Synthesis of Multi Bursts-Mode Controllers using Generalized C-Elements

Duarte Lopes de Oliveira[*], **Marius Strum**[+], **Wang Jiang Chau**[+], **Wagner C. Cunha**[*]

[*]**Departamento de Eletrônica Aplicada**
Instituto Tecnológico de Aeronáutica

Praça Marechal Eduardo Gomes, 50 - CEP 12228-900 - São José dos Campos, SP - Brazil

[+] **Laboratório de Microeletrônica**
Escola Politécnica da USP

Av. Prof. Luciano Gualberto, Trav 3, 158 - CEP 05508-900 - São Paulo, SP- Brazil

Resumo

Este artigo apresenta uma técnica de síntese automática para controladores assíncronos no modo multi-bursts, e tendo como alvo a arquitetura gC. Esta arquitetura manuseia eficientemente sinais sensíveis a nível (condicional) não monotônico, que são importantes quando projetamos circuitos assíncronos para sistemas heterogêneos. Uma importante característica de nossa solução é a sua capacidade de manusear entradas multi-bursts. Quando comparado com o controlador gC gerado pela ferramenta 3D, os nossos resultados mostram uma melhora em área e velocidade (medida como uma concorrência E/S estendida)

Palavras-chave: Circuitos assíncronos; risco; corrida crítica; elemento gC; síntese de controladores e sistemas heterogêneos.

Abstract

This paper presents an automatic synthesis technique for multi bursts-mode asynchronous controllers targeting the gC architecture. This architecture handles efficiently non-monotonic (conditional) level sensitive signals, which are important when designing asynchronous circuits for heterogeneous systems. An important feature of our solution is its capability of handling input multi-bursts. When compared to the gC controllers generated by the gC-3D tool, our experimental results show an improvement in area and speed (measured as an extended I/O concurrency).

Keywords: asynchronous circuits; hazard; critical race; gC element; synthesis of controllers and heterogeneous systems.

1. Introduction

Considerable interest has been given to asynchronous designs due to their potential advantages when compared to their synchronous counterparts: no clock skew, lower power consumption, larger modularization and more robustness against temperature variations and electromagnetic interactions. Their main disadvantage is the difficulty of their design, free from hazards and critical races [1-4].

A rich set of design tools and the design maturity favor the use of the synchronous paradigm. Even so, the interfaces among the system components do not strictly adhere to the synchronous paradigm because of the cost benefit of mixing modules operating at different clock rates and modules with asynchronous interfaces. (Extended) burst-mode asynchronous controllers present excellent features to be used as an interface between these two worlds.

Three techniques were proposed for the synthesis of (extended) burst-mode asynchronous controllers: Minimalist [20], 3D [5-8] and Miriã [15,17,18]. In all cases the target architecture is based on basic logic gates (AND, NAND, OR, NOR, INV).

The Minimalist tool synthesizes controllers starting from a burst mode specification [19].

The 3D tool generates starts from an extended burst mode (**XBM**) specification generating extended Huffman machines. In XBM, state transitions may be described in terms of: 1) edge sensitive signals, ESS; 2) level sensitive signals, LSS, and 3) directed don't-care signals, DCS. When DCS are present, I/O concurrently may occur.

The Miriã tool generates controllers as gRS architectures. It starts from a multi-burst graph **(MBG)** specification, that extends the XBM specification allowing to describe OR causality and concurrency between pairs of state transitions [15].

Minimalist and 3D, were used to produce a few high performance circuits, like Cache [10] and SCSI [11]. However, when the objective is to synthesize circuits whose specifications contain strict requirements, such as the differential equation solver [13] and portions of Intel's RAPPID [14], it was found that two-level circuits may be inefficient [21]. A substantial performance improvement can be achieved through the use of architectures based on complex gates.

Three techniques were proposed for the synthesis of (extended) burst-mode asynchronous controllers, targeting such architectures.

