XIII SIGE ISSN: 1983 7402 ITA, 27 a 30 de setembro de 2011

A new Memory Element for Synthesis by Direct Mapping of Asynchronous FSMs from XBM Specification

Duarte L. Oliveira, Lester de A. Faria e Noé Alles
Divisão de Engenharia Eletrônica do Instituto Tecnológico de Aeronáutica – ITA – IEEA
Praça Mal Eduardo Gomes, 50 – Vila das Acácias – C EP 12.228-900 – São José dos Campos – SP – Brazil

***Abstract*—This paper proposes a new memory element (control cell) for synthesis by direct mapping of extended burst-mode asynchronous finite state machines (XBM_AFSM). The great advantage of the synthesis by direct mapping is the achieved simplicity in methodology because it does not demand any knowledge of asynchronous logic concepts. The direct mapping starts from a well known extended burst-mode (XBM) specification and allows implementing large specifications with just little computational effort. The proposed memory element presents a better performance if compared to the existing ones that are used in the synthesis by direct mapping of XBM_AFSM, showing a reduction of 50% in the area, 29% of LUTS and 15% at the latency time.**

***Key-words* — control cells, finite state machine, asynchronous logic, direct mapping, XBM specification**

I. Introduction

In recent years, there has been a growing interest in asynchronous circuits due to the complexity and the increasing performance of digital systems [1,2]. Asynchronous circuits present several potential advantages over their synchronous counterparts: they tend to be faster and smaller, dissipate less power, do not present clock skew problems and are more robust in relation to temperature variations and electromagnetic interactions [2,3]. All of these features become this kind of circuits a great and special opportunity to aeronautical and aerospace applications, where low size, low power and high performance are highly desirable.

A. *Classes of asynchronous circuits*

Asynchronous controllers (AC) may be classified according to three different kinds of delay models, where “Bounded delay” relates to a well-known range and “unbounded delay” relates to finite but unknown ones [4,5]. These AC can be classified as:

- Speed independent (**SI**) circuits - They operate according to the unbounded gate and zero wire delay model;
- Delay insensitive (**DI)** circuits - They operate according to the unbounded gate and use the wire delay model; and
- Asynchronous finite state machines (**AFSMs**) - Originally proposed by Huffman [6] and extended as burst mode asynchronous finite state machines in [7,8], they operate according to the bounded gate and wire delay model. Their synthesis starts from a burst mode (BM) specification, as proposed by Nowick [7]. The BM specification is very familiar to VLSI designers. Later, it was generalized to the, so called, extended burst mode specification (**XBM_AFSM**), as proposed by Yun [9,10], and then to the multi-burst graph specification (**MBG_AFSMs**), as proposed by Oliveira et al. [11].

SI and **DI** circuits operate in the I/O mode. As soon as an output transition occurs, the circuit is ready for a new input event. Such circuits are modular, presenting a simple interface and are very robust.

SI circuits are synthesized from a signal transition graph (STG), a Petri-net like specification [12] [13]. Restrictions must be imposed to the circuits, in order to incorporate a “free of hazards” behavior in such descriptions. STG specifications are not familiar to VLSI designers, tending to be confused for large specifications. Also, the description of a conditional signal by level is not natural (level sensitive signal).

DI circuits are synthesized from a specification in a concurrent language [14] [15]. The resulting circuits are netlists of complex gates, tending to be large and slow (poor optimization) [14] [15].

XBM_AFSMs operate in the generalized fundamental mode (GFM). A new transition may start only when the circuit has reached a stable state. Input and/or output bursts are allowed. The XBM_AFSMs presents good performance, especially for those applications that interact with a slow environment. However, they are not modular and present timing and interface problems [16].

B. *Asynchronous controllers (AC): approach*

The two main approaches to the designing of asynchronous controller are: logic synthesis [17] and direct mapping [18-20].

Logic synthesis works with the low-level specifications, which capture the behavior of a circuit at the level of a signal

Duarte L. Oliveira, duarte@ita.br, Lester A. Faria, lester@ita.br, Noé Alles, noealles@ita.br

transition. In this approach, the goal is to obtain the minimized Boolean equations of the output signals and the next state. This optimization requires an exploration of all possible events of the specification. The logic synthesis methods are well established and supported by automatic synthesis tools, as Petrify [21], Minimalist [22], and 3D [9,10,23]. However, it suffers from excessive computation complexity and memory requirements. These limitations make it difficult to be applied to large specifications. The tools Minimalist and 3D, respectively, synthesize BM_AFSMs and XBM_AFSMs, which operate in the GFM mode. For STG specification, there is a variant method that reduces the computational effort of the synthesis, called direct synthesis [24,25].

