# Java-based Instructional Materials for Introductory Logic Design Courses

*Sally Wood*
*Ronald Danielson*
*School of Engineering*
*Santa Clara University*
*Santa Clara, California 95053*

***Abstract*** *- Introductory logic design is one of the fundamental topics in the lower division curriculum in computer engineering, computer science, and electrical engineering. Since it provides a basic foundation for a large portion of the undergraduate curriculum in ASIC design, data acquisition and signal processing systems, graphics controllers, and architectures for general purpose computing and communication systems, it is important that students learn the material well. The range of backgrounds and preferred learning styles of today's students can make it difficult to effectively convey these concepts. This paper describes the structure and content of a set of Web-based tutorials to supplement the spectrum of other learning experiences offered by the typical introductory logic design course. It also presents comments on selection and design of Java classes for such tutorials.*

## Introduction

Introductory logic design [4, 7] is one of the fundamental topics in the lower division curriculum in computer engineering, computer science, and electrical engineering. It typically includes both fundamental concepts of logic and architecture that are important for all technologies, as well as some focus on currently used technology and corresponding design tools. Like the first circuits course, this course is a basic foundation for a large portion of the undergraduate curriculum in ASIC design, data acquisition and signal processing systems, graphics controllers, and architectures for general purpose computing and communication systems.

Effectively teaching fundamental concepts in the context of current technology is a growing challenge because of the diversity of the students' backgrounds and because the styles of student learning have changed noticeably in recent years. Techniques that use interactive visual presentation of examples are better matched to student learning styles than "top down" textbook and lecture presentations. However, bombarding the students with specific examples does not necessarily lead to generalization and concept understanding, which are essential for application of the material in later courses and professional tasks. A new type of instructional tool is needed which creates appropriate examples interactively but is embedded in a structured framework that reinforces concept development.

Computer-based tools aimed at logic design courses have been developed by several authors, based on ideas as diverse as finite state machine simulation [3], assembly-level design languages [5], and "virtual breadboards" [6].

The Web-based tutorials for the introductory logic design course described here are one part of a spectrum of learning experiences that also includes lectures, traditional labs, and use of CAD tools and FPGAs. The tutorials offer explanatory text as well as directed exercises that create an integrated presentation by, for example, simultaneously showing the circuit description as a truth table, a two-level algebraic expression of selectable form, and a circuit schematic. The exercises allow students to visually trace the flow of logic levels through the circuits, see the effect of Karnough and Quine-McCluskey minimization techniques, and even explore more sophisticated implementation techniques using multiplexers and decoders.

## Design of Java-based Instructional Materials

Previous experience from the design, implementation, and use of interactive instructional materials as Motif applications [3] was used to guide the design decisions for the new Java-based approach. Java is far more portable and accommodating to different platforms than Motif. It is also much easier to combine hypertext with the graphical presentations of interactive applets than with previous systems, so more customization is possible. Given these advantages, investment in extension and expansion of interactive tutorials to Java-based tutorials was reasonable.

Three levels of design were important for the definition of this project. The scope of each component and the structure for accessing each component was determined first. Since there is some overlap between courses in any curriculum and different instructors progress along a variety of paths through the course material, a full course was determined to be too large a unit for design. Instead "topics" were chosen for greater flexibility since one topic might support several courses, and topics can be activated in any order.

For the second level of design the content of the first group of topics was specified in terms of the fundamental concepts to be described and demonstrated. Finally, a set of Java applets was defined to support these topics. In many cases an applet can support multiple topics and small

variations on an applet can extend its useful range. The development of several topics used to support a first course in logic design is described below. Lessons learned from this experience will be used to design new families of applets which will support additional lower division courses as well as technology courses for nonmajors [4].

## Structure

The structure of the interactive instructional material is based on a collection of "topics", each of which covers a concept that might be taught as the focus of a single lecture. Each topic combines discussion and review text, guided examples, problems, and an interactive Java applet with graphical presentations of relationships and interactions. The cost of designing, implementing, and testing the Java applets is high compared to the other hypertext-based components of the tutorials, so the applets are designed to support multiple topics and multiple presentation styles of individual topics. The lower cost of development for the hypertext components allows relatively easy updating, extension, and customization to complement a wide variety of course organizations.

