

Umsetzung eines Laborversuches auf LON-CAPA mit automatischer Bewertung von ABEL-Programmieraufgaben

von Sebastian Maier und Oliver Rod¹

Abstract: Im Labor für Datentechnik, FB Elektrotechnik der Hochschule Ostfalia, wurde für die Vermittlung von Hardwareprogrammierung durch die Programmiersprache ABEL ein Versuch abgeändert. Hierbei wurden bestehende Toolbook-Versuche auf die automatische Bewertung und Vermittlung mittels LON-CAPA und Praktomat umgestellt.

Gründe für eine Erneuerung des Versuches sind zum einen, den alten Versuch plattformunabhängig zu gestalten. Zum anderen soll gleichzeitig die Verwendung neuester Webtechnologien ermöglicht werden. Probleme mit Windows-Updates sorgten in der Vergangenheit für einen erhöhten Aufwand bei der Sicherstellung der Funktionstüchtigkeit des Laborversuches. Die Evaluation der Studentinnen und Studenten sollte ebenfalls erleichtert und eine Alternative zu den hohen Lizenzkosten des alten Versuchs gefunden werden. In diesem Paper wird gezeigt, wie die Umsetzung erfolgt und welche Vorteile dieser Umstieg mit sich bringt. Im Folgenden werden die Funktionsweise und der Ablauf des Labors vorgestellt.

1 Einleitung

Das Labor, in dem der Versuch durchgeführt wurde, befindet sich im Elektrotechnik Hauptstudium. Ein Team aus zwei Studierenden führt einmal pro Woche einen praktischen Versuch durch. Die Versuche sollen den Studentinnen und Studenten einen einfachen Zugang in verschiedene Bereiche vermitteln. Diese bestehen aus Grundlagen in Assembler Programmierung, der Logikanalyse, des Synthetisieren eines FPGAs und der Hardwareprogrammiersprache ABEL. Die jeweiligen Versuche bestehen aus drei Teilen: dem Grundlagenteil, mit anschließendem Abfragen des Wissens, dem Programmiereteil, zum Erlernen der Sprache und dem anschließenden Projekt zum Programmieren mit ABEL zu einem bestimmten Hardware- Aufbau.

Der Laborversuch zur Vermittlung von ABEL wurde 1994 im Rahmen einer Diplomarbeit erstellt. Teilweise wird für die Durchführung eine DOS-Umgebung benötigt. Die Einführung der Studierenden in die Thematik wurde mittels Toolbook² durchgeführt.

Nicht nur die Pflege und Korrigierbarkeit der Toolbook-Versuche sondern auch die Bereitstellung der Programmierumgebung ist durch Betriebssystem-Updates immer schwieriger geworden.

¹ Fakultät Elektrotechnik / Zell, Ostfalia HS

² Toolbook ist eine kostenpflichtige e-Learning Software die seit 1990 am Markt ist. Sie wird bis heute gepflegt und durch die Firma SumTotal Systems weiterentwickelt.

Im Anschluss an den Grundlagenteil zur ABEL-Programmierung steht eine größere Programmieraufgabe an. In dieser sollen die Studierenden einen PLD (Programmable Logic Device) programmieren und einen vorhandenen Aufbau mit ihrer erstellten Logik versorgen. Im alten Versuch wurden die Einreichungen mit Hilfe von Batch-Skripten kompiliert. Anschließend erfolgte ein Vergleich der Prüfsummen des Kompilates (JDEC-Datei) mit der Musterlösung, um die fachliche Richtigkeit der Aufgabe sicherzustellen.

Die Anforderungen an das neue System sind sehr unterschiedlich, es soll:

- als Alternative zu den Toolbook-Versuchen dienen,
- den Abel-Compiler für eine automatische Überprüfung nutzen,
- die einfache Pflege der Aufgaben gewährleisten,
- keine Lizenzgebühren beinhalten, besser unter Verwendung freier bzw. offener Software laufen, und
- bestehende Infrastruktur der Hochschule nutzen.

Im Kapitel 2 werden die verwendeten Systeme und deren technischer Ablauf beschrieben. Folgend wird im Kapitel 3 dessen Ergebnis ausgewertet und die Evaluation der zwei Testgruppen diskutiert. Abschließend folgt in Kapitel 4 der Ausblick.