In Direct Mapping approach, the graph specification of a system is translated into a circuit net list. Each node of the graph corresponds to a circuit element and the arcs between these elements represent the interconnections of the circuit. Different methods for direct mapping have been proposed. In [18] and [19] the AFSM was adopted, as limitation the assumptions GFM. In [20] the adopted specification was the STG. Oliveira et al. [26] proposed a method for XBM_AFSMs synthesis by direct mapping that operates in the input burst/ output burst (I_b/O_b) mode, obeying the bounded gate and wire delay model. While the I/O mode accepts new inputs when one of the output signals is changing (concurrency of inputs and outputs), in the (I_b/O_b) mode a new input burst is only accepted when all the output signals have changed their value. The direct mapping approach in [26] starts from the extended burst mode specification and produces XBM_AFSMs which are more modular, presenting a simpler interface. The great advantage of the method is the simplicity of the procedure. It does not require previous knowledge of the theory of “free of hazard” circuits and of the critical race. Another advantage of the method is that it allows the implementation of large specifications with just a little computational effort. In the direct mapping approach, each state of the XBM specification requires a memory element (control cell) [26].

In this paper it is proposed a memory element for XBM_AFSMs synthesis by direct mapping. Compared to the memory element proposed by Oliveira et al. [26], this memory element has a similar area but with a lower latency time, i.e., presents a better performance. Once it presents the same behavior of the memory element used in [26], it s possible to replace one by the other, achieving better results.

The paper is organized as follows. In section II it is described the extended burst mode specification. In section III the proposed memory element is presented, while in section IV the direct mapping approach is focused. In section V it is showed an example of the cell functioning and in section VI the achieved results of the proposed memory element are discussed. Finally, we present some concluding remarks in section VII.

II. Extended Burst-Mode Specification

The burst mode is represented by a graph (BM_graph) whose vertices represent the stable states while the arcs represent the state transitions. An initial state must exist. Transitions occur when one or multiple inputs/outputs (bursts) change their logic level, 0→1 or 1→0 (transition sensitive signals). While there is no input change, the machine remains in the same stable state. The input/output bursts must be monotonic, i.e., they can change only once during each transition. Three restrictions must be obeyed by the burst mode specifications: 1. signal polarities, 2. unique entry points and, 3. maximal set property [7]. Yun proposed the extended burst mode specification adding two more features: directed don’t-care signals (which allow an input signal to change concurrently with output signals) and conditionals (which depend on level sensitive signals) [9]. For the Ib/Ob operation mode, more three restrictions are required:

- The output burst must never be empty. If it happens in the original behavior, an acknowledgement output signal must be added.
- Level sensitive transitions must become stable during the previous state.
- Directed don’t care signals must remain constant during the next-state transition.

Fig.1 shows an extended burst mode specification (benchmark SCSI) with 4 inputs (**Cntgt1, Frin, Ok, Rin**), 2 outputs (**Aout,Frout**) and an initial state 0. The description **Rin+ Frin- / Aout+** in transition 5→3 means that the output (**Aout**: 0→1) will follow the input burst (**Rin**: 0→1 AND **Frin**: 1→0). The level sensitive signal **cntgt1** is used to describe the mutual exclusion between transitions 3→6 and 3→4. The directed “don’t care” signal **Rin*** in transition 4→5 means that **Rin** may either change its value or remain in its old value.