There are three hierarchical levels within the tutorial structure to allow easy access to the topics, and many topics may be reached using multiple paths. At the highest level (shown in Figure 1), the user is presented with three panels containing a list of supported courses on the left, a list of all topics in the center, and brief instructions for usage on the right. Students may access topics in two ways. All topics are listed in the center frame alphabetically, so the user who knows what specific topic should be reviewed will always have direct access to any topic using this index style approach.

Alternatively, the student looking for topics related to a particular course may wish to browse a list containing a subset of relevant topics. The list on the left of supported courses allows the user to select a specific course. Then the appropriate list of relevant topics is written over the instruction panel on the right, and the user may select a topic from that list. Within a course list, topics are listed in a developmental order. This list may also include links to other support information such as text-only descriptions, tutorials on CAD tools used for design, simulation, and implementation, and vendor information for commonly used components. These lists will contain major headings and subheadings to facilitate navigation.

When a topic is selected from either the course list of topics on the right or the index list in the center, a new window is created which contains the applet supporting this topic along with a narrow text column in a panel on the right. This is shown in Figure 2 for the “Truth Table Definition” topic. Initially the text column contains operating instructions, but navigation buttons above the column allow a choice of operating instructions, discussion, examples, or problems. All four of these text columns are reasonably concise, and each contains navigational information at the top so a specific part of the text may be located efficiently.

- The operating instructions include specific directions about how to use the interactive inputs. Although the user interface is designed to be intuitive and operating instructions may seem redundant, it is important that students feel there is no learning curve investment required for the tutorials. The instructions also include tips on the best screen size, adjusting borders of the frame panels, and how to exit.
- The discussion option presents a description of the topic organized as answers to a series of short questions about the topic. This is also shown in Figure 2. Each question serves as a heading for a two or three sentence answer. The discussion section includes definitions of terms and relationships as well as the significance of the topics in context. All of the questions also appear at the top of the column with links to the answers so that a student may quickly navigate to find the answer to a specific question. The discussion section also includes links to other related topics. For example, in the answer to the question about the ordering of the inputs in a truth table there is a link to the binary number representation topic.
- Examples are designed to progressively illustrate specific aspects of the topic and are grouped so that following a series of two to five examples will result in the desired demonstrations. Each example specifies the inputs that the user should select and then describes what should be observed and what aspect of the concept is being demonstrated. This approach is preferable to automatically setting all the input conditions for each example and presenting the example sequence as a slide show because the student will observe the response to each input as he changes it. At any time the student may choose his own input selections and spontaneously create his own examples.
- The problems are similar to problems typically listed at the end of a text chapter and can be answered using the applet, the discussion text, and the examples. One type of problem can be answered by using specified inputs and making the required observations about the output. A second kind of problem requires that the student figure out how to specify the inputs necessary to answer the question.

The applets are designed to always show relationships and multiple perspectives so that the student can transfer insight from a representation which is comfortably intuitive to alternate representations that may be more appropriate for specific objectives. The design and function of the applets will be discussed in a later section.

## Content of Topics

For each course supported by the Java-based instructional material, a set of topics has been selected to explain and demonstrate important concepts in a developmental order. For example, the entry level logic design course has topics listed in five basic groups. The first group covers basic definitions, operations, and representations used in all the following topics. Useful combinational components form the second group, which covers arithmetic circuits and other components at a medium scale integration level such as multiplexers and decoders. Specific tutorials for schematic capture and simulation design tools are also included. More advanced groupings include register structures and general sequential circuits.

For the beginning logic design student it is important to establish and reinforce the multiple equivalent representations of logic functions. Although these different representations are completely equivalent and interchangeable, each has a "niche" where it is most natural or appropriate, and each best matches the cognitive style of a subset of students. The applets show the relationships of these representations, how to use each effectively, and how to convert from one to another. These topics might also be included in other entry-level courses in engineering or courses designed for nonmajors.

The "Truth Table Definition" topic shown in Figure 2 is one of the most basic topics. The truth table is an exhaustive list of all possible input combinations and the corresponding outputs, and is often used for function specification from control inputs. This representation can be attractive to students who are not yet comfortable with Boolean algebraic operations. The truth table can be used to verify the equivalence of algebraic expressions simply by evaluating each rather than applying theorems. However, this approach becomes intractable as the number of input variables increases, so it is important to let the student start here at a reasonable comfort level, but migrate toward algebraic and schematic representations.

