Hindawi Publishing Corporation
International Journal of Reconfigurable Computing
Volume 2008, Article ID 926851, 10 pages
doi:10.1155/2008/926851

Research Article

Burst-Mode Asynchronous Controllers on FPGA

Duarte L. Oliveira,[1] Marius Strum,[2] and Sandro S. Sato[1]

[1] *Electronic Engineering Division, Aeronautics Institute of Technology, Praça Marechal Eduardo Gomes 50, 12228-900 São José dos Campos, SP, Brazil*
[2] *Microelectronic Laboratory, Polytechnic School, University of São Paulo, Avenida Prof. Luciano Gualberto, Trav 3, 158, 05508-900 São Paulo, SP, Brazil*

Correspondence should be addressed to Duarte L. Oliveira, duarte@ita.br

Received 5 July 2008; Revised 8 October 2008; Accepted 30 October 2008

Recommended by Gustavo Sutter

FPGAs have been mainly used to design synchronous circuits. Asynchronous design on FPGAs is difficult because the resulting circuit may suffer from hazard problems. We propose a method that implements a popular class of asynchronous circuits, known as burst mode, on FPGAs based on look-up table architectures. We present two conditions that, if satisfied, guarantee essential hazard-free implementation on any LUT-based FPGA. By doing that, besides all the intrinsic advantages of asynchronous over synchronous circuits, they also take advantage of the shorter design time and lower cost associated with FPGA designs.

1. Introduction

Due to the increasing complexity of digital systems combined with the market drive for higher performance, there has been an increased interest about asynchronous circuits [1, 2]. Asynchronous circuits do not present clock distribution related problems like clock skew. The circuits have low power consumption, better modularity, robustness toward variations in temperature, and low emission of electromagnetic radiation [3]. One known weakness of asynchronous circuits has been the difficulty to design hazard-free circuits and to solve the critical races [3]. Furthermore, asynchronous circuits frequently cannot benefit from the use of FPGAs due to the extra difficulty imposed by their fixed architecture to deal with hazards [4].

Asynchronous circuits can be classified according to different criteria like its function (controller—datapath); delay model (delay insensitive—quasi-delay insensitive—speed independent—generalized fundamental mode (GFM)) [2]; styles (global asynchronous local synchronous—self-timed systems—micropipeline—speed-independent controllers—burst-mode controllers) [5–9].

Burst-mode asynchronous controllers proposed by Nowick [9, 10] are a popular class of finite state machines. They allow multiple inputs changes. They operate according to the GFM, meaning that a new state transition may only start when the whole circuit (gates and lines) is stable. This paper addresses burst-mode asynchronous controllers. Their advantages are the use of basic gates, similarity with synchronous design. These controllers have been adopted in important industrial and academic designs [11–13].

FPGAs are popular components for prototyping and production of digital circuits due to their low cost and short design time. Their focus has been on synchronous digital circuits. There have been some recent efforts to prototype asynchronous circuits on both commercial [14–17] and academic FPGAs [4, 18–20].

Burst-mode controllers are usually designed using a logic-driven design methodology [21]. There are two reasons why off-the-shelf FPGAs are not fit for burst-mode asynchronous controllers [4, 14, 22].

(1) The *mapping process* of burst-mode Booleans functions (equations of next state—controllers) to logic blocks (macrocells) may introduce *logic hazards*.

(2) The *internal routing* among logic blocks may introduce significant delays that may result in *essential hazards*.

1.1. Avoiding Logic Hazards in Burst-Mode Controllers

The burst-mode specification proposed by Nowick is *functional-hazard free* [23]. Nowick also proposed a method to produce *logic-hazard free* burst-mode Boolean functions [24]. Furthermore, Siegel et al. [25] proposed a technique to decompose large fan-in burst-mode Boolean functions without introducing *logic-hazards*. Finally, Maheswaran and Akella [15] and Hauck et al. [4] showed that if Booleans functions are *functional-hazard free* then they can be mapped on ordinary LUT-based FPGAs without presenting *logic hazards* [26].

