessors. The students performed their

experiments on hardware kits and assessed programs written by one another. The peer-assessment exercise helped the students to learn better and score higher in examination. There was a moderately strong correlation between the ratings received by the students from their peers and the score they received in the post-intervention test. The students advised their peers on improving the efficiency and readability of their programs. Although the students agreed with the ratings they received from their peers, they felt that they learnt more while assessing programs written by others than from the advice they received from their peers.

## Acknowledgement

## References

[1] Al-Dhaher, A. H. G. (2001) Integrating hardware and software for the development of microcontroller-based systems. Microprocessors and Microsystems, 25(7), 317-328.

[2] Cadenas, J. O., Sherratt, R. S., Howlett, D., Guy, C. G. and Lundqvist, K. O. (2015) Virtualization for cost-effective teaching of assembly language programming. IEEE Transactions on Education, 58(4), 282-288.

[3] Cevik, Y. D. (2015) Assessor or assessee? Investigating the differential effects of online peer assessment roles in the development of students' problem-solving skills. Computers in Human Behavior, 52, 250-258.

[4] Diab, H. B. and Demashkieh, I. (1991) A computer-aided teaching package for microprocessor systems education. IEEE Transactions on Education, 34(2), 179-183.

[5] Hamrita, T. K. and McClendon, R. W. (1997) A new approach for teaching microcontroller courses. International Journal of Engineering Education, 13(4), 269-274.

[6] Jeon, J. W. (2000) A microprocessor course: Designing and implementing personal microcomputers. IEEE Transactions on Education, 43(4), 426-433.

[7] Joseph, D., Kaur, G. and Chakraborty, P. (2016) An exercise on hardware/software codesign following the RISC model. Computer Applications in Engineering Education, 24(2), 305-312.

[8] Kanika, Chakraverty, S. and Chakraborty, P. (2020) Tools and techniques for teaching computer programming: A review. Journal of Educational Technology Systems, 49(2), 170-198.

[9] Larruscain, G., Tapia, G., Susperregui, A. and Martinez, M. I. (2018) Student-tailored final year project on microcontroller-based hardware-in-the-loop speed control of a wind generator. International Journal of Electrical Engineering & Education, 55(3), 213-233.

[10] Merat, F. L. and Chung, D. (1997) World Wide Web approach to teaching microprocessors. Proceedings of the Twenty-seventh Annual Conference on Frontiers in Education, vol. 2, pp. 838-841.

[11] Qaralleh, E. A. and Darabkh, K. A. (2015) A new method for teaching microprocessors course using emulation. Computer Applications in Engineering Education, 23(3), 455-463.

[12] Reinholz, D. (2016) The assessment cycle: A model for learning through peer assessment. Assessment & Evaluation in Higher Education, 41(2), 301-315.

[13] Rodriguez-Sanchez, M. C., Chakraborty, P. and Malpica, N. (2020) International collaborative projects on digital electronic systems using open source tools. Computer Applications in Engineering Education, 28(4), 792-802.

[14] Sitthiworachart, J. and Joy, M. (2003) Web-based peer assessment in learning computer programming, Proceedings of the Third IEEE International Conference on Advanced Learning Technologies, pp. 180-184.

[15] Sluijsmans, D. M. A., Brand-Gruwel, S. and van Merrienboer, J. J. G. (2002) Peer assessment training in teacher education: Effects on performance and perceptions. Assessment & Evaluation in Higher Education, 27(5), 443-454.

[16] Smith, H., Cooper, A. and Lancaster, L. (2002) Improving the quality of undergraduate peer assessment: A case study from psychology. Innovations in Education and Teaching International, 39(1), 71-81.

[17] Trengove, E. (2017) Peer interaction as mechanism for providing timely and accessible feedback to a large undergraduate class. International Journal of Electrical Engineering Education, 54(2), 119-130.

[18] Vallejo, F., Harbour, M. G. and Gregorio, J. A. (1992) A laboratory for microprocessor teaching at different levels. IEEE Transactions on Education, 35(3), 199-203.

[19] Wanner, T. and Palmer, E. (2018) Formative self- and peer assessment for improved student learning: The crucial factors of design, teacher participation and feedback. Assessment & Evaluation in Higher Education, 43(7), 1032-