Theoretical and Applied Informatics
ISSN 1896-5334
Vol. 26 (2014), no. 1, 2
pp. 25 - 37

A Modern Approach to the Asynchronous Sequential Circuit Synthesis

Piotr CZEKALSKI[a], Krzysztof TOKARZ[a], Bolesław POCHOPIEŃ[a]

[a] Institute of Informatics
Silesian University of Technology
ul. Akademicka 16, Gliwice, Poland
piotr.czekalski@polsl.pl

Abstract: This paper presents novel approach to the Huffman's asynchronous sequential circuit two valued Boolean switching system design. The algorithm is implemented as software using distributed, service oriented application model with means of the web service component design. It considers method implementation challenges, both towards Moore and Mealy structures with particular respect to the estimation of the Huffman's minimization algorithm computational complexity. The paper provides implementation details, theoretical model estimation and experimental results that acknowledge the theoretical approach in practice. This paper also examine the multistep design process implementation and its problems inherent in web service based environment both for development and educational purposes.

Keywords: logic circuits, automatic control, web service

1. Introduction

The sequential, switching circuit design is still one of the leading methods of the finite state automata design. In case of the complex digital systems it is supplanted by the embedded systems, microcomputers, personal computers and accompaniment software, but in critical industrial processes, when reliability and simplicity is required, classical, wired logic circuits are developed, still playing a key role as control, monitoring and managing devices.

The switching circuit design is composed of the two classes of the devices: combinational circuits and sequential circuits. The class of the sequential circuits contains various classes, including the most general one: asynchronous circuits [10][11][15].

While combinational circuit design is covered mostly by the Boolean function minimization and related techniques (i.e. Karnaugh maps [1][5], Kazakov [4][5][10], Quine-McCluskey [6][7][8], Espresso[9][14]), asynchronous, bounded-delay sequential circuit design is covered by the two low-level, leading algorithms: Switching Sequence Table method [3][10][11] and Huffman's method [2][10][11].

The Huffman's method is a multi-step process of delivering the complete asynchronous circuit description, including so called Memory Block (MB - a state machine) switching equations and Output Combinational Blocks (OCB). Since the asynchronous sequential circuit principals were presented by D. A. Huffman in 1954, the development and education of this method was provided using pen and paper mostly, while in 90's with dedicated, desktop-based PC software. By the Huffman's Circuits there exist others i.e. Hollaar Circuits (breaking a single input change assumption) [16], Burst-Mode Circuits [15] [16], given by subclass definition based on restrictions of input-output signals [15] by Unger [18] and higher level design concepts i.e. those specified through the use of the Communication Channel Model [17]. Latest research on asynchronous circuits discussed in the annual IEEE International Symposium on Asynchronous Circuits and Systems show in the modern optimization techniques on existing methods [19][20][21] as well as high speed and low power consumption implementations of the modern controllers [22][23]. Concluding current research, modern asynchronous circuit design using computer-aided techniques are essential when high speed, low power consumption and low radio noise controllers are required.

The rapid development of the Internet and World Wide Web for the last 10 years leads to the conclusion that web-based implementation of the methods, where everyone can access the algorithm as a service using unified, XML based SOAP protocol [12] is reasonable. The Huffman's method is one of the various components constituting ZMITACSIM – a digital circuit design software for research and education purposed, developed at the Silesian University of Technology. The web based approach provides flexibility for the implementation, but currently at the early development stage, all components including Huffman's method are implemented using Microsoft technologies, particularly .NET Framework and C# language.

2. Huffman's method – the model

The Huffman's method covers preparation of the optimal switching circuit that contains memory, implemented as bounded-delay state signal loopbacks and the appropriate Boolean functions that constitute the translation between machine Internal State Vector (ISV) and Output Vector (OV). The Huffman methods delivers two type of the structure to implement: Moore machine and Mealy machine. In any case the machine is composed of two blocks. Also in any case the MB is driven by the Input Vector (IV) data change (internally also by the loopbacks of the ISV signal bits). The OCB connection schema varies between machines. The Moore's machine structure (Fig. 1) requires change on the internal state to enforce output change. In case of the Mealy's machine, the OCB is driven by both the ISV and IV (Fig. 2). The Mealy's Machine features capability that the output may experience changes that are not eligible to the state machine change (i.e. gating of the signals by external trigger). Output vector may be changed directly from the input, via the OCB module without the necessity of the ISV change. This may provide more

compact design of the MB, while increasing complexity of the OCB. It also has impact on the process of the Huffman's method design, as presented later in this chapter.

