FPGA based Optimisation and Implementation of Nondestructive Identification Procedures

Stefan Uting[*,**], **Michael Brutscheck**[*,**], **Steffen Becker**[**], **Andreas Thomas Schwarzbacher**[**]

[*] *Department of Electrical, Mechanical and Industrial Engineering, Anhalt University of Applied Sciences, Bernburger Str. 57, Köthen, Germany, Email: m.brutscheck@emw.hs-anhalt.de*

[**] *School of Electronic and Communications Engineering, Dublin Institute of Technology, Kevin Street, Dublin 8, Ireland, Email: andreas.schwarzbacher@dit.ie*

Abstract — **The correct and fast identification of unknown ICs is an important issue for current as well as future IC technology. The authors have previously presented an algorithm which has demonstrated for the first time the correct non-invasive identification of unknown ICs [1]. Here, the analysis procedure was implemented into an analysis environment using MATLAB. However, the handling and the overall calculation time was not practical. Moreover, ICs to be analysed become more and more complex and therefore, large amounts of data have to be stored to complete an analysis.**
This paper describes a single Field Programmable Gate Array (FPGA) optimised implementation of the analysis algorithm to overcome these problems. To show the improvement the traditional analysis procedure was implemented into a Cyclone-II ALTERA FPGA using Quartus-II [2] and a Nios II soft-core microprocessor [3]. To demonstrate the correct operation the algorithms were implemented, fully tested and validated on ISCAS-85, ISCAS-89 and ISCAS-99 IEEE benchmark models of real ICs. The results are presented in this paper. It will be shown that the overall calculation time can be improved by a factor of 25000 compared to the traditional procedure.

Keywords — **Sequential Finite State Machines, Digital CMOS ICs, FPGA, Nios II, Quartus, MATLAB.**

I Introduction

The identification procedures presented in [1] showed that it is possible to identify several unknown ICs. The algorithm can separate the type of automaton and automatically determine the number of states. Thereby, the identification procedure works dependable. However, traditional analysis approaches have several disadvantages. The first disadvantage is the long calculation time. This can be seen by an increasing complexity of automata. Another disadvantage besides the calculation time is the handling between MATLAB and Quartus. To set and get information a Universal Asynchronous Receiver Transmitter (UART) is used which requires most of this calculation time. Therefore, a novel system without an UART was built. Thereby, the identification procedure is adaptable for several systems. This solution makes the identification procedures comfortable and fast which is an important advantage in contrast to the traditional destructive [4] and the non-destructive [5] analysis strategies.

This paper will discuss the particular problem of the identification of unknown nonlinear CMOS ICs as represented by the implementation of the algorithm. To demonstrate the improvement the traditional analysis procedure was implemented into a Cyclone-II ALTERA FPGA by using Quartus-II [2] and a Nios II soft-core microprocessor [3]. An important issue when identifying complex unknown nonlinear CMOS ICs is to reduce the calculation time. The handling and the overall calculation time will show that this version is more efficient and faster.

In Figure 1 the traditional procedure can be seen. The procedure was divided into three parts. First, the pin types of the device under test are determined, which was described in detail by the authors in [6]. The next part is the separation into Mealy or Moore. Finally, the identification of unknown IC starts and is finished by the identification procedures.

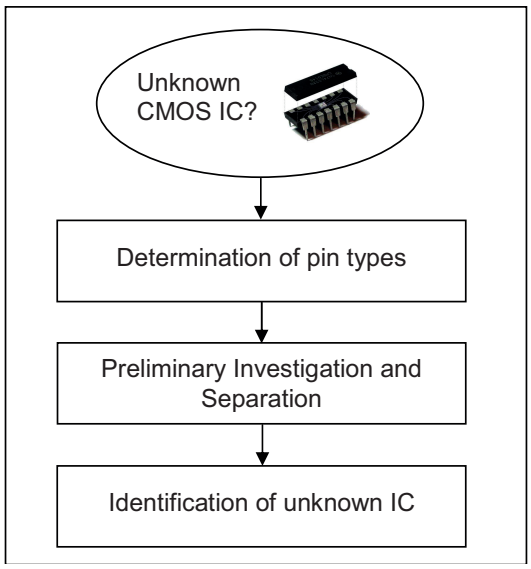

Fig. 1: The Analysis Flow

In Section III the optimisation of the traditional procedure will be explained. The novel procedure contains a fast and efficient algorithm that was implemented into a FPGA using a soft-core microprocessor. Afterwards, the results of the optimised identification procedure implemented into a FPGA are presented in Section IV.

