DEVELOPMENT AND PERFORMANCE EVALUATION OF A TERNARY FUNCTIONS MINIMIZATION TOOL, IMPLEMENTED USING MATLAB

Emilia SIPOS*, Lucian Ovidiu CICA**, Costin MIRON*, Laura IVANCIU*
**Technical University of Cluj-Napoca; Faculty of Electronics,Telecommunications and Information Technology*
G. Baritiu Street,No 26-28, 400027, Cluj-Napoca, Romania
emilia.sipos@bel.utcluj.ro
*** EBS Romania, Pavel Rosca Street,No.9, 400118, Cluj-Napoca, Romania*

Abstract: This paper presents a tool designed to minimize both completely and incompletely specified ternary functions with two, three and four input variables. The tool is tested using 33 ternary functions and performance such as minimization time and number of output terms is evaluated. One main advantage of ternary system over binary ones is the reduction of chip area required to implement logic functions. Using a minimization process for ternary functions, the required area is even smaller. For the ternary function minimization tool implemented using Matlab, the minimization process is entirely ternary and the ternary-to-binary and binary-to-ternary conversions used in well know minimization software for ternary functions, such as MVSIS and Espresso MV, are eliminated. For tested functions, the minimization time increases with one second, for every new input variable and the maximum minimization time was 2.9 seconds. The smallest number of output terms is obtained in 40% of the cases for one expression of minimized function, in 42% of the cases for the other expression and in 6% of the cases; the number of output terms is the same for both expressions.

Keywords: *minimization tool, ternary functions, performance evaluation, Matlab*.

I. INTRODUCTION

One fundamental problem of system-on-chip (SoC) design is the interconnection between different modules integrated on the chip [1]. The minimization of product terms obtained by logic minimization has a direct impact on the physical area required to implement the logic function [2].

A ternary system has several important advantages over a binary one, such as: reductions in the interconnections required to implement logic functions, thereby reducing chip area; more information can be transmitted over a given set of lines; less memory required for a given data length. Besides, the serial and some serial-parallel operations can be carried out at higher speed. Its advantages have been confirmed in applications like memories, communications, digital signal processing etc [3].

The first minimization software tools, SIS [4] and Espresso [2], were dedicated exclusively to binary functions. Along with the increasing interest about multivalued logic, software minimization tools for multivalued functions were developed, such as MVSIS (Multi-Valued SIS) [5], [6] and Espresso MV [7]. Both tools are developed based on the ones for binary functions (SIS, Espresso); ternary-to-binary and binary-to-ternary conversions are needed to minimize ternary functions and the minimization process remains binary.

A new minimization tool developed in Matlab which can be used to minimize completely specified ternary functions with two, three and four input variables is presented in [9]. The minimization process is entirely ternary; both conversions (ternary-to-binary and binary-to-ternary) are eliminated and the possible errors occurring during the successive conversion are eliminated as well. The minimized function, denoted *Z*, is provided in SOP (Sum Of Products) terms and is expressed in two different ways: first, using the expression well know in literature (combining the partial results Z_1 and Z_2) and second, using the partial results Z_0 and Z_1.

This paper shows the developments made over the minimization tool implemented using Matlab [8]. Their purpose was to extend the possibility to minimize both completely and incompletely specified ternary functions. The minimization tool is tested using 33 ternary test functions and the performance such as minimization time and number of the output terms are evaluated.

The paper is organized as follows: section 2 briefly describes the minimization tool implemented in Matlab for functions with completely specified input variables; section 3 presents the developments pointing out that the tool can be used to minimize incompletely specified ternary functions, by means of a relevant example - ternary function with four incompletely specified input variables. Section 4 focuses on the performance evaluation of the minimization tool from the point of view of the minimization time and the number of the output terms for both expressions of minimized function. The paper ends with section 5, which draws some conclusions and sets the main future work directions.

Manuscript received October 29, 2013; revised December 4, 2013

II. SOFTWARE MINIMIZATION TOOL FOR TERNARY FUNCTION WITH COMPLETELY SPECIFIED INPUT VARIABLES DEVELOPED IN MATLAB

In [8] a minimization tool for ternary functions with two, three and four completely specified input variables was presented. The block diagram for the minimization tool is given in Fig. 1.