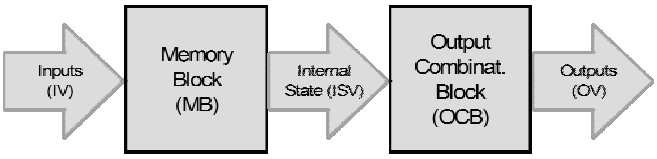

Fig. 1. Moore's machine data flow model

In any case, the current state machine is presented by the ISV of the binary values that uniquely identify the context of the operation the automata is performing at the moment. The machine is in the Stable State (a particular value of the ISV) when no change on the IV occurs and all the remaining bounded-delay operations have been already finished. The Unstable State appears for the short moment of time (a Transition) when machine switches between its Stable States.

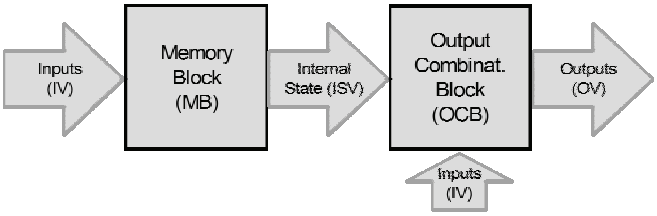

Fig. 2. Mealy's machine data flow model

The Huffman's method rely on the flow tables where relation between current and following Stable States through Unstable State Transition is juxtaposed. It is worth to mention that Huffman Circuits are coupled with single-input-change in opposition to i.e. Hollaar Circuits [16]. The modern Huffman's method circuit design is composed of the following steps [2][10][11]:

- Preparation of the Moore machine Primitive Program Table (PPT, observe Tab. 1 for sample) where each row is occupied by the single Stable State, Unstable State(s) and supplemented by the OV values (some of them may be unspecified, however).
- Reducing of the PPT by merging of the equivalent and pseudo-equivalent states. The rules are solid here and do not distinguish between Moore and Mealy machines, as presented below:
  - Two or more Stable States are equivalent when their IVs are same, OVs are non-contradictory and all Transitions (Unstable States) are non-contradictory or contradictory conditionally and the contradiction condition is fulfilled.

- Merging of the rows to compress the MB and the length of the ISV. The rules vary, depending on the target structure (Moore vs Mealy), as presented below:
  - Two or more rows are merge able, when no contradictions on the Stable and Unstable states occur.
  - In case of the merging towards Moore machine structure there is requirement of the non-contradictory outputs. This is not the case when working towards Mealy's structure.
- Encoding the rows of the MB with particular respect to the Race [2][10][11] phenomena – it is essential to ensure that each Transition between two ISVs is logically neighbour.

The remaining steps are related to the implementation of the asynchronous circuit that are not limited to the Huffman's method - its description is above the scope of this paper.

The Huffman's method delivers MB description in the form of the binary coded previous-state-next-state flow table and OCB functions coded table. Those tables used to be minimized using Karnaugh maps when working manually. In case of the asynchronous system design it is common that specification of the Boolean functions that describe the changes of the ISV is weak (as well as the OCB functions) so iterative methods of the minimization applies here.

