Fostering an Entrepreneurial Mindset in "Digital Systems" Class through a Producer-Customer Model

Nozar Tabrizi
Department of Electrical and Computer Engineering
Kettering University
Flint, USA
ntabrizi@kettering.edu

***Abstract*—We introduce a Producer-Customer model and use it to add an innovative portion to the "Digital Systems" course materials. In this work, we have developed and tested ten Intellectual Property (IP) cores, real products, which are either *defective* or *improvable*. A *defective* product does not function as it should. An *improvable* product works in accordance with its specification; however, it has potential for improvement. Two different directions are possible for improvements: *performance* or *area*. Students will work in the customer and then producer settings to diagnose and then resolve the issues, respectively. This work is an attempt to fulfill the Kern Entrepreneurial Engineering Network (KEEN) Students Outcomes, hence help instill an entrepreneurial mindset in engineering undergraduates. We have received encouraging evaluations for our innovative work from students.**

Keywords—Digital systems; entrepreneurial mindset; FPGA; innovation in education; KEEN; VHDL; producer-customer model

I. INTRODUCTION

"Digital Systems" is an undergraduate core course taken by Electrical/Computer Engineering as well as Computer Science students worldwide. In this course, students learn how to design digital circuits and use them to build larger systems. This course is a must to understand the hardware design of revolutionizing digital systems such as microprocessors, which are ever-increasingly and inevitably entering our daily lives.

In this paper, we introduce a Producer-Customer model and then use it to add an innovative portion to the "Digital Systems" course materials. To implement this model, we have developed and tested ten Intellectual Property (IP) cores, **real products**, which are either *defective* or *improvable*:

- A *defective* product suffers from some implementation deviations from its specification, i.e., such a product does not function as it should.
- An *improvable* product works in accordance with its specification; however, it has potential for improvement. Two different directions are possible for improvements: *performance* or *area*. A product in this category will work faster should it be improved performance-wise. The improvement in the second direction saves hardware, and hence makes the product smaller in size.

Students will be provided with the products' descriptions written in VHDL, a hardware description language. Students will then compile and map the VHDL descriptions into a Field Programmable Gate Array (FPGA) chip with *a single click* to build the physical products. Each product comes with a *user guide* to teach how the product works and how to use it, and *prerequisite material* to help students understand the operation theory of the product.

Each team of students will first work on a product in a *customer-team* setting to exercise curiosity and uncover the issues with the product through creative thinking. Then another group of students in a *producer-team* setting will be challenged to resolve the issues identified by the customer, and make the necessary changes to the product to achieve *customer satisfaction*. The producer may also resolve the customer's possible misunderstandings. This way, students will develop innovative mentality to *see* opportunities for value creation. The proposed approach provides students with a broad horizon to demonstrate a great deal of enterprise that will help them develop a new mindset and move towards the following KEEN Students Outcomes (KSOs) **as elaborated in Section V**:

- (Multilevel) teamwork, collaboration, communication and connection
- Applying creative thinking to ambiguous problems
- Applying system thinking to complex problems
- Stimulating curiosity and engineering judgment
- Creating value
- Identifying personal passions and a plan for professional development
- Fulfilling commitments in a timely manner

KEEN, The Kern Entrepreneurship Education Network [1], is a collaboration of a growing number of U.S. universities that support the development of an entrepreneurial mindset in undergraduate engineering students. KEEN strives to empower students to see the world as a place full of opportunities to create value. KEEN is sponsored by The Kern Family Foundation, a grant making organization [2].

This work was supported by a KEEN Topical Grant. Our proposal, built on our innovative idea of the Producer-Customer Model, was reviewed by KEEN and finally accepted

after the proposal had been endorsed by *three* external faculty members.

One unique selling point of our work is that it only needs basic lab equipment (a computer and a FPGA board) that generally EE/CE/CS Departments possess. So, every institution can readily use the deliverables; i.e., our work is *transferable*.

The rest of this paper is organized as follows: In Section II background work is reviewed. The products are explained in Section III. You will read two examples of improvability and defect in Section IV. **Section V discusses how the KSOs are established.** In Section VI our assessment methods and conclusion are summarized and students' opinions are shown.

II. BACKGROUND WORK

A work comparable to ours was recently reported in [3] with one major difference: our project modules are based on our own innovative *producer-customer* model, while in [3] Liu et al developed some (mechanical engineering) exercises based on the well-known Problem-Based Learning (PBL) pedagogy to improve students' entrepreneurial skills.