The applet used in Figure 2 allows the user to specify the number of inputs for a logic circuit, but limits the number of outputs to one. The user can interactively set the values in the truth table, complement the truth table output values, or initialize the truth table from a library of arithmetic and logic functions. The "majority" circuit has been used in Figure 2. The same applet is used in the "Maxterm and Minterm" topic which is basic to the conversion from truth table representation to algebraic representation.

The Karnaugh map or K-map is similar to the truth table in that it explicitly represents all possible input combinations and the output associated with each. However, the physical arrangement of the data makes algebraic relationships more visible. The applet shown in Figure 3 is based on the applet used for the "Truth Table Definition" topic, but displays the K-map arrangement of output values rather than the list of maxterms and minterms. This applet is used to define the K-map, define logical adjacency, and demonstrate the simplification or adjacency theorem. In addition to the student selections available in the previous applet, the student can also select size of the group of adjacent terms, whether the grouping represents products or sums, and whether the map is created in row or column order to match various textbook styles. In Figure 3, there are groups of 1, 2 and four minterms that are adjacent. The student has selected 2 as the size to show, so pairs are circled and the algebraic representation of all pairs is listed below the K-map.

Algebraic representation of logic functions is powerful, versatile, and essential for specification using hardware description languages. Students are initially uncomfortable with algebraic manipulations and the lack of uniqueness, but the topics described above take the student from a basic truth table representation to a canonical algebraic representation and then to a more efficient representation using the adjacency theorem. Additional topics cover the K-map minimization techniques, definitions of prime implicants and essential prime implicants, the Quine McCluskey tabular minimization technique, and iterative minimization techniques. Figure 4 shows an example of the use of the Quine-McCluskey method based on the original truth table applet. The user can step through the minimization process using individual buttons to create the implicant table, create the covering table, identify the essential prime implicants, and then identify the minterms covered by the prime implicants. All minimized functions are shown in the horizontal field above the tables.

Circuit schematics provide a third representation of logic functions, and schematic capture and simulation are often used in design. Applets are used to show circuits in both canonical and minimized forms based on the truth table specification. The applet in Figure 5 allows the user to select a set of input values in the circuit panel and then step through the circuit, level by level, with red lines indicating a logic 1 and green lines indicating a logic 0. (Undetermined values are shown in black.) This applet shows the truth table, algebraic representation, and circuit together.

After the basic representations are fully understood, they are used in components that become the building blocks of more highly integrated circuits. Often the multiplexer or decoder is the first component of this type to be investigated. The applet in Figure 6 is the first of the multiplexer applets used for functional definition. It shows the compressed truth table using nine lines instead of 128, a schematic symbol, a corresponding graphic symbol reinforcing the concept of switching exactly one of the inputs to an output, and a logic circuit using the three basic operators AND, OR, and NOT. When the user changes the MUX input values, the switch adjusts its position, and the red, green, and black lines in the schematic are changed in response to the input change. The user can choose the number of select inputs for the multiplexer and whether the enable is high, low, or not

present. Additional applets show how to create hierarchies of multiplexers and how to use them as a basic building block for arbitrary logic functions.

## Applet Design

Our experience in implementing the Java applets that are at the heart of the instructional materials described here emphasizes the importance of carefully selecting and defining the Java classes that embody the concepts being explained. Previous efforts at implementing applets for a much smaller set of logic design concepts, as well as computer network concepts and differential equation animations [1, 5] had produced workable applets that were difficult to maintain and even more difficult to extend beyond their initial application. Knowing in advance that we would be implementing a series of tutorials that led students from introductory ideas to more advanced concepts, we wanted to create a set of "pluggable" components that could be used across all the tutorials, and potentially contribute to new materials beyond the scope of the work described here.

Unfortunately we have no guaranteed design methods to ensure applet reuse. However, an explicit, predefined strategy and a long-range view regarding possible uses is essential. We devoted a great deal of thought to considering the topics to be covered, various representations for a logic function, the representations we thought most useful for various topics, and the transformations that would be necessary between the possible representations. We also storyboarded possible screen layouts appropriate for the various topics, to define a user interface that remains relatively consistent across topics, even as the functions that must be performed through the interface change.

We decided to use a sum of products expression as an internal representation for the logic function a table implements. Tables are characterized by the number of inputs and outputs, which define the table's shape. For each module there are upper bounds on both, which constrain the screen area needed to display the table. Functions are provided to convert between sum of product and product of sum expressions, to display a graphical representation of the truth table, and to accept student input of an expression in algebraic form or by clicking on output values to toggle them between 0 and 1. The interface contains three distinct areas: an interaction area at the top for user/applet communication (e.g., choice of predefined logic expressions, size of table, operations provided by the module), a window on the left for displaying the truth table for the logic function, and a window on the right for topic-specific displays (Figures 3 - 6).