The technique proposed in [23,24] starts from the BM specification, extracts sum-of-product functions and implements them using CMOS complex gates. However, it has been shown that the resulting circuits were not logical hazard free [21]. Kudva solved this problem [12] by adopting the requirements stated in [19] during the synthesis procedure. Yun improved the controllers' performance by replacing the complex gates by the gC architecture composed of generalized C-elements, shown as a pair of inverters in figure 1, and transistor stacks for the corresponding *set* and *reset* functions [9, 21].

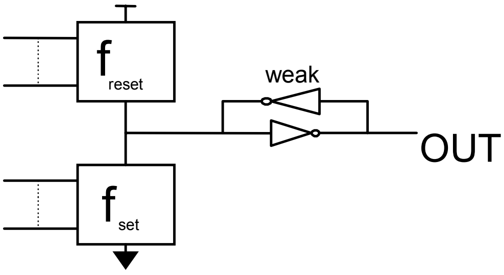


Figure 1 – The gC Architecture

In this article we present a modification of the Miriã tool that allows to synthesize gC architectures (instead of gRS architectures) for multi-burst applications. Two **operators,** CONCURRENCE and OR [15] may be used to describe special relations between pairs of bursts, which accounts for the multi-burst operation. Multi-burst operation enhances the performance of asynchronous controllers [16].

In section 2 we present an example containing multi-bursts. In section 3 we explain our synthesis methodology. Section 4 presents our experimental results. Finally in section 5 we present our conclusions.

2. Multi-Burst-Graph Description

Figure 2 illustrates the MBG specification of a circuit with four inputs [a,b,c,d] and 2 outputs [x,y].

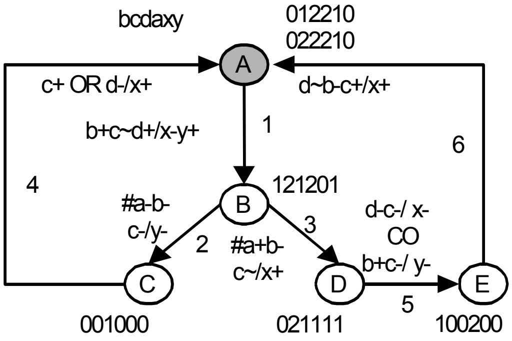


Figure 2 - A MBG specification including 2 burst operators.

The initial state is **A**. Multi-burst transitions 4 and 5 contain the OR and the CO operators respectively. The notation b+ (b–) means the rising (falling) edge of signal b. The bar (/) is used to delimit each input burst. Input signal *a* is a LSS while all others are ESS. Thus, #a–b–c–/y- means that if (a=0 AND b: 1→0 AND c:1→0) then the output will be (y: 1→0). The OR operator generates a super-state (A in figure 2). The OR operator in arc 4 means that if (EITHER c+ OR d- OR ELSE c+ AND d-) then the output will be x+. The CO operator in arc 5 means that the transitions (c– AND d– /x–) and (b+ AND c– /y–) may occur concurrently.

A special type of input signal, namely undetermined value signal has been introduced into the MBG specification. It consists of ESS, which are not active in a state transition, but their values are unknown at that transition. Signal c in Figure1 has an unknown value during the state transition 1, which is denoted as c~.

3.- The Synthesis Methodology

The Miriã synthesis tool performs three steps [18]:

1. Transformation of the MBG into an intermediate format (set of **transition cubes, TC** [6, 7]).
2. State variable optimization. This step is performed solving three tasks:
   a. State conflicts analysis.
   b. Determination of the minimum classes of compatible states [17].
   c. State assignment [4].
3. Logic minimization of the *Set* and *Reset* functions for each non-input signal. This step is performed solving four tasks [22]:
   a. Trigger cubes extraction.
   b. Intersection cubes extraction.
   c. Violation cubes extraction.
   d. Minimum cover (binate and unate).

Due to the functional difference between a RS latch and a C element, two tasks were modified in the Miriã tool to generate the gC architecture: the conflict analysis (2a) and the intersection cube extraction (3b).

