A computer based teaching program for the design and analysis of digital counter circuits

A computer based teaching program for the design and analysis of digital counter circuits

Charles Hacker* & Renate Sitte†

*School of Engineering, C.Hacker@mailbox.gu.edu.au
†School of Information Technology, R.Sitte@mailbox.gu.edu.au
Griffith University, Gold Coast Campus, Qld, Australia.

ABSTRACT: Digital Flip-Flop counters are widely used for control logic applications, such as timers, clocks, remote controls, and process controllers. Flip-flop circuits are standard memory elements, and form the basis of sequential digital circuits. The teaching of sequential digital logic is core in the curriculum of many university digital electronic subjects. Students in first year digital electronic subjects often have difficulties grasping the theoretical concepts in the design and analysis of Flip-Flop counter circuits. This has prompted the author to developed a Microsoft Windows™ compatible software, *WinCounter*, that provides a tutorial style teaching aid for undergraduate students in understanding the mechanisms of digital counter design. The animated computer screen of the MS-Windows environment is ideal for visually conveying the required design and analysis procedures, allowing students to better visualise the techniques. The software was developed for both lecture demonstration and for students educational use, allowing students to study privately at their own pace. The software has been utilised in a digital electronic subject lectured by the author, and student feedback has indicated the software was educationally useful and

INTRODUCTION

Flip-Flop counter circuits form the fundamental digital electronic components of most logic control applications. Examples of the types of systems that use digital counter circuits are: electronic timers, electronic clocks, remote controls, and industrial process controllers.

Flip-flop circuits are digital electronic components that are standard memory elements. They are utilised in computer RAM circuits, and other data storage circuits. Counter circuits also require memory elements, to store the current counter state. Counter circuits thus use flip-flop circuits for this task.

The study of digital electronics covers the two broad areas of combinatorial logic (circuits with no memory) and sequential logic (circuits with memory). Teaching combinatorial and sequential digital logic circuits are part of the core in the curriculum of many university digital electronic subjects.

Students in first year digital electronic subjects often have difficulties grasping the theoretical concepts in the design and analysis of sequential Flip-Flop counter circuits. This has prompted the author to developed a Microsoft Windows™ compatible software, *WinCounter*, that provides a tutorial style teaching aid for undergraduate students in understanding the mechanisms of digital counter design. The animated computer screen of Microsoft Windows provides an easy to use, user friendly, environment that is ideal for visually conveying the required design and analysis procedures, allowing students to better visualise the techniques.

The software was developed for both lecture demonstration and for students personal use. During lecture demonstrations detailed intricate explanations are necessary to sufficiently convey the design and analysis strategies. During the explanation, it is easy for students to become confused and bewildered in the barrage of new information and technical terms. The designed software's animated output is ideal for anticipating possible confusion in the explanation, allowing students to observe the strategies during the explanation. The software was also intend for students personal use, providing the ability for a student to design and analyse their own counter circuit. This allows students to study the design and analysis procedures privately, at their own pace.

The Flip-Flop counter software complements other software previously developed by the author, for the design and analysis of combinatorial digital logic circuits. The existing software provides a tutorial that covers the design and analysis of Boolean Algebra (Hacker & Sitte, 1997; Hacker & Sitte, 1998), and the logic minimisation (Hacker & Sitte, 1999).
The Flip-flop counter software enhances the educational use of the existing combinatorial logic teaching package, by extending the existing software to include memory sequential circuits.

DIGITAL COUNTER CIRCUITS

A digital counter utilises Flip-Flop circuits to store the current state of the counter. Digital counters therefore usually form the functional hardware of a *Finite State Machine*. A finite state machine are abstract models of machines with a primitive internal memory. Each state of the finite state machine is a specific set of values stored in the memory elements, and represents the condition or state the machine current resides.

A finite state machine can be represented in tabular form, which is referred to as a *State Transition Table*. This table lists the *Present State*, *Control Logic inputs*, and the *Next State*. (An example of which is shown in Figure 2). The finite state machine can also be represented as a directed inter-connected diagram, referred to as a *State Transition Diagram*. The diagram consists of nodes, which in this case are the counter states, which are linked with arrows to the next counter state. An example of a State Transition Diagram, that have hexadecimal (0 to F) values for the states, is shown in Figure 8.