II The traditional Procedure

The traditional procedure in [1] demonstrated the correct non-invasive identification of unknown ICs for the first time. Here, the analysis procedure was implemented into an analysis environment using MATLAB. The analysis was divided into two parts, the user interface (UI) and the development board as shown in Figure 2.

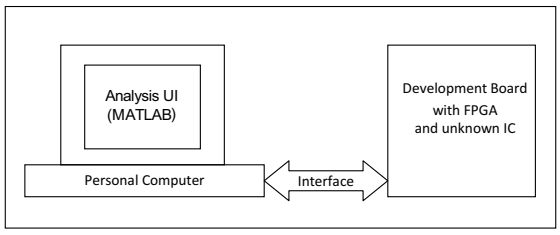

Fig. 2: Traditional Structure of Analysis Environment

As can be seen in Figure 2 the traditional analysis environment consists of a personal computer and a FPGA. The user interface was realised using MATLAB and is used for a priori determination of information which are needed for the analysis process. This information will be specified at the beginning of the analysis process. Then, this information is transmitted to the FPGA. Here, the models of unknown ICs were implemented to validate the analysis algorithm.

The identification procedure can be divided into three major parts. The first part describes the separation into Moore and Mealy. After the type of automaton is determined the next major part is the preparation algorithm. The traditional algorithm can be divided into several process steps. It consists of several blocks and works similar for Moore and Mealy automata which is schematically shown in Figure 3. First, the identification procedure must find one initial state of the automaton. Then, the identification algorithm is carried out which is the next major part of the analysis procedure for nonlinear finite state machines (FSMs).

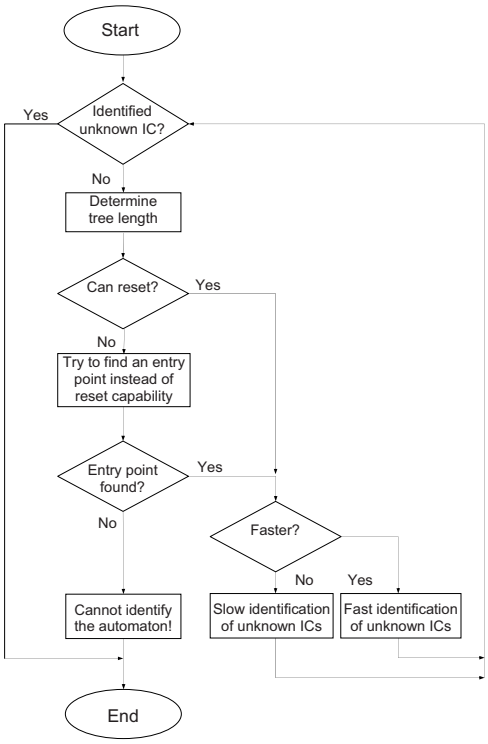

Fig. 3: Traditional Structure of Analysis

The traditional procedure is an efficient algorithm to identify unknown ICs. However, the handling and the overall calculation time is not realistic for practical use. Moreover, as the ICs to be analysed become more and more complex increasingly large amounts of data have to be stored in the memory of the running system. Therefore, the traditional procedure must be even more efficient and faster.

III The optimised identification procedure

In this section the further optimisation of the nonlinear analysis procedure will be explained as shown in Figure 4.

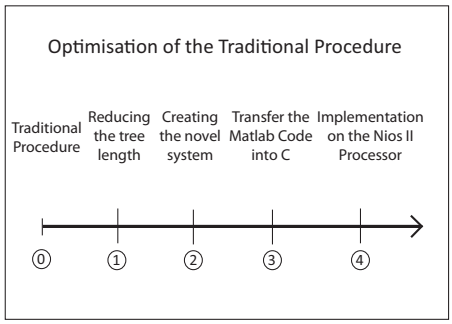

Fig. 4: The Optimisation Flow

First, the reduction of the tree length will be explained. Then, the implementation of a Nios II soft-core microprocessor [3] into a Cyclone-II ALTERA FPGA using Quartus-II [2] will be explained. Finally, it is necessary to transfer the traditional procedure from MATLAB into C. This makes it possible to fully run the algorithm on the soft-core microprocessor.