As in the gC-3D tool, the conversion of the Set and Reset logic functions into stacks of transistors, and the analysis of the need to insert delays to avoid essential hazards, are not part of our automatic procedure.

3.1 Generating the intermediate format

Initially the MBG specification is transformed into a table of transition cubes (**TC**). A TC is a cube representing all paths in the state transition table, corresponding the set of sub-cubes that cover each state transition of an MBG [6, 7]. When no burst operator is present there are two sub-cubes, that we call I_T (representing the activation of the input signals) and O_T (representing the activation of the output signals). When an OR operator is present there are 3 sub-cubes: one I_T and 2 O_T [15]. When a state transition follows another that contains an OR operator, there are 3 sub-cubes: 2 I_T and 1 O_T. When a concurrent operator is present there are 4 sub-cubes: one I_T and 3 O_T [15]. The sizes of I_T and O_T are proportional to the number of (input + output) signals. Figure 3 shows the TC table for the example in figure 2, were ‘2’ means don't-care and ‘3’ means undetermined.

ARC	Transition Cubes
1	232210+212210+131222
2	221001+001002
3	231101+031121
4	022200+012220+020220
5	222211+102222 /10+ 200222 /01+100222 /00
6	223200+013220

Figure 3 Table of TC

3.2 State Variable Optimization

States transitions are not compatible (conflict situation) if any common intermediate state either present the different outputs or they do not satisfy the requirements presented in [19].

Two conditions, described in figure 4, that cause conflicts when synthesizing a gRS architecture, are not present in the gC architecture because, they are filtered by the C element [9].

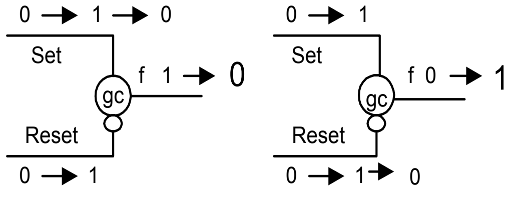
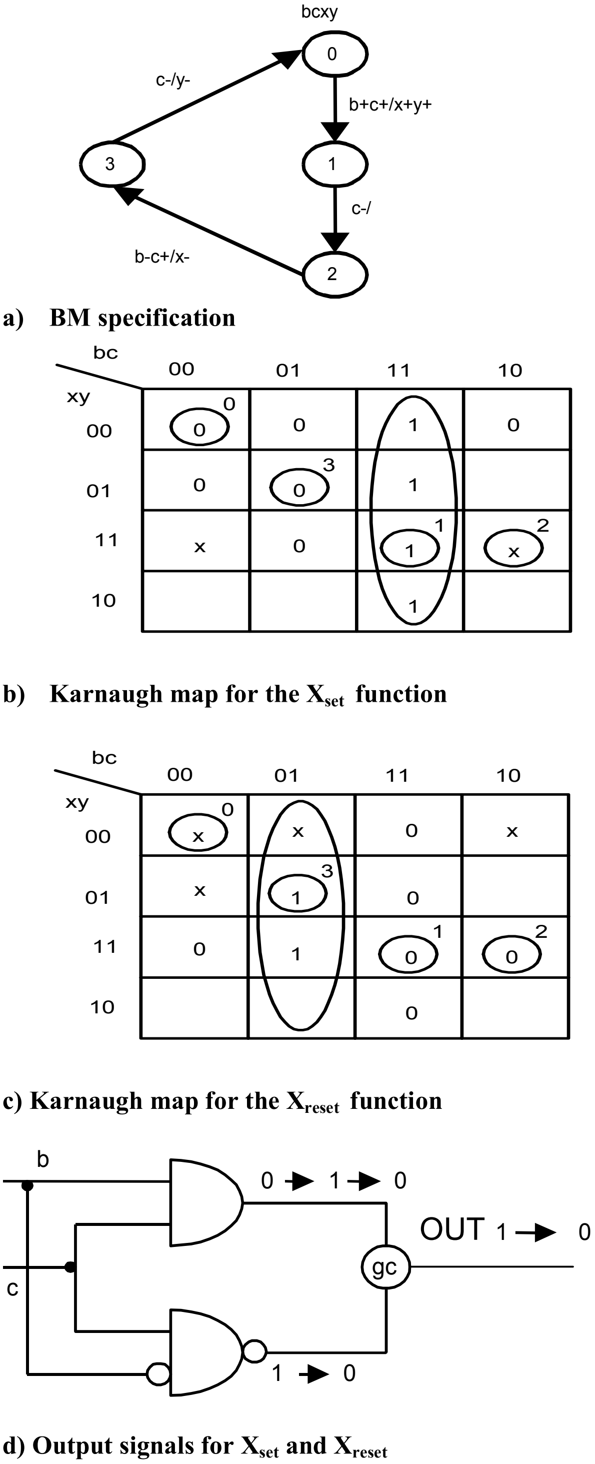


