

XBOOLE AND THE EDUCATION OF ENGINEERS

B. Steinbach¹, Ch. Posthoff²

¹Freiberg University of Mining and Technology, Freiberg, Germany

²The University of The West Indies, St. Augustine Campus, Trinidad & Tobago

This paper emphasizes the educational aspect of discrete devices. The growing complexity of such systems requires both extended skills of the engineers and extended methods in their education. The details of modern discrete devices are created by computers. Hence, computers must be included in the education process. Together with an extended educational concept we show in this paper how specialized software supports the educational process. As example we use the XBOOLE monitor and explain its application to solve tasks related to discrete devices.

Introduction

Many academic institutions all over the world pay more and more attention to "computer-supported" or "computer-assisted" learning. The range of these concepts is very broad, it starts very often with PowerPoint presentations and ends with a fully Internet-based presentation of the teaching material. However, the didactics of such an approach, the advantages and disadvantages, the desired and the actual outcome are often very unclear and deserve careful and serious consideration.

First of all it has to be emphasized that "computer-supported" not only considers the computer per se, but also the existence of appropriate software or even nowadays an appropriate (digital) multimedia environment which is specialized with regard to the topic to be taught. This combination of hard- and software will be the necessary assumption for the problems to be considered.

As a second basic assumption we emphasize that it is **not** intended to replace the classroom or the conventional teaching. Many people understand the computerization of teaching or learning mainly in replacing the taking of notes in the classroom by slide shows, by handing over files with the text to the students etc. If this would be the intention, then a file of the respective textbook could be copied to each student, and this would be the solution.

1. Goals and Methodology

It is our goal to increase the quality of learning considerably by examples and exercises which can be handled on a much higher level when computer support can be provided. Sometimes, and in application-oriented areas all the time, computer-based solutions are the **only** possibility of solving such problems. We demonstrate this approach by using the book on "Logic Functions and Equations" [1] as the theoretical background (possibly as a textbook for a course "Logic Design" or "Logics for Computer Science" or similarly) and the software package XBOOLE as the learning software for a student. Other courses (from Mathematics, Languages, ...) and other packages (such as MAPLE, MATCAD, MATEMATICA, ...) could be used as well.

- *First assumption* The course material has been presented to the student in a proper way (by textbook, in class, by tutorials, assignments, ...) such that the student has a given understanding of the area which has to be extended, deep-

ened and made ready for the solution of real-world problems (or at least more relevant for the solution of real-world problems). Very often in class only small examples will or can be presented which have more or less the character of toys.

- *Second assumption* The educational software must have a user interface which is sufficiently simple and problem-oriented. Nowadays everybody is using many sophisticated software packages, and all of them require a good amount of knowledge to be used properly. It is frustrating to sit before the screen in order to solve a problem and to struggle with the peculiarities of the application program, instead of solving relevant problems.
- *Third assumption* A student must be able to transfer a given problem into the given learning system. He/she must be able to split the problem into a sequence of subproblems which can be solved by using the (XBOOLE) software and the solution of which can be combined into a solution of the overall problem. The best method to learn and to manage this approach is the **learning from examples**, i.e. the presentation and explanation of a sufficiently large number of examples. Hence, a second book will be provided now that has the same structure as the original textbook. Each chapter will contain typical problems related to the contents of this chapter which has been represented in the first book. It starts with some (two or three) examples that are presented and explained very much in detail, followed by some examples with short explanations, and finally a longer list of test examples (with and without solutions) can be used by the student to test his skills.

It is quite easy to understand that, by using this methodology, at least two achievements can be stated:

1. The student will be free of a lot of routine calculations that are error-prone and do not really contribute to the problem-solving process; the solution process goes far beyond problems that can be solved by hand; in this way the educational process will be more relevant for real-life problems, hence, the educational process will be more relevant for the later professional life.
2. The student has more time and more possibilities and/or facilities to explore the solutions, to understand the meaning of the solutions, to represent functions graphically etc. which results in a much better understanding of the problem area, the solution process, the applicability of theories etc.