With an increasing complexity of the automata to be investigated the calculation time of the algorithm also increases. To reduce the analysis time the tree length must be minimised. This was achieved by changing the examination of the results at the end of each investigation. By using the investigation procedure the actual tree length is always computed using the number of distinguishable states plus two. Furthermore, the change of the number of states is the only variable in this computation. The alternative procedure now includes the number of states found as well as the previous tree length as shown in Equation (1). Here, the new tree length can be determined by the number of current states (NoS), the number of found states (NoFS) and the tree length of the last investigation cycle as follows.

$$length_{tree}(t) = NoS(t) - NoFS(t-1) + length_{tree}(t-1) \quad (1)$$

The newly investigated number of states is decreased by the number of found states of the last investigation cycle. Moreover, the previous tree length is added to the result. However, after the last investigation cycle more states could be distinguished in contrast to only using a fixed number of states. The determination of the number of distinguishable states requires only a tree length of two. For a possible modified discrimination the value two is not sufficient. It can be seen that this method reduces the tree length significantly as well as the calculation time. Here, for the correct determination the old value has to be replaced by a tree length which is equal to the new minimum investigation depth.

The handling and the overall calculation time of the traditional analysis is not realistic for practical use. Therefore, a novel functional construct was realised. The traditional construct contained the IC which was realised on a Cyclone-II ALTERA FPGA. Here the algorithm was implemented in MATLAB where the unknown IC was connected to MATLAB/SIMULINK by a RS–232 interface. The analysis was divided into two parts, the user interface (UI) and the development board as shown in Figure 2.

However, this interface is to slow for complex ICs. The resources and performance the of Cyclone-II ALTERA FPGA makes it possible to implement the algorithm and the ISCAS–85, ISCAS–89 and ISCAS–99 benchmark models of real ICs on it. The resources of the whole system take under 10 percent of the Cyclone-II ALTERA FPGA. This shows how compact the whole system is. The novel construction was realised on a Cyclone–II ALTERA FPGA which contains the IC as well as the algorithm. The algorithm was implemented into the Nios II soft-core microprocessor.

The Nios II architecture describes an instruction set architecture (ISA). Additionally, the processor does not include peripherals or connected logic outside the FPGA. It includes only the circuits required to implement the Nios II architecture which is shown in Figure 5.

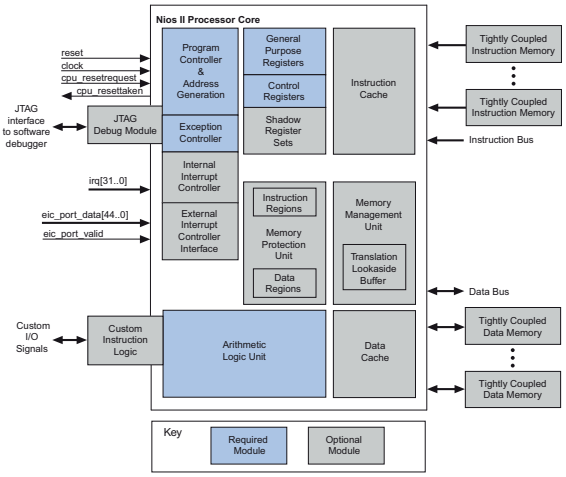

Fig. 5: Nios II Processor Architecture [3]

The Nios II processor is a general-purpose RISC processor core and provides a full 32-bit instruction set, data path and address space, 32 general-purpose registers and optional shadow register

sets. With 32 interrupt sources and an external interrupt controller interface for more interrupt sources interrupt service routines (ISR) can be handled. The single-instruction 32 × 32 multiply and divide producing a 32-bit result, dedicated instructions for computing 64-bit and 128-bit products of multiplication and some more that can be stand on the same level as conventional ASICs. The software development environment is based on the GNU C/C++ tool chain and on the Nios II Software Build Tools (SBT) for Eclipse. The processor performance is up to 250 DMIPS which makes it possible to implement complex algorithm [3].