	IV (binary)				
Row	**00**	**01**	**11**	**10**	**OV**
1	1	2			0
2		2	3		0
3		4	3		1
4	1	4			1

Tab.1. Sample primitive program table (Stable States are underlined)

3. The software implementation

The implementation of the Huffman's method is a part of the ZMITACSIM – a component based software solution both for circuit design and education purposes. Front side web application to the ZMITACSIM is a website hosted on the Microsoft Internet Information Services (IIS) server. The backend is composed of the set of computational web services implementing algorithm logic. The only part of the ZMITACSIM accessible through the Internet is an IIS based website (Appendix A for URL). The computational web services, implementing directly (among others) Huffman's and Kazakov algorithms are hidden to the Internet and accessible only through the Campus Intranet. All of the Huffman's algorithm is implemented as a standalone, stateless web service, utilized both by the front side web application and 3rd party components – i.e. the Huffman's service by itself uses Kazakov web service that delivers Boolean function minimization capabilities to the solution in the final step of the obtaining MB state equations and OCB's OV equations (Fig.3). Such a design is required to ensure system security, stability and

scalability, as each component can be hosted independently on the separate PC machine. Moreover, uniform TCP communication support free composition of the computation network, not limited to the single location nor domain.

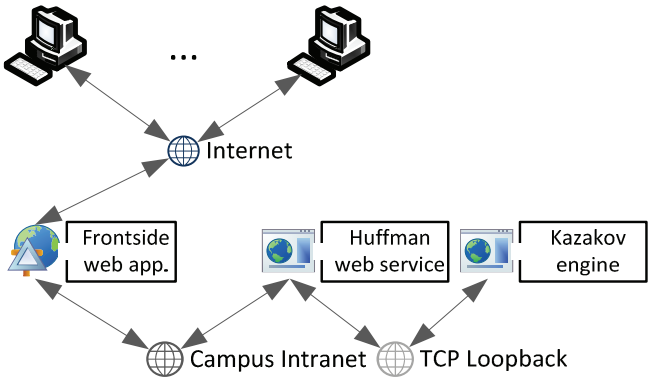

Fig. 3. Implementation schema

At the web development level, all web services, participating in the system are stateless, thus a full set of results is obtained within single remote function call via the uniform SOAP communication. The input data delivered to the Huffman web service is presented as a sequence of the switching, using dedicated grammar (Appendix B for details).

The sample timing diagram of the thermostat controller related to the PPT given by the Tab. 1 is presented on the Fig. 4. Its sequence of the switching as input string to the Huffman web service is presented on the Fig. 5. The sequence is composed of the 5 segments.

The implemented Huffman's web service delivers two kinds of data layers on its output, utilized in two different scenarios:

- The computation layer, presenting the final MB and OCB equations, used for computation and digital circuit design.
- The detailed step – by – step results of each of the steps composing Huffman's algorithm, including detailed data on PPT reducing, row merging and encoding of the rows, used for debugging and educational purposes mostly.

Both layers are delivered to the caller as a single XML file using SOAP standards over the TCP socket.

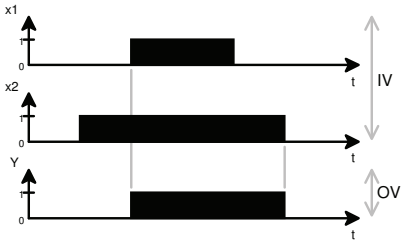

Fig. 4. Sample thermostat controller timing diagram

```
-x1-x2-_Y|+x2|+x1+_Y|-x1|-x2-_Y
```

Fig. 5. Sample thermostat controller switching sequence

The Huffman's computational service is hosted as a session-less web service, working synchronously within single request-response fashion. That limits the possible problems with concurrency and coordination between front side website and the service, as well as simplifies implementation.

4. The complexity analysis

The computational complexity and efficiency is essential when processing multidimensional problems. In case of the Huffman's algorithm, there is more than one dimension of the input data to be considered: n - IV length, m - OV length, s - number of the segments in the PPT equal to the number of the Stable States.

The analysis of the complexity of each step constituting the Huffman's algorithm is provided in the following chapters. The preparation of the PPT and binary row encoding (steps 1 and 4 of the Huffman algorithm) are omitted as those steps are not a long lasting task and they are directly related to the length of the input switching sequence string provided by the service.