Figure 4 Transitions allowed in the element C

Consider the specification shown in figure 5a. Figures 5b and 5c show the Karnaugh maps of the X_{set} and X_{reset} output functions. Consider the state transitions 0➔1 and 2➔3. The input transition cube bcxy=2211 intercepts the output transition cube bcxy=1122 at the bcxy=1111 intermediate state. As cube 1122 does not cover the initial state 2, a conflict condition arises (according to [5,19]. The glitch that results from this conflict in the gRS architecture is shown in figure 5d. In order to avoid such a conflict, a state variable has to be added. This conflict is filtered (avoided) by the C element.



a) BM specification

xy \ bc	00	01	11	10
00	0^{0}	0	1	0
01	0	0^{3}	1	
11	x	0	1^{1}	x^{2}
10			1	

b) Karnaugh map for the X_{set} function

xy \ bc	00	01	11	10
00	x^{0}	x	0	x
01	x	1^{3}	0	
11	0	1	0^{1}	0^{2}
10			0	

c) Karnaugh map for the X_{reset} function

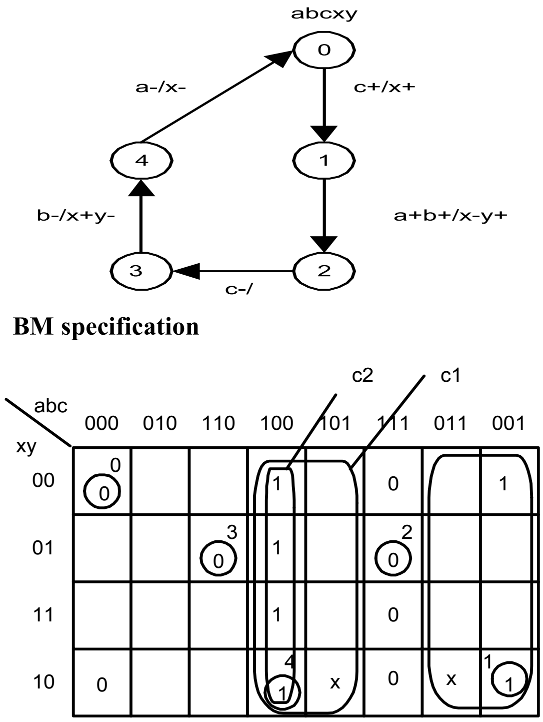
d) Output signals for X_{set} and X_{reset}

Figure 5 Example of a glitch due to an unsolved conflict condition

Whenever state variables are introduced, steps 2b and 2c must be performed in the same way as presented in [18]. One estate variable was needed for the example in figure 2.

