2006-2239: STUDENT MISCONCEPTIONS IN AN INTRODUCTORY DIGITAL LOGIC DESIGN COURSE

Craig Zilles, University of Illinois-Urbana Champaign

James Longino, University of Illinois-Urbana Champaign

Michael Loui, University of Illinois-Urbana Champaign

Student Misconceptions in an Introductory Logic Design Course

Abstract

In order to improve student learning, instructors should identify concepts that are difficult for students to understand. Instructors can then change course material or teaching methods to focus on these difficult concepts. Researchers can develop assessment tools based on common student misconceptions to measure the effects of pedagogical changes.

This paper describes the results of interviews with students who took an introductory logic design course in the Spring or Summer of 2005 at the University of Illinois at Urbana-Champaign. These interviews revealed many common misconceptions students have after completing a sophomore-level course on logic design. This paper also describes the results of an assessment test based on the interviews and administered to students taking an introduction to logic design course at the end of the Fall semester of 2005 at the University of Illinois at Urbana-Champaign.

Introduction

Engineering and computer science faculty have a growing number of reasons to assess student learning outcomes. ABET accreditation requires "a system of ongoing evaluation that demonstrates achievement of these objectives and uses the results to improve the effectiveness of the program." [1] Faculty interested in improving their teaching require an objective, reliable tool to evaluate the effects of different teaching methods. Education researchers need a standard tool to compare pedagogies. Classroom assessment can be used to achieve institutional, faculty, and research assessment objectives. [2] One assessment tool that has proven valuable in science and engineering fields is the concept inventory, a short, multiple-choice tool used to determine how students think about concepts in a field. In particular, the Force Concept Inventory (FCI), which probes conceptual understanding of Newtonian mechanics, has revolutionized how introductory physics courses are taught. Because concept inventories test student misconceptions in a field, designing new concept inventories requires an understanding of the important concepts in the field and how students understand those concepts.

In this paper, we describe our initial work toward the development of a concept inventory for logic design. In particular, we document student misconceptions that we identified through a series of interviews with students who had recently completed an introductory course in logic design. To check whether other students held these misconceptions, we developed and administered a multiple-choice assessment test. While this test has limited coverage and has not been validated sufficiently to be considered a concept inventory, both the test and the student misconceptions that we have identified represent a promising first step. Because this work is only a first step, we focus primarily on identifying student misconceptions and make no attempt at recommending teaching practices to address those misconceptions.

We conducted our research at the University of Illinois at Urbana-Champaign, which offers two introductory logic design courses: ECE290: Computer Engineering I offered by the Department of Electrical and Computer Engineering and CS232: Computer Architecture I offered by the Department of Computer Science. Both courses are sophomore-level courses that are required for the departments' undergraduate majors. The two courses cover almost identical material: representation of information, both combinational and sequential circuit analysis and design, and computer organization and control. Both courses use the same text[3], but they are not jointly offered, primarily due to their large enrollments of 200 and 120 students per semester respectively. In both courses, students attend three hours of class every week; two hour-long lectures are taught by faculty and one hour-long recitation is taught by graduate teaching assistants. Students complete schematic-capture and simulation-based laboratory assignments on a bi-weekly basis. Students submit written homework every other week for CS231 and every week for ECE290. In addition, students complete frequent web-based homework problems and receive instant feedback. During the semester, students take two mid-term exams and a comprehensive final exam. Further information can be found at: http://courses.ece.uiuc.edu/ECE290 and http://www.cs.uiuc.edu/class/cs231.

The first section of this paper presents a brief overview of the literature on concept inventories. The second section discusses the interview and assessment test experimental method. The final section provides an overview of the results of the interviews and the end-of-semester review and detailed discussion of some of the common misconceptions we found.

Background

The FCI was developed to demonstrate the extent to which students were failing to become "Newtonian thinkers" in spite of extensive instruction in Newtonian mechanics. [4-5] The results of initial administrations of the FCI were very disappointing to physics professors because many students who received high grades in courses scored poorly on the concept inventory even though inventory problems appeared trivial to physics faculty.

The FCI has been shown to be useful for evaluating courses and pedagogical approaches, measuring student knowledge gain over a semester, and, when coupled with a math test, placing students in honors courses.[4-8] The FCI has had a pronounced effect on perception of pedagogy within physics.[9] In an attempt to replicate the success of the FCI and produce a similar pedagogical change in other disciplines, concept inventories are being developed for other science and engineering fields including electromagnetics, thermodynamics, signals and systems, circuits, and fluid mechanics.[10]

Previous research in concept inventories for logic design is sparse; we are aware of only one prior attempt. This concept inventory, developed at The University of Massachusetts at Dartmouth and The University of Alabama, consists of 13 problems over seven different topics.[11] However, this concept inventory remains incomplete, some questions are outside our intended scope, and other questions rely too deeply on formalism or algorithms that can be applied without a firm understanding of the underlying concepts.