2 Integration von LON-CAPA und Praktomat

Mit LON-CAPA steht an der Hochschule bereits ein kostenfreies Lern-Management-System zur Verfügung. LON-CAPA ist quelloffen und wird ständig weiterentwickelt. Darüber hinaus bietet es eine Vielzahl von Aufgabentypen (z.B. Multiple-Choice, Click-in-Bild). Weitere Arbeitsmittel bestehen beispielsweise aus Stundenplänen, vielen Kommunikationswerkzeugen (E-Mail, Chat und Foren) aber auch Möglichkeiten, innerhalb von Kursen Informationen zum Selbstlernen bereitzustellen. Eine wichtige Stärke ist zudem die Fähigkeit der Wiederverwendung und des Austausches von Aufgaben zwischen den Dozenten und verschiedenen Einrichtungen innerhalb des LON-CAPA Netzwerkes (vgl. [Ko08]).

Die Toolbook-Versuche konnten erfolgreich in LON-CAPA übertragen werden. Die Vermittlung von Inhalten mit anschließender Überprüfung durch Multiple-Choice-Fragen wurde beibehalten. Hierbei ist darauf geachtet worden, dass die Auswahl an Antworten auf drei beschränkt wird. LON-CAPA wählt dabei per Zufall aus einer größeren Anzahl von Antworten für die jeweilige Frage aus. Einfache Betrugsversuche des Abschauens und des Auswendiglernens von Antworten sind somit nur noch unter größerem Aufwand möglich.

Eine automatische Überprüfung der Programmieraufgabe ist mittels LON-CAPA nicht möglich. Für diesen spezialisierten Anwendungsfall kam das Bewertungssystem Praktomat³ zum Einsatz. Der Praktomat ist ein eigenständiges, webbaserendes und modulares

³ <https://github.com/KITPraktomatTeam/Praktomat> Weiterentwicklung vom Karlsruhe Institute of Technology (KIT)

Bewertungssystem. Es unterstützt verschiedene Programmiersprachen wie Java, Python und C++. An der Ostfalia HS wurde eine Schnittstelle zwischen LMS (LON-CAPA) und Grader (Praktomat) entwickelt und eingesetzt (vgl. [PJR12a]). Im Rahmen des Projektes wurden nur Formative Evaluation in den Programmiersprachen Java und Python verwendet.

Eine der Hauptschwierigkeiten bei der Umstellung des Laborversuchs lag in der Umsetzung eines automatisierten Testverfahrens für die Überprüfung der ABEL-Programmaufgabe. Die modulare Struktur der Tests (Checker) im Praktomaten ermöglichten es, einen eigenständigen ABEL-Test zu schreiben. Der ABEL-Kompiler und die Batch-Skripte benötigen jedoch eine DOS-Umgebung. Diese wurde auf dem Praktomaten mittels DOS-Box⁴ emuliert.

Der Fokus der DOS-Umgebung wurde in erster Linie auf das Spielen gelegt und nicht auf den Serverbetrieb. Eine Anpassung der Konfigurationsskripte ermöglichte den Betrieb auch ohne Windows-System. Der Bedarf an Arbeitsspeicher und CPU-Zeit konnte somit verringert werden.

Die DOSBOX läuft ohne Administratorenrechte und kann als Benutzerprozess aufgerufen werden. Die einzelnen Tests werden im Praktomaten immer in einer eigenen Benutzerumgebung aufgerufen. Somit ist die Sicherheit des Gesamtsystems durch Fehler in der DOSBox nicht gefährdet.

In der Abbildung 1 ist eine Aufgabe aus dem umgesetzten Laborversuch gezeigt. Nach einer Beschreibung der Aufgabe sind die Studierenden aufgefordert ihre Lösung in einem HTML-Textfeld zu entwickeln. Die Studentinnen und Studenten sollen bei dieser Aufgabe aus dem Programmierbeispiel lernen und das angeeignete Wissen anschließend in einer Transferaufgabe umsetzen.