The digital counter (finite state machine) transitions to a new state depending on the current state and the next state function. This provides for an orderly reproducible path through the counter states. The next state function is provided by digital logic decoding of the current state values. Thus standard logic gates, (AND, OR, NAND, NOR, etc.) are used for the decoding. Boolean Algebra and gate minimisation techniques are utilised in the determining the required logic gates to achieve the Next State decoding.

The Flip-Flop memory circuits can be wired in a number of modes. These modes are; Data mode, Toggle mode, and JK mode. The Data mode is more commonly used in data storage applications, such as RAM circuits. The Toggle mode is commonly used in counter applications, where transitions are frequent. The JK mode can be switched between the Data and Toggle modes. A counter circuit can utilise a Flip-Flop that is wired in any of these modes, and hence the software allows for this.

DESIGNED SOFTWARE

The software provides an interactive tutorial for both the design of a flip-flop digital counter, and for the analysis of a flip-flop digital counter. The design is achieved by the user input of a count sequence, while the analysis is performed on a user input counter circuit. The flow diagram of the software structure is shown in Figure 1. The remaining figures in this paper are presented in the order of the *WinCounter* software flow diagram sequence.

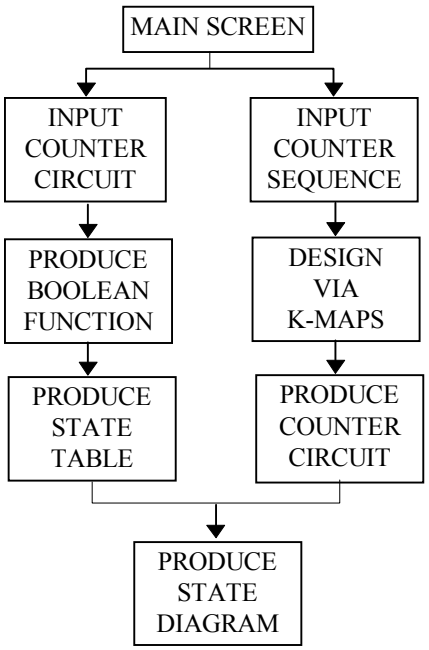
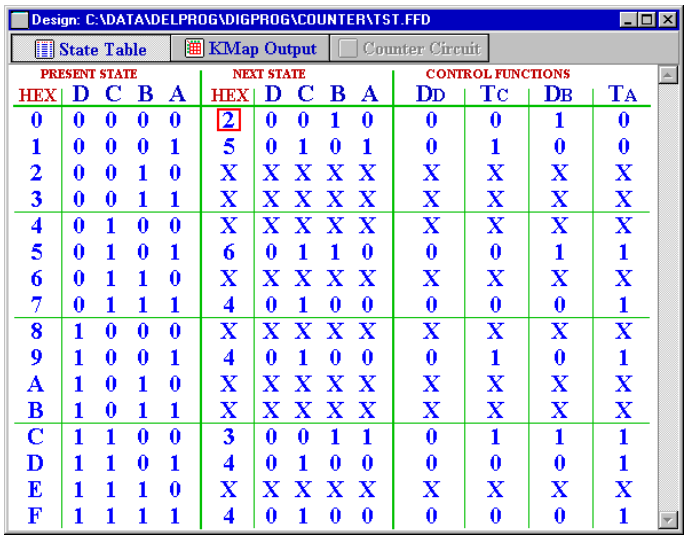


Figure 1: Flow diagram of the Counter Software

When designing the flip-flop counter, the user is prompted to input the desired counter sequence, and the modes for the flip-flop elements, (for example *Data* or *Toggle* mode). This determines the present and next state values for the State Transition Table. The control functions required to transition the counter from the present state to the desired next state is then determined by the program. An example of this State Transition Table screen is shown in Figure 2.



PRESENT STATE					NEXT STATE					CONTROL FUNCTIONS			
HEX	D	C	B	A	HEX	D	C	B	A	DD	TC	DB	TA
0	0	0	0	0	2	0	0	1	0	0	0	1	0
1	0	0	0	1	5	0	1	0	1	0	1	0	0
2	0	0	1	0	X	X	X	X	X	X	X	X	X
3	0	0	1	1	X	X	X	X	X	X	X	X	X
4	0	1	0	0	X	X	X	X	X	X	X	X	X
5	0	1	0	1	6	0	1	1	0	0	0	1	1
6	0	1	1	0	X	X	X	X	X	X	X	X	X
7	0	1	1	1	4	0	1	0	0	0	0	0	1
8	1	0	0	0	X	X	X	X	X	X	X	X	X
9	1	0	0	1	4	0	1	0	0	0	1	0	1
A	1	0	1	0	X	X	X	X	X	X	X	X	X
B	1	0	1	1	X	X	X	X	X	X	X	X	X
C	1	1	0	0	3	0	0	1	1	0	1	1	1
D	1	1	0	1	4	0	1	0	0	0	0	0	1
E	1	1	1	0	X	X	X	X	X	X	X	X	X
F	1	1	1	1	4	0	1	0	0	0	0	0	1

