*The International Workshop
on Discrete-Event System Design, DESDes'01,
June 27÷29, 2001; Przytok near Zielona Gora, Poland*

IMPLEMENTATION OF THE FSM INTO FPGA

Hana KUBÁTOVÁ

Department of Computer Science and Engineering, Czech Technical University in Prague,
Karlovo nám. 13, 121 35 Prague 2, Czech Republic, *Phone*: +42 2 2435 7281
Fax: +42 2 2492 3325, *e-mail: kubatova@fel.cvut.cz*

Abstract. *This paper deals with the possibility of description and decomposition of the finite state machine (FSM). The aim is to obtain better placement of a designed FSM to the selected FPGA. It compares several methods of coding of the FSM internal states with respect to the space (number of the CLB blocks) and time characteristics. It evaluates the FSM benchmarks and looks for such qualitative properties to choose the best method for coding before performing all FOUNDATION algorithms because this process is time consuming. The new method for coding of the internal FSM states is presented. All results are documented by experiments.*

Key Words. *Finite state machine (FSM), Hardware Description Language (VHDL), state transition graph (STG), code method, decomposition, benchmark*

1. INTRODUCTION

Most research reports and other materials devoted to searching of the "optimal" coding of the internal states of *FSM* are based on minimal number of internal states and sometimes also on minimal number of used flip-flops in their hardware realization. The only method how to get the really optimal results is testing of all possibilities, [1]. But sometimes "wasting" of the internal states or flip-flops is better solution due to speed of the designed circuit. The most coding methods are not based on recently used structures, like different types of FPGA or CPLD. Therefore we try to compare several types of sequential circuit benchmarks to search the relation between the type of this circuit (number of the internal states, inputs, outputs, cycles, branching) and the coding method with respect to their implementation by *XILINX FPGA*.

We have worked with the CAD system *XILINX FOUNDATION* v2.1i during all our experiments. We have used the benchmarks from the Internet in KISS2 format, some coding algorithms from *JEDI* program and system *SIS 1.2* [7]. First of all we have classified the FSM benchmarks to know the quantitative characteristics of them: number of internal states, inputs, outputs, transitions (i.e. the number of arcs in the state transition graph - STG), maximal number of input arcs, maximal number of output arcs to and from STG nodes, etc. We have compared eight coding methods: "one-hot", binary, Johnson and Gray that are implemented in *FOUNDATION* CAD system; we have implemented Fan-in and Fan-out oriented algorithms and the algorithm "FAN" connecting Fan-in and Fan-out ones [1], [5] and our original method called "own" that will be presented in this paper. The second group of our experiments has been directed to the decompositions of the FSM. The final results (number of the CLB blocks and maximal frequency) were obtained for concrete FPGA implementation (Spartan XCS05-PC84).

2. METHODS

2.1. Coding methods

"One hot" method uses the same number of bits as the number of internal states - the great number of internal variables is the main disadvantage of this method. The states, that have the same next state for a given input, should be given adjacent assignments ("Fan-out oriented"). The states, that are the next states of the same state, should be given adjacent assignments ("Fan-in oriented"). The states, which have the same output for a given input should be given adjacent assignments (this will help to cover the 1's in the output Karnaugh-maps; "output oriented"). Very popular and frequently used method is the binary code, that uses the minimum number of internal variables, and the Gray code with the same characteristics and adjacent codes for a sequence of states. First partial results based on these 7 methods and benchmarks characteristics were presented in [3]. We have found out, that binary coding is better than "one hot" coding for those FSM, which fulfil the following condition: STG that describes the FSM should be complete or nearly complete. If the ratio of average output degree of a node to the number of states is greater than 0.7, than it is better to use the binary coding. On the contrary, when this ratio is low, "one hot" coding is better. This qualitative characteristic property of the FSM benchmarks is defined as:

$$AN = \text{AverageOutEdges}/(\text{NumberOf States} - 1) \tag{1}$$

The value $AN = 0.7$ were verified on benchmarks and on our specially generated testing FSM [4].