2. Steps to an Implementation

This approach must be accompanied by (at least) the following steps:

- The educational process must put great(er) emphasis on modeling aspects. Students must be able to formalize real-world problems, to find limitations for the models, to evaluate the correctness of solutions etc. As a logical consequence, courses such as "Modeling and Simulation", "Scientific Computing" etc. must be introduced as soon as possible, compulsory for all Science and Engineering students.

- More specialized areas must be covered by appropriate software packages (which very often have to be created by the teaching academics themselves, see XBOOLE).
- The examination process also must take these developments into consideration. Computer-based examinations are much more appropriate to test the students' knowledge and skills than (trivial) calculations by hand and memorizing facts that can be stored on (tiny) hard disks.

The final solution which we have in mind for the use of XBOOLE comprises

- the textbook "Logic Functions and Equations" [1],
- the book "Logic Functions and Equations - Examples and Exercises" (to be finished),
- the software package XBOOLE that can be downloaded (for free),
- a collection of examples and solutions on a web site that can be growing and growing (by contributions of students, academics, relevant applicants etc.).

As an illustration of the method see the following example.

3. Set up the Working Environment for XBOOLE

The XBOOLE monitor can be downloaded by everybody for free from the following web page.

<http://www.informatik.tu-freiberg.de/xboole>

On the left side of this page the link XBOOLE Monitor can be found. A click on this link leads to the download page of the XBOOLE monitor. In order to download the XBOOLE monitor, the button labeled by XBOOLE Monitor and located at the bottom of this page must be pressed. In the opened dialog File Download the button SAVE must be pressed in order to start the download of the file XBOOLEMonitor.zip into a directory of your choice. After the download of this file the dialog Download completed appears. In this dialog the button Open Folder should be pressed. The opened folder includes the file XBOOLEMonitor.zip downloaded before. In the context menu after a right click the item Extract all can be chosen to extract all zipped files into a new directory XBOOLEMonitor. Of course, other procedures to extract all zipped files can be used as well. The XBOOLE monitor is now ready for use.

The directory XBOOLEMonitor includes the unzipped files of Table 1. It can be seen that the XBOOLE monitor is prepared for use in English or German. The executable file of the XBOOLE monitor is xbm32.exe. It can be started without any further installations and selects automatically the language of both all representations and the help environment depending on the language of the used Windows operating system.

Table 1
Meaning of the files in the directory XBOOLEMonitor

| Name of the File | Meaning of the File |
|------------------|--|
| xbm32.cnt | content structure of the used help file |
| xbm32.exe | executable file of the XBOOLE monitor |
| xbm32.hlp | used help file |
| xbm32_e.cnt | content structure of the English help file |
| xbm32_e.hlp | help file in English |
| xbm32_g.cnt | content structure of the German help file |
| xbm32_g.hlp | help file in German |

All further explanations relate to this XBOOLE monitor. Thus, it is recommended that the reader applies this given procedure for downloading the XBOOLE monitor to the computer available to perform all further logic calculations.

Fig. 1 shows a screen shot of the XBOOLE monitor considerably reduced in its size to save space in this article. The size of this window can be resized as each other window on the screen.