Die Abbildung 2 veranschaulicht in Form eines Sequenzdiagramms den Ablauf einer studentischen Einreichung. LON-CAPA sendet die studentische Einreichung mit der zugehörigen Identifikationsnummer der Aufgabe (TaskId) an den Praktomaten. Dieser startet den Grading-Prozess in welchem alle nötigen Programme wie Kompiler und Batch-Skripte in den Arbeitordner, zu der eingereichten Datei, kopiert werden. Der Ordner wird anschließend in der DOSBox gemountet um Zugriff darauf zu erhalten. Mit Hilfe von Batch-Skripten wird die Einreichung kompiliert. Anschließend erfolgt ein Vergleich der Prüfsummen des Kompilates (JDEC-Datei) und der Musterlösung. Dies stellt die fachliche Richtigkeit der Aufgabe sicher. Das Ergebnis (GradingResult) wird an LON-CAPA zurückgesendet. Die Studierenden bekommen eine nicht bearbeitete Ausgabe des Compilers zurück, wodurch sie Fehler im Programm-Code finden und korrigieren können.

3 Auswertung und Diskussion

Die Umsetzung des Versuches auf LON-CAPA und Praktomat ermöglicht nun eine All-in-one-Lösung für die Studierenden. Sie können ihre Fragen direkt in LON-CAPA bearbeiten.

⁴ <http://www.dosbox.com/information.php>

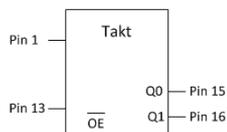
Das folgende Beispiel beschreibt einen 2-Bit Zähler ($n=2$). Allgemein: $n=id(\text{max. Zählstufen})$

```

module Dateiname
Dateiname device 'p20v8';

Takt,IOE pin 1,13; "Eingänge"
Q0,Q1 pin 15,16 istype 'reg'; "Ausgänge"
    count = [Q1,Q0];
equations
    count := count + 1;
end

```



Das Funktionsblattschaltbild wird vom Quelltext (oben) beschrieben

Wie zu sehen ist, läßt sich ein Zähler sehr leicht beschreiben.

Aufgabe: Entwerfen Sie, ausgehend von dem gegebenen Quelltext, einen 3-Bit Zähler. Deklarieren Sie dazu einen weiteren Ausgang (z.B Q2) für Pin 17.

[Hinweise zur Benutzung](#)

Geben Sie im folgenden Feld Ihre Lösungsdatei ein:

```

1 module ZaehTeil
2 ZaehTeil device 'p20v8';
3 ...
4 equations
5 ...
6 end
7

```

Abb. 1: Ausschnitt einer Abel-Programmieraufgabe aus dem LON-CAPA-Kurs

Bei der vorherigen Lösung musste ein weiteres Programm herangezogen werden um die Aufgaben zu lösen. Die vielseitigen Möglichkeiten von LON-CAPA erleichtern das Verschmelzen von erlerntem und anschließend angewandtem Wissen. Dem Kursleiter bietet sich eine Vielzahl an Möglichkeiten das Lernerlebnis der Studenten positiv zu gestalten.

Der Versuch wurde bereits an zwei studentischen Gruppen getestet. Dabei stellte sich heraus, dass die Anzahl der Versuche zur Beantwortung der Multiple-Choice-Fragen aus dem Grundlagenteil, mit einem Antwortversuch pro Frage, etwas knapp gewählt ist. Die Studierenden sollen die Möglichkeit erhalten, ihre falsche Antwort zu korrigieren nachdem sie den Teil noch einmal in den Grundlagen nachgelesen haben. In LON-CAPA lässt sich die Anzahl der Versuche schnell anpassen.

Eine Gruppe von Studierenden hatte wenige Wochen zuvor den alten Versuch absolviert und war von den Neuerungen angetan. Die Gruppe wies darauf hin, dass sie es als störend empfinde die Parameter der Programmieraufgabe jedes mal neu eingeben zu müssen. Dies ließ sich schnell umsetzen und wird, sofern es sich bei weiteren Versuchsdurchläufen als positive Hilfe erweist, beibehalten oder entfernt, falls nicht.