3.3 Logic minimization

The ATACS algorithm presented in [25] has been adapted to minimize the set and reset functions for Miriã. However, the intersection cubes extractions task had to be modified. Consider the BM specification of figure 6a. There are no states conflicts. Figure 6b shows the Karnaugh map for the Xset function. If cube c1 is used to cover the 3➔4 transition, a glitch would propagate through the gRS architecture. This is due to the fact that c1 covers the abcxy=10110 intermediate state, but does not cover the abcxy=00110 intermediate state which is the initial state of the 1➔2 transition. In order to avoid the glitch, cube c2 has to be used. This problem does not arise in the gC architecture because the C element filters the glitch that would be produced. State abcxy=10110 is an intersection cube for gRS but not for gC.



a) BM specification

xy \ abc	000	010	110	100	101	111	011	001
00	0^{0}			1		0		1
01			0^{3}	1		0^{2}		
11				1		0		
10	0			1^{4}	x	0	x	1^{1}

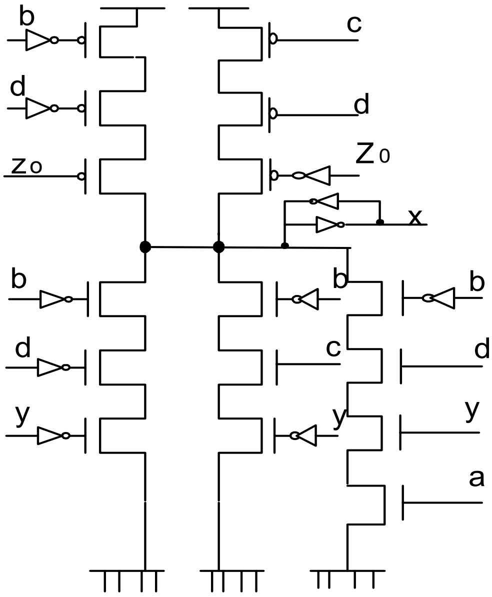
b) Karnaugh map for x (X_{setr}) output signal

Figure 6 Violation of the covering requirement

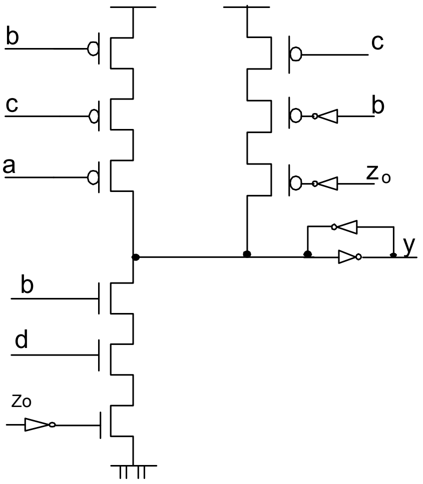
After logic minimization, the *SET* and *RESET functions* for the example in figure 2 are:

$$\mathbf{X_{set} = b'\ d'\ y' + b'\ c\ y' + b'\ d\ a\ y}$$
$$\mathbf{X_{reset} = b\ \ d\ \ Zo' + c'\ d'\ Zo}$$
$$\mathbf{Y_{set} = b\ \ d\ \ Zo'}$$
$$\mathbf{Y_{reset} = b\ c\ '\ Zo + \ b\ c\ a}$$
$$\mathbf{Zo_{set} = b'\ \ d\ \ y\ \ a}$$
$$\mathbf{Zo_{reset} = b'\ c\ \ y'}$$

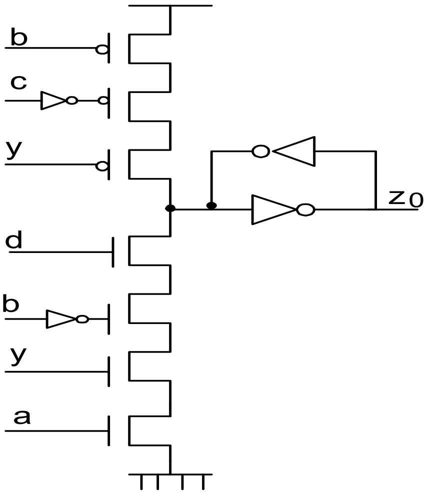
Figure 7 shows the transistor stacks for each signal.



a) Transistor stack of the x signal



b) Transistor stack of the y signal



c) Transistor stack of the z0 signal

Figure 7 Transistor stacks for example in figure 2.