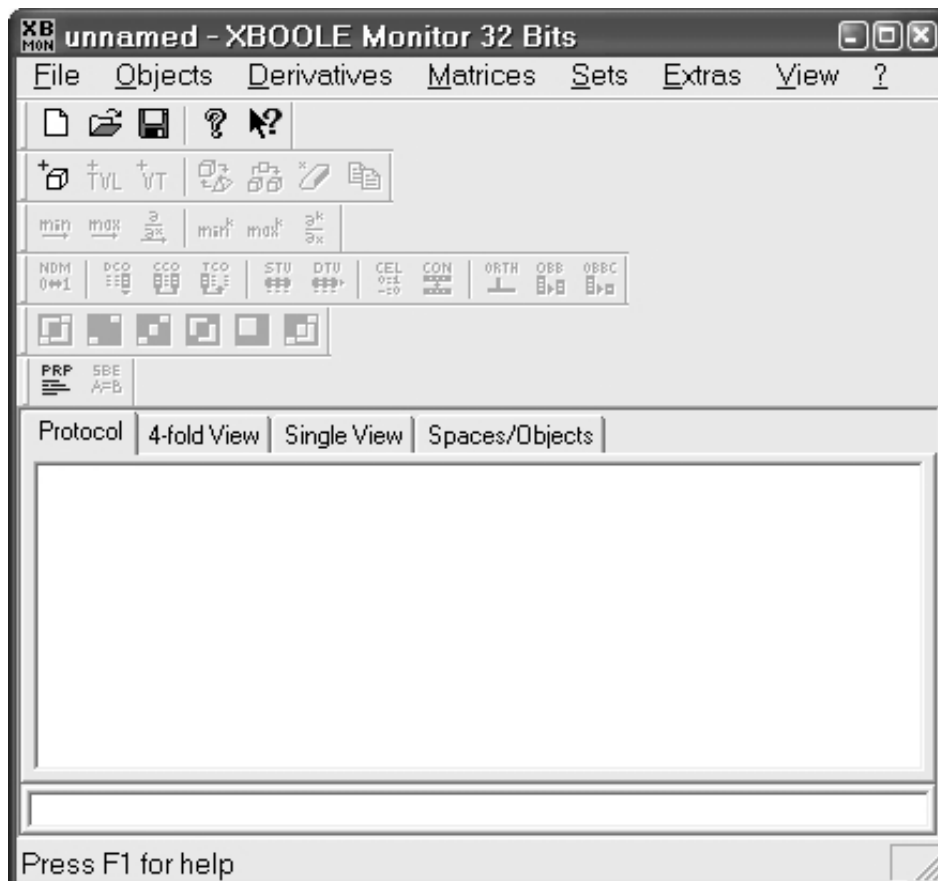


Fig. 1. Complete window structure of the XBOOLE monitor

The window of the XBOOLE monitor comprises several parts. In the following the main purpose of these parts will be explained.

The headline of the XBOOLE monitor window shows the icon of this program, the name of the file lastly opened or stored, and the title of the program *XBOOLE Monitor 32 Bits*. The term unnamed as name of the file means that no file is used until now. Otherwise a file name having the extension `sdt` is shown. Such an `sdt`-file allows to break the work with the XBOOLE monitor. After loading a stored `sdt`-file the work with the XBOOLE monitor based on previous data can continue.

The menu bar is located below the headline of the XBOOLE monitor. The menu is structured like a tree and allows controlling the behavior of the XBOOLE monitor completely. Typically the menu items will be selected using the mouse. Alternatively the ALT-key in connection with further keys of the keyboard can be used. Additional information for the selected action must be specified in most cases by using special dialogs.

Below the menu bar a set of toolbars is located. Each of these toolbars can be shown or hidden separately. Their position can also be arranged by the user. All these adjustments are stored in the windows registry database automatically such that the adjustments of the toolbars do not change after the next start of the XBOOLE-monitor. The toolbars were introduced to shorten the handling of the XBOOLE monitor. Instead of selecting the whole path between the root and the leaf in the menu tree step by step, the action associated to the wanted leaf can be selected by a single click on the associated icon in a toolbar. Note: toolbar icons are defined only for most often used actions. Thus the actions controllable by the toolbars are a subset of all possible actions.

The part below the toolbars covers the main space of the XBOOLE monitor window. The XBOOLE monitor visualizes different types of information in this area, but allows partial interactions with the user, too. This part is structured by a tab control. A click with the mouse brings the page associated to the tab in the foreground.

