Pedagogy and Technology to Enhance Formative Assessment and Feedback: Pilot Study for an Undergraduate Large Class in Digital Fundamentals

AKASH Kumar*, Rajesh C. PANICKER and Ashraf KASSIM

Department of Electrical and Computer Engineering, National University of Singapore, Singapore 117583

*Corresponding Author’s E-mail: akash@nus.edu.sg

Keywords: Formative Assessment, Feedback, Digital Fundamentals, Web-based Toolkit, Undergraduate, Large Classes

Extended Abstract

Introduction

This paper aims to (a) justify the need for an intervention in teaching digital fundamentals to a large undergraduate class, (b) describe how formative assessment and feedback could be enhanced in the module by adopting the principles of good feedback by Nicol and McFarlane (2004), and with the support of a web-based toolkit.

The module EE2020 “Digital Fundamentals” is taught to undergraduates at the Department of Electrical and Computer Engineering. It is a first course (with an average class size of 300 students) that introduces fundamental digital logic, digital circuits, and programmable devices to the students. One of the learning outcomes of EE2020 is to enable students to design, simulate and realise simple circuits using selected integrated design environments (IDEs) based on their understanding of the fundamentals of digital design.

However, most students taking EE2020 are being challenged by the steep learning curve as they attempt to familiarise themselves with the currently available IDEs such as Xilinx Webpack or Altera Quartus, in order to use them as tools for editing and simulating designs within the six to eight weeks during the semester when the topic is taught. Another challenge the instructor would encounter when teaching the module to a large class is the difficulty in providing immediate feedback to all students when they are writing programming code, a critical element when learning programming. There are instances where students think that they have the correct answer for all scenarios, when in fact the timing, which is very important in hardware design, may be incorrect. While manual checking of the code and its resulting simulation waveforms is the most desirable solution in such cases, it is not feasible for large classes.

With the aim of solving these two issues encountered in teaching this module, a user-friendly web-based toolkit was developed to serve as an alternative IDE for students to edit and simulate designs. The key affordance of the toolkit is for students to have the opportunity of practicing and applying the principles of fundamental design they learnt with sufficiently timely feedback provided. The toolkit is lightweight, has an installer with a file size of less than 20MB and is multi-platform, including running on MacOSX. It is equipped with an automated checker that allows students to submit the code to a server and receive feedback on whether they have written the correct programming code. For the instructors, the toolkit has the following features which makes it feasible to implement in large classes:

- A lightweight cross-platform open source IDE for editing, compiling and simulating VHDL¹ programmes.
- A client-server system (integrated into the IDE) to do automated checking of VHDL programmes.
- An interface which teaching staff can use to update the database of problems whose solutions can be automatically checked.
- A logging system to keep track of student submissions, which can be useful in programming tests.

In order to effectively enhance the module's formative assessment and feedback for students, the toolkit has been introduced into EE2020 by applying the principles of good practice in providing feedback developed by Nicol and MacFarlane (2006).

**Formative Assessment in EE2020**

Formative assessment, according to Sadler (1989), refers to ‘assessment that is specifically intended to generate feedback on performance to improve and accelerate learning’. Such feedback, whether conveyed in class or on assignments, enable students to restructure their understanding/skills and build more powerful ideas and capabilities (Nicol & Macfarlane, 2004; Zimmerman, 2002; Pintrich & Zusho, 2002). The construct of self-regulation refers to the degree to which students can regulate aspects of their thinking, motivation and behaviour during learning (Pintrich & Zusho, 2002). There has been a constant demand in higher education for graduating students to be equipped with the capacity for lifelong learning that enable learners to continually upgrade their skills and knowledge through their own self-motivation and learning (Bennett, Dunne & Carre, 1999; Dearing, 1997). An important aspect of achieving this goal is to help students take greater responsibility when it comes to managing their own learning; this is done by helping them become more strategic learners through self-regulated learning (McMahon & Luca, 2001). Being equipped with skills such as self-regulation is important because a major function of education is the development of lifelong learning skills (Zimmerman, 2002).
Formative assessment and feedback, incorporated into EE2020 via the web-based toolkit, have been designed based on the following principles of good feedback, developed by Nicol and MacFarlane (2006):

1. **Helps clarify what good performance is (via goals, criteria, expected standards).** Students are shown ample examples of model answers to both circuit design and VHDL programming. These solutions are posted on the web for easy access by the students.

2. **Facilitates the development of self-assessment (reflection) in learning.** The VHDL tool and the feedback plug-in is an excellent means for students to quickly write VHDL programmes and get feedback on the correctness through the automated tool. This self-assessment allows students to improve their design to handle all possible input scenarios.