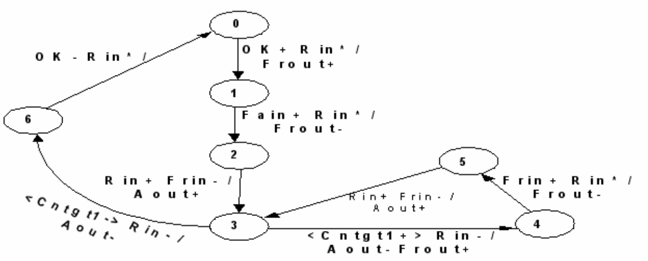


Fig. 1. Extended burst-mode specification

III. Memory Element

The direct mapping uses one memory element (control cell) for each state of the XBM specification. The function of the control cell is to enable and disable the sequence of states that are being processed. Fig. 2 shows the memory element proposed in [26]. It is implemented in the architecture based on the latch RS, having its behavior described by the timing diagram shown in Fig. 3. The new control cell proposed in this paper follows the same behavior of that presented in Fig. 3. The control cell has the inputs [**Req-in, Ack-in**] and outputs [**Req-out, ack-out**]. The input signal **Req-in** enables the present state. The output signal **Req-out** starts the activation of next state. The output signal **Ack-out** starts the disabling of previous state. The input signal **Ack-in** disables the state. Fig. 4 shows the proposed control cell specified in the MBG [11]. It is implemented in a Huffman machine with feedback output. Fig. 5 shows the Multi-Burst flow table free of critical race. Fig. 6 shows the logic “hazard-free” extraction of the signals **Ack-out** and **Req-out**. Finally, Fig. 7 shows the logic circuit.

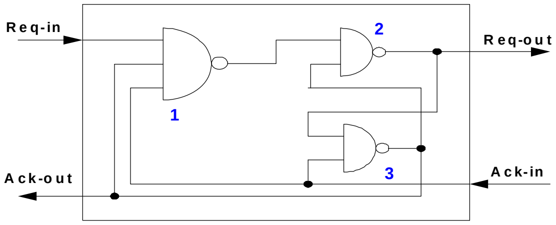


Fig. 2. Memory element of [26].

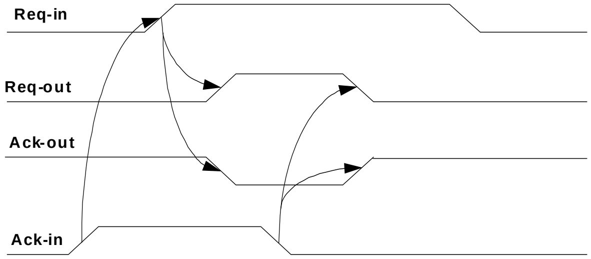


Fig. 3. Timing diagram of control cell.

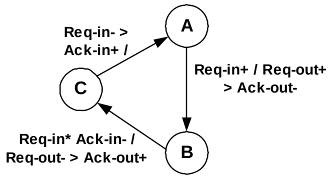
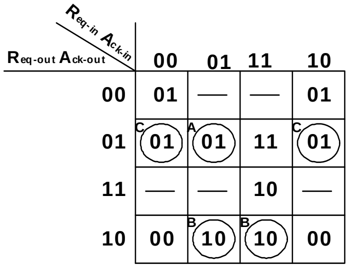
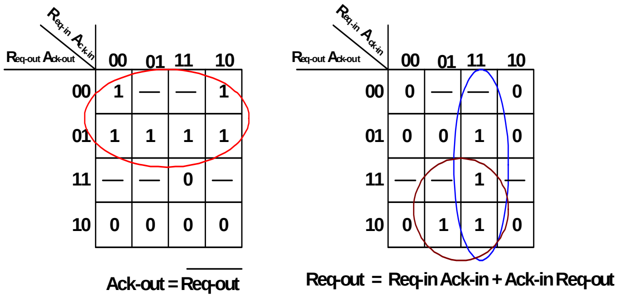


Fig. 4. Multi-burst graph specification.



Req-out Ack-out \ Req-in Ack-in	00	01	11	10
00	01	—	—	01
01	C 01	A 01	11	C 01
11	—	—	10	—
10	00	B 10	B 10	00

Fig. 5. Multi-burst flow table.



Req-out Ack-out \ Req-in Ack-in	00	01	11	10
00	1	—	—	1
01	1	1	1	1
11	—	—	0	—
10	0	0	0	0

$\text{Ack-out} = \overline{\text{Req-out}}$

Req-out Ack-out \ Req-in Ack-in	00	01	11	10
00	0	—	—	0
01	0	0	1	0
11	—	—	1	—
10	0	1	1	0

$\text{Req-out} = \text{Req-in Ack-in} + \text{Ack-in Req-out}$

Fig. 6. Karnaugh map: next state equations.