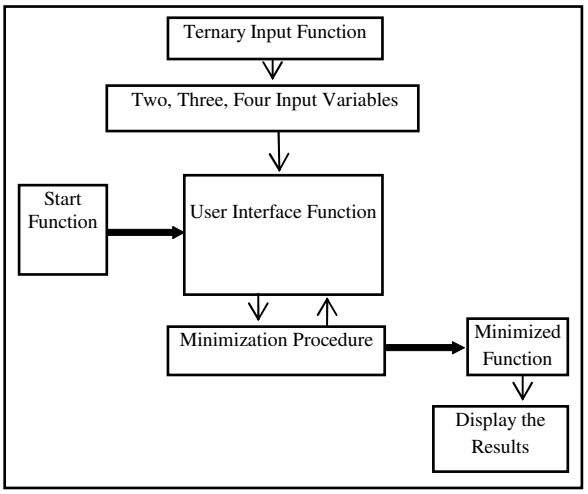

Fig.1 The block diagram for the minimization algorithm [8]

The following paragraphs describe the blocks of the diagram.

Start Function

The minimization algorithm starts by choosing the number of input variables.

Ternary Input Function

The input function is entered by the user (choosing the values "0","1" or "2" for every input variables and for the output variable), as an SOP form. The user only has to insert the input terms for either two values of the output variable; the remaining input terms (for the third value of the output variable) are automatically generated by the minimization tool.

User Interface Function

The user interface is implemented using the GUIDE editor. The interface is designed to display error messages whenever incompatible options with the input data are selected. One of the first settings of the minimization algorithm implemented in the user interface is to specify the number of input variables for the input ternary function.

Minimization Procedure

The minimization procedure is with respect to the reduction procedure for ternary functions proposed by Hurst [9]. For the minimization tool implemented using Matlab, the minimization process is achieved in parallel for all three values of the output variable.

Miminized Function

The implemented algorithm computes the minimized expressions for all the three values of the output variable, denoted partial results Z_0, Z_1, Z_2. All three partial results are displayed.

Display the Results

The minimized function is obtained by combining the expressions for two values of the output variable (two partial results). The algebraic expression of the minimized function depends on the particular partial results.

As a result, the software tool returns the minimized function using two different expressions. Both expressions are with respect to the truth table for the ternary input function. The first one is the well known expression [10] and is obtained with the partial results Z_1 and Z_2,

$$Z = 1 \cdot Z_1 + Z_2 \tag{1}$$

The second expression is obtained by using the partial results Z_0 and Z_1,

$$Z = STI(Z_0 + 1 \cdot Z_1) \tag{2}$$

The ternary operators used to obtain both expressions for Z are: (.) – minimum ternary function; (+) – maximum ternary function and (STI) – Simple Ternary Inverter.

III. MINIMIZATION OF INCOMPLETELY SPECIFIED TERNARY FUNCTIONS. EXAMPLE

III. I. Development

In order to improve the tool to minimize completely and incompletely specified ternary functions, *Start Function*, *Ternary Input Function* and *Minimization Procedure* blocks from are modified. The block diagram for the improved minimization tool is given in Fig. 2.

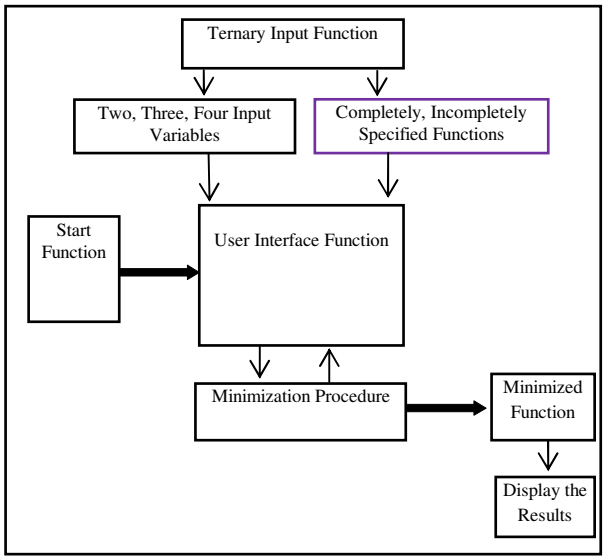

Fig. 2 The block diagram for the improved minimization algorithm

Start Function

First, the input interface is added new field corresponding to the type of the input variables. For the improved minimization tool, the minimization algorithm starts by choosing both, the number and the type of input variables (complete or incomplete variables).