Concept inventory development relies on knowledge of student misconceptions. While the FCI relied on Hestenes' prior research on early misconceptions of physics students[12], such a characterization of misconceptions is not readily available in every field. While instructors often think they know which concepts are the ones that students find the most challenging, concept inventory developers generally believe that the best approach to diagnosing the source of student misconceptions is through student interviews.[13-14] In the next section, we describe our interview method, based on this prior work.

Methods

We interviewed eight students during the fall semester of 2005; each interview lasted about one hour and was videotaped. Student volunteers were identified through three rounds of email solicitation. In the first round, we contacted students who had taken ECE290 and CS231 during the Summer 2005 session, and only students with A grades responded. To achieve a diversity of achievement levels in our interview subjects, we contacted students who had taken these courses during the Spring 2005 semester and received course grades of “C and lower” and “B/B-” in the second and third solicitations, respectively. Of the eight students whom we interviewed, three had taken CS231 and five had taken ECE290. Two students were women.

Students were first asked to reflect on their experiences in the course and to share their perceptions of the most important, the easiest, and the most difficult topics in the courses. We provide no data on these responses because they did not reveal useful information about misunderstandings or difficult topics. Even when we notified students ahead of time that we would ask this question their responses were too general to be useful. The time between instruction and interview may have contributed to the lack of meaningful responses.

For the remainder of the interview, students were asked to explain digital logic concepts and perform “think alouds” during which students would solve logic design problems – either on the whiteboard or on paper – while verbalizing their thought process. We asked follow-up questions to probe why students chose particular techniques and how they arrived at their answers. We focused on the digital logic concepts described in the course objectives for ECE290 (included as Appendix A); computer organization course objectives covered in the courses were not included as they are beyond the scope of this work. We focused on these concepts because the ECE290 course objectives, written for ABET accreditation, had been specifically developed to capture the key concepts of the course. These concepts are also consistent with important concepts taught in CS231 and in the textbook. Examples of both questions and problems can be found in Figures 1-3.

What is the complement of $wx(y'z + yz') + w'x'$?

Figure 1 – An example interview problem

Why can we design any combinatorial system using only NAND gates?

Figure 2 – An example interview question

Synthesize the following state diagram using any type of flip-flop.

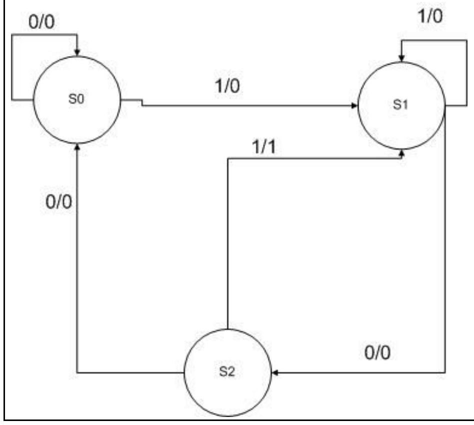


Figure 3 – An example interview problem

Initially, many problems were similar to problems students would encounter in a textbook or course problem set. The initial set of problems consisted of at least one problem testing each course objective. As the interviews progressed, we removed from the problem set the problems that students correctly answered most often. The remaining problems contained some concept that students found difficult. We tuned each of these problems to focus on one misconception, sometimes using a context that differed from what students may have seen in the courses.

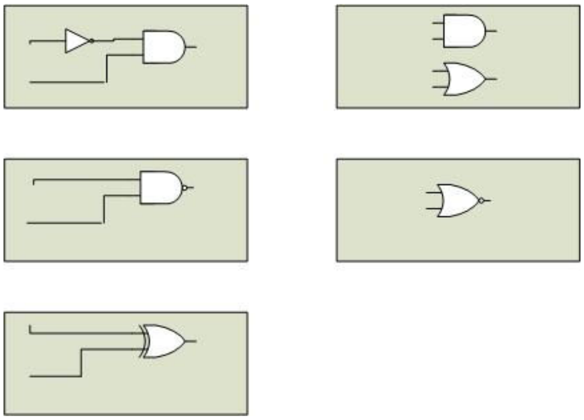
During the interviews, students voiced some misunderstandings with enough regularity that we decided to check whether other students had the same misconceptions. For these misconceptions we developed multiple-choice problems where each problem is intended to test for a "deep", conceptual understanding of a single concept and to require little or no calculation. Possible answer selections were answers common in student interviews (to test for distracting wrong answers), non-obvious or non-intuitive (to test for missed correct answers), or the result of our experience that the answer could be a significant distracter. Some problems had more than one correct answer, and students were instructed to select all correct answers. In some cases, we provided many possible answers in order to identify which were compelling distracters. We created a test consisting of twelve items and administered this test to 28 students in two recitation sections of ECE 290 as an end-of-semester review during the final class period of the semester. Examples of test items are shown below in Figures 4 and 5.