Significant amount of work has been carried out to improve Digital Systems teaching/learning process; however **there is not much work aiming at instilling an entrepreneurial mindset in students**. Therefore, **we believe that our work is an attempt to take a first step for filling this gap in the existing literature**. Poplawski designs and implements a GUI-based simulator for digital systems [4]. The tool provides beginners with a friendly environment to create and simulate their digital circuits. The tool is easy to learn, and this is one of its selling points compared to commercial products. Vainio et al use a unified FPGA platform to support a series of courses such as Basic Digital Circuits, Digital Design, Implementation of Digital Systems, and Digital Design for FPGA [5]. They have used the Altera DE2 education board. This way, students do not have to learn a new environment for each course hence will significantly improve their class performance. The need for a second undergraduate in Digital Logic Design is argued in [6], and then the datapath and controller approach is utilized for this course. This approach helps students better understand some design aspects such as resource estimation, pipelining and area versus speed tradeoffs. In [7], Powell H. C. et al propose to vertically integrate three EE standard core courses, namely Linear Circuits, Electronics and Signals and Systems, and produce three new courses each of which is comprised of selected materials from each of the three original courses. The interesting concept of vertical integration of some related courses may be generalized to a series of computer engineering courses including Digital Systems. A FPGA-based training system for the design and test of combinational digital circuits is reported in [8]. The system provides the user with virtual logic gates. A virtual gate is created in the FPGA chip. To introduce and then map a logic gate you need to plug the associated IC model into one of the four slots of the system. A microcontroller detects the model and then informs the FPGA of the type of the gate. The user then manually makes appropriate interconnections between the IC models to create and test different combinational circuits. Smith proposes and implements an asynchronous logic design portion for the undergraduate computer engineering curriculum [9]. In this methodology, clock distribution and timing closure concerns are ruled out. It also results in significant saving in power consumption as well as reducing the noise generated by signal transitions. A major challenge with asynchronous logic design is the delay-related assumptions that have to be made regarding the interconnects and logic gates used in the circuit. Through introducing VHDL modeling and GPGA synthesis, Rodriguez-Ponce takes an approach in [10] to modernize the digital-design course materials in those universities that still use outdated design techniques. Deeds [11], developed at the University of Genoa, is an integrated and user-friendly learning environment for digital systems. In [12] Donzellini et al add a FPGA extension to Deeds to enable the new system to map the projects created by Deeds into a FPGA chip. This extension minimizes students' interactions with FPGA-specific tools and therefore helps students be exposed to FPGAs at an early stage of their undergraduate studies. In [13] Burch introduces Logisim, a simple but powerful digital-circuits simulation environment for beginners. Logisim [14], with a steep learning curve, is a multi-platform tool, a feature that makes it more transferable to other schools. Pang proposes an integration of online tools for digital circuit design to provide students with an active learning environment hence stimulate their thoughts [15]. Logicly, Multisim, Modelsim and a FPGA-based design software are considered in her work. Al-Zoubi et al use LabVIEW in [16] to implement a virtual remote lab for digital design course. This allows students form every university worldwide to get access to the lab setup of the host university hence carry out digital design experiments online. Manual test of large digital circuits are time consuming and tedious. The issue is addressed in [17], where Kurmas adds an extension called JLSCircuitTester to JLS [18], a digital logic simulator, to provide the users with an environment to test their digital circuits automatically. This not only helps students verify their designs in a reasonable amount of time, it saves the instructor significant amount of grading time as well.

III. PRODUCTS

There are ten products in this work classified as improvable or defective. This section is not of course the user manuals of products. The main goal is to illustrate how *complex* but still *manageable* the products are for a sophomore-level course.

A. Improvable Products

Five of the ten products are improvable, three of which suffer from redundant hardware, while the other two, No 2 and No 4, are improvable in the performance domain.

1) MSI8Bimp, Eight-bit signd multiplier: MSI8Bimp is an 8-bit integer signed multiplier based on Booth algorithm. The MSI8Bimp takes two 8-bit signed numbers and then generates a 24-bit product in (8 + 1) clock cycles.

Fig. 1 illustrates the inputs/outputs of MSI8Bimp. Switch15 (SW15) through SW8 are the first multiplication operand, and SW7 through SW0 are the second operand. Set the switches properly. The operands will be displayed on Displays 7:6 and 5:4, respectively. Slide SW17 down and then hit KEY3. This is the initialization stage. Now press KEY3 eight more times; the

four (out of six) least significant digits of the product will be displayed on Displays 3:0. Press and hold KEY0 to see the two most significant digits of the product on Displays 3:2.

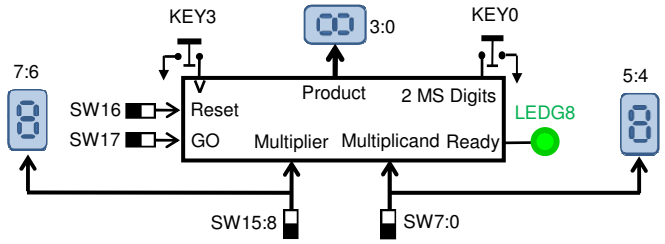

Fig. 1. MSI8Bimp's inputs and outputs

2) CN4x16, 4-Channel synchronous serial Bus: CN4x16 is a unidirectional 1-bit synchronous bus with one transmitter-type bus master and 4 receiving channels, 3 through 0. Fig. 2 illustrates the inputs/outputs of CN4x16. The transmitter checks (polls) the receiving channels one at a time to see if they are ready to receive data. Each channel has two cycles to respond. From the Bus Master's viewpoint, this gives rise to three scenarios as follows:

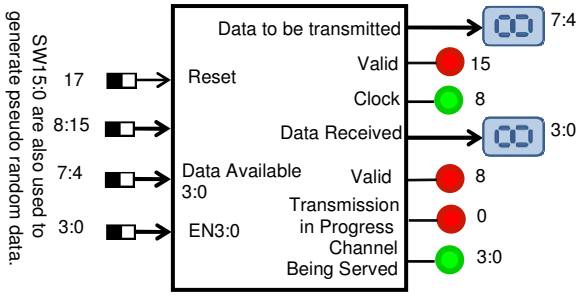

Fig. 2. CN4x16's inputs/outputs

- If the channel being polled acknowledges during the first cycle, the transmitter sends it a 16-bit data, one bit at a time and over the single-bit bus in 16 clock cycles. LEDR0 will light up while data transmission is in progress.
- If the channel being polled acknowledges during the second cycle, it will lose its turn and the Bus Master will therefore move on to the next channel. However, the Bus Master records the channel's request; so that when that channel is polled next time, it will unconditionally be considered *ready* and be treated as explained above.
- If the channel being polled does not acknowledge during these two cycles, the transmitter will move on to the next channel with no further attention until the channel is polled again.