3. **Delivers high-quality information to students about their learning.** Good quality external feedback is information that helps students troubleshoot their own performance and to self-correct. As mentioned earlier, the toolkit has an automated checking feature which can give students immediate feedback about the code that they have written. Figure 1 shows the jEdit editor portion of the toolkit with a custom plugin designed for VHDL compilation, simulation and so on. The editor area shows a VHDL code snippet with highlighted syntax. The custom plugin is docked below the editor area, with user interface (UI) controls for performing various tasks. The developed IDE plugin supports automated checking. Once the programme has been compiled, students can submit it after indicating the corresponding problem identifier. The server compiles the submitted programme, carries out the simulation using the input test cases for the problem and compares it with the corresponding output cases. For each test case, a response is sent to the client indicating the correctness. Figure 1 shows the response for problem 2001, where the program produced correct output for three out of the four test cases.

4. **Encourages teacher and peer dialogue through learning.** The web-based toolkit includes an online discussion forum as shown in Figure 2. The forum allows students to view the solutions posted by the teachers and other students. The forum also allows them to comment on the posts by other students to encourage participation and peer learning.

5. **Encourages positive motivational beliefs and builds learners’ self-esteem.** Studies suggest that “motivation and self-esteem are more likely to be enhanced when a course has many low-stakes assessment tasks, with feedback geared towards providing information about progress and achievement, rather than high-stakes summative assessment tasks where information is only about success or failure, or about how students compare with their peers (e.g. grades)” (Nicol & MacFarlane, 2006). One of the strategies adopted in EE2020 is to provide peer feedback to students when they are designing small circuits for tutorial problems. This is further enhanced by immediate automated feedback when they start using the VHDL toolkit.

6. **Provides opportunities to close the gap between current and desired performance.** According to Sadler (1989), “the only way to tell whether learning results from feedback is for students to make some kind of response to complete the feedback loop” (p. 123). With the help of VHDL toolkit, students get an opportunity to improve their solutions to cover all the corner cases until their programme is correct.
7. Provides teachers with information that can be used to help shape the teaching. All student attempts and corresponding programmes are logged on the server for later analysis. This gives a detailed report to the module lecturers. For example, the average number of attempts required for a particular problem before the correct solution was provided, can be seen by the lecturers. This allows them to adapt their teaching and provide more help in topics which are perceived as challenging by students.

Figure 1. Integrated development environment with the plugin docked at the bottom.

Figure 2. Online forum for viewing, posting and discussing solutions.
Discussion

A survey was conducted to evaluate the usability and usefulness of the tool. Most questions were evaluated on a Likert scale from 1 to 5, with 5 being the most desirable. Table 1 shows the various survey questions and the average of the responses. Most users found the installation process rather easy with our custom installers for different platforms. None of the users found it difficult to install with 50% users finding it extremely easy to install. Most users (63% with a score of 4 and above) felt it was important to have a simple VHDL editor since existing tools are very bulky. Close to three-quarters of respondents found it easy (score of 4 or 5) to use the editor. The automated checking feature was well-received with close to 90% respondents feeling a strong need for it. A similar number of people felt that such a simple editor and automated checking feature can motivate and enhance the learning process. Some useful suggestions to improve the interface were also received during the feedback which will be incorporated in the future versions of the toolkit.

In the future, we plan to explore the possibility of using the IDE for the programming test conducted as a part of continuous assessment for EE2020. This will make the administration of the test easier, and the evaluation more objective. Since the accuracy of the program can be checked instantly, the evaluation workload of teaching assistants will be reduced. Further, we are working on an online platform which does away with the need for installing any software locally altogether. The students would be able to login to a portal which stores all their programmes and they can use any device (including mobile phones/tablets) to edit, compile, simulate and submit the solutions.

Table 1. Survey results of the developed toolkit.

<table>
<thead>
<tr>
<th>Question</th>
<th>Average Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>How would you rate the installation process</td>
<td>4.1</td>
</tr>
<tr>
<td>How important is it to have a light-weight IDE</td>
<td>3.8</td>
</tr>
<tr>
<td>How usable is the provided editor</td>
<td>3.9</td>
</tr>
<tr>
<td>How important is to have automated checking feature</td>
<td>4.4</td>
</tr>
<tr>
<td>How usable is the automated checking feature</td>
<td>3.9</td>
</tr>
</tbody>
</table>
Endnote
VHDL refers to VHSIC Hardware Description Language.

Acknowledgments
The authors would like to thank Ms Liew Shin Dee and Mr Alan Soong for providing extremely valuable input to improve the quality of the abstract.

References