4. Results and Discussion

In this section we compare our results with those obtained by the Gc-3D tool. The comparison is made in terms of state transitions, number of literals and products.

4.1 Multi-burst Controllers with CO operator

Tables 4.1 and 4.2 show the gC-3D and Miriã synthesis results of 15 known benchmarks (that do not present any LSS) and for the Scsi-targ-send benchmark (that contains two LSS).

Miriã produced 20% less state transitions than gC-3D, due to the presence of the CO operator. However, the area performance of the Miriã tool, was worse than the gC-3D (17% for the number of literals and 38% for the number of state variables) which can be explained as the price to pay for a larger degree of concurrence. For the number of products the performance of the Miriã tool was best than the gC-3D in 3%.

The best results occurred for the Call-procedure macro-module [25] (50% increase in concurrency, 5% increase in the number of literals and 72% reduction in the number of products) and Pe-send-ifc benchmark from HP [23] (7% increase in concurrency and 17% reduction in the number of literals).

	XBM Specification		gC-3D	
	Int/Out	State /Tran	State Var	Lit / Prod
Alloc-Outbound	4/3	8/9	3	28/8
Call-procedure	3/3	12/16	0	32/12
Chu150	3/3	5/5	0	14/5
Dramc	7/6	12/14	0	38/9
It-Control	5/7	9/11	1	48/15
Nak-Pa	4/5	6/6	1	26/7
Nowick	3/2	6/6	1	12/3
Pe-HP	5/5	7/7	2	23/8
Pe-Send-Ifc	5/3	11/14	2	57/14
Pipeline-Hand	2/2	4/4	0	5/1
Pscsi-Ircv	4/3	6/7	2	26/4
Pscsi-Trcv-bm	4/4	7/9	2	45/8
Re-Setup	3/2	6/6	1	9/3
Scsi-Targ-Send	6/4	11/13	2	70/14
Stetson-P3	4/2	8/11	0	8/2
Two-Ticks-If	4/3	7/10	0	8/2
Total	-	125/148	17	449/115

Table 4.1 Results of the gC-3D tool

	MBG Specification		gC-Miriã	
	Int/Out	State /Tran	State Var	Lit / Prod
Alloc-Outbound	4/3	7/8	3	28/6
Call-procedure	3/3	4/8	2	34/4
Chu150	3/3	4/4	1	20/6
Dramc	7/6	8/10	2	62/10
It-Control	5/7	8/10	1	48/13
Nak-Pa	4/5	5/5	1	26/6
Nowick	3/2	5/5	1	18/6
Pe-HP	5/5	5/5	2	33/10
Pe-Send-Ifc	5/3	10/13	2	45/14
Pipeline-Hand	2/2	2/2	1	6/0
Pscsi-Ircv	4/3	5/6	1	21/4
Pscsi-Trcv-bm	4/4	6/8	2	46/9
Re-Setup	3/2	5/5	2	17/4
Scsi-Targ-Send	6/4	10/12	2	77/12
Stetson-P3	4/2	7/10	2	17/3
Two-Ticks-If	4/3	5/8	2	18/5
Total	-	96 /119	27	516 /112

Table 4.2 Results of the gC-Miriã tool

4.2 Multi-burst Controllers with OR operator

The OR condition, as described in section 2, is not present in any of the known benchmarks. Therefore we created 4 new benchmarks (EX1-1-or, EX2-2-or, EX3-3-or and EX4-2-OR) in which such behavior occurs. We described these benchmarks using both, XBM and MBG. Figures 8 and 9 show these descriptions for EX1-1-or benchmark. Tables 3 and 4 show the synthesis results, for the gC-3D and the Miriã tools, of these benchmarks, as well as of the example in figure 2. The concurrency increase is demonstrated by a reduction of 65% in the number of state transitions (78 in the table 3 and 28 in the table 4). The area results for the Miriã tool were also better: 10% reduction in the number of literals, 15% reduction in the number of products and 20% reduction in the number of state variables.