A sequential state diagram with n states and requires at least m flip-flops. If a different state diagram has 2*n states, what is the minimum number of flip-flops?

a.) m
b.) m + 1
c.) m + 2
d.) 2*m
e.) 2*m + 1
f.) m^2
g.) 2*m + 1
h.) None of the above

Figure 4 – An example test item

Which of the following are complete logic families (i.e., all possible combinational logic circuits can be implemented using just these gates and the constants 0 and 1).



There may be more than one right answer.

Figure 5 – An example multiple-choice problem

Results

Overall, we found that students fully understood many course concepts. Students correctly solved many basic problems such as converting a 2's complement binary number to decimal, completing combinatorial timing diagrams, and converting a logic diagram to a Boolean expression. Most students could accurately describe the functionality and use of basic logic gates, medium-scale integration (MSI) components, and flip-flops, although almost uniformly students could recall only D- and T-type flip-flops. Furthermore, most students correctly solved problems based on simple algebraic identities, basic MSI designs, and designs using two-input logic gates.

Nevertheless, we identified the following student misconceptions:

Karnaugh Maps: Because Karnaugh maps are a major topic in both ECE 290 and CS 231, we were surprised to find that students were generally reluctant to use them in all but the most obvious circumstances. In particular, it appears that students tend to associate using Karnaugh maps with particular problem formulations. Two questions required the use of Karnaugh maps to complete them quickly. The first question was used in the interviews and is shown in Figure 6.

Find a two-level multiple-output AND-OR gate network to realize the following functions using 6 gates.
a'c + a'd' + b'c and c'd' + ab' + ac'

Figure 6 – Interview problem easily solved by Karnaugh maps

Although the problem in Figure 6 can be solved algebraically, it is most easily solved by constructing the Karnaugh maps for both functions and selecting loops that overlap. Students presented with this problem usually tried other methods, such as Boolean algebra or truth tables, and generally got stuck. One student in particular told the interviewers he was looking for overlapping terms, but decided that a truth table was a better tool than a Karnaugh map. Only the top students instinctively used a Karnaugh map for this problem, though other students were able to find the correct answer after some leading questions suggested a Karnaugh map would be a good way to solve this problem.

A closed-form problem is shown below in Figure 7. This problem was presented to students both in interviews and in the assessment test. The number of students choosing an answer on the assessment test is given in parenthesis to the right of the answer choice. Correct answers are marked with an "X" to the right of the number of students choosing that answer.

If you implement the following truth table as a sum of products, what are the minimum required number of AND and OR gates. Assume you have inputs and their complements available as inputs to the circuit. *(attempted by 22 students)*

```
 A B C | Output
-------+--------
 0 0 0 |    1
 0 0 1 |    1
 0 1 0 |    1
 0 1 1 |    0
 1 0 0 |    1
 1 0 1 |    0
 1 1 0 |    1
 1 1 1 |    1
```

a.) one 2-input gate (0)

b.) one 3-input gate (0)

c.) two 2-input gates (0)

d.) two 3-input gates (1)

e.) one 2-input and one 3-input gate (0)

f.) three 2-input gates (0)

g.) three 3-input gates (1)

h) two 2-input gates and one 3-input gate(14 X)

i) one 2-input gate and two 3-input gates (0)

j) three 2-input gate and one 3-input gate (1)

k) one 2-input gate and three 3-input gates (0)

l) two 2-input gates and two 3-input gates (2)

m) None of the above (3)

Figure 7 – A test item focusing on Karnaugh maps

This problem more obviously leads to the use of a Karnaugh map as it asks for the minimum number of gates, phrasing that is commonly associated with Karnaugh maps in an introductory logic design course. Still, many students did not attempt the question and only half chose the correct answer. Examination of the review sheets showed that some students were not correctly identifying minimal loops in their Karnaugh maps.