Ternary Input Function

After the selection of number and type for input variables, the possible values for input variables become "0","1", "2" or "inactive"; the possible values for the output variable remains "0","1" or "2". Fig. 3 presents the code for generating the truth table with terms considered active if the value "inactive" is chosen, for functions with two incompletely specified input variables. The value "3" is considered in Matlab for the "inactive" state of the input variable.

```
function tab=genelemtab2var(truthtable2,tabv2)
n=size(tabv2);
 m=size(truthtable2);
 for i=1:n(1)

   if tabv2(i,1)<3 & tabv2(i,2)==3
     for j=1:m(1)
       if truthtable2(j,1)==tabv2(i,1)
         truthtable2(j,3)=tabv2(i,3);
       end
     end
    end

   if tabv2(i,1)==3 & tabv2(i,2)<3
     for j=1:m(1)
       if truthtable2(j,2)==tabv2(i,2)
         truthtable2(j,3)=tabv2(i,3);
       end
     end
    end

   if tabv2(i,1)<3 & tabv2(i,2)<3
     for j=1:m(1)
       if truthtable2(j,1)==tabv2(i,1)& truthtable2(j,2)==tabv2(i,2)
         truthtable2(j,3)=tabv2(i,3);
       end
     end
    end
  end
tab=truthtable2;
```

Fig. 3 Matlab function for activating the "inactive" terms

Minimization Procedure

The Matlab functions that form the *Minimization Procedure* block must consider a big number of extra terms in order to minimize the functions with incompletely specified input variables compared with the minimization of functions with completely specified input variables; the missing variable take all the three possible values (0, 1 and 2) [9].

For example, to minimize ternary functions with two incompletely specified variables, the maximum number of possible input terms is obtained using the next equation:

$$N_{2\max} = C(n,n-1)\cdot 3^{n-1} + C(n,n)\cdot 3^{n}$$
$$N_{2\max} = C(2,1)\cdot 3^{1} + C(2,2)\cdot 3^{2} = 15 \quad (3)$$

where
n is the number of input variables
$C(n,k)$ is the number of k-combinations from a given set of n elements.

The maximum number N_{max} is in this case 15, while the maximum number of input terms for completely specified functions is 9; the number of possible terms increases with 66% (6/9).

The maximum number of possible input terms for ternary functions with three incompletely specified variables is:

$$N_{3\max} = C(n,n-2)\cdot 3^{n-2} + C(n,n-1)\cdot 3^{n-1} + C(n,n)\cdot 3^{n}$$
$$N_{3\max} = 3\cdot 3^{1} + 3\cdot 3^{2} + 3^{3} = 63 \quad (4)$$

For these functions the maximum number of input terms increases with 133% (63 terms compared with 27).

Considering the functions with four incompletely specified variables the maximum number of possible input terms is:

$$N_{4\max} = C(n,n-3)\cdot 3^{n-3} + C(n,n-2)\cdot 3^{n-2} + C(n,n-1)\cdot 3^{n-1} + C(n,n)\cdot 3^{n}$$
$$N_{4\max} = 4\cdot 3^{1} + 6\cdot 3^{2} + 4\cdot 3^{3} + 3^{4} = 255 \quad (5)$$

The number of the input terms increases with 215% compared with the functions with four completely specified input variables, where the maximum number of input terms are 81.

III. II. Minimization example for a function with incompletely specified input variables

Due to the fact that the biggest number of input terms is obtained for functions with four input variables, a minimization example is presented next, for a randomly chosen four input ternary function.

The function is provided by the terms corresponding to the values "0" and "2" of the output variable (Z_0 and Z_2 terms). The terms for the third value of the output variable, Z_1, are not provided by the user, being automatically generated by the software.

The ternary input function is given by equation (6) and has 15 terms for the value "0" at the output, 14 terms for the value "2" at the output, and the rest of the terms for the value "1" at the output (226 terms):

$$Z_0 = A_2B_2C_2 + B_2C_0D_1 + B_2C_2D_1 + B_2D_1 + A_0 + B_0 + C_0 + C_1 + C_2 + D_0 + D_1 + A_0B_0C_0D_0 + A_1B_2C_2D_1 + A_2B_2C_2D_1 + A_2B_2C_2D_2$$

$$Z_2 = A_0B_1C_2 + A_0C_0D_1 + A_1C_0D_1 + B_0C_2D_1 + B_1C_0D_1 + B_1C_2D_1 + A_0C_2 + A_0D_0 + A_0D_1 + A_0D_2 + C_0D_1 + A_0B_1C_2D_2 + A_0B_2C_0D_1 + A_2B_1C_2D_2 \quad (6)$$

The terms corresponding to all three values "0", "1" and "2" of the ouptut variable are completely and incompletely specified.