4.1 Reducing of the PPT - merging of the equivalent and pseudo-equivalent states

This step involves checking each of the Stable States against another with presenting previously criteria. Regarding the number of the segments, the pessimistic case is given by the total number of comparisons that may be performed to estimate merging schema for one Stable State, eq. (1):

$$O_{I,1}(s^2) \leq \frac{s(s-1)}{2} \tag{1}$$

Comparing of any two rows involves verification of the possible contradictions on the Transitions for all IV combinations and OV combinations given by n and m

respectively. That leads to the final, pessimistic order, eq. (2):

$$O_{I,1,pes}(s^2 \cdot n \cdot m) \tag{2}$$

The common asynchronous systems present the rule that each row of the PPT contains only one Unstable State. This lowers average complexity order for the formula given by eq. (3):

$$O_{I,1,avg}(s^2 \cdot m) \tag{3}$$

Finally merging of all of the rows within PPT, requires checking of any Stable State against another, so the most complex and pessimistic case is given by the eq. (4):

$$O_{I,pes}(s^2 \cdot n \cdot m \cdot \log_2 s) \tag{4}$$

4.2 Reducing of the rows - Moore's structure

The second step of merging varies with the output condition checking between target Moore and Mealy structures. The merging of the rows to produce t compact MB for the Moore machine requires merging try for each row against the other. Moreover, each single checking requires verification of the Unstable States on every IV binary combination and for every OV binary value as well. This leads to the conclusion that the total pessimistic order is same as when PPT merging, eq. (5):

$$O_{II,Moore,pes}(s^2 \cdot n \cdot m \cdot \log_2 s) \tag{5}$$

4.3 Reducing of the rows - Mealy's structure

In case of the Mealy machine, the row merging conditions, lack the requirement of non-contradictory outputs as in case of the Moore. This leads to the simplified order of the complexity given by eq. (6):

$$O_{II,Mealy,pes}(s^2 \cdot n \cdot \log_2 s) \tag{6}$$

4.4 Complexity analysis summary

The most critical part of the algorithm complexity relates to the steps 2 and 3 of the Huffman algorithm. The final estimated pessimistic order is approximated by eq. (5) for Moore machines and eq. (6) for Mealy machines.

5. Complexity verification by performance tests

A set of integration and performance tests has been performed to ensure correctness of the system and the estimation of the orders [13]. The integration tests included low complexity samples that are manually solvable by the tester (varying from elementary systems, as presented on Fig. 4, finishing on the 3 input, 3 output 10 segment system). Once ensured the correctness of the algorithm implementation, the performance tests acknowledging the estimated orders of complexity were performed.

5.1 Testing methodologies

Regarding the form of the orders eq. (5) and eq. (6) tests were performed in three areas: variable length of the IV while constant OV and the number of the Stable States (Table 2), variable OV while constant IV length and fixed number of Stable States (Table 3), variable number of the Stable States with fixed length of the IV and OV (Table 4). The I1 test is shared between all performance test sessions as common for both (Table 3 and Table 4).

Test set	Stable States (s)	IV length (n)	OV length (m)
I1	20	5	2
I2	20	10	2
I3	20	15	2
I4	20	20	2

Tab. 2. Performance test sets on IV length

Test set	Stable States (s)	IV length (n)	OV length (m)
I1	20	5	2
O1	20	5	4
O2	20	5	6
O3	20	5	8
O4	20	5	10

Tab. 3. Performance test sets on OV length

Test set	Stable States (s)	IV length (n)	OV length (m)
I1	20	5	2
S1	40	5	2
S2	60	5	2
S3	80	5	2
S4	100	5	2

Tab. 4. Performance test sets on segments

Test set	Moore, avg. [ms]	Mealy avg. [ms]
I1	4.8589	6.4084
I2	5.4456	7.4759
I3	7.3178	10.4242

I4	9.6523	13.5704
O1	5.0678	7.0649
O2	5.5732	7.3152
O3	6.0076	7.6956
O4	6.3420	8.1194
S1	43.0218	51.1353
S2	185.3492	201.9617
S3	552.1152	590.0618
S4	1306.9410	1363.13

Tab. 5. Average processing time

6. Summary

The implementation of the Huffman's asynchronous circuit design using web service is used during the teaching process, participating as an element of the distant and on-site learning. The future work includes implementation of the other components
of the ZMITACSIM system where the Huffman's method is a computation component among others. The resulting average processing time for each performance test set is juxtaposed in the Table 5. The implementation and computational complexity estimation validated the theoretical model of the Huffman's one. The graphical representation of the tests also proves correctness of the theoretical estimation of the order of the Huffman's method complexity (Fig. 5, Fig. 6 and Fig. 7).