3) DUI8B9, 8-bit unsigned divider: DUI8B9 is an 8-bit unsigned integer divider based on the "Shift-Left and Subtract" algorithm. Fig. 3 illustrates the inputs/outputs of DUI8B9. KEY3 is the manual clock generator. Use SW15:8 and SW7:0 to enter the dividend the divisor, respectively. The values that you choose will be displayed on Displays7:6 and 5:4, respectively. Slide SW17 down and hit KEY3. The dividend will be also displayed on Displays3:2 as well as LEDG7:0. Slide SW17 up and hit KEY3 eight more times. You will see the quotient on Displays3:2 and LEDG7:0. You will see the remainder on Displays1:0 and LEDR7:0. With a minor change, DUI8B9 can operate in an auto mode.

4) DUI8B17, 8-bit unsigned divider: DUI8B17 is a performance-improvable version of the previous product.

5) 8-Channel 8-bit TDM system: TDM88 is an 8-chammel 8-bit time-division multiplexing system. This product consists of a transmitter side, tSide (shown in Fig. 4), and a receiver side, rSide (shown in Fig. 5), which can communicate through 12 one-way shared lines called tdmBus. There are eight transmitters (Tx0 through Tx7) in tSide and eight receivers (Rx0 through Rx7) in rSide.

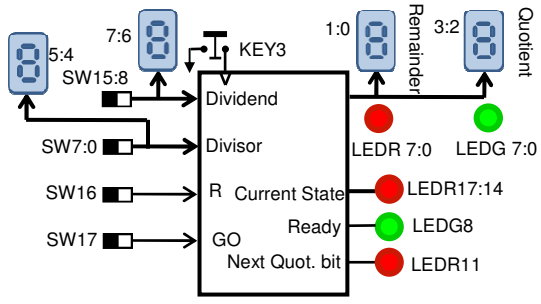

Fig. 3. DUI8B9's inputs/outputs

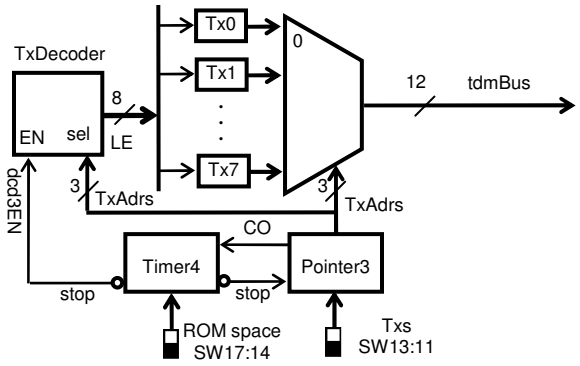

Fig. 4. TDM88's transmitter side

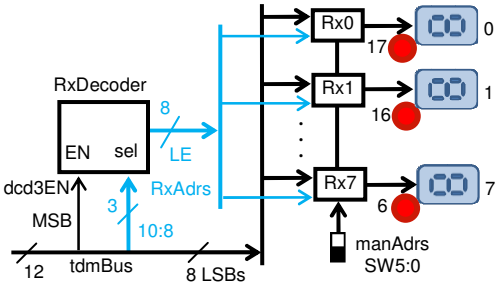

Fig. 5. TDM88's reciver side

There is a 16x12 ROM in each transmitter. The memories are user programmable. The eight LSBs of each 12-bit word

of each ROM are the data bits, the MSB is reserved, and the remaining 3 show the destination of that word as depicted in Fig. 6. As an example, word 53A in a transmitter means that the 8-bit data 3A will be sent to receiver No 5 and be seated in the next available location of the local memory of this receiver. Each receiver has a 32x8 memory. The contents of the receivers' memories may be displayed on eight seven-segment displays.

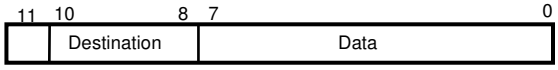

Fig. 6. Word format in TDM88, bit 11 is resereved

Each transmitter can periodically get the ownership of tdmBus for one clock period and use it to send one word to the receiver specified in that word. This will repeat as many times as you specify, so that eventually each transmitter will be able to send a block of data to the receiver side.

Use SW13:11 to specify the transmitters that will participate in the transmission cycle. For example, if you set the switches to 100, then only Tx0 through Tx4 will send their data to the receivers. Additionally, the memory space (of each selected transmitter) that is going to be transmitted is specified through SW17:14. So, if they are set to 1100, then the words located at ROM addresses 0 to 12 of each participating transmitter will be sent.

B. *Defective Products*

1) MSI8B, 8-bit signed multiplier: MSI8B is an 8-bit signed integer multiplier based on Booth multiplication algorithm. MSI8B takes two 8-bit signed numbers and then generates a 16-bit product in 8 + 1 clock cycles.

Fig. 7 illustrates the inputs/outputs of MSI8B. KEY3 is the manual clock. Slide SW16 down and then hit KEY3; Displays 3:0 will be reset to 0 and LEDG8 will light up to indicate that the multiplier is ready to start. Slide SW16 up to come out of the Reset mode. You may use SW16 anytime to reset the multiplier and start over.