The first tab is labeled by `Protocol`. On this page a protocol about each action executed by the XBOOLE monitor is automatically recorded. The representation of the protocol does not depend on the way of initializing the action. There are three possibilities to activate an action: first by an item of the menu, secondly by an icon of the tool bar, and thirdly by a command of the command line. The protocol is written as a sequence of commands. There is a possibility to store the protocol and execute this sequence of commands later on again. Such a sequence of commands is called `problem program` or shortly `PRP`.

The second tab, labeled by `4-fold View`, and the third tab, labeled by `Single View` provide basically the same behavior. The difference between these pages is that the 4-fold view consists of a 2x2-matrix, where each subsheet of this matrix has the same behavior as the whole single view. In such a view a selected list of ternary vectors (TVL), an associated Karnaugh-map, or a selected tuple of variables (VT) can be shown. There is an edit mode in such a view which allows editing the elements of a TVL. If there is not enough space, scroll bars appear automatically and allow the selection of each part of the data shown. It is suggested to use the single view for very large objects. Otherwise the 4-fold view is preferred because four objects can be seen at the same time.

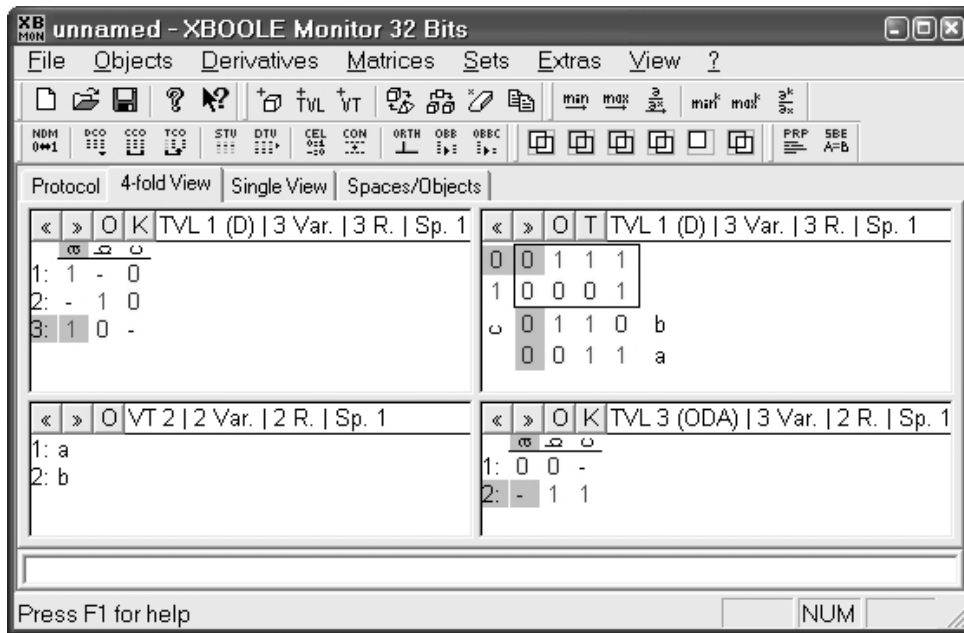


Fig. 2. Example of the 4-fold View in the XBOOLE - monitor

Figure 2 shows an example of the 4-fold view in the XBOOLE monitor. The top left part shows the first object, a TVL in disjunctive form, depending on 3 variables, consisting of 3 rows, and defined in the first space. The top right part shows the Karnaugh-map of this logic function. The button labeled by K switches to the Karnaugh-map and the button labeled by T to the TVL, respectively. The bottom left part shows the second object, a VT of two variables in the first space. The bottom right part shows the third object, a TVL in orthogonal disjunctive antivalence (ODA) form, depending on 3 variables, consisting of 2 rows, and defined in the first space. This TVL was calculated as complement of the first TVL.

The fourth tab is labeled by Spaces/Objects. This page is divided vertically into two views. The left view shows a list of all details about the Boolean spaces defined by the user. The right view shows a list of all TVLs and VTs recently stored in the XBOOLE monitor, where each of these objects is uniquely identified by its object number. In addition to the number of the object and its type, information about the form, the space, and numbers of variables, rows and boxes is given.