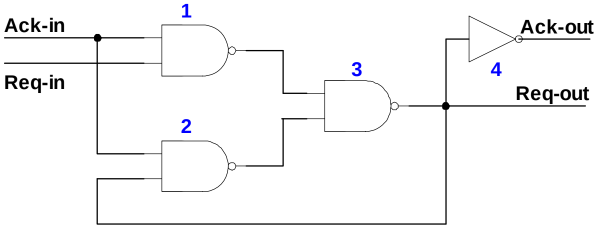


Fig. 7 Logic circuit of the new control cell.

A. *Timing analysis*

Latency time is defined as the interval between the activation of the input burst and the activation of the output signal. On the other hand, the cycle time is defined as the interval between the activation of input burst and the stabilization of the cell. Figure 8 shows the communication between memory elements, involving a state transition.

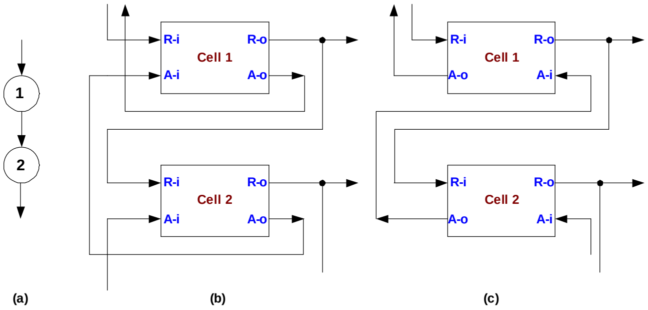


Fig. 8. General behavior: a) state transition; b) comunication of proposed cells: c) comunication of cells of [26].

Considering the memory elements of Fig. 2 and 7 and their communication shown in Fig. 8, it could be possible to extract the latency time and the cycle time. For cycle time, the signal **Ack-out** involves two cells.

The following next four equations ((1) to (4)) are related to the proposed cell and the last four equations ((5) to (8)) are related to the cell presented in [26], where TP is the maximum propagation time of a gate.

$$T_{LATENCY\_R\text{-}out\text{-}2} = T_{P2\_NAND\text{-}2\text{-}1} + T_{P2\_NAND\text{-}2\text{-}2} \quad (1)$$

$$T_{LATENCY\_A\text{-}out\text{-}2} = T_{P2\_NAND\text{-}2\text{-}1} + T_{P2\_NAND\text{-}2\text{-}2} + T_{P2\_INV\text{-}4} \quad (2)$$

$$T_{CYCLE\_R\text{-}out\text{-}2} \geq T_{P2\_NAND\text{-}2\text{-}1} + T_{P2\_NAND\text{-}2\text{-}2} + T_{P2\_NAND\text{-}2\text{-}3} \quad (3)$$

$$T_{CYCLE\_A\text{-}out\text{-}2} \geq T_{P2\_NAND\text{-}2\text{-}1} + T_{P2\_NAND\text{-}2\text{-}2} + T_{P2\_INV\text{-}4} + + T_{P1\_NAND\text{-}2\text{-}1} + T_{P1\_NAND\text{-}2\text{-}2} + T_{P1\_INV\text{-}4} \quad (4)$$

$$T_{LATENCY\_R\text{-}out\text{-}2} = T_{P2\_NAND\text{-}3\text{-}1} + T_{P2\_NAND\text{-}2\text{-}2} \quad (5)$$

$$T_{LATENCY\_A\text{-}out\text{-}2} = T_{P2\_NAND\text{-}3\text{-}1} + T_{P2\_NAND\text{-}2\text{-}2} + T_{P2\_NAND\text{-}2\text{-}3} \quad (6)$$

$$T_{CYCLE\_R\text{-}out\text{-}2} \geq T_{P2\_NAND\text{-}3\text{-}1} + T_{P2\_NAND\text{-}2\text{-}2} + T_{P2\_NAND\text{-}2\text{-}3} \quad (7)$$

$$T_{CYCLE\_A\text{-}out\text{-}2} \geq T_{P2\_NAND\text{-}3\text{-}1} + T_{P2\_NAND\text{-}2\text{-}2} + + T_{P2\_NAND\text{-}2\text{-}3} + T_{P1\_NAND\text{-}2\text{-}3} + T_{P1\_NAND\text{-}2\text{-}2} \quad (8)$$

IV. Synthesis Procedure

The behavior of an extended burst mode asynchronous state machine is initially captured as a state transition diagram (XBM_graph), as the one illustrated in Fig. 1. The synthesis procedure proposed by Oliveira et al. [26] consists of seven steps:

1. Each state of the XBM_graph will be represented by a control cell. Paths with two state transitions must have auxiliary control cells.