SW15:8 are the first multiplication operand and SW7:0 are the second operand. Set the switches properly. The operands will be displayed on Displays 7:6 and 5:4, respectively. Slide SW17 down and then hit KEY3. This is the initialization stage. Press KEY3 eight more times; the product will be displayed on Displays 3:0. During these 8 cycles LEDG8 will remain off. To start a new multiplication, slide SW17 up, hit KEY3. LEDG8 will turn on. Then slide SW17 down and press KEY3 again should you have the correct operands properly set. You may also use SW16 to reset the MSI8B for a new multiplication.

In the *Adjust* mode, LEDG5:4 are lit. One red LED is assigned to each display as illustrated in Fig. 8. Exactly one such LED at a time is on. Using KEY3 or KEY0, you may move the ON red LED to the next or previous digit, respectively. The digit with the ON red LED will increment should you hit KEY2. You may continue to hit KEY0 or KEY3 to eventually turn off all the above red LEDs. This will take you to the Time mode.

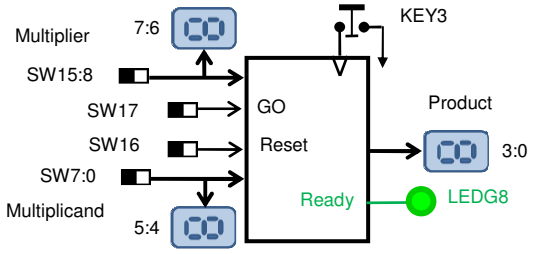

Fig. 7. MSI8B's inputs and outputs

2) Digital Clock: DCA6 is a digital clock with hours, minutes and seconds. Fig. 8 shows the inputs and outputs of DCA6. It has two operation modes, adjust and time. In the adjust mode, you may adjust each digit of hours, minutes and seconds. In the time mode, DCA6 shows the time.

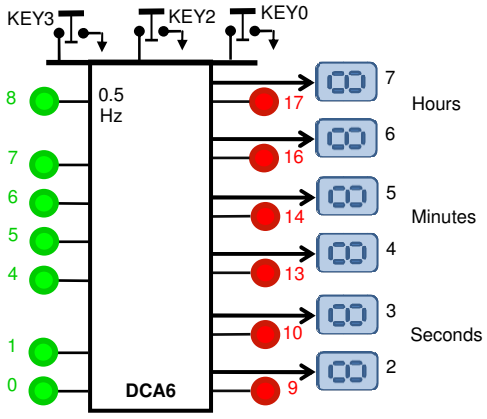

Fig. 8. Inputs/Outputs of DCA6

In the Time mode, green LEDs 0, 1, 6 and 7 will turn on to indicate that you may push either KEY3 or KEY0 at any time to leave the Time mode and enter the Adjust mode. LEDG8 will always be blinking.

3) MUI10B, 10-bit unsigned multiplier: MUI10B is a 10-bit unsigned integer multiplier based on the Add-and-Shift-Right multiplication algorithm. It takes two 10-bit numbers and generates a 20-bit product. MUI10B has 2 operation modes, manual and automatic. LEDG0 is the mode indicator. MUI10B is in the auto mode if LEDG0 is ON. Use KEY2 to switch the mode.

Manual Mode: Fig. 9 illustrates the inputs/outputs in the manual mode. KEY3 is virtually the manual clock. You use SW9:0 to enter the first and second multiplication operands one at a time. They will be displayed on Displays 2:0 and 7:5, respectively. Now hit KEY3 ten times, then slide SW17 up and press KEY3 one more time. The final product will be displayed on Displays 4:0, while the second operand is still displayed on Displays 7:5.

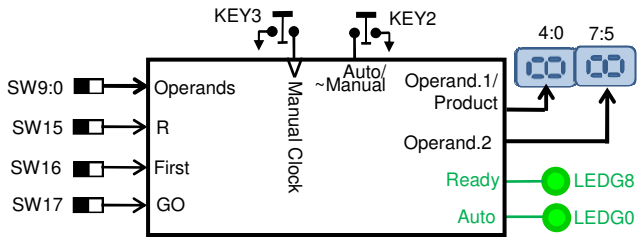

Fig. 9. MUI10B in manual mode

Automatic Mode: The two operands are still applied through SW9:0. Once you properly set SW9:0, hit KEY0 to store the first operand. The same ten switches now become the second operand. Hit KEY3; the product will be displayed on Displays 4:0 *immediately.*

4) PWG1D, Programmable square wave generator: PWG1D is a square wave generator. Its output high time, *Mark*, and low time, *Space*, are programmable. These numbers will be in the range of 1 to 16 seconds. Fig. 10 shows an example of the output waveform in which Mark = 4 seconds and Space = 2 seconds.

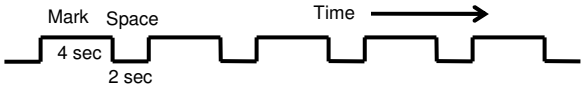

Fig. 10. Example of PWG1D's output waveform

Fig. 11 illustrates the inputs/outputs of PWG1D. This product has four operation modes, namely *Learn* or *Program*, *Ready*, *Generate* and *Reset* as described below:

In the Program mode, the user specifies the Mark width and the Space width by pushing and holing KEY2. Five 7-segment displays help the user to take the necessary steps. KEY3 is the reset key. Once the user is done with the programming, the wave generator enters the Ready mode. PWG1D will start generating the output waveform should KEY2 is pressed one more time. Display 0 starts with the Mark width. It will decrement by one *every second* until it hits 1. During this time interval (Mark time), LEDR0 will be on and LEDG7 will be off. Then a similar cycle starts automatically and continues for Space. During this time interval (Space time), LEDG7 will be on and LEDR0 will be off. This way, Mark alternates with Space forever unless you hit KEY3 to reprogram PWG1D.