Since Karnaugh maps are a fundamental tool of logic design, students should know how to use them correctly and at the appropriate times. It is possible that Karnaugh maps are used in these courses primarily in problems that specifically ask for them or in larger design problems that are presented close to when students learn Karnaugh maps. Since students generally see Karnaugh maps in only one context, they may find it difficult to know apply Karnaugh maps in different kinds of problems. The ability to use knowledge across different contexts, known as *knowledge transfer*, is well documented in educational psychology research.[15] We believe that students' reluctance to use Karnaugh maps in unfamiliar situations results from a lack of knowledge transfer.

Duality: Students generally had difficulty deriving the dual of an expression. During the interviews, very few students correctly solved this problem the first time. The types of errors varied widely. Some errors were merely typographical in nature, such as swapping two variables in copying from one line to another. Other students confused taking the dual with complementing the expression; they inverted variables in the various steps. Students also misapplied operation precedence. The expression was complicated enough to require an understanding of operation precedence, but did not contain a constant 1 or 0 to see if students remembered to complement constants.

What is the dual of $wx(y'z + yz') + w'x'(y' + x)(y + z')$?

Figure 8 – An interview problem on duality

It is unclear exactly why students had so much difficulty with this problem. Most likely, students thought that duality is unimportant. Also, duality is covered very quickly at the beginning of the term and usually is not seen again. However, there is a fair amount of formalism involved in the problem, and students might understand some of the concepts behind the dual while not being able to derive it. While we want students to understand both the implications and the derivation of the dual of an expression, we recognized students' difficulty with the derivation and decided to test whether students knew the implications of duality. The problem on the end-of-semester review questions shown in Figure 9 probed for students who understood one of the main implications of duality without actually having to derive the dual.

Let F, G, and H be functions of x, y, and z. The truth tables of F and G are the same. Which of the following must be true? There may be more than one answer.
(*attempted by 26 students*)

a.) F = G	(26 X)	e.) FH = HG	(23 X)
b.) F' = G'	(26 X)	f.) FH = GH	(24 X)
c.) Dual(F) * G = 0 (Dual of F AND G)	(3)	g.) Dual(F) = Dual(G)	(21 X)
d.) F'G=0	(24 X)		

Figure 9 – A test item examining implications of duality

Specifically, answer choice g tested whether students understood that, if two expressions are equal, then the duals of those expressions are also equal. Most students correctly identified all correct statements. While this apparent understanding of a main idea of duality is encouraging, it is possible that students selected answer g because they saw the same operation being performed on both sides of the equation and not because they understood duality. More investigation is needed to understand if students understand duality and just forget the process to derive the dual or if they do not fully understand duality.

Non-Trivial Output: In real design applications, it's not always possible to find devices with the exact number of input lines required. Some input must go into these lines, but students are not always confident about what kind of input is needed. This misunderstanding first came up as interviewed students tried to solve a problem with a limited set of devices, shown in Figure 10.

f(x,y,z) = AND(M1, M4, M6)
g(x,y,z) = OR(m1, m3, m5)
h(x, y, z) = x XOR z

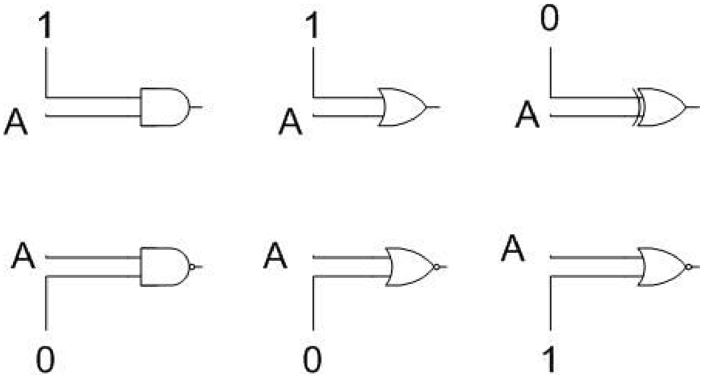
Draw a circuit which implements the three functions f, g, and h. You may use only one 3-to-8 decoder, two 4-input OR gates, and one 4-input NOR gate.

Figure 10 – An interview question where one gate has an unused input

One student in an interview, needing an additional input to the NOR gate, decided to use a high logic value since an OR gate requires a 0 to result in a non-trivial output. We developed the problem in Figure 11 to test students' knowledge on what inputs to use for particular gates.