To test and validate the algorithm the traditional analysis procedure was implemented into a Cyclone-II ALTERA FPGA using Quartus-II [2]. Here, the novel hardware solution has several advantages when handling data. The first advantage is that this hardware solution consists of one standalone system which is only realised in Quartus. This makes the implementation flexible and compact. The second advantage is that the algorithm which was firstly realised in MATLAB/SIMULINK by a RS-232 interface is directly connected to the hardware. This resulted in a significant decrease in calculation time. Therefore, the slow serial interface which is used in the traditional solution is no longer needed in this implementation. This reduced the time to write and read information to and from the IC.

The significant advantage is that the system is compatible with all FPGAs that can be used by Quartus. To implement the traditional procedure, a Nios II soft-core microprocessor was implemented into a Cyclone-II FPGA. Figure 6 shows the whole hardware solution that was implemented on the FPGA.

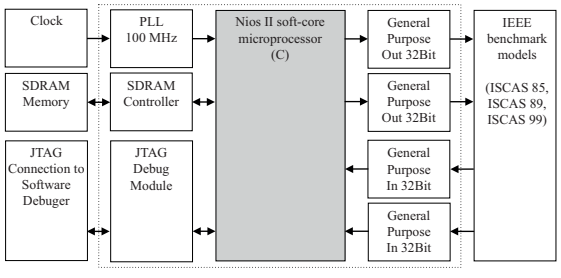

Fig. 6: Novel Structure of Single FPGA Optimised Analysis Nios II Environment

Figure 6 shows the PLL that is used to provides the clock frequency for the Nios II processor. With a clock of 50 MHz the PLL provides a frequency of 100 MHz for the processor. It is the highest frequency for the Nios II to run correctly. To program the microprocessor the IDE Nios II Software is used. The JTAG Debug Module makes it possible to transfer information between the IDE and the Nios II processor and to run the processor in a debug modus. The SDRAM-Controller realises the connection between the SDRAM and processor for the programme storage. It is also possible to use the on chip memory of the FPGA if the program size is small. Its size depends on the used FPGA. Here, the Cyclone-II ALTERA FPGA has 250 M4K (4,096 memory bits per block) RAM blocks. For the connection between the algorithm and the tested and validated hardware the GPIO (General Purpose Input/Output) are connected to the Nios II processor. All components used here are implemented on the FPGA. This makes it efficiently and compact, especially when the settings or components will be changed.

Before the Nios II Processor was programmed, the algorithm had to be transferred from MATLAB into C. After that, the novel C procedure was used in the IDE Nios II software to program the Nios II processor on the FPGA. To transfer the traditional procedure from the MATLAB code into the novel code C procedure some points have to be observed. Between this two high-level languages there are no significantly difference. Differences are the handling of matrices, global variables and the using of functions. Additionally, the structure of both codes are nearly identical which makes the transfer manageable.

One way is to convert all MATLAB functions which often consists of a separate .mat-file into a function in C. This is important to get a clearly structure and manageable code. After that, the functions can be used and the code can be handled in the same way as the traditional procedure. To realise global variables get and set functions are used. In MATLAB the dynamic matrices can be realised using a standard array. In C it is not possible to directly realise dynamic matrices. Therefore, it is important to include set and get function with a variable to obtain the size of the array used in the procedure. This enables the implementation to handle the size of the array. Firstly, the array must be initialised with an optional size. This size can be compared with the temporary size of the array. If that size is reached the array must be copied into a larger array. Therefore, it does not change the temporary size. Additionally, most of the MATLAB code can be transferred directly be into C and can gradually be used to run on the Nios II processor. This makes it possible to quickly change a large MATLAB code into C code. In addition to the traditional algorithm Equation (1) was implemented which makes the identification procedure more efficient.

To demonstrate the correct operation the implemented algorithms were fully tested and validated on the ISCAS-85, ISCAS-89 and ISCAS-99

benchmark models of real ICs. This novel construction is highly flexible and has a directly connection between the IC under investigation and the algorithm. Additionally, it is possible to include and remove components or change parameters very fast. The important advantage is that the overall calculation time was dramatically improved. Furthermore, the reduced tree length which can be also optimised the calculation time of the algorithm. This will be shown in the next section were the results are presented for the implemented traditional and novel procedure on the Nios II processor.