2.2. Method "own"

Our original method combines the "one-hot" and binary coding methods. It is based on the partially FSM internal state decomposition. The global algorithm could be described as follows:

a) All FSM internal states Q_i are placed to the set S_0 – not yet classified states

b) From all S_0 elements select the state Q_i with the most number of transitions to the another disjoint states from S_0. This state Q_i is taken away from S_0 and becomes the first member of the new set S_{group}

c) Construct the set of neighbour internal states of all members of S_{group} – $S_{neighbour.}$ Compute the score [4], that expresses the placement suitability for a state Q_j into S_{group}, for all states from $S_{neighbour}$. The state with the highest score add to S_{group} . The score is a sum of:

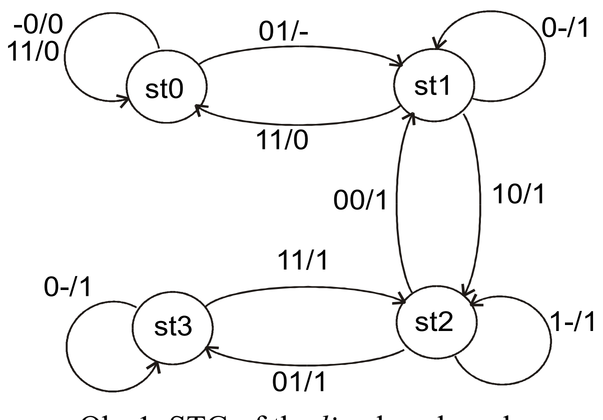
- The number of the transitions from Qj to all states from S_{group} multiplied by the constant 10;
- The number of such states from S_{group} the transition exists from Qj into those ones multiplied by the constant 20;
- The number of the transitions from Qj to all neighbour internal states from S_{group} (i.e. to all states from $S_{neighbour}$) multiplied by the constant 3;

- The number of such states from $S_{neighbour}$ the transition exists from *Qj* to those ones multiplied by the constant 6;
- The number of the transitions from all internal states from S_{group} to *Qj* multiplied by the constant 10;
- The number of such states from S_{group} the transition exists from those ones into *Qj* multiplied by the constant 20;
- The number of the transitions from all neighbour states of S_{group} (placed in $S_{neighbour}$) to *Qj* multiplied by the constant 3;
- The number of the neighbour states in $S_{neighbour}$ the transition exists from those ones to *Qj* multiplied by the constant 6;

d) Compute the *AN* (1) ratio for S_{group}. When this ratio is grater then the "border ratio" (the input parameter of this algorithm, according our experiments usually 0.7) the state *Qj* becomes the real member of S_{group}. Now continue by step c). When the ratio is less then the "border ratio", state *Qj* is discarded from the S_{group} and this set is closed. Now continue by step b).

e) When all internal states are placed into some set S_i and S_0 is empty, the internal state code can be constructed. It is connected from the binary part (serial number of the state in its set in binary notation) and the one-hot part (serial number of a set in one-hot notation). The number of binary part bits is equal to b where 2^b greater or equal to the maximum number of states in sets. The number of one-hot part bits is equal to the number of sets S_i.

Example (*lion* benchmark [7], border ratio 0.7):



Obr.1. STG of the *lion* benchmark

a) All FSM internal states Q_j are placed to the set S_0 – not yet classified states

$$S_0 = \{st0,\ st1,\ st2,\ st3\}$$

b) For all S_0 elements compute the number of transitions to the another disjoint states from S_0 (st0…1, st1…2, st2…2, st3…1). Choose the state with the highest value and construct the new set S_1:

$$S_0 = \{st0,\ st2,\ st3\},\ S_1 = \{st1\}$$

c) Construct the set of neighbour internal states of all members of S_1 – $S_{neighbour}$:

$$S_0 = \{st0,\ st2,\ st3\},\ S_1 = \{st1\},\ S_{neighbour} = \{st0,\ st2\}$$

Compute the score for all states from $S_{neighbour}$:

$$st0_{score} = 1.10+1.20+2.3+1.6+1.10+1.20+2.3+1.6 = 84$$

$$st2_{score} = 1.10+1.20+1.3+1.6+1.10+1.20+1.3+1.6 = 78$$

Choose the state with the highest score and add it to S_1:

$$S_0 = \{st2, st3\}, S_1 = \{st0, st1\}, S_{neighbour} = \{st2\}$$

d) Compute the *AN* (1) ratio for the elements from S_1: $AN = 1.0$. *AN* is grater then 0.7, therefore the state *Qj* becomes the real member of S_1. Now continue by step c).

c) Try to add the state *st2* into S_1 and compute the *AN*. Because $AN = 0.66$ state *st2* is discarded from the S_1 and this set is closed. Now continue by step b).