Figure 2: State Transition Table Input

The digital logic required to implement the control functions is then determined by Boolean logic simplification. The software performs the simplification by the Karnaugh Map technique, which results in the final digital logic functions to implement the next state functions. This simplification technique is shown in Figure 3.

A student may be requested to analyse an existing counter circuits operation, in addition to designing a counter circuit. In this case a student may actually be presented with a counter circuit, and be requested to determine the count sequence that results from this circuit. The software is thus designed to perform this analysis, from an initial user input counter circuit.

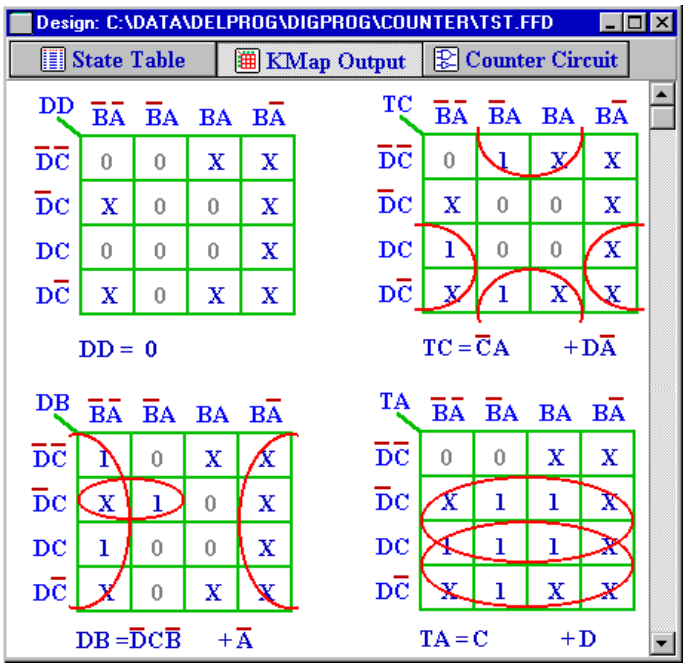


Figure 3: K-Map Simplification of the Next Sate Function

The Boolean functions determined by the K-Map simplification represent the logic gates required to implement the Next State decoder functions. The Next State decoder logic has inputs of the Present State, which is derived from the flip-flop memory outputs. The final counter circuit diagram is then produced. An example of this is shown in Figure 4.

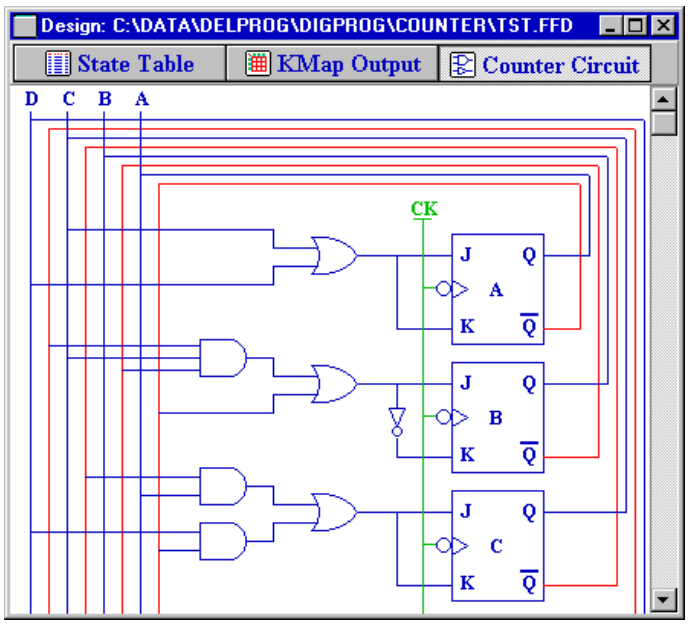


Figure 4: Final designed circuit to implement the counter

The user enters a counter circuit in the analysis screen. To achieve this the top menu toolbar offers logic gate types for selection, that can be placed in the circuit at any desired user location. A wire toolbar button can then be selected, to wire the logic gates in any user desired arrangement. An example of a user input counter circuit diagram is shown in Figure 5.