A single Java class (TruthTableImg) extends Java's Canvas class [2] to provide screen output facilities, and implements both the interface and the behavior of the truth table. Implementing a module for a specific topic (for example, Karnaugh map minimization), involves coding one or more applets that extend Canvas. These topic-specific applets provide appropriate functionality (e.g., identifying minterm pairs) as well as methods to display explanatory diagrams on the screen. Then TruthTableImg is modified, either by extending the class or by copying and editing the original text, to add interaction controls needed for the new topic and to use the topic-specific applets to display the explanatory diagrams.

We have used this approach to implement ten different modules. Java code related to truth table functionality remains essentially unchanged across modules. Much of the code for handling interaction remains fundamentally unchanged, except for different placement of buttons and different content of predefined logic functions. The topic-specific additions for displaying conceptual information (drawing K-maps, for example) change for almost all modules. Depending on the complexity of that display, TruthTableImg may expand very little or nearly double in size.

## Summary

Java-based Web tutorials offer an attractive means of expanding student opportunities to explore concepts in a discipline outside of typical learning experiences as lectures and traditional laboratories. We have created a set of such tutorials for an introductory logic design course that provide multiple ways of accessing the topics covered, multiple formats for presenting a topic, and multiple perspectives on each topic to accommodate student preferences or pedagogical objectives. By paying careful attention to the organization of topics, the concepts to be presented, and the functions provided by Java applets used in the tutorials, these materials can be reorganized to support other classes relatively easily. Similar attention to the detailed design of the applets reduces the effort required to extend them to related topics and functions.

## References

[1] Danielson, R. and S. Wood, "Stimulating Introductory Engineering Courses with Java," *Proceedings of 1998 Frontiers in Education Conference*, IEEE Press, 1998, pp. 897 - 902.

[2] Flanagan, D., *Java in a Nutshell, 2nd Edition*, O'Reilly and Associates, Sebastapol, CA, 1997.