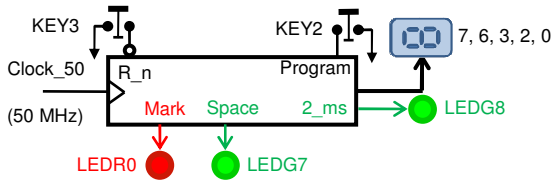

Fig. 11. PWG1D: inputs and outputs

5) TRX8S-112, Serial transmitter/receiver: TRX8S-112 is an 8-bit serial synchronous transmitter (shown in Fig. 12) and a receiver (shown in Fig. 13). The transmitter and the receiver communicate through a single line, SO, shown in these fugures. Each communication side has 112 bits of memory organized as a 16 x 7 register file (RF). The transmitter serializes the 7-bit words of its RF using a PISO and sends the 16 words one bit at a time. Each 7-bit word is preceded by its parity bit during transmission. This results in a total number of 16 x 8 bits in each complete data transfer cycle. The receiver parallelizes the serial input bits to form 7-bit words and places the resulting 16 words in its memory, RF, and exactly in the same order that they are received. The parity bits are checked for possible errors but not stored.

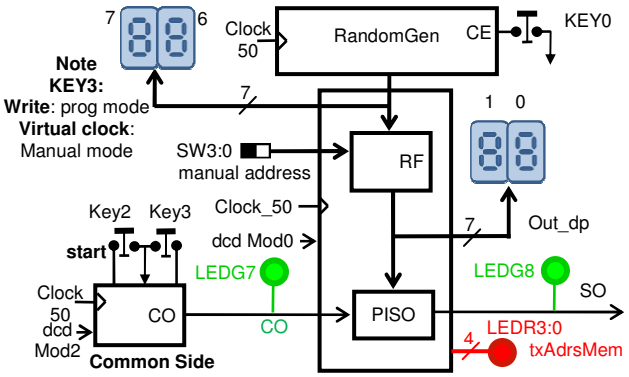

Fig. 12. Transmitter side of TRX8S-112

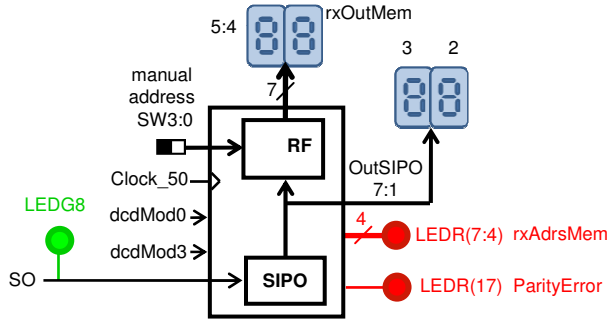

Fig. 13. Receiver side of TRX8S-112

TRX8S-112 has four operation modes, namely *program*, *manual*, *automatic* and *test*. In the manual mode, the clock signal is applied manually; so data is transferred from the transmitter to the receiver one bit per click. You may use this mode for troubleshooting purposes. In the automatic mode, the memory in the transmitter is copied into a same-size memory in the receiver in one click. Writing data into the transmitter memory is easy. To facilitate this, TRX8S-112 additionally has a *program* mode, in which you can enter data generated by a random generator into any location of the transmitter's memory. In the test mode, the user can examine the contents of different registers on the receiver side.

IV. Two Examples of Improvibility and Defect

The challenge in this work was twofold: First we had to come up with ten *application-oriented* (real-world) ideas for ten *complex* products while still *understandable* to

sophomores. We believe that non-real-world ideas will not appeal to students. Second, we had to identify frequently-made engineering design mistakes and incorporate them into the products. In our work, we see that uncovering such engineering mistakes helps facilitate a deep learning experience for students. In this section, we present one example for improvability and one example of defect. Due to the page limit we are not able to cover more examples. In order not to disclose the solutions to prospective students, the following products are not exactly what we have used in this work.

A. *Example of improvibility*

The product: "A single-cycle RISC processor has a data cache memory made up of flip flops."

Discussion: If a cache location is not going to be written while the same location is being read, then latches can be used instead of flip flops. In other words, latches can be used to build the cache if the following scenario is ruled out (resulting in significant hardware optimization):

```
add     memroyLoc1, memoryLoc1, secondOperand
```

Since the product is a RISC processor, the customer team should know and then advise the producer that it is not possible to *write in and read from* the cache in the same cycle, let alone write into a location while the *same* location is being read. Therefore, the producer team needs to replace the flop-based cache with a latch-based one. Moreover, the instructor may challenge the producer to determine whether the new cache should be transparent when clock is low or high!

B. *Example of defect*

The simplified product: "There are two registers, R1 and R2, in the product. When pushbutton PB is pressed, R1 will be reset. If the button is hit again, R2 will be reset as well."

Discussion: While the customer team plays with the product, they notice that it sometimes works, but sometimes does not! More specifically, when they hit PB, sometimes only R1 is reset, but sometimes both R1 and R2 are reset.