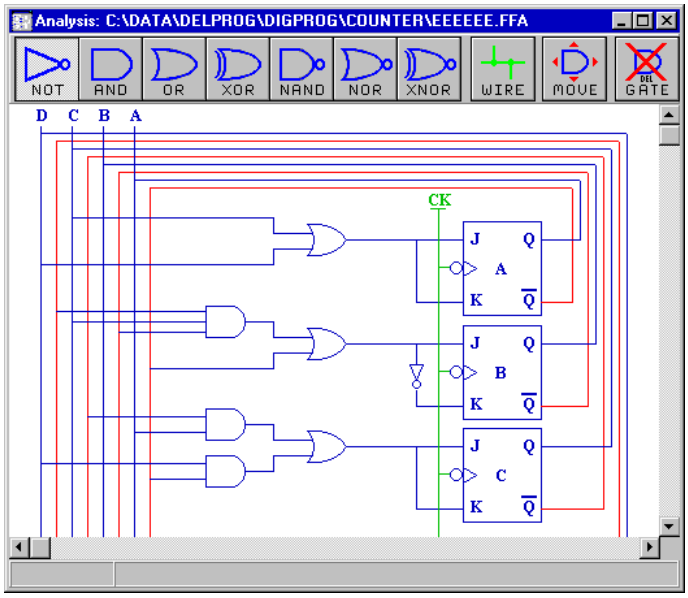


Figure 5: User input Counter circuit for Analysis

Once the circuit is input, the user can request the software to determine the Boolean logic of the Next State decoder function. An example of the software output of the Next State decoder Boolean functions are shown in Figure 6.

The software will then use the Next State Decoder Boolean functions, to determine the control functions that directly controls the Next State value from the Present State value. The software displays this output in the State Transition table, which is shown in Figure 7.