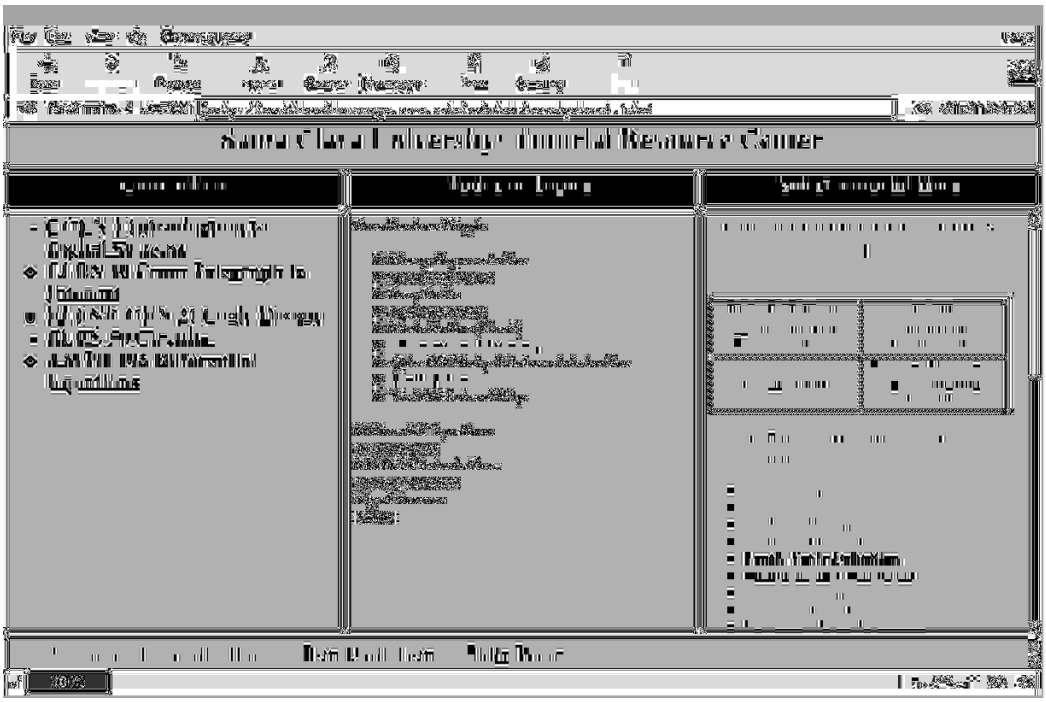

*Figure 1. High-level Student Selection Page*

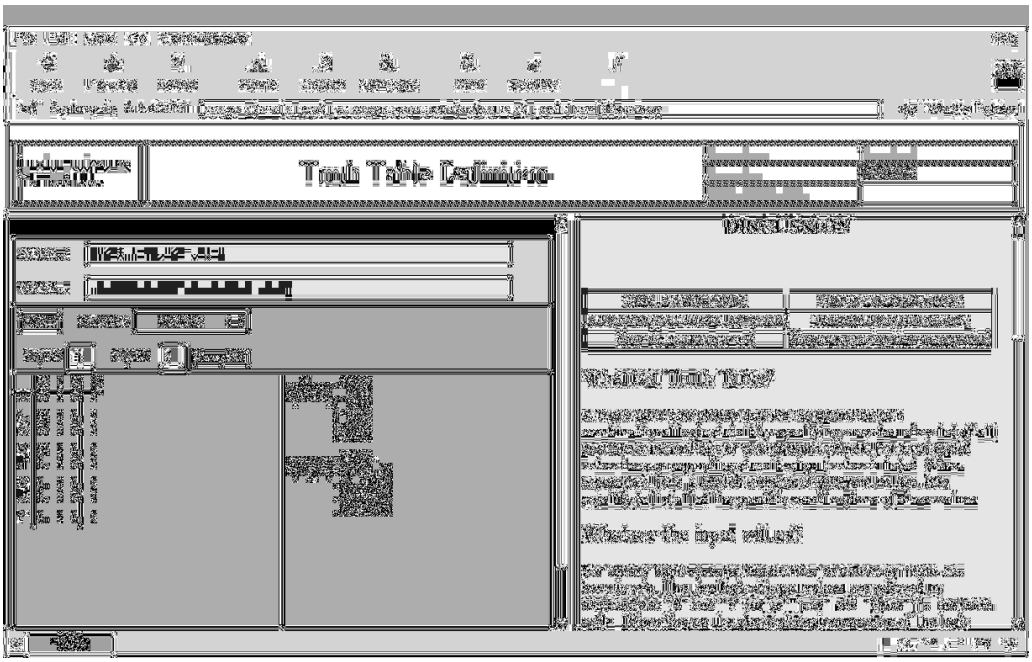


*Figure 2. Truth Table Definition Topic*

[3] Hacker, C. and R. Sitte, "Implementing Finite State Machines in a Computer Based Teaching System," *Proceedings of the SPIE 1999 Education in Microelectronics and MEMS Conference*, SPIE, 1999, pp. 110 – 117.

[4] Katz, R., *Contemporary Logic Design*, Addison-Wesley, Reading, MA, 1993.

[5] Maurer, P., "Enhancing the Hardware Design Experience for Computer Engineers," *Proceedings of the 1998 Frontiers in Education Conference*, IEEE Press, 1998, pp. 60 - 63.

[6] Serra, M., E. Wang and J. Muzio, "A Multimedia Virtual Lab for Digital Logic Design," *Proceedings of the 1999 IEEE International Conference on Microelectronic Systems Education*, IEEE Press, 1999, pp. 39 – 40.
[7] Wakerly, J. and H. Stone, *Digital Design: Principles and Practices*, Prentice Hall, Englewood Cliffs, NJ, 1999.
[8] Wood, S., "A New Approach to Interactive Tutorial Software for Engineering Education," *IEEE Transactions on Education*, 39(3), 1996, pp. 399–408.
[9] Wood, S., "A Concept Oriented Freshman Introductory Course Utilizing Multimedia Presentations and Group Laboratory Experience," *Proceedings of the 1998 Frontiers in Education Conference*, IEEE Press, 1998, pp. 824 - 829.
[10] Wood, S. and R. Danielson, "Web–Based Enrichment Courseware for Introductory Engineering Students," *Proceedings of the International Conference on Computers and Advanced Technology in Education*, International Association of Science and Technology for Development, 1998, pp. 121 - 125.