The command line is located below the main part of the XBOOLE window. While the menu bar and the toolbar allow a simple intuitive handling of the XBOOLE monitor, the application of the command line requires knowledge of the commands to control the actions. The benefit of the command line is that XBOOLE operations can be activated faster, because interactions in dialog windows are omitted. The details about all commands can be studied in the XBOOLE help system.

The bottom of the XBOOLE window is built by the status bar. In this bar help information about the action selected by a menu item or toolbar icon will be displayed.

4. Analysis of a Combinational Circuit

Let us take the combinational circuit of [1], page 271, as example. Fig. 3 repeats this example.

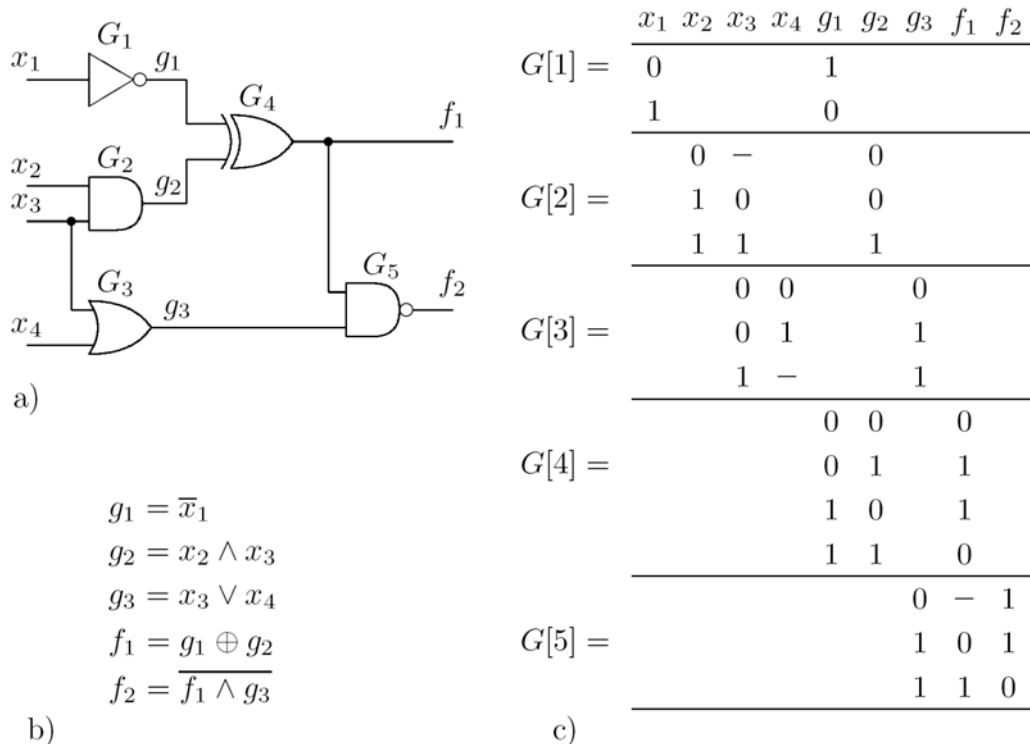


Fig. 3. Structure of a combinational circuit:
a) diagram; b) system of logic equations; c) set of local lists of phases

The behavior of each gate can be expressed by a logic equation (see Fig. 3 b) or a list of phases (see Fig. 3 c), respectively. The global task to be solved is the analysis of the given circuit. That requires the calculation of the behavior that may be expressed by a global list of phases. This task can be solved in different ways. The aim of this exercise is not only the calculation of the behavior. The students should learn additionally the relationships between logic equations, the list of phases as their solution, the methods for solving equations and substitutions using XBOOLE.