IV Results

In this section the results of the nonlinear identification procedure as well of its optimisation will be discussed. Especially, the theory of the optimised identification procedure in cases of reduced tree length and the novel procedure in C on the Nios II soft-core microprocessor will be described. The optimised identification procedure presented in this paper was verified using both simulation and real hardware tests. The IC models were analysed having unknown as well as known number of internal states using optimised identification procedure. For clarity the following tables only show the results of the simulation and the hardware analysis of the nonlinear identification procedure where the number of internal states of the IC models to be investigated is unknown. As can be seen in Table 1 the results of the traditional and the novel procedure for the hardware analysis. It shows distinctly a significant improvement.

IC Name	Type of Finite State Machine	Number of States	Overall Evaluation Time (Traditional)	Overall Evaluation Time (Novel)	Improvement Factor
EC1	Mealy	1	7713,6s = 2,14h	1,0s	7714
ELS1	Mealy	8	62110,0s = 17,25h	8,9s	6979
ENLS1	Mealy	8	62127,0s = 17,26h	8,0s	7766
S27	Mealy	5	615580,0s = 7,12d	24,6s	25024
B06	Moore	13	11456s = 3,18h	1,4s	8183

Tab. 1: Traditional and Novel Results of Analysing

The comparison of the traditional and the new time of the nonlinear identification procedure shows an dramatically improved analysis time by a factor ranges from 7000 to 25000. The traditional analysis approaches have several disadvantages. One of them is the long calculation time for complex ICs. The second disadvantage is the handling between the algorithm and the hardware. Therefore, the data transfer is not practical enough to manage the analysis of complex ICs. In comparison to the hardware analysis the calculation time of the simulation is significantly faster.

Additionally to the standard identification procedure the optimised procedure using the novel procedure in C on the Nios II soft-core microprocessor was fully implemented and tested. All logic simulation and hardware results of the standard algorithm were fully confirmed. Furthermore, a significant reduction of the calculation time was achieved. Figure 7 shows the results of the standard identification procedure of the simulation compared to the novel procedure in C. The standard algorithm is highlighted in black, while the gray graph shows the calculation time of the novel simulation procedure in C.

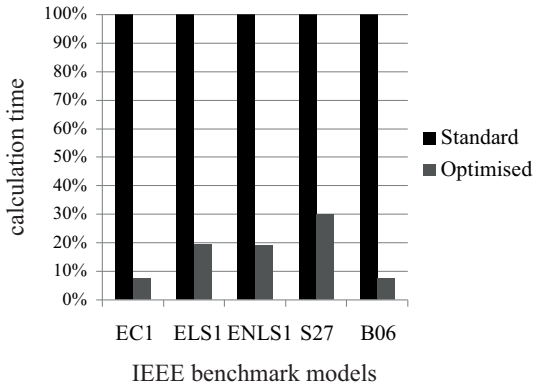

Fig. 7: Improvement of the Novel Simulation Procedure

Using the novel hardware procedure the results are shown in Figure 8. The standard algorithm is also highlighted in black, while the gray graph shows the calculation time after novel hardware procedure in C. Because of the dramatically improvement Figure 8 shows only a particular part of the standard algorithm.

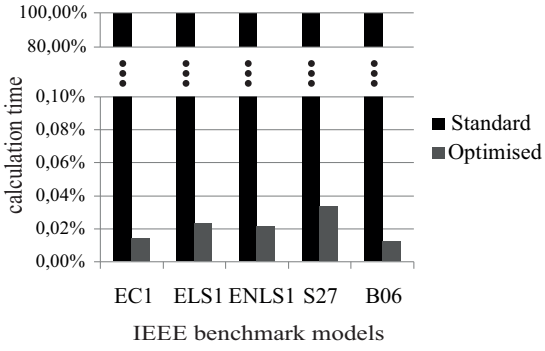

Fig. 8: Improvement of the Novel Hardware Procedure

Using the novel optimised version with a reduced tree length it is possible to reduce the calculation time by over 80 percent when compared to traditional approaches. Furthermore, the results contain the correct automata as well as the right number of states found. Moreover, the state transition table and the output function were successfully found in this significantly reduced time.