Which of the following will result in nontrivial output (not always 0 or 1)?
There may be more than one right answer.
(attempted by 26 students)

a.) (25 X) b.) (0) c.) (25 X)



d.) (2) e.) (22 X) f.) (2)

g.) None of the above (0)

Figure 11 – A test item on non-trivial outputs

Students correctly solved this problem in the interviews and at the end-of-semester review. Most students misunderstood gates with complemented outputs. Students were very inconsistent with answers e and f. Three students chose neither, one student chose both, and one student chose f instead of e. With such a small data set, it is not clear if these data points result from carelessness or some deeper misunderstanding. It is clear from both interview results and the review that students do not always understand NOR and NAND gates, sometimes attributing the complement to the input instead of the output.

The problem statement in Figure 11 may artificially increase the number of correct answers. As presented, it is very easy for students to draw a quick truth table and get the correct answer. While drawing a truth table is a valid way of solving this problem (it may even be the fastest way), this method does not expose a real understanding of how complemented gates work. This problem should be examined more closely to see if it can be stated in a way that will elicit responses based more on a conceptual understanding than the ability to create a two-input truth table.

Minimum number of flip-flops needed in a minimum state diagram: The creation of a sequential circuit design from a state diagram is taught in most introductory logic design courses, including ECE 290 and CS 231. This area of the course is open to misunderstandings since transforming a state diagram into a sequential circuit design is an algorithmic task and can be applied without a firm understanding of why or how the transformation works. We designed the problem shown in Figure 12 to capture a part of students' conceptions of this process.

A sequential state diagram with n states requires at least m flip-flops. If a different state diagram has 2*n states, what is the minimum number of flip-flops?
(attempted by 24 students)

a.) m	(0)	e.) m^2	(2)
b.) m+2	(2)	f.) 2*m^2	(0)
c.) 2*m	(7)	g.) None of the above	(12 X)
d.) 2*m + 1	(1)		

Figure 12 – A test item relating number of states to number of flip-flops

This problem probes students' understanding of the relationship between states and flip-flops. Many students remember that the minimum number of flip-flops needed for n states is $\log_2(n)$, but this problem cannot be solved by recalling that fact. Because the correct answer (m+1) has a different form than the distracters, we chose not to list it as an option to avoid students deducing it as the correct answer.

As suspected, answers b, c, and e were common distracters. Students who chose answer b may have been trying to apply the log rule, but did not fully understand the log function. Students who chose answers c, d, and e show a fundamental misunderstanding of how states in a state diagram are related to flip-flop outputs. This problem was mostly used in the end-of-semester review, so we did not get to ask why students chose answers c or e.

These results should be interpreted carefully. Because students could not pick the correct answer, it is not possible to determine if students chose answer g with the real answer in mind or a different incorrect answer. Further, the process of going from a state diagram to a flip-flop design isn't always taught with the goal of a minimum flip-flop design. While many of the basic conceptions are the same regardless of the goal, this particular measure may not be appropriate for other kinds of instruction.

MSI components: Some students had difficulties explaining MSI components and, somewhat surprisingly, we found more confusion with multiplexers than other MSI components. For example, multiple students indicated that multiplexers choose one input to appear as the output, but failed to mention a selection input. In a multiple-choice problem asking students to identify correct implementations of an expression, more students identified correct decoder implementations than correct multiplexer implementations (92% and 71% for two decoder implementations versus 43% and 17% for correct multiplexer implementations). This is true for both decoder implementations (fewer than 65% correctly chose the right answer) using minterms and logic gate implementations (63% correct) using a Karnaugh map.

Flip-flops and Sequential Design: Students commonly misunderstood the distinction between flip-flops and latches. On a multiple-choice question, students performed poorly (40% correct) on a problem involving characterizing the output waveform of a simple circuit involving two latches and two gates, but we need to verify that this poor performance is the result of a lack of understanding and not merely due to careless assumption that the circuit elements were flip-flops.

Furthermore, sequential design is a topic of significant difficulty for many students, but one that we have yet to characterize to our satisfaction. In interviews, students either finished sequential design problems quickly or got lost and did not complete the problems. Because we failed to identify students' misconceptions about sequential design problems, we are conducting further interviews.

Conclusions

Our preliminary results revealed some digital logic concepts that many students misunderstand. While we anticipated some of these misconceptions, others were very surprising. Interaction with students has been very helpful in understanding which concepts are difficult to understand and why they are difficult to understand. Because our approach has been successful for previous inventories, we believe that understanding of student misconceptions requires a high level of student interaction, especially if there is little or no research on student misconceptions in a field.