1.2. Avoiding Essential Hazards in Burst-Mode Controllers

Yun and Dill [27] and Nowick and Coates [10] proposed the insertion of delay elements on the feedback wires to avoid essential hazards in burst-mode controllers. However, this solution is not adequate for FPGAs because these components are not designed to ease the insertion of delay elements. Furthermore, delay elements degrade the circuit cycle time, area, and reliability.

In this paper, we demonstrated a *sufficient condition* that guarantees *essential hazard-free operation* of *any type of burst-mode controller* when mapped on *any type of LUT-based FPGA* component without the need of extra delay elements. The proof is based on two new concepts: (1) *essential signals*; (2) *essential super states*. The essential hazard-free operation is guaranteed if the following conditions are satisfied:

(1) essential hazard-free specification: for all state transitions in a burst-mode specification, if the label contains a nonempty output burst, it must also contain at least one *essential* input signal;

(2) essential hazard-free implementation: starting from an essential hazard-free specification, while building the burst-mode flow map, all *single states* whose incident state transitions are labeled with nonempty output bursts must be transformed into *essential super states*.

Furthermore, whenever a burst-mode specification does not satisfy the first condition, we present two *functional transformations* that *create essential input signals* without altering the original functionality:

(1) reduction of input concurrency: transforms concurrent transitions into sequential transitions whenever acceptable (but there is a latency penalty);

(2) addition of dummy input signals (but there is an area penalty).

2. Hazard-Free BM Conditions

This paper is divided in four sections. Section 3 briefly explains the burst-mode specifications. Section 2 presents the essential signal and essential super-state concepts and explains the two functional transformations. Section 4 presents our method and illustrated with an example. Section 5 shows our experimental results presenting the latency and area penalties found on nine known and one homemade benchmark. Section 6 presents our conclusions and future work.

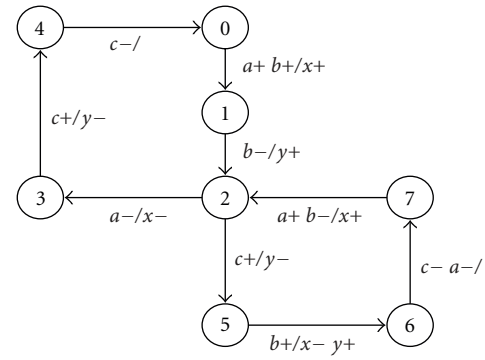

Figure 1: BM specification.

$a\,b\,c$ / $x\,y$	000	010	110	100	101	111	011	001
00	(0) 00	00	10	00	–	01	–	(4) 00
01	(3) 01	(7) 01	01	11	–	(6) 01	01	00
11	01	–	–	(2) 11	10	01	–	–
10	–	–	(1) 10	11	(5) 10	01	–	–

Figure 2: BM flow map.

3. Burst-Mode Specification

The BM specification is represented as a state transition diagram. Each transition is triggered by an input burst (single- or multiple-input changes) causing the occurrence of an output burst (that may be empty or nonempty). It is necessary to define an initial state. State transitions are represented by arcs, which are labeled with their corresponding input/output bursts. The signals are always transition sensitive ($0 \to 1$, or $1 \to 0$). Input bursts may not be empty. The input signals are monotonic, changing only once during each state transition. The BM specification has to obey the polarity property, the unique entry point and the maximal set property [23].

Figure 1 shows a BM specification. The input signals are a, b, and c while the output signals are x and y. For example, state transition $7_{[a+\,b-/x+]} \to 2$ means that if a changes from 0 to 1 and b changes from 1 to 0, the output x will change from 0 to 1. State 0 is the initial state. Figure 2 shows the corresponding burst-mode flow map (2D map) [27]. Several tools, like Minimalist [28], 3D [27], and ATACS [29] have been proposed to synthesize controllers from a textual description of the burst-mode specification. These

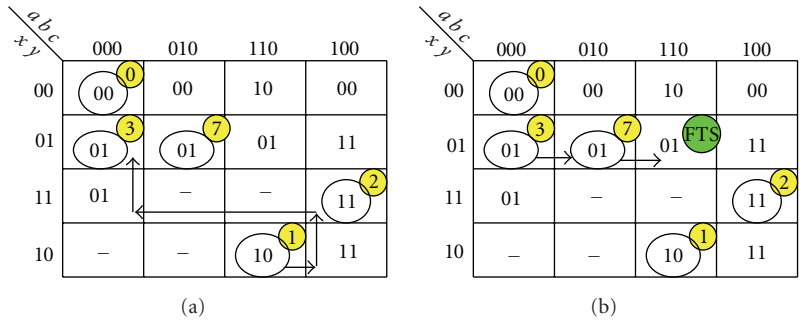

Figure 3: Part of the BM flow map of Figure 2: (a) path T2; (b) path T3 (final total state—FTS).

tools generate an independent *netlist* of the technology (next-state equations of the type sum of products).