2. Each control cell is associated to a block CL (condition logic) with output in **Req-in**. Cells of auxiliary control do not need of the block CL.

3. Perform the connections ***Req-out***→*CL* for each state transition, where **Req-out** is the initial state and CL is the final state.

4. Performe connections ***Ack-out***→***Ack-in*** for each state transition in the opposite direction. When there are two or more connections arrining in **Ack-in,** it generates a joint.

5. For each state *j* of the CL_j, extract the Boolean equation of type "sum-of-product". Each state transition auto coming at state *j* and generates a product with input burst.

6. For each joint, replace it by a gate AND.

7. Using signals **Req-out**, get the minimum Booleans equations of type "sum-of-product" of output signals.

Fig. 9 shows the logic structure for each state of the XBM_graph. Fig. 10 shows the target architecture for the proposed controllers.

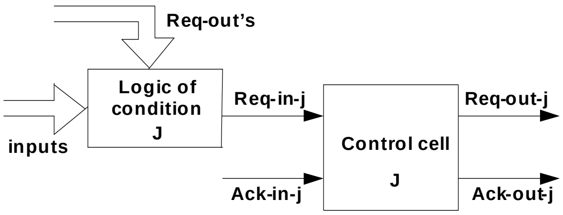


Fig. 9. Logic struture of each state.

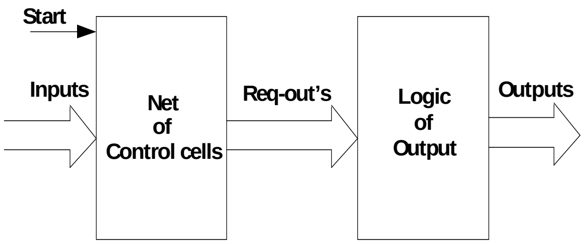


Fig. 10. Target architecture.

V. Example Implementation

In this section it is illustrated, through an example, the approach of [26] with the benchmark Sbuf-send-pkt2-core using the proposed memory element instead of the original one. The XBM specification of this example is shown in Fig. 11. Fig. 12 shows the step one. How is the path of two state transitions (1→3→1) we must insert an auxiliary control cell between the state transition 3→1. Fig. 13 shows the 2nd and 3rd steps (insertion of connections for enable the state and the block CL). Fig. 14 shows the step four (the insertion of connections that disables the states). The steps five, six and seven (get the Boolean function CL (Ri's), solution of join (AND gate) and the Booleans functions of output signals (**Ack, Sendline**)) are shown through the equations.

R-i-0 = Start + Ackline Req' R-o-2

R-i-1 = R-o-0 Req + Ackline' R-o-31

R-i-2 = R-o-1 done Ackline

R-i-3 = R-o-1 done' Ackline

R-i-31 = R-o-3

Ack = R-o-0' R-o-2

Sendline = R-o-2' R-o-1 R-o-3'

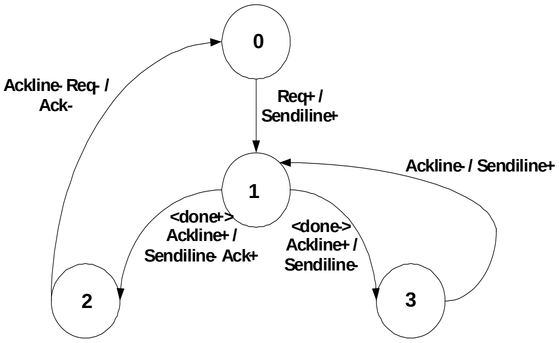


Fig. 11. XBM specification.