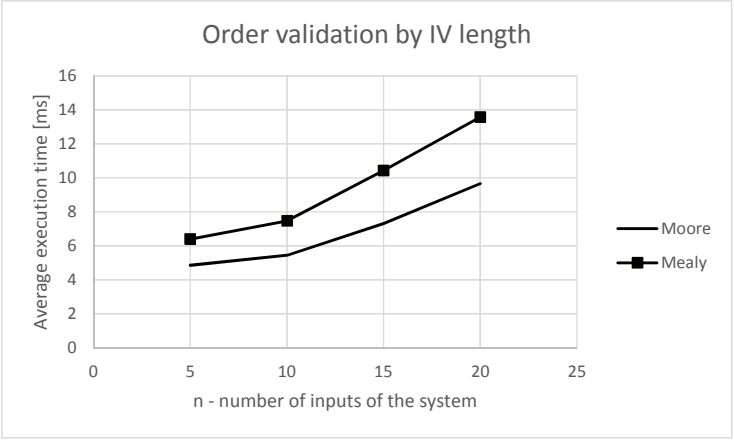

Fig. 5. Average execution time - variable IV length

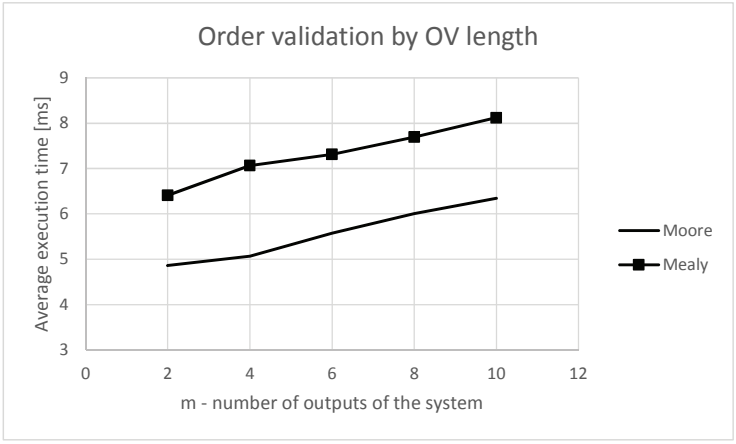

Fig. 6. Average execution time – variable number of the Stable States

7. Conclusion

The experimental results prove the theoretical analysis of the computational complexity. Regarding the complexity as presented before, the large number of the input signals, states and output signals may lead to longer computation times, thus increasing the timeouts on the connection may be required. In case the synchronous design is no more acceptable because of the long response times, the asynchronous implementation would be necessary then. The authors are aware of this problem. All the experiments were provided within the reasonable complexity of the Huffman's systems. This limitation does not impact the overall experiments and its results, however. It also has no impact on the experimental validation of the theoretical model.

8. Acknowledgements

This material is based upon work supported by Silesian University of Technology under the project BK-266/RAu2/2014/502.

Appendix A

The ZMITACSIM front side website is accessible over the Internet at: http://zmitacsim.zmitac.aei.polsl.pl .

Appendix B

The web service input data grammar [13] in BNF:

- <digit>::=0|1|2|3|4|5|6|7|8|9

- <l_letter>::=a|b|c|d|e|f|g|h|i|j|k|l|m|n|o|p|q|r|s|t|u|v|w|x|y|z
- <u_letter>::=A|B|C|D|E|F|G|H|I|J|K|L|M|N|O|P|Q|R|S|T|U|V|W|X|Y|Z
- <sign>::=- | +
- <iv_id>::=<l_letter>|<digit>|<iv_id><l_letter>|<iv_id><digit>
- <ov_id>::=_<u_letter>
- <siv_id>::=<sign><iv_id>
- <sov_id>::=<sign><ov_id>
- <seg>::=<siv_id>|<sov_id>|<seg><siv_id>|<seg><sov_id>
- <start_seg>::=<siv_id><sov_id>
- <formula>::=<start_seg>|<formula>&<seg> - where & stands for the | sign

within the input protocol which is segment separator.