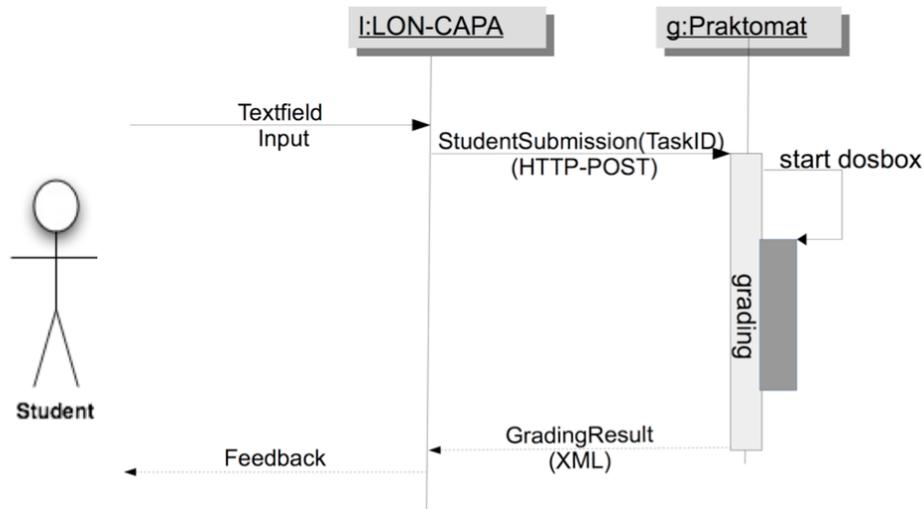


Abb. 2: Sequenzdiagramm der Kommunikation zwischen LON-CAPA und Praktomat

Ein weiterer Hinweis bezog sich auch auf die Programmieraufgabe. Wenn die Programmieraufgabe erfolgreich absolviert wurde, ist der Inhalt der JDEC-Datei im Textfenster angezeigt worden. Der Inhalt musste umständlich in eine Textdatei eingefügt werden. Mittels Javascript und einem HTML5-Downloadlink wurde dieser Aufwand eingespart. Die Gruppe kann die Datei nach erfolgreicher Kompilierung herunterladen und im Anschluss ihren Baustein programmieren.

Während des Hardwareprogrammierteils kam es im Rahmen dieser Arbeit zu einer enormen Erleichterung der Administrierung des Versuches. Die Entwicklungsumgebung auf den Laborarbeitsplätzen konnte eingespart und der Versuch kann von einem beliebigen Arbeitsplatz durchgeführt werden. Ein Update des Betriebssystems führt nicht mehr dazu, den Versuch im Anschluss auf dessen Funktion überprüfen zu müssen. Die Integration einer DOS-Umgebung konnte ebenfalls erspart werden.

In LON-CAPA ist zusätzlich eine IP-Begrenzung eingestellt worden, damit die Bearbeitung nur im Labor stattfinden kann. Dies dient einerseits der Sicherheit vor Angriffen auf den Versuchsumgebung. Andererseits können die Studierenden so die Antworten auf die Fragen nicht aus dem Antwortpool herausarbeiten. Bei Schwierigkeiten oder Fragen steht ihnen im Labor immer ein Laborbetreuer zur Verfügung.

Aus der DOSBox heraus gibt es keine direkte Möglichkeit die Ausgabe eines Fehlers z.B. beim Kompilieren in LON-CAPA auszugeben. Beim Aufruf des Batch-Skriptes wird die Ausgabe des DOSBox-Fensters in einer Textdatei gespeichert. Diese Datei wird durch den Praktomaten an LON-CAPA weitergeben und angezeigt. Der Praktomat besitzt die Funktion, einen Log-Eintrag zu erstellen und die Ausgabedatei darin ausgegeben. Auch im Falle einer fehlerfreien Kompilierung erscheint eine Ausgabe über den Log-Eintrag.

Die Studierenden bekommen somit eine Rückmeldung ob eventuell der Syntax oder die Pinbelegung falsch ist.

Es werden zwei Batch-Skripte verwendet, die den Kompilierungsvorgang steuern. Das erste Skript überprüft, ob der Dateiname, der zu kompilierenden Datei, dabei ist. Bei einem Fehler erfolgt eine Ausgabe in der DOSBox mit einem konkreteren Fehlernamen. Im Falle eines fehlerfreien Durchlaufens wird AHDL2PLA.exe aufgerufen. Es kompiliert die Datei und führt gleichzeitig eine Syntaxprüfung (vgl. [Ko01]), sowie eine Expansion der Macros durch. AHDL2PLA.exe reagiert auf Direktiven und synthetisiert das Design. Zunächst wird die Datei in das intern verwendete PLA-Format konvertiert. Darauf folgend wird die zweite Batch-Skript aufgerufen in der die weiteren Programme aufgerufen werden. In der Reihenfolge wird anfangs PLAOPT.exe, dann FUSEASM.exe und letztlich JEDSIM.exe von dem Batch-Skript gestartet. PLAOPT.exe führt eine Logikminimierung durch, die bei Hardwareprogrammierung anzuwenden ist, um die Größe des Hardwareaufwands so gering wie möglich zu halten. FUSEASM.exe erzeugt eine Dokumentationsdatei und zusätzlich eine JEDEC-Datei, die zur Programmierung des PLDs genutzt wird. Darauf folgend wird mit JEDSIM.exe die Simulation der JEDEC-Datei durchgeführt und mit eventuellen Testvektoren getestet.