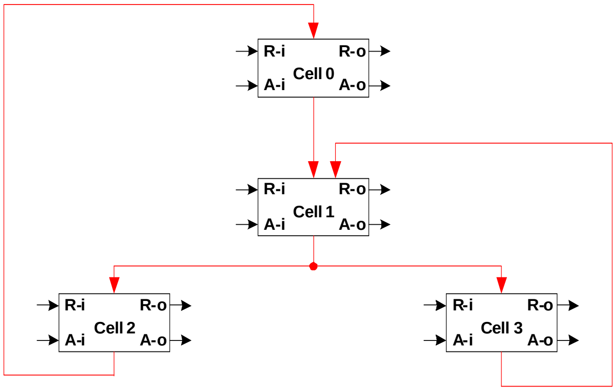


Fig. 12. Net of control cells: initial.

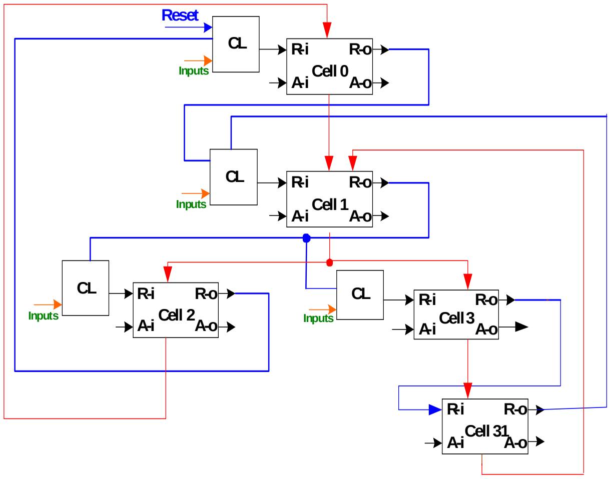


Fig. 13. Net of control cells: connections

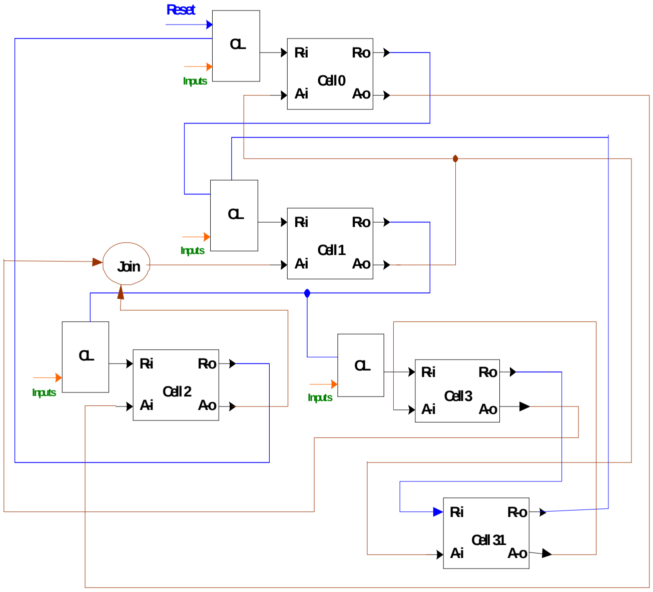


Fig. 14. Net: finals connections.

VI. Discussion

The direct mapping approach for synthesis of XBM_AFSMs shows interesting properties. The XBM_AFSMs allow operating in the I_b/O_b mode. Besides that, the proposed synthesis method requires no previous knowledge about logic synthesis algorithms and asynchronous concepts, as state assignment and logic minimization [27]. These kinds of algorithms generally require high computational effort, especially for large specifications, what is optimized by the use of the proposed method. The direct mapping approach consists of easy steps (algorithms), not requiring high computational effort, therefore, becoming easier the synthesizing of large specifications.

The proposed memory element was simulated as an example with the same previously topology of [26]. It was simulated in software QUARTUS II version 9.1, family CYCLONE III, in device EP3C25F324C6. The simulations didn't show any hazard problem and the circuit operated as predicted in its specification. Figure 15 shows the hazard-free waveforms extracted from the simulation of the benchmark Sbuf-send-pkt2-core controller synthesized in the section V. This example occupied 3% of the pins of this device (6), less than 1% of combinational LUTS (10) and 0% of memory LUTS. The latency time was 7.6 ns. Comparing the synthesis of this benchmark, to that one using the memory element of [26], it was obtained a reduction of 29% of LUTS.