Assume the input-output behavior is desired. Hence, the internal signals g_i , $i = 1, 2, 3$ must be eliminated. This can be done manually by substitutions. From Fig. 3b) we get (1) and (2), respectively.

$$f_1 = \bar{x}_1 \oplus (x_2 \wedge x_3) \tag{1}$$

$$f_2 = \overline{(\bar{x}_1 \oplus (x_2 \wedge x_3)) \wedge (x_3 \vee x_4)} \tag{2}$$

These equations depend on 4 or 5 variables only, but it takes time to find a solution. In order to focus on more important topics, the XBOOLE monitor is used to solve these equations. First we need a Boolean space large enough to carry all variables. Several Boolean spaces can be defined using the menu item - Define Space ..., the SPACE button of the toolbars, or the command `space [vmax[sno]]`. A favored order of the variables can be assigned using the menu item Objects - Attach Variables ... or the command `avar [sni]`. After this preparation of the environment, the equation can be solved separately using the menu item Extras - Solve Boolean Equation ..., the SBE button of the toolbars, or the command `sbe [sni[tni]]`. The expression of an equation can be edited directly in the dialog window that is open to solve the equation. The operation

signs can be assigned using the option button. The last solved equation can be reloaded using the \Leftarrow button in the dialog window to solve the equation. It is suggested to save an edited equation as txt-file using the SAVE button. Such a file can be reloaded using the OPEN button and changed later on. The real work to solve the equations (1) and (2) using the XBOOLE monitor is to type the equations

$$f1 = /x1\#(x2\&x3) \tag{3}$$

$$f2 = /((/x1\#(x2\&x3))\&(x3 + x4)) \tag{4}$$

in the dialog window and press the SOLVE button. Fig. 4 shows the results. On the left-hand side the list of phases of the function f_1 is shown, at the top as TVL and below the associated Karnaugh-map.