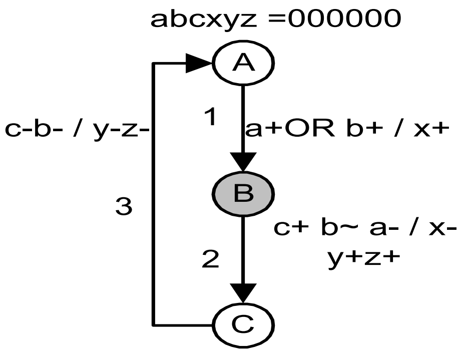


Figure 8 - MBG specification - Ex1-1-or

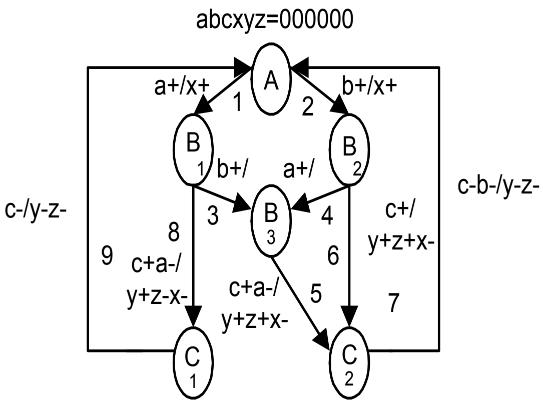


Figure 9 - XBM specification – Ex1-1-or

	XBM Specification		gC-3D	
	Int/ Out	State/ Tran	State-Var	Lit / Prod
EX1-1-OR	3/3	6/9	0	14/3
EX2-2-OR	4/3	9/16	1	23/4
EX3-3-OR	4/3	13/23	3	57/14
EX4-2-OR	4/3	10/15	0	40/10
Figure 2	4/2	10/15	1	30/9
TOTAL	--	48/78	5	164/40

Table 4.3 gC-3D results

	MBG Specification		gC-Miriã	
	Int/ Out	State/ Tran	State-Var	Lit / Prod
EX1-1-OR	3/3	3/3	0	14/3
EX2-2-OR	4/3	5/6	2	35/7
EX3-3-OR	4/3	6/7	1	34/8
EX4-2-OR	4/3	5/6	0	41/10
Figure 2	4/2	5/6	1	32/7
TOTAL	--	24/28	4	156/35

Table 4.4 gC-Miriã results

5. Conclusion

This paper presented the modifications introduced in the Miriã design tool [18], required to synthesize multi-burst asynchronous controllers targeting the gC architectures Instead of the original gRS architecture). Two modified synthesis tasks, conflict analysis and intersection cube extraction were explained and illustrated. The experimental results show an improvement of the resulting machine's performance compared to the gC-3D solution in terms of I/O concurrency and area.

References

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

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

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

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

[5] Kenneth Y. Yun, "Synthesis of Asynchronous Controllers for Heterogeneous Systems", Ph.D. thesis, Stanford University, 1994.

[6] Kenneth Y. Yun and D. L. Dill, "Automatic Synthesis of Extended *Burst-Mode* Circuits: Part I (Specification and Hazard-Free Implementation)," IEEE Trans. on CAD of Integrated Circuit and Systems, Vol. 18:2, February 1999, pp. 101-117.

[7] Kenneth Y. Yun and D. L. Dill, "Automatic Synthesis of Extended *Burst-Mode* Circuits: Part II (Automatic Synthesis)," IEEE Trans. on CAD of Integrated Circuit and Systems, Vol. 18:2, February 1999, pp. 118-132.

[8] Kenneth Y. Yun and D. L. Dill, "Unifying synchronous /asynchronous state machine synthesis," In Proc. of the IEEE/ACM International Conference on CAD, November 1993, pp. 255-260.

[9] Kenneth Y. Yun, "Automatic synthesis of extended burst-mode circuits using generalized C-elements," in Proc. European Design Automation Conference, September 1996, pp. 290-295.