Sollte bei den Vorgängen ein Fehler auftauchen, wird dieser in dem Log-Eintrag von LON-CAPA ausgegeben. Bei einem erfolgreichen Durchlauf wird die JEDEC-Datei in eine Textdatei umbenannt und vom Checker des Praktomaten nach einer einzigartigen Checksumme durchsucht. Diese Checksumme ist bei einer einzelnen Programmieraufgabe immer gleich, da bei Hardwareprogrammierung die Minimierung der Logik stets für die kleinstmögliche Hardwarezusammenstellung sorgt. Beim Erstellen einer Aufgabe muss diese Checksumme händisch generiert und im Praktomaten, in ein dafür eingerichtetes Feld des Checkers, eingetragen werden.

4 Ausblick

Durch die Verlagerung des Laborversuches von einem lokal installierten System hin zu einer Onlinelösung, ist es gelungen, den Versuch zu konservieren und noch einige Jahre plattformunabhängig zu nutzen. Für den kleinen Rahmen des Labors und der geringen Komplexität der Aufgaben, reicht die beschriebene Lösung vollkommen aus.

Als Ausblick erscheint eine externe Entwicklungsumgebung für Abel zum Üben und Probieren für interessierte Studentinnen und Studenten sinnvoll. Die implementierte Lösung könnte auch im Rahmen eines reinen Selbststudiums genutzt werden. Hierfür müsste der Kursaufbau und die erlaubte Anzahl von Einreichungen angepasst werden.

Ein zusätzliches Feedback um den Umgang mit dem Compiler zu erleichtern wäre möglich, ist derzeit jedoch nicht erwünscht, weil die Studierenden mit den Compilermeldungen umgehen lernen sollen und bereits Erfahrungen mit anderen Programmiersprache haben.

Eine genaue Analyse von Lasttest könnte bei einer höheren Nutzung des Versuches von Interesse sein. Durch die Verwendung von DOSBox für jeden Bewertungsprozess gibt

es auch klare Anforderungen an die virtuelle Maschine auf der der Praktomat läuft. Im Rahmen des Labors mit einer maximalen Anzahl von zwei Gruppen, die gleichzeitig den Versuch bearbeiten, ist ein Lasttest von geringerem Interesse.

Die Verwendung des Aufgabenformates ProFormA (vgl. [St15]) würde den Austausch und die Wiederverwendung des Programmieranteils sicherstellen. Viel Aufwand wäre hierfür nicht notwendig, da der Praktomat mit LON-CAPA bereits zu den Unterstützten Systemen gehört. Diese Umstellung würde auch den Konfigurationsaufwand auf dem Praktomaten reduzieren (vgl. [PJR12b]).

Literaturverzeichnis

- [Ko01] Programmierbare Logikbausteine, https://www.fbi.h-da.de/fileadmin/personal/e.komar/public_html/DGT-Labor_v3Abel-neu.pdf, Stand: 04. Juni 2001.
- [Ko08] Kortemeyer, Gerd; Kashy, Edwin; Benenson, Walter; Bauer, Wolfgang: Experiences using the open-source learning content management and assessment system LON-CAPA in introductory physics courses. *American Journal of Physics*, 76(4):438, 2008.
- [PJR12a] Priss, Uta; Jensen, Nils; Rod, Oliver: Software for E-Assessment of Programming Exercises. In: *GI-Jahrestagung*. S. 1786–1791, 2012.
- [PJR12b] Priss, Uta; Jensen, Nils; Rod, Oliver: Software for Formative Assessment of Programming Exercises. *elearning Baltics*, S. 63–72, 2012.
- [St15] Strickroth, Sven; Striewe, Michael; Mller, Oliver; Priss, Uta; Becker, Sebastian; Rod, Oliver; Garmann, Robert; Bott, J. Oliver; Pinkwart, Niels: ProFormA: An XML-based exchange format for programming tasks. *elead*, 11(1), 2015.