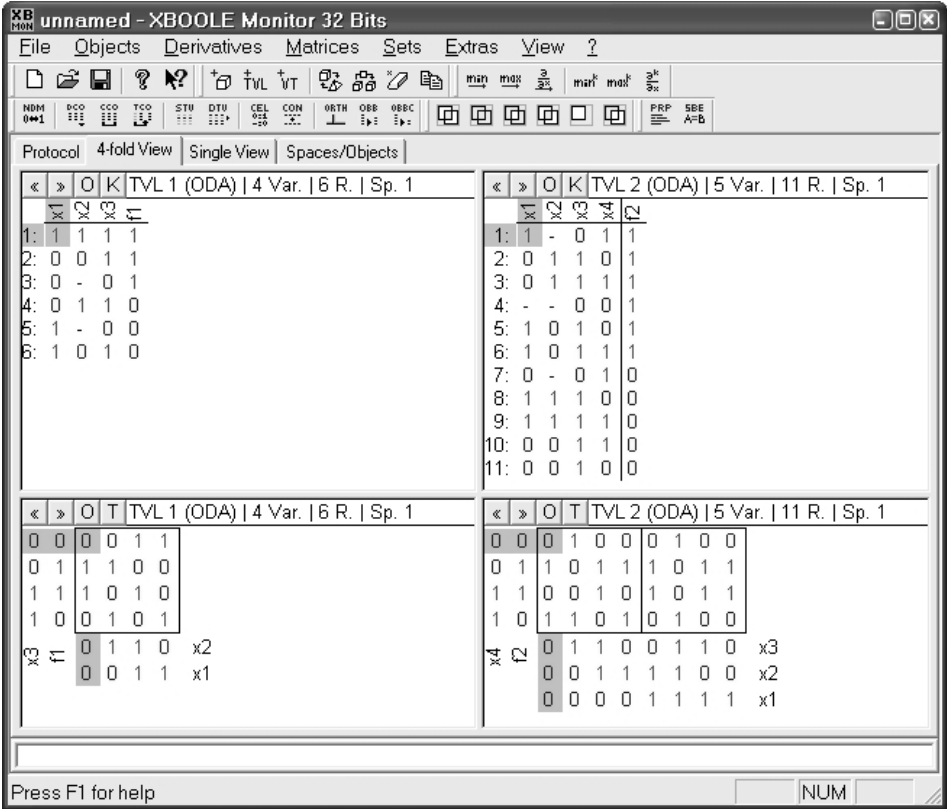


Fig. 4. Separate behaviors of the functions f_1 and f_2 of Fig. 3

The result of Fig. 4 should be evaluated in detail. It shows that for each input pattern exactly one output value for each function exists. The ternary representation helps to save space. The 6 vectors in TVL 1 express 8 phases, and the 11 vectors in TVL 2 express 16 phases. TVL 2 can be compressed even more from 11 to 7 rows using the OBB operation. Execute this compression and compare the TVLs.

As next the complete input-output behavior is required. This behavior solves both equation (1) and equation (2). Thus, the equations (1) and (2) must be solved conjointly as a system of logic equations. There are two methods in order to calculate this solution. First, both equations can be solved simultaneously as a system of equations. This can be done as before, but both equations must be connected by a comma sign (,) into a system of equations:

$$\begin{aligned}
 f1 &= /x1\#(x2\&x3), \\
 f2 &= /((/x1\#(x2\&x3))\&(x3 + x4))
 \end{aligned}
 \tag{5}$$

Secondly, the solution vectors must belong to both the solution of equation (1) and the solution of equation (2). Thus the intersection of TVL 1 and TVL 2 leads directly to the desired solution. The intersection can be executed using the menu item `Sets - ISC - Intersection`, the `ISC` button of the toolbars, or the command `isc tni1 tni2 [tno]`. It is suggested to execute both methods and compare the results. Notice: the order and the merge of ternary vectors may be different for TVLs which express the same function. The content of two TVLs is equal to each other if their symmetrical difference (XBOOLE operation SYD) is empty. Fig. 5 shows in the top left view the list of input-output phases and in the top right view the associated Karnaugh-map. This result should again be studied carefully.