ECE 290 and CS 231 have very similar course objectives, but are taught very differently. ECE 290 uses large problem sets with lots of repetition, while CS 231 assigns less homework to students. We are interested in the effect the different structures have on students' conceptual understanding of course material. Students from both courses participated in the interviews, but the data set is so small that no significant conclusions can be drawn. Furthermore, only ECE290 students took the review test. We will study the effect of taking the CS version of the course in future work.

Future work will also focus on understanding student misconceptions about sequential circuits and translating these misconceptions into multiple-choice questions. Finally, we intend to refine the questions and answer choices to create a draft concept inventory that can be taken by a larger group of students at the end of spring 2006.

References

1. Criterion 2.d from ABET. (1997). Engineering criteria 2000: Criteria for accrediting programs in Engineering in the United States. *ASEE Prism, 6,* 41-42.
2. Angelo, T.A. and Cross, K.P. (1993). *Classroom Assessment Techniques*. San Francisco: Jossey-Bass.
3. Mano, M.M. and Kime, C.R. (2003). *Logic and Computer Design Fundamentals, 3rd Edition.* Upper Saddle River: Pearson Prentice Hall
4. Hestenes, D., Wells, M., and Swackhamer, G. (1992). Force Concept Inventory. *The Physics Teacher, 30*, 141-158.
5. Hestenes, D. and Halloun, I. (1995). Interpreting the Force Concept Inventory. *The Physics Teacher, 33*, 502-506.
6. Henderson, C. (2002). Common concerns about the Force Concept Inventory. *The Physics Teacher, 40,* 542-547.
7. Savinainen, A., and Scott, P. (2002). The Force Concept Inventory: A tool for monitoring student learning. *Journal of Physics Education, 37(1)*, 45-52.
8. Savinainen, A., and Scott, P. (2002). Using the Force Concept Inventory to monitor student learning and to plan teaching. *Journal of Physics Education, 37(1),* 53-58
9. Mestre, J.P. (2005). Facts and myths about pedagogies of engagement in science learning. *Peer Review, 7(2)*, 24-27.
10. http://foundationcoalition.org/home/keycomponents/concept/index.html
11. http://www.foundationcoalition.org/home/keycomponents/concept/computer.html
12. Halloun, I. and Hestenes, D. (1985). The initial knowledge state of college physics students. *American Journal of Physics, 53,* 1043.
13. Martin, J., Mitchell, J., and Newell, T. (2003). Development of a concept inventory for fluid mechanics. *Proceedings, 2003 ASEE/IEEE Frontiers in Education Conference*, T3D-23-T3D-28.
14. Jacobi, A., Martin, J., Mitchell, J., and Newell, T. (2003). A concept inventory for heat transfer. *Proceedings, 2003 ASEE/IEEE Frontiers in Education Conference,* T3D-12-T3D-16.
15. Bransford, J. D., Brown, A., & Cocking, R. (2000). (Eds.), *How People Learn: Mind, Brain, Experience and School, Expanded Edition.* Washington, DC: National Academy Press. (Co-author).

APPENDIX A

Representation of information

- Convert between decimal, binary, octal, and hexadecimal representations of integers
- Determine the number of errors that a code can detect or correct
- Understand two's complement representation of integers and determine whether overflow occurs in arithmetic operations
- Distinguish between a variety of decimal and alphanumeric codes

Design and analysis of combinational networks

- Understand the operation of discrete logic gates
- Analyze a combinational network using Boolean expressions
- Convert a verbal specification into a Boolean expression
- Understand basic properties of Boolean algebra: duality, complements, standard forms
- Apply Boolean algebra to prove identities and simplify expressions
- Use Karnaugh maps to find minimal sum-of-products and products-of-sums expressions
- Design combinational networks that use NAND, NOR, and XOR gates
- Design with MSI components such as encoders, decoders, multiplexers, adders, arithmetic-logic units, ROMs, and programmable logic arrays
- Calculate delays in ripple carry adders and combinational arrays

Design and analysis of sequential networks

- Understand the operation of latches; clocked, master-slave, and edge-triggered flip-flops; shift registers; and counters
- Plot and interpret timing diagrams
- Determine the functionality of sequential circuits from state diagrams and timing diagrams
- Translate sequential circuit specifications into state diagrams
- Design sequential circuit components (latches, flip-flops, registers, synchronous counters) using logic gates
- Synthesize general sequential circuits
- Understand tradeoffs in register and counter design