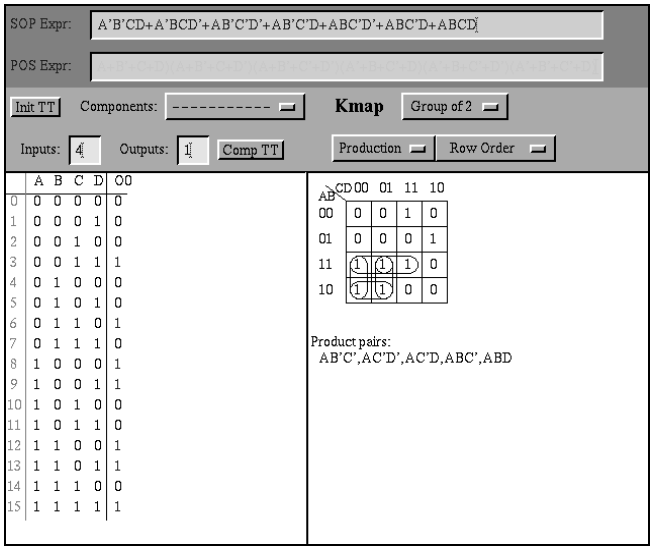


*Figure 3. Karnaugh Map*

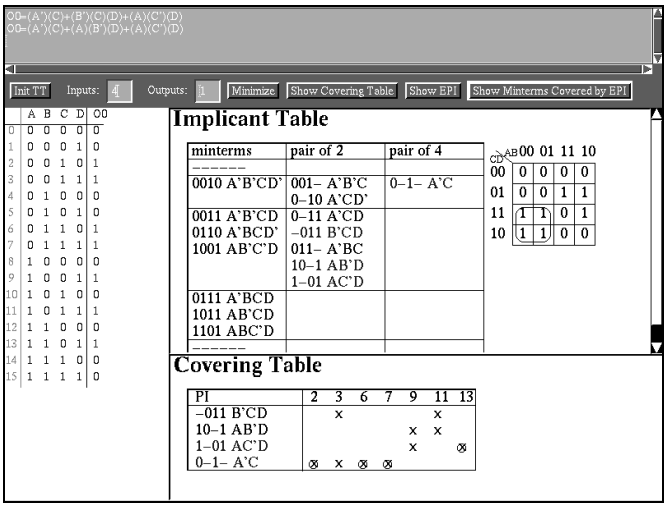


*Figure 4. Quine-McCluskey Method*

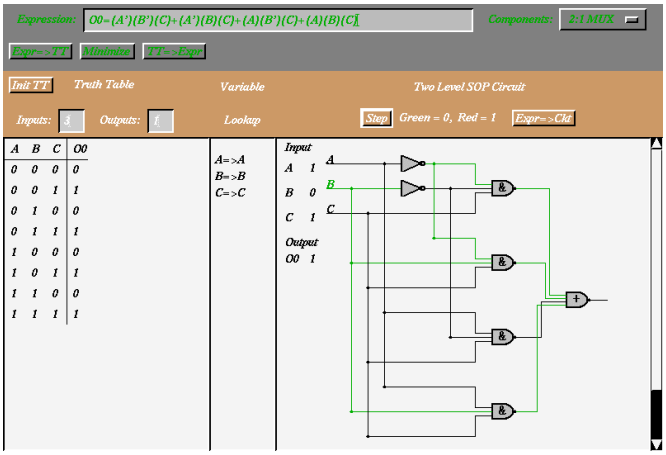


*Figure 5. Circuit Schematics*

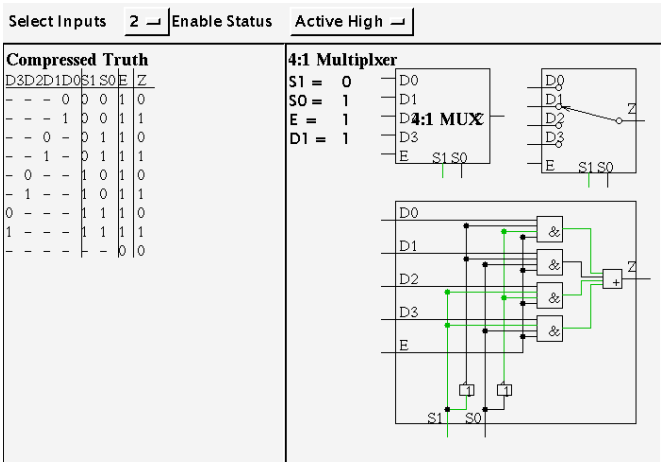


*Figure 6. Multiplexer Functional Definition*