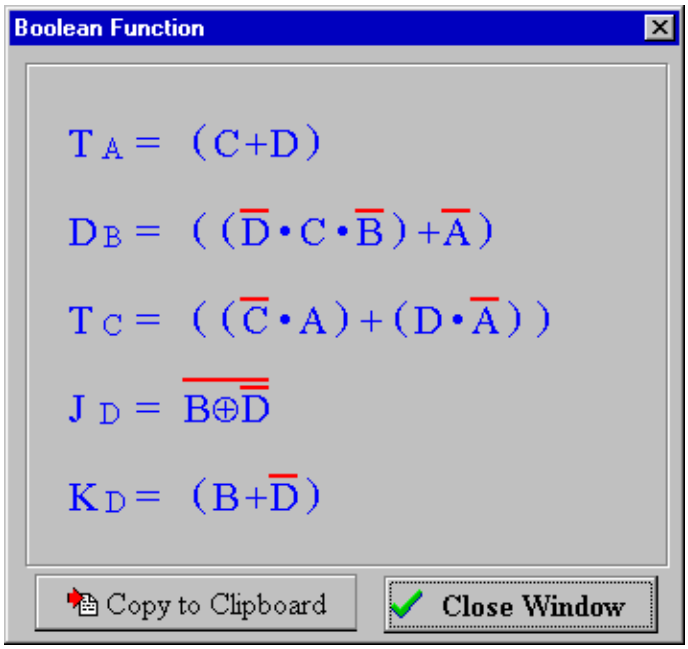


Figure 6: The Next State Decoder Boolean Functions

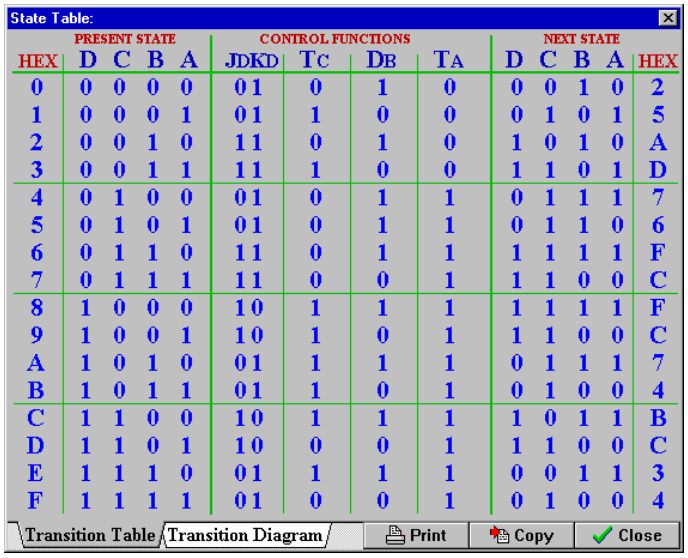


Figure 7: The State Transition Table of the input Circuit

To complete the analysis of the input counter circuit, the final count sequence is determined. The software extracts the Present State values, and the associated Next State values from the State Transition Table, and uses this information to derive the State Transition Diagram. This State Transition Diagram succinctly displays the count sequence that will be performed by the input counter circuit. An example State Transition Diagram output is shown in Figure 8.

PROGRAM EVALUATION

Investigations into the educational use of the software was undertaken by incorporating the software as part of the digital electronics curriculum in a second year subject. The software enabled the students to quickly obtain digital counter circuits for laboratory testing, and provided the answers to tutorial

problems by supplying the full design steps.

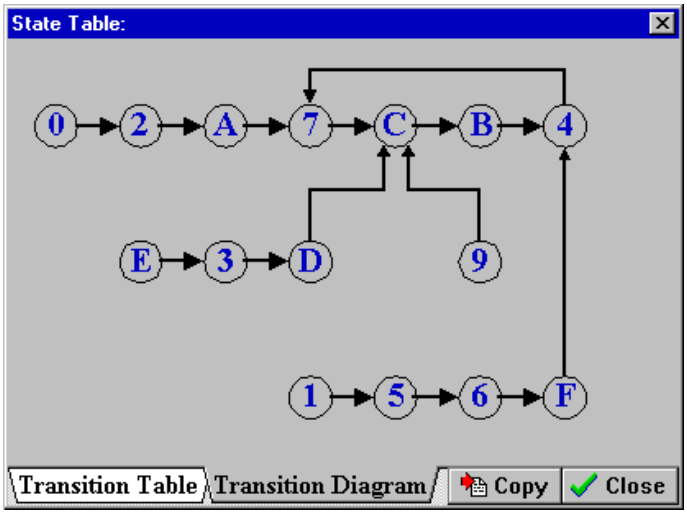


Figure 8: Final State Transition Diagram

Continuous feedback was given by the students throughout the subject, and a final evaluation questionnaire was administered. The evaluation revealed; that the students agreed the program was educationally useful for tutorial, laboratories, and examinations. The students were either neutral or in agreement that the software was easy to use, and user friendly.

The best features of the software were stated by the students as; the software allowed for easy design of digital counter circuits, it was a good study aid, and that results/outputs could be easily copied to other applications (such as word processing documents).

The students suggested improvements were to allow an option for the circuits to snap to a grid, to extend the current *copy-to-clipboard* menu option to allow for coping all displayed outputs and pasting into other files (such as student assignments), and to improve the general user friendliness of the program.

CONCLUSION

This paper outlined the development of a computer based tutorial for design and analysis of flip-flop digital counters. The software's graphical interface performs the design of a flip-flop counter, to achieve some user defined count/logic sequence. The software also allows the direct input of a flip-flop counter circuit, which is then fully analysed to determine the counters operation. The graphical screen of MS-Windows allows input and output of data that closely match the students initial understanding of digital counter circuits. The software has been utilised in a digital electronic subject lectured by the author, and student feedback has indicated the software was educationally useful and conveyed the desired information.

REFERENCES

1. Hacker, C. and Sitte, R., *A Computer Based Tutorial, for demonstrating the Solving of Digital Electronic Circuits*, Proceedings of the 10th Annual Australasian Association for Engineering Education (AaeE98), Waves of Change, September 1998, pp. 509-519, 1998
2. Hacker, C. and Sitte, R., *Development of a computer program, to electronically design Digital Logic Circuits using Boolean Algebra*, Proceedings of the 9th Annual Australasian Association for Engineering Education (AaeE97), December 1997, pp. 353-357, 1997
3. Hacker, C. and Sitte, R., *Implementing the 'Espresso - two level logic minimiser' algorithm in the MS-Windows environment*, 2nd Asia-Pacific Forum on Engineering & Technology Education, UNESCO International Conference in Engineering Education (UICEE99), July 1999, pp. 124-127, 1999