When the producer team draws a state graph for the controller, they get a graph like the one shown in Fig. 14a. Let us assume that R1 and R2 are reset while the machine leaves states A and B, respectively. In Fig. 14a when PB is pressed, the machine leaves A resetting R1. However, if the button is pushed close to a clock edge or it is held for sufficiently long time, then the machine may take one more step and reach state

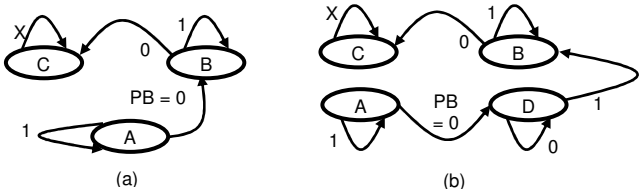

Fig. 14. (a) State graph for defective product, (b) functional graph

C resetting R2 as well. To resolve this issue the producer team may add state D to the graph as shown in Fig. 14b. (We assume that the clock frequency is sufficiently high; otherwise even R1 will sometimes not be reset!)

V. DISCUSSION

The big picture of our work is as follows: **We were able to successfully expose students to complex digital circuits in a 10-week introductory course**; and **this was facilitated through our novel *producer-customer* model.** The overwhelming challenge that we were facing in this work was twofold: on one hand we had to fulfill two main criteria, namely making the products *real* and *considerably complex* while they were still *understandable* to the target students. On the other hand, we had to make the products either *reasonably defective* or *reasonably improbable*. Failure to meet these (sometimes seemingly contradictory) requirements would have not allowed our work to direct students towards the growth mindset that they need to develop KEEN Students' Outcomes (KSOs), or the skillset that they need as a successful engineer. In the rest of this section, **we will comment on how this work fulfills the following KSOs:**

A. *(Multilevel) teamwork, collaboration, communication and connection*

Our work facilitates a novel multilevel *team-based work* hence *communication* and *collaboration* between students: As the first step, the members of each customer team identify either the implementation deviations from the specification, or the improvable components of the product. They work independently of each other and then share their findings at a meeting or two. This *intra-team* level of communication makes each teammate aware of the *perspectives* of others, and let him/her understand their motivations and the way that they look at the product. At this level, the team members will uncover and understand their possible misunderstandings and misinterpretations, and this in general improves the way students *think*.

After the first stage, each team, say A, will collaborate with two other teams, B and C. Team A, as a customer, will communicate with the producer team B, and provide team B with the customer's findings about the product. This initiates an inter-team level work. Note that now each producer-team member may integrate information from some four sources (members of the customer team) to gain insight. This level of communication will help the teams properly handle the possible *ambiguities* through collaboration.

While customer A is assigned to producer B, team A is assigned as a producer for another product having team C as a customer. This way, another level of teamwork is formed between team A on one side and team C on the other side, and therefore another level of inter-team communication is created.

Finally, the members of team A will work on the product that was handed over by team C to resolve the issues identified by team C. At this level of teamwork, the producer-team members need to *substantiate* the correctness of their claims once they have come up with their individual solutions. This substantiation is very *critical*, as it is very likely that although a solution may appear to work, at the same time it may also

cause a new malfunction; this is not something that the producer team wants to happen!

Every team will experience similar inter- and intra-team activities explained above, and be directed towards a mindset that would not be developed in a traditional curriculum.

End of Project Anonymous Quote from Students: ... However, this project had opportunities for me to show my group just how valuable I can be, and what they were missing out on when they were ignoring all of my input.

B. Apply creative thinking to ambiguous problems

In this section and as an example, the basic idea used in one of the products is described to further clarify how creative thinking may be applied to ambiguous problems in our work. The product introduced here is not a real product in order not to disclose the solution to the prospective students.

Sixteen-bit data are generated and written in a memory 16 bits at a time. However, the memory contents do not match the original data. At first, several causes could be imagined for this malfunction; in other words there is an inherent *ambiguity* in this and similar problems. To resolve the issue, students need *creative thinking*: The problem would not look that overwhelming should an error pattern be identified. In other words, students need to correlate between the contents of the memory and the original data. When a few 16-bit patterns are examined, student will be able to uncover a rule that causes bit corruptions. For example, they will find out that bit 0, bit 6, bit 14 of each pattern are repeated 4 times, 2 times, and 5 times in different locations of the same pattern. This finding will direct the students to locate the faulty part.

End of Project Anonymous Quote from Students: I applied creative thinking by examining an open ended problem, and determining our product needs to be a Mealy machine.
Based on the ambiguity of it, we all had to think outside of the box.

Students may apply *creative thinking* to *ambiguous problems* in a second way as well. This happens while they apply *system thinking* to *complex problems* as explained in the next section:

C. Apply system thinking to complex problems

System thinking is infused while students work on a product: They first need to identify a subsystem of the product that contains the problematic component(s) and then understand the way that the subsystem's components influence each other. The problem of locating such a subsystem is *ambiguous*, and therefore students need to apply *creative thinking* to identify the right subsystem.

End of Project Anonymous Quote from Students: It was interesting to work on a complex digital system for the first time.

D. Stimulating curiosity and engineering judgment

Let us take a look at the different stages of students' activities one more time: Each team member is initially provided with a *complex* product along with its user guide, and is supposed to uncover either the implementation deviations from the specification or the improvable parts of the product (depending on the category of the product). The immediate and natural byproduct of this unique setup is *stimulation* of students' *curiosity*. Students are directed to make an engineering judgment and demonstrate that the way a product behaves is not what it is expected, or the way that a product is built is not an *economical* one. In other words, they want to *see* something that is *barely visible*. This is a basic step towards demonstrating curiosity about our changing world.