They were compared also the area and latency time of the two memory elements, being simulated in Quartus II version 9.1 to the family CYCLONE III in device EP3C25F324C6. The memory element of [26] occupied two LUTs and the latency time was 8,10ns. The proposed memory element occupied only one LUT and presented a latency time of 6,88ns. It leads to a reduction of 50% in the area and 15% at the latency time. Fig. 15 and 16 show the compatibility between the generated circuit and the predicted functions.

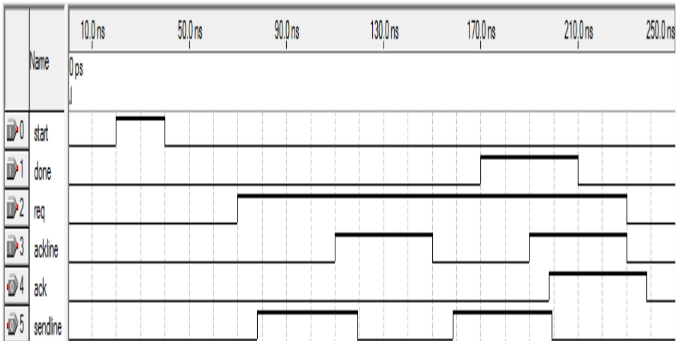


Fig. 15. Simulation results: sbuf-send-pkt2-core.

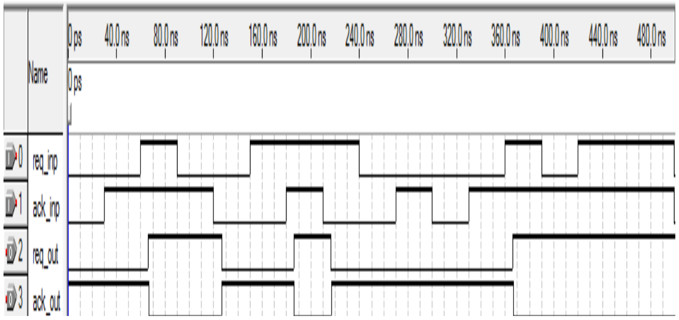


Fig. 16. Simulation results: memory element.

VII. Concluding Remarks

In this paper it was proposed a new memory element for direct mapping approach for synthesis of XBM_AFSMs. The direct mapping approach presents several advantages if compared with the logic synthesis approach, like very simple procedures and non-necessity of previous knowledge about hazard-free circuits theory and critical race. This approach allows synthesizing large specifications with very little computational effort. The synthesized XBM_AFSMs operate in mode I_b/O_b. Such circuits present superior performance when compared to generalized fundamental mode asynchronous synthesized circuits obeying the same delay model (bounded gate and wire delay). Theses circuits are faster because new input bursts are allowed sooner than in the alternative solutions (it is not necessary to wait for a stable state in order to accept a new input burst) while keeping a simple interface with the external world (good modularity). The new memory element was proved to improve the performance of the XBM_AFSMs and, through an example, showed to achieve similar results as the previous one.

As future work it is proposed the development of the direct mapping tool which will allow the testing of larger examples. Likewise, it is necessary to study the reduction of the memory element, a full-custom version and to propose a method for direct mapping full-custom

References

[1] S. B.Furber, “Breaking Step the Return of Asynchronous Logic”, *IEE Review*, July 1993, pp.159-162.

[2] C. J. Myers, “*Asynchronous Circuit Design*,” Wiley & Sons, Inc., 2004, 2^a edition.

[3] S. M. Nowick et. al, “The Design of a High Performance Cache Controller: A Case Study in Asynchronous Synthesis” *Integration, the VLSI Journal*, Vol. 15, no 3, pp. 241-262, October 1993.

[4] S. H. Unger, “Hazards, Critical Races, and Metastability”, *IEEE Transaction on Computer*, June 1995, Vol. 44:6, pp. 754-768.

[5] S. Hauck, “Asynchronous Design Methodologies: An Overview”, *Proc. of the IEEE*, January 1995, Vol. 83:1 pp.69-93.

[6] D. H. Huffman, “The Synthesis of Sequential Switching Circuits,” *J. Franklin Ins*., Vol. 257, pp. 161-190, March 1954 e pp. 275-303, April 1954.