There is required that the <start_seg> contain all input and output variables with the appropriate values.

References

[1] M. Karnaugh: *The Map Method for Synthesis of Combinational Logic Circuits.*, in Transactions of the American Institute of Electrical Engineers part I, 72(9), pp. 593–599, 1953.

[2] D. A. Huffman: *The Synthesis of Sequential Switching Circuits.*, in Journal of The Franklin Institute, 257(3-4), 1954.

[3] J. Siwiński: *Układy przełączające w automatyce.*, WNT, 1968.

[4] V. D. Kazakov: *Minimalizacija łogiczeskich funkcji bolszogo czisła pieriemiennych.* in Awtomatika i Telemechanika., 9, 1962.

[5] P. Kerntopf, A. Michalski: *Wybrane zagadnienia syntezy kombinacyjnych układów logicznych.* PWN, 1972.

[6] E. J. McCluskey Jr.: *Introduction to the Theory of Switching Circuits.*, McGraw-Hill Book Co., 1965.

[7] E. J. McCluskey Jr.: *Minimization of the Boolean Functions.*, Bell Systems Tech. J., pp. 1417-1444, 1956.

[8] M. V. Quine: *The Problem of Simplifying Truth Functions*., in Amer. Math. Monthly, 59, pp. 521-531, 1952.

[9] R. Brayton, G.D. Hachtel, L. Hemanchandra, R. Newton, A. Sangiovanni-Vincentelli: *A Comparison of Logic Minimization Strategies Using ESPRESSO: An APL Program Package for Partitioned Logic Minimization*. in Proc. of the Int. Symposium on Circuits and Systems, Rome, pp. 42-48, 1982.

[10] H. Kamionka-Mikuła., H. Małysiak, B. Pochopień: *Teoria układów cyfrowych, T. I. Układy kombinacyjne, T. II. Układy sekwencyjne*. Wydawnictwo Politechniki Śląskiej, 2013.

[11] U. Stańczyk, K. Cyran, B. Pochopień: *Theory of logic circuit. Vol. 1. Fundamental issues. Vol. 2. Circuit design and analysis.* Silesian University of Technology, 2007.

[12] World Wide Web Consortium, SOAP Specification. W3C, 2004. [Online]. Available: http://www.w3.org/TR/soap/

[13] P. Mroczkowski: *System ZMITACSim – Implementacja metody Huffmana dla układów Moore'a i Mealy'ego.* M.S. Thesis, Dept. Aut. Control, Electron. and Comp. Science, Silesian University of Technology, 2013.

[14] T. Łuba: *Synteza układów logicznych.* Oficyna Wydawnicza Politechniki Warszawskiej, 2005.

[15] R. Fuhrer, M. Nowick: *Sequential Optimization of Asynchronous and Synchronous Finite-State Machines.* Kluwer, Boston, 2001.

[16] J. Brzozowski, C. Seger: *Asynchronous Circuits.* Springer Verlag, New York 1995.

[17] C. J. Myers: *Asynchronous Circuit Design.* John Wiley and Sons, New York 2001.

[18] S. H. Unger: *Asynchronous Sequential Switching Circuits*. Wiley-Interscience, New York, 1969.

[19] B. Sheikh, R. Manohar: *An Operand-Optimized Asynchronous IEEE 754 Double-Precision Floating-Point Adder.* in Proc. of the 2010 IEEE Symp. on Async. Circuits and Systems, Grenoble, pp. 151-162, 2010.

[20] N. Jamadagni, J. Ebergen: *An Asynchronous Divider Implementation.* in Proc. of the 2012 IEEE 18th Int. Symp. on Async. Circuits and Systems, Lyngby, pp. 97-104, 2012.