The *Input* section displayed during the input terms entering process is presented in Fig. 4.

Fig. 4 Ternary function with four incompletely specified input variables – Input section

The minimization process starts along with pressing the "End Terms Entering" button. Two types of minimization results are presented: three partial results, Z_0, Z_1 and Z_2 - *Output* section, and two minimized functions *Z* - *Minimized Function Z* section in Fig. 5.

The minimization process minimize individually three different functions (corresponding to the three values of the output function - "0", "1" and "2"). Due to the fact that the input terms are completely and incompletely specified, the all three partial results can have the same minimized terms (Fig. 5). This is an odd situation; the minimization software doesn't make the difference between an incompletely specified input term and a minimized one; the difference must be made by the user.

To start another minimization process (for another input functions), the user only presses the "Back" button and the input interface with number and type of input variables selection is reloaded.

IV. PERFORMANCE

To measure the performance of the improved minimization tool, from the point of view of minimization time and number of output terms for both expressions of minimized function, 33 test functions provided by [11], [12] are used. The test functions are provided in two different logics– Galois (G) and Modulo (M).

We evaluated the improved minimization tool performance using a common computer configuration, a notebook with dual-core processor at 2.3GHz and 3GB RAM. The minimization time measurement starts when the "End Terms Entering" (END) button is pushed, and ended when the minimized function is displayed. Table 1 presents the minimization time (in seconds) obtained for all test functions.

The minimization tool displays at the same time both expressions for minimized function *Z*. The minimized function *Z* expression obtained using equation (1) is denoted *E1* and the expression obtained with equation (2) is denoted *E2* – see Table 1. The number of output terms for both

Fig. 5 Minimization results – Output and Minimized Function Z sections

expressions are also available in Table 1.

The minimization time is between 0.3 sec - 0.5 sec for two inputs functions, between 1.5 sec – 1.9 sec for three inputs functions and between 2.3 sec - 2.9 sec for four input ternary functions. The minimization time increases with 1 second for every new input variable.

For 14 functions, the number of output terms obtained after simulations is smaller if the expressions *E2* is used, for 13 functions the number is smaller if the expressions *E1* is used and in 6 cases the number of output terms is the same for both expressions.

Table 1 Performance of developed minimization tool

Ternary test functions			Minimization time [sec]	Number of output terms	
Number	Name	Expressions		E1 Z=1.Z1 +Z2	E2 Z=STI(Z0+1.Z1)
1	2cyG2	ab – G	0.494	4	4
2	2cyM2	ab - M	0.471	4	5
3	3cyG2	ab+bc+ca - G	1.53	12	11
4	3cyG3	abc - G	1.425	8	9
5	3cyM2	ab+bc+ca - M	1.88	12	12
6	3cyM3	abc - M	1.732	8	12
7	4cyG3	abc+abd+acd+bcd - G	2.803	32	30
8	4cyG4	abcd - G	2.375	16	12
9	4cyM2	ab+ac+ad+bc+bd+cd - M	2.935	36	32
10	4cyM3	abc+abd+acd+bcd - M	2.505	32	36
11	4cyM4	abcd - M	2.495	16	19
12	a2bccG	aa+bc+c - G	1.894	12	12
13	a2bccM	aa+bc+c - M	1.711	5	4
14	avgG2	int [(a+b)/2] - G	0.434	6	8
15	avgG3	int [(a+b+c)/3] - G	1.982	11	16
16	avgG4	int [(a+b+c+d)/4] - G	2.307	38	57
17	prodG2	ab -G	0.494	4	4
18	prodG3	abc - G	1.425	8	9
19	prodG4	abcd - G	2.375	16	12
20	prodMin2	ab - M	0.471	4	5
21	prodMin3	abc - M	1.723	8	12
22	prodMin4	abcd -M	2.495	16	19
23	sqsumG2	aa+bb - G	0.486	8	5
24	sqsumG3	aa+bb+cc - G	1.592	18	15
25	sqsumG4	aa+bb+cc+dd - G	2.43	48	57
26	sqsumM2	aa+bb -M	0.484	5	4
27	sqsumM3	aa+bb+cc - M	1.811	12	8
28	sqsumM4	aa+bb+cc+dd - M	2.817	19	16
29	sumG2	a+b - G	0.364	6	6
30	sumG3	a+b+c -G	1.587	18	18
31	sumMax2	a+b - M	0.484	5	4
32	sumMax3	a+b+c - M	1.811	12	8
33	sumMax4	a+b+c+d - M	2.817	19	16