[7] S. M. Nowick, “Automatic Synthesis of Burst-Mode Asynchronous Controllers,” Ph.D thesis, Stanford University, 1993.

[8] S. M. Nowick e Bill Coates, “UCLOCK: Automatic Design of High-Performance Unclocked State Machines,’ Proc. ICCD, pp. 434-441, 1994.

[9] K. Y. Yun, “Synthesis of Asynchronous Controllers for Heterogeneous Systems”, Ph.D. thesis, Stanford University, 1994.

[10] K. Y. Yun and D. L. Dill, “Automatic Synthesis of Extended *Burst*-Mode Circuits: Part I (Specification and Hazard-Free Implementation) and Part II (Automatic Synthesis),” *IEEE Trans. on CAD of Integrated Circuit and Systems*, Vol. 18:2, February, pp. 101-132, 1999.

[11] D. L. Oliveira, et al., “Miriã: a CAD toll synthesize multi-burst controllers for heterogeneous systems,” *Microelectronics Reliability*, 43 (2003) 209-213.

[12] T. –A. Chu, “On the Models for Designing VLSI Asynchronous Digital Systems,” *INTEGRATION, the VLSI Journal* 4, pp.99-113, 1986

[13] T. -A. Chu, “Synthesis of Self-Timed VLSI Circuits from Graph-Theory Specifications,” Ph.D. thesis, June, 1987, Dept. of EECS, MIT.

[14] A. J. Martin, “Compiling Communicating Process into Delay-Insensitive VLSI Circuits,” *Distributed Computer*, vol.1, no.3, pp.226-234, 1986.

[15] E. Brunvand e R. F. Sproull, “Translating Concurrent Programs into Delay-Insensitive Circuits,” Proc. ICCD, pp.262-265, 1989.

[16] Wei-Chum Chou, et. all, “Average-Case Technology Mapping of Asynchronous Burst-Mode Circuits,” *IEEE Trans. on CAD of Int. Circuits and Systems*, vol.18, N0. 10, October 1999, pp. 1418-1434.

[17] S. H. Unger, *Asynchronous Sequential Switching Circuits*, John Wiley & Sons Inc, 1969.

[18] R. David, “ Modular design of asynchronous circuit defined by graphs,” *IEEE Trans. Computer*, vol. C-26, no. 8, pp.727-737, August 1977.

[19] L. A. Hollaar, “Direct implementation of asynchronous control unit,” *IEEE Trans. Computer*, vol.C-31, no. 12, pp.1133-1141, Dec., 1982.

[20] D. Sokolov, A. Bystrov and A. Yakovlev, ”Direct mapping of low-latency asynchronous controllers from STGs,” *IEEE Trans. CAD of Integration Circuits and Systems*, vol. 26, no. 6, June 2007.

[21] J. Cortadella, et al., “Petrify: A tool for manipulating concurrent specifications and synthesis of asynchronous controllers,” *IEICE Trans. Inf. Syst*., vol.E80-D, no. 3, pp.315-325, March 1997.

[22] R. Fuhrer, et al., “Minimalist: An environment for the synthesis, verification and testability of burst-mode asynchronous machines,” Columbia Univ., NY, Tech. Rep. TR-CUCS-020-99, July 1999.

[23] K. Y. Yun e D. L. Dill, “A High-Performance Asynchronous SCSI Controller,” Proc. Int. Conf. Computer Design (ICCD), pp. 44-49, 1995.

[24] E. Pastor, et al., “Structural methods for the synthesis of speed-independent circuits,” *IEEE Trans. Computer-Aided Design*, vol. 17, pp. 1108-1129, November 1998.

[25] J. Carmona, “Structural methods for the synthesis of well-formed concurrent specifications,” PhD thesis, Software Dept., Universitat Politécnica de Catalunya, Barcelona, Spain, March 2004.

[26] D. L. Oliveira, et al., “Synthesis by Direct Mapping of Asynchronous Controllers from Extended Burst-Mode Specification,” XVI Workshop Iberchip, Foz de Iguaçu – Brazil, 2010.

[27] S. M. Nowick e D. L. Dill, “Exact Two-Level Minimization of Hazard-Free Logic with Multiple-Input Changes,” *IEEE Trans. on CAD of Integrated Circuits and Systems*, Vol. 14, no 8, August 1995.