Because of this the calculation time of the simulations and hardware procedure is nearly the same. Therefore, Figure 9 shows the results of the novel identification procedure of the hardware compared to the novel identification procedure with the reduced tree length. The novel identification procedure is highlighted in black, while the gray graph shows the calculation time after reducing the tree length.

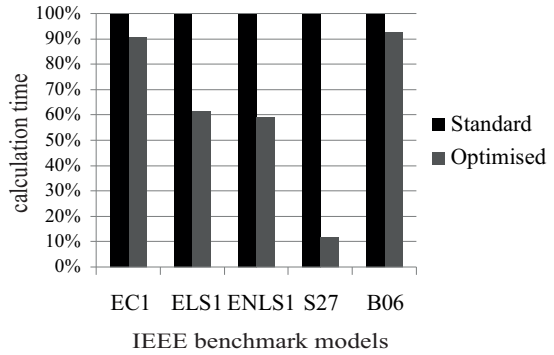

Fig. 9: Improvement of Reduced Tree Length

In each case the evaluation time is significantly reduced. Reasons for these results are the compact implementation and the optimised identification procedure. In summary, it can be stated that the evaluation time is reduced by a factor over 5000 compared to the traditional hardware procedure.

V Conclusion

This paper has presented a FPGA based implementation of a novel optimised identification procedure to fully determine nonlinear ICs which possesses several improvements compared to the traditional procedure. The handling and the overall calculation time of the traditional procedure is not realistic for practical use. Moreover, the ICs to be analysed become more and more complex and therefore, large amounts of data have to be stored for the analysis. This novel procedure is able to identify even very complex unknown ICs without any prior knowledge. To realise the novel procedure the construction is implemented on a FPGA of a Nios II soft-core microprocessor. This makes the novel procedure compact and efficiently for change in performance and components. The algorithm is running on a C running system which can be implemented in several systems. These are important advantages to the traditional identification methods which are either destructive or is very expensive.

In summary, a novel optimised identification algorithm was developed, which consists of three main parts. The reducing of the tree length is the first part and is based on the traditional procedure. The implementation in a Nios II soft-core microprocessor is the second part. Here the traditional algorithm was transferred from MATLAB to C. The reduced tree length is included in the novel identification algorithm. The optimised identification algorithm is implemented on a FPGA which further reduces the overall complexity of the proposed procedure.

This paper was described the FPGA based optimisation and implementation of non-destructive identification procedures. All analysis steps described were implemented into the Nios II soft-core microprocessor. The correct operation was verified through the implementation of several IEEE benchmark ICs as well as user defined IC models. Moreover, the traditional identification procedure was optimised using the novel identification algorithm by using a reduced tree length in combination with a Nios II soft-core microprocessor to identify the unknown IC. The optimisation shows that the calculation time is reduced while the outcome of the analysis remained the same. Therefore, in conclusion this paper has presented a novel FPGA based implementation of a optimised fast and non-invasive reverse engineering procedure for structured analysis of unknown nonlinear ICs.

References

[1] Brutscheck, M., Schmidt, B., Franke, M., Schwarzbacher, A. Th., Becker, St.: "Optimisation and implementation of a nonlinear identification procedure for unknown ICs". *Irish Signals and Systems Conference*, Cork, Ireland, June 2010.

[2] TERASIC Corporation [Online]. Available: http://www.terasic.com.tw/cgi-bin/page/archive.pl?Language=English&No=226 [Accessed: Jan. 5, 2011].

[3] ALTERA Corporation [Online]. Available: http://www.altera.com/literature/hb/nios2/n2cpu_nii5v1.pdf [Accessed: Feb. 1, 2011].

[4] Jarzabek, St., Keam, T. P.: "Design of a generic reverse engineering assistant tool". *Proceedings of 2nd Working Conference on Reverse Engineering*, Toronto, Canada, July 14-16, 1995, pp. 61-70.

[5] Lee, D., Yannakakis, M.: "Principles and methods of testing finite state machines - a survey". *Proceedings of the IEEE*, vol. 84, no. 8, August 1996, pp. 1090-1123.

[6] Brutscheck, M., Franke, M., Schwarzbacher, A. Th., Becker, St.: "Determination of pin types and minimisation of test vectors in unknown CMOS integrated circuits". *13th EDS IMAPS CS Conference*, Brno, Czech Republic, September 2006, pp. 64-69.