[10] S. M. Nowick, M.E. Dean, D. L. Dill, and M. Horowitz. "The design of a high-performance cache controller: case study in asynchronous synthesis," INTEGRATION, the VLSI journal, 15(3):241-262, October 1993.

[11] Kenneth Y. Yun and D. L. Dill, "A high-performance asynchronous SCSI controller," In ICCD, pp. 44-49, October 1995.

[12] P. Kudva, G. Gopalakrishman, H. Jacobson, and S. M. Nowick. "Synthesis of the hazard-free customized CMOS complex-gate networks under multiple-input changes," In Proc. ACM/IEEE Design Automation Conference IEEE Computer Society Press, June , 1996

[13] Kenneth Y. Yun, Peter A. Beerel, Vida Vakilotojar , Ayoob E. Dooply, and Julio Arceo, "The design and verification of a high-performance low-control-overhead asynchronous differential equation solver," IEEE Transactions on VLSI Systems, vol. 6, no 4, pp.643-655, Dec.1998.

[14] Shai Rotem, Ken Stevens, Ran Ginosar, Peter A. Beerel, Chris Myers, Kenneth Y. Yun, Rakefet Kol, Charles Dike, Marly Roncken, and Boris Agapiev, "RAPPID: An asynchronous instruction length decoder,"in Proc. International Symposium on Advanced Research in Asynchronous Circuits and Systems, April, 1999, pp.60-70.

[15] D. L. Oliveira, M. Strum and W. C. Cunha, "Grafo Multi-*Burst*: Uma Especificação para Máquinas de Estado Assíncronas," Proc. do VI Workshop Iberchip, March 2000, São Paulo, Brazil, pp. 371-380.

[16] D. L. Oliveira, M. Strum and W. C. Cunha, "GMB-Direto: Um novo método de sintese de Máquinas de Estado Assíncronas," Proc. do VI Workshop Iberchip, March 2000, São Paulo, Brazil, pp. 381-390.

[17] D. L. Oliveira, M. Strum, W. J. Chau and W. C. Cunha, " Um Novo Método para Minimização de Máquinas no Modo Burst,", Proc. do VII Workshop Iberchip, March 2001,Uruguay, (midia eletrônica).

[18] D. L. Oliveira, M. Strum, W. J. Chau and W. C. Cunha, " Synthesis of Multi-Burst Controllers as Generalized RS Architecture for Heterogeneous Systems,", Proc. do VIII Workshop Iberchip, March 2002,Mexico, (midia eletrônica).

[19] S. M. Nowick, "Automatic Synthesis of *Burst-Mode* Asynchronous Controllers," Ph.D. thesis, Stanford University, 1993.

[20] R. M. Fuhrer, S. M. Nowick, "MINIMALIST: An Environment for the Synthesis and Verification of *Burst*-Mode Asynchronous Machines," in Proc. IEEE/ACM Int. Workshop Logic Synthesis , 1998.

[21] Kevin W. James and Kenneth Y. Yun, "Average-case optimized transistor-level technology mapping of extended burst-mode circuits," In Proc. 4th Symp. Advanced in Asynchronous Circuits and Systems, 1998, pp.70-79.

[22] Hans Jacobson, Chris Myers, and Ganesh Gopalakrishnan, "Achieving Fast and Exact Hazard-Free Logic Minimization of Extended Burst-Mode gC Finite State machines, " Proc. ICCAD, November, pp.303-310, 2000.

[23] A. Davis, B.. Coates, and K. Stevens, "The post office experience: designing a large asynchronous chip," In Proc. of the 26th annual Hawaiian int. conf. System sciences, volume 1, January, 1993.

[24] K. S. Stevens, " Practical Verification and Synthesis of Low Latency Asynchronous Systems", Ph.D. thesis, University of Calgary, Alberta, September 1994.

[25] G. Gopalakrishnan, P. Kudva, and E. Brunvand, "Peephole Optimization of Asynchronous Macromodule Metworks", IEEE Trans. On Very Large Scale Integration (VLSI) Systems, vol. 7, Nro 1, March 1999.