At the end all internal states are placed into 2 groups:

$$S_1 = \{st0, st1\}, S_2 = \{st2, st3\}$$

Now internal state code is connected from the one bit binary part and the two bits one-hot parts:

st0 … 0/01

st1 … 1/01

st2 … 0/10

st3 … 1/10

3. EXPERIMENTS

The conversion program between KISS2 format and *VHDL* was necessary to build - we have implemented the converter *K2V_DOS* (in *C++* by translator *GCC* for *DOS OS*) [3], [4]. The *K2V_DOS* program allows an acquisition of information about the FSM like e.g.: node degree, number of states, number of transitions, etc. The FSM in the *VHDL* description, that was created by the *K2V_DOS* program, can be described by different ways (with different results):

- one big process sensitive to the clock signal and to the input signals (one *case* statement is used in this process - it selects active state and in each branch of the *case* there are *if* statements, which define next states and outputs - this is the same method, like the *XILINX FOUNDATION* uses for conversion between STG and *VHDL* [8]);
- three processes (*next-state-proc* for implementation of the next-state function, *state-dff-proc* for asynchronous reset and using D flip-flops and *output-proc* for the FSM output function realization). To overcome the *XILINX FOUNDATION* optimization for the "one-hot" coding method we have used direct code assignment, too.

The *K2V_DOS* program system can generate our special testing FSM (for more precise setting of the "border ratio" *AN*). We have generated the Moore type FSM with the determined number of internal states and mainly the determined number of the transitions from the internal states. Our FSM has the STG with the strictly defined and the same number of transitions from all states. The resulting format is the KISS2 format – e.g. 4.kiss testing FSM has the STG with four edges from every internal state (node). Both the first and also the next state connections are generated randomly to overcome the *XILINX FOUNDATION* optimization for the counter design.

The *K2V_DOS* program can generate different FSM internal state coding by methods binary, Gray, Johnson, one-hot, Fan-in, Fan-out and FAN and "own". All benchmarks were processed by *DECOMP* program to generate all possible types of decompositions (in KISS2 format due to using the same batch for FPGA implementation).

4. RESULTS

We have performed about 1000 experiments with different types of coding and decomposition methods for 50 benchmarks. We have obtained the great amount of the results processed to the visual graphs. One of them expressing the comparison of the "one-hot", binary and "own 0.7" coding methods with translation of them into three VHDL processes and direct code assignment is presented on Fig. 2.

We can present the following conclusions based on our experiment results:

- the binary coding method gives the best results for FSM with few internal states (5) and for FSM with *AN* > 0.7 (the state transition graph with many cycles)
- "one-hot" coding methods is better for other cases and mostly generates the faster circuits (but the *XILINX FOUNDATION* uses optimization methods for "one-hot" coding)
- the original "own" method is universal one because it combines the advantages of both "one-hot" and binary methods (see Fig. 2)
- for such FSM implementation where the majority of the CLB blocks are used (e.g. 90%) the "one-hot" methods gives better results mainly with respect to the maximum working frequency due to easier wiring
- all FSM decomposition types are not advantageous to use in most cases due to great information exchange - the parallel decomposition is the best one (when it exists)
- the different strategy for looking for the partitions – the best FSM partition is not that one with minimal number of internal states but that one with the minimal sets of input and output symbols – could be used for FPGA implementation

5. ACKNOWLEDGEMENTS

This paper is based on the results of several student projects supervised by the author during two years. This research was in part supported by the internal CTU grant.

REFERENCES

[1] Ashar, P., Devadas, F., Newton, A.R.: "Sequential Logic Synthesis*", Kluwer Academic Publishers*, Boston/Dordrecht/London 1992

[2] Hrdý, T.: "Influence of the FSM decomposition to their implementation", *Diploma thesis*, CTU Prague, FEL, 2001 (in Czech)

[3] Kubátová, H., Hrdý, T., Prokeš, M.: "Problems with the Encoding of the FSM with the Relation to its Implementation by FPGA", *ECI2000 Electronic Computers and Informatics, International Scientific Conference*, Herl'any 2000.