[21] T. Liu, J. Rabaey: *Statistical Analysis and Optimization of Asynchronous Digital Circuits.* in Proc. of the 2012 IEEE 18th Int. Symp. on Async. Circuits and Systems, Lyngby, pp. 1-8, 2012.

[22] N. Onizawa, W. Gross, T. Hanyu: *A Low-Energy Variation-Tolerant Asynchronous TCAM for Network Intrusion Detection Systems.* in Proc. of the 2013 IEEE 19th Int. Symp. on Async. Circuits and Systems, Santa Monica (CA), pp. 8-15, 2013.

[23] M. Davies, A. Lines, J. Dama et al.: *A 72-Port 10G Ethernet Switch/Router Using Quasi-Delay-Insensitive Asynchronous Design.* in Proc. of the 2014 IEEE 20th Int. Symp. on Async. Circuits and Systems, Potsdam, pp. 103-104, 2014.

Synteza asynchronicznych układów cyfrowych – nowe horyzonty

Streszczenie

Synteza asynchronicznych układów cyfrowych w logice szytej, bazująca na dwuwartościowej algebrze Boole'a, doczekała się licznych modeli i metod implementacji. Najpopularniejsze z nich to metoda Huffmana [2] oraz metoda Tablicy Kolejności Łączeń [3]. Metody te, jakkolwiek powstały w połowie XX wieku, są stosowane również obecnie, zarówno w procesie projektowania przemysłowego jak i w edukacji [10][11][14]. Metoda Huffmana pozwala na syntezę układów o strukturach Moore'a (Fig.1) i Mealy'ego (Fig.2). Proces projektowania z wykorzystaniem metody Huffmana jest wieloetapową metodą o deterministycznym przebiegu. W niniejszym opracowaniu przedstawiono nowatorskie podejście do zagadnienia automatycznego projektowania cyfrowych układów asynchronicznych, bazujące na wytworzeniu oprogramowania o budowie komponentowej oraz z zastosowaniem zunifikowanej komunikacji w sieci WEB z wykorzystaniem protokołu SOAP. Na podstawie teoretycznego modelu opracowano rozwiązanie programowe, składające się z bezstanowego serwisu obliczeniowego oraz aplikacji sieciowej WWW, która służy, jako interfejs użytkownika dla usługi obliczeniowej. W zastosowaniach edukacyjnych przewidziano zwracanie szczegółowych informacji z wszystkich etapów syntezy układów metodą Huffmana, co umożliwia ich wizualizację oraz poruszanie się po nich w przód i w tył, w celu prezentacji zależności i sposobu działania, co stanowi również element zdalnego nauczania - aplikacja WWW jest bowiem widoczna w globalnej sieci Internet (adres znajduje się w Appendix A). Ponieważ metoda Huffmana udostępnia wyniki w postaci zredukowanej i zakodowanej siatki bloku pamięci i bloku kombinacyjnego wyjściowego, serwis obliczeniowy korzysta z uprzednio przygotowanej, zewnętrznej usługi, implementującej funkcjonalność minimalizacji funkcji Boolowskiej metodą Kazakowa. W ramach prowadzonych badań przeanalizowano teoretyczną, pesymistyczną złożoność obliczeniową poszczególnych etapów metody Huffmana, a także łączną, pełną złożoność obliczeniową metody, w szczególności względem liczby sygnałów wejściowych, liczby stanów wewnętrznych oraz liczby sygnałów wyjściowych, otrzymując pełny model złożoności obliczeniowej metody Huffmana. Przeprowadzono szereg eksperymentów, aby potwierdzić oszacowaną teoretycznie złożoność obliczeniową w sposób doświadczalny (Tab. 2-4). Przeprowadzone eksperymenty potwierdziły zgodność z teoretycznie wyznaczonym modelem złożoności obliczeniowej, zarówno dla układów Moore'a jak i Mealy'ego (Tab. 5, Fig. 5 i 6). Na potrzeby sformalizowania zapisu wykresu czasowego opracowano gramatykę zapisu przełączania sygnałów (Appendix B). Stanowi ona format wewnętrzny danych wejściowych dla usługi obliczeniowej implementującej metodę Huffmana.