Students' curiosity is further excited when they move on to the next stage and work as a producer. They are now facing a complex system, which is not either functioning as it is supposed to or cost-effective. The producer team is now challenged to identify/work on either the malfunctioning or improvable components, which could be more than one. Curiosity stimulation is again the very first and inevitable must of this scenario, which is an attempt to develop a growth mindset to look at our surrounding world differently and be prepared for value creation to be discussed next.

End of Project Anonymous Quote from Students: They helped foster my ability to invent solutions and dodge the pitfalls of working on something I initially did not create.

E. Creating value

In the second stage of the project each team becomes a producer who is expected to either identify the malfunctioning components and fix them accordingly or analyze the improvable components and then improve the product. The fact is that the products in this work are pretty *complex* relative to the students' class standing. Therefore, it is an ambitious achievement when the students manage to get the job done, and therefore naturally get the feeling of *creating a value* once they see that they have successfully handled such a sophisticated product. Note that it is too unrealistic to assume that even highly motivated students may design and build such a sophisticated product from scratch in two weeks or so.

In this process, students have to read the user guide and operate the product to make connections between the issues and the structure of the product. So students normally make mistakes in identifying the problematic components or in properly handling them. However due to the ambitious goal of value creation that they naturally and gradually set for themselves, students should be sufficiently motivated to *persist through* and *learn from failure*.

End of Project Anonymous Quote from Students: It felt great to finally fix it. It gave a great sense of accomplishment.
Feeling that you fix something broken is one of the best feelings.

F. Identify personal passions and a plan for professional development

Having been involved in such complex and real projects, students will see that a new horizon opens up towards the real world of Digital Systems. They will then find out how close this new avenue is to their carrier ideas, and therefore identify *personal passions* and *a plan for professional development*.

G. Fulfill commitments in a timely manner

In a traditional curriculum, students have to submit their projects by the scheduled date, or they will be penalized. In the first stage of our work, however, there is one more strong

reason for students to fulfill their commitments as a customer in a timely manner: a producer team will be waiting for them on the date that all the customers are scheduled to turn over their products to the producers. Should they miss the deadline, they would negatively affect the work of another team!

VI. Assessment: Methods and Conclusion

We have utilized five different methods to assess our work's *outcome*: students' presentations, first-hand information from students' meetings/discussions, students' questions, one-on-one discussions/interviews and students' anonymous and written evaluations.

Students' presentations were videotaped in June 2015, and submitted to KEEN as part of our deliverables. The presentations show how the Producer-Customer model has motivated the students to handle such *complex systems* and how this model has directed the students towards *value creation* in this sophomore-level *introductory class*.

We randomly witnessed students' teamwork and how enthusiastic they were in the discussions. Our observations revealed an encouraging level of students' *curiosity* and *creativity*. We also realized how the project encouraged them to *connect* with different resources and search for knowledge that could help them get innovative answers to their questions.

We had close interactions with students during their work as a customer as well as a producer. Their questions were a key for us to understand the impact of our model on their mentality. It was a challenge for us to answer their questions to sufficiently help them while leaving enough room for them to think out of the box, and eventually let them feel a high-level of creativity once they manage to resolve the issues.

One-on-one meetings with students, although time consuming, provided us with a valuable opportunity to better understand how useful our model was in successfully exposing sophomores to complex digitals systems. This also helped us better understand students' individual contributions.

Student anonymous evaluations of the above projects have been encouraging. The following are more End of Project *Anonymous* Quotes from the students:

... However, this project had opportunities for me to show my group just how valuable I can be, and what they were missing out on when they were ignoring all of my input. ... but I proved my value to myself as well as the group when I was able to work on the problem instead of sit in the background.

By completing these projects, I felt like I was doing something actually useful. ... By solving these problems, I feel much more confident that I will be able to do my job well, no matter where I am.

It made me realize how much more I know than the tests have.

It forced us to think outside of the box on how to fix the code.

Every problem has something in common with the reality, so it was really good to apply creative thinking in the projects.

Keeping in mind that this project could be something I created makes me feel confident to create more and more.

Learning what we did and the process to get there really was a great value.

It was exciting when we finally got the program to work. I know that some of my thoughts and ideas went into solving a problem.

I feel like I was able to successfully enhance the performance of my machine. This was a good first feeling of being an engineer.

I felt proud and amazed to see everything come together.

It help me think outside the box and use ideas creatively.

When we were done I felt I had accomplished something big.

It was rewarding to see the functional clock after working on it for so long.

It was a great feeling to finish creating our project. We actually celebrated at the end because we had experienced so much failure before.

When I discovered/fixed what was wrong with the code I felt that I created something new.

I felt that something I thought was impossible to fix actually something I fixed. I was very excited.

... It was very cool to see the entire system come together and actually work. It felt like we had created a much larger system even though we had only repaired a portion of one.

They helped me feel that I was learning the class material, and showed me how much I was learning while in class.

The project made us respect the amount of work that had to be put in the products.

When we successfully fixed our product, it gave us all a great feeling of accomplishment.

It taught me that the everyday use products like digital clocks were designed by people just like me with similar knowledge of digital systems.

We took something that seemed complicated at first and then improved or fixed it and it caused a feeling of satisfaction which also created a sense of wonder about what else coding could be used for.

These projects helped instill a feeling of value creation by making us put in a lot of effort until a successful result. This made us proud when we finally succeeded.

More than anything, they made me feel successful when finally 1. Discovering the problem, and 2. Fixing the problem.