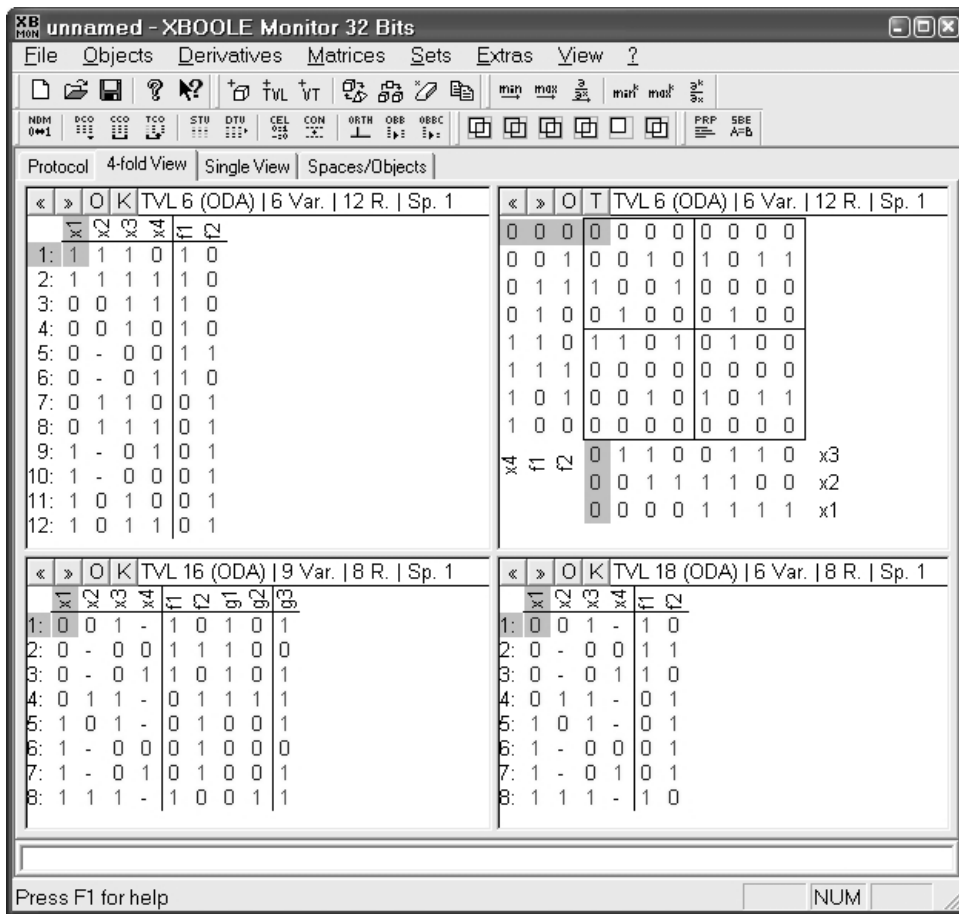


Fig. 5. Common behavior of the circuit of Figure 3

As next the complete behavior of the circuit of Fig. 3 is desired. The values of all internal wires and all outputs must be given in a list of phases. This can be done by solving the system of equations of Fig. 3 b as shown above. Another thought leads to a very simple solution method. The local behavior of the gates can be described by local TVLs as shown in Fig. 3 c. The phases of the complete behavior must fit into all these behaviors. Hence, the intersection (XBOOLE operation ISC) of all local lists of phases (TVL) covers the global list of phases. The benefit of this approach is that the local lists of phases can be taken from a library (see [1] page 270) and must only be adapted to the variables.

All required steps for this calculation of the complete behavior are included in a PRP, see Figure 6. The objects are labeled by numbers. The lists of the local gates $G[i]$, $i = 1, \dots, 5$ are associated to the objects number 11, \dots , 15. The global list of phases is calculated by four `isc` commands. The final `obb` command compresses the global list of phases in object number 16.

```

tin 1 11          tin 1 13          tin 1 15
x1 g1.           x3 x4 g3.         g3 f1 f2.
01              000                0-1
10.            011                101
tin 1 12        1-1.              110.
x2 x3 g2.       tin 1 14          isc 11 12 16
0-0             g1 g2 f1.         isc 16 13 16
100            000                isc 16 14 16
111.           011                isc 16 15 16
              101                obb 16 16
              110.

```

Fig. 6. PRP (slivered in three columns) that calculates a compressed list of complete phases

The PRP of Fig. 6 can be edited using any text editor. The execution of the prp can be started using the menu item `Extras - Execute PRP` or the PRP button of the toolbars. Figure 5 shows in the bottom left view the complete behavioral information about the circuit of Figure 3.

Finally, the input-output behavior should be calculated based on the complete behavior. That means, all input-output vectors are desired that appear in combination with any values of the internal signals g_i , $i = 1, 2, 3$. The m -fold maximum of the Boolean Differential Calculus fits to the statement *there is any* and can therefore solve this task. In order to indicate the variables g_i , $i = 1, 2, 3$ a tuple of variables can be used. The XBOOLE command

```

vtin 1 17
g1 g2 g3.

```

creates the tuple of variables $\langle g1, g2, g3 \rangle$ in Boolean space 1 as object number 17. The desired input-output behavior can be calculated using the XBOOLE command `maxk 16 17` 18. Fig. 5 shows in the bottom right view the behavioral information about the circuit of Fig. 3 restricted to the inputs and outputs using the XBOOLE operation MAXK.

Conclusions

We have shown in this paper that computer-supported education must be embedded in classical methods for education and that there is a real challenge in order to achieve the required skills of next generation engineers. In addition to the required basic hardware appropriate teaching software is necessary. We demonstrate by a small number of examples how such a teaching software can be used. The XBOOLE monitor is an appropriate teaching software.

References

1. Ch. Posthoff, B. Steinbach: Logic Functions and Equations - Binary Models for Computer Science. Springer, Dordrecht, The Netherlands, 2004.