[4] Prokeš, M: "Influence of the FSM internal states codind to their implementation", *Diploma thesis*, CTU Prague, FEL, 2001 (in Czech)

[5] Studenovský, J.: Coding of internal states of synchronous sequential circuit. *Diploma thesis*, CTU Prague, FEL, 1999 (in Czech)

[6] Výmola, V.: Decomposition of a finite state automaton. *Diploma thesis*, CTU Prague, FEL, 2000 (in Czech)

[7] Benchmarks test
ftp://ftp.mcnc.org/pub/benchmark/Benchmark.dirs/LGSynth93/LGSynth93.tar

[8] The Programmable Logic Data Book. XILINX Tenth Anniversary, 1999
http://www.xilinx.com

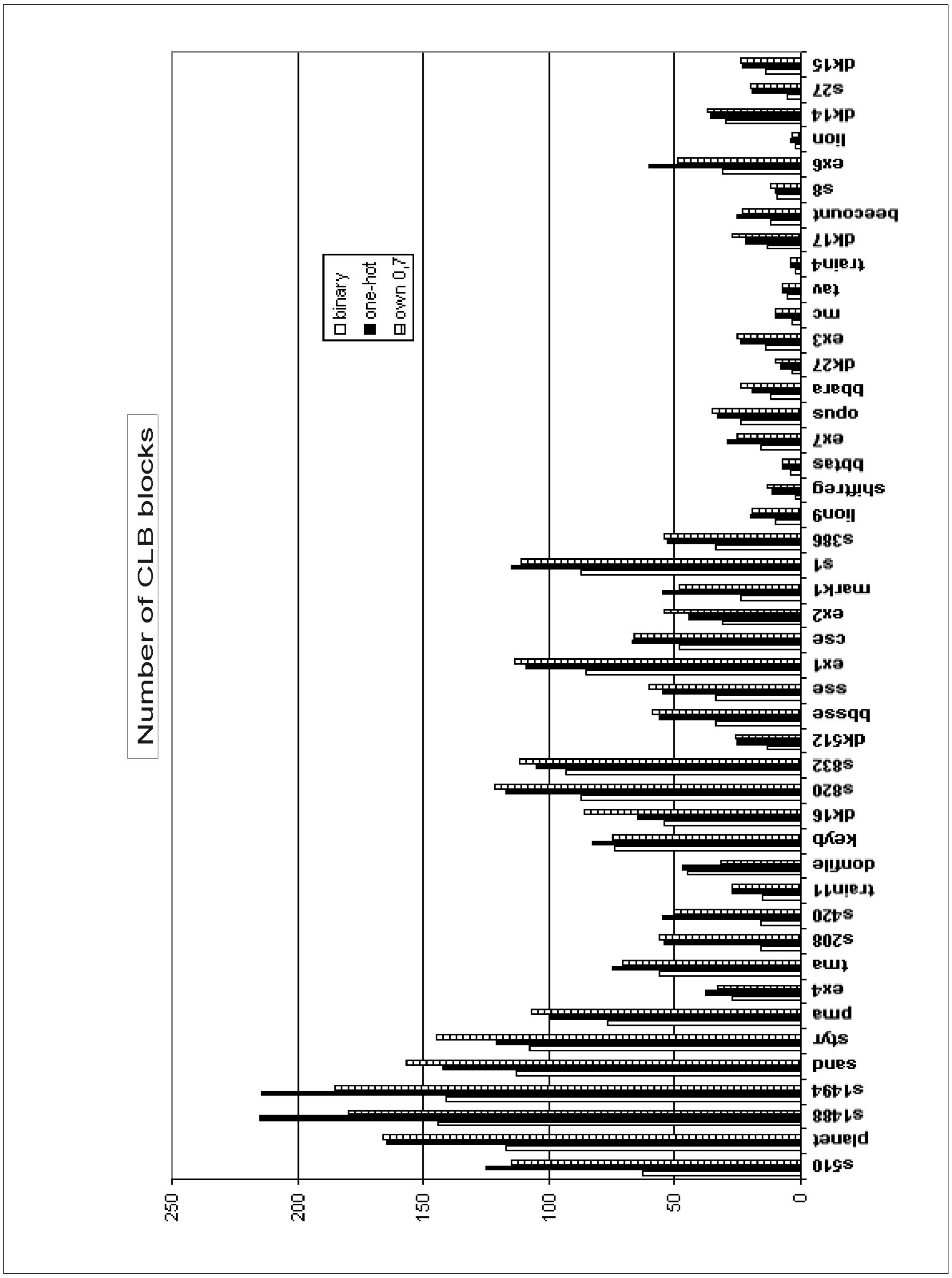


Fig. 2. Number of used CLB blocks for all processed benchmarks