In conclusion, students' performance has been encouraging since we incorporated this work into the class materials of Digital Systems I in our department in the winter of 2015. Some 80% of students (on average) were able to completely finish or almost finish their work. Some 70% of students showed significant enthusiasm and hard work in this new portion of the curriculum and eventually were happy with their achievements. They demonstrated a satisfactory level of curiosity, creative thinking and efficient teamwork. ("Thinking outside the box" was a metaphor that some of the students used as a key factor to their success in the project.) They made so-called *professional* presentations of their work that were beyond our expectation. Additionally, their good work unintentionally pushed a significant percentage of the under-motivated students towards the goals. Although students had different and possibly tight schedules, they were able to adequately meet with their teammates to discuss their work in a timely manner. In many cases we witnessed their close cooperation to teach each other. Although the majority of under-motivated students had tight schedules, and therefore were unable to spent sufficient amount of time on these challenging projects, we believe that these students were able to adequately benefit from their work.

ACKNOWLEDGMENTS

This work was supported by a Topical Grant from KEEN, the Kern Entrepreneurial Engineering Network.

I would like to extend my sincere appreciation and gratitude to my colleagues (listed alphabetically)

Dr. Yanqing Ji
Department of Electrical & Computer Engineering
Gonzaga University

Dr. Claudio Talarico
Department of Electrical and Computer Engineering
Gonzaga University

Dr. Xiaofang (Maggie) Wang
Department of Electrical and Computer Engineering
Villanova University

for reviewing my grant proposal. I am so honored that my proposal was endorsed by these faculty members.

My special thanks also go to Dr. Mo Torfeh of Kettering University, who was always supportive during my work on developing the proposal.

REFERENCES

[1] The Kern Entrepreneurship Education Network
http://www.keennetwork.org/

[2] The Kern Family Foundation
http://www.kffdn.org/entrepreneurial-mindset/

[3] Liping Liu et al, 2015. "Fostering the entrepreneurial mindset in the junior and senior mechanical engineering curriculum with a multi-course problem-based learning experience," *Proceddings of 2015 Frontiers in Education Conference* (FIE), pp 1 – 5, 21-24 Oct. 2015, El Paso, TX.

[4] D. A. Poplawski, 2007. "A Pedagogically Targeted Logic Design and Simulation Tool," *Workshop on Computer Architecture Education,* WCAE '07, June 9, 2007 San Diego, CA

[5] O. Vainio et al, 2010. "Teaching digital systems using a unified FPGA platform," *12th Biennial Baltic Electronics Conference* (BEC), pp 137-140, 4-6 Oct. 2010, Tallinn

[6] M. A. Thornton et al, 2003. "A Second Undergraduate Course in Digital Logic Design: The Datapath+Controller-Based Approach," *Proceedings of American Society for Engineering Education*, Southeastern Section Conference, 6-8 April 2003, Macon, Georgia

[7] H. C. Powell et al, 2015. "Restructuring an Electrical and Computer Engineering Curriculum: a Vertically Integrated, Laboratory/Lecture Approach," *Proceedings of American Society for Engineering Education*, April 12-14, 2015, Gainesville, FL

[8] S .Sothong et al. 2010. "Design of Combinational Logic Training System Using FPGA," *40th ASEE/IEEE Frontiers in Education Conference*, October 27 - 30, 2010, Washington DC

[9] S. C. Smith, 2006. "Integrating Asynchronous Digital Design into the Undergraduate Computer Engineering Curriculum," *Proceedings of the 2006 Midwest Section Conference of the American Society for Engineering Education*, September 13-15, 2006, Kansas City, Missouri

[10] R. Rodriguez-Ponce et al, 2013. "Integrating VHDL into an undergraduate digital design course," *IEEE International Conference on Teaching, Assessment and Learning for Engineering* (TALE), pp 172 – 177, 26-29 Aug. 2013, Bali

[11] Deeds: Digital Electronics Education Design Suite;
http://www.esng.dibe.unige.it/deeds/

[12] G. D. Donzellini et al, 2013. "From gates to FPGA: Learning digital design with Deeds," *2013 3rd Interdisciplinary Engineering Design Education Conference* (IEDEC), pp 41 - 48, 4-5 March 2013, Santa Clara, CA

[13] C. Burch, 2002. "Logisim: a graphical system for logic circuit design and simulation," *Journal on Educational Resources in Computing* (JERIC), Volume 2 Issue 1, March 2002, Pages 5-16

[14] Logisim: An educational tool for designing and simulating digital logic circuits;
http://sourceforge.net/projects/circuit/

[15] J. Pang 2015. "Active Learning in the Introduction to Digital Logic Design Laboratory Course," *Proceddings of 2015 American Society for Engineering Education*, (Zone III).
https://www.asee.org/documents/zones/zone3/2015/Active-Learning-in-the-Introduction-to-Digital-Logic-Design-Laboratory-Course.pdf

[16] A.Y. Al-Zoubi et al, 2008. "Integration of an Online Digital Logic Design Lab for IT Education," *Proceedings of the 9th ACM SIGITE conference on Information technology education*, SIGITE'08, pp 237-242, October 16 - 18, Cincinnati, OH, USA,

[17] Z. Kurmas, 2008. "Improving student performance using automated testing of simulated digital logic circuits," *Proceedings of the 13th annual conference on Innovation and technology in computer science education*, ITiCSE'08, pp 265-270, June 30 - July 02, 2008, Madrid, Spain

[18] JLS: A Pedagogically Targetted Logic Design and Simulation Tool;
http://www.cs.mtu.edu/~pop/jlsp/bin/JLS.html