V. CONCLUSIONS

The development of the minimization tool, in order to minimize both completely and incompletely specified ternary functions, was presented in the paper. First, the input interface was modified, another possible "value" for input variable denoted "inactive" was introduced and the truth tables for all ternary functions were expanded.

To minimize functions with incompletely specified input variables, the number of possible terms for input functions with four variable increases with 215% relative to the function with four completely specified input variables (255 terms vs. with 81 terms).

The improved minimization tool was tested using 33 test functions. The parameters under evaluation are the minimization time and the number of output terms, for both expressions of minimized function *Z*.

Analyzing the minimization time, the minimization time increases with 1 second, for every new input variable, from ~0.4 sec for ternary functions with two input variables to ~2.5 sec for ternary functions with four input variables. The type of input variables (completely or incompletely specified) has no impact over the minimization time for functions with the same number of input variables.

Comparing the two expressions used to display minimized function *Z* from the point of view of output terms number, in 40% of the cases, the smallest number of output terms is obtained if the expression from literature is used (*E1*), in 42% of the cases, the expression *Z=STI(Z0+1.Z1)* generates the smallest number of output terms, and in 6% of the cases, the number of output terms is the same for both expressions.

The results obtained using a computer with regular processing performance (processor speed, memory) proves that the minimization tool can be used even by users with no access to powerful minimization workstations.

The performance obtained until now encourage us to implement a general minimization tool, for ternary functions with a large number of input and output variables, so that the tool can also be used to minimize ternary functions corresponding to combinational and sequential circuits.

REFERENCES

[1] Winter, M., & Fettweis, G. (2006). Interconnection Generation for System-on-Chip Design. International Symposium on System-on-Chip. 1-4. doi: 10.1109/ISSOC.2006.321975.

[2] Brayton, R. K., Hachtel, G. D., McMullen, C. T., Sangiovanni-Vincentelli, A. L. (1984). *Logic Minimization Algorithms for VLSI Synthesis*. Kluwer Academic Publishers.

[3] Dhande, A.P., & Ingole, V.T. (2005). Design Of 3-Valued R-S & D Flip – Flops Based on Simple Ternary Gates. *Transactions on Engineering, Computing and Technology*, V4.

[4] Sentovich, E. M., et al. (1992). SIS: A System for Sequential Circuit Synthesis. Technical Report of the UC Berkeley Electronics Research Lab.

[5] M. Gao, J.-H. Jiang, Y. Jiang, Y. Li, S. Sinha, and R. K. Brayton, "MVSIS", Int. Workshop Logic Synthesis, 2001.

[6] MVSIS, http://www- cad.eecs.berkeley.edu/mvsis.

[7] Rudell, R. L. (1986). Multiple-valued logic minimization for PLA synthesis. Technical Report M86/65, UCB.
[8] Sipos, Emilia, Cica, L., & Miron, C. (2012). Software minimization of ternary functions using Matlab. IEEE International Conference on Automation Quality and Testing Robotics (AQTR), 284 – 289. doi: 10.1109/AQTR.2012.6237718.
[9] Hurst, S. L. (1968). An extension of binary minimisation techniques to ternary equations. *The Computer Journal*, 11 (3), 277-286.
[10] Herrmann, R. L. (1968). Selection and implementation of a ternary switching algebra. Proceedings of AFIPS Joint Computer Conference,.Spring Joint Computer Conference, 283-290.
[11] Khan, M. H. A., Perkowski, M. A., Khan, M. R., & Kerntopf, P. (2005). Ternary GFSOP Minimization using Kronecker Decision Diagrams and Their Synthesis with Quantum Cascades. *Journal of Multiple-Valued Logic and Soft Computing*, 11 (5-6), 567-602.
[12] Denler, N., Yen, B., Perkowski, M., & Kerntopf, P. (2004). Synthesis of Reversible Circuits from a Subset of Muthukrishnan-Stroud Quantum Realizable Multi-Valued Gates. Proceedings of IWLS 2004, 321-328.