BM asynchronous controllers may be subject to sequential hazards. Essential hazards, like transient essential hazard or steady-state essential hazard, are inherent to the sequential function and are not necessarily associated to a particular implementation of the circuit. The concept of essential hazard has been originally defined by Unger [30] in connection with fundamental-mode controllers.

This concept has been generalized for BM controllers and may be explained using the total state concept. A total state $A(I, O)$ is a vector composed of all the input (I) and output (O) signal values in the specification. A total state corresponds to one single cube (cell) on a burst-mode flow map (see Figure 2). For example, the total state 2 of the Figure 2 is $(a, b, c, x, y) = 10011$. There may be $n!$ paths on the BM flow map corresponding to the transition from total state $A(I_1, O_1)$ to total state $B(I_2, O_2)$ (labeled with an input/output burst I_b/O_b), n being the number input signals in I_b. These paths cover the set $\{A, B, C, \ldots, N\}$, where A is the initial total state, B is the final total state, and $C, \ldots, N$ are intermediary total states.

3.1. Essential Hazard

Generalized Unger Rule [30] (GUR): the Triple Sequential Input Burst

Let $A(I_1, O_1)$ and $B(I_2, O_2)$ be two total states in the BM flow map and I_b/O_b the input/output burst that activates the transition $A \rightarrow B$. Let N be the number of the input burst signals. Consider the following transitions sequence:

T1: $A_{[Ib]} \rightarrow B$; (transition 1 is $A \rightarrow B$ activated by I_b)

T2: $B_{[Ib\_\text{inverted\_polarity}]} \rightarrow C_i$ ($i = 1, \ldots, k$ are possible final states);

T3: $C_{[Ib]} \rightarrow D_j$ ($i = 1, \ldots, n$ are possible initial states), ($j = 1, \ldots, m$ are possible final states).

Definition 1. There is a potential *steady-state essential hazard* in the $A \rightarrow B$ transition if, applying the GUR rule, any final total state D_j ($j = 1, \ldots, m$) $\neq B$.

Definition 2. There is a potential *transient essential hazard* in the $A \rightarrow B$ transition if, applying the GUR rule, there is a total state I ($\neq A$ and $\neq B$) on any path of transitions $B \rightarrow C_i$ or $C_i \rightarrow D_j$ that produces an output signal different from any value occurring on any path of transition $A \rightarrow B$.

A potential essential hazard can be detected applying the GUR rule from any initial state. For example, Figures 3(a) and 3(b) show two paths for the $0 \rightarrow 1$ state transition on the BM flow map of Figure 2. Consider the $0_{[b+a+/x+]} \rightarrow 1$ path on Figure 3(a). According to the GUR rule we must apply the following activation sequence: T1($b + a+$), T2($b - a-$), T3($b + a+$). The corresponding paths on the BM flow table are

T1: $abcxy = \{00000 \rightarrow 01000 \rightarrow 11000 \rightarrow 11010\}$,

T2: $abcxy = \{11010 \rightarrow 10010 \rightarrow 10011 \rightarrow 00011 \rightarrow 00001\}$,

T3: $abcxy = \{00001 \rightarrow 01001 \rightarrow 11001\}$.

As the final total state (11001) after the last activation (T3) is different from the final total state (11010) after the first activation (T1), then a steady-state essential hazard has occurred. Figure 3(a) shows the path T2 and Figure 3(b) shows the path T3.

3.2. BM-EHF Condition

An input signal in a BM specification is a *context signal* in an $A \rightarrow B$ transition if it does not change during this transition (it is not on the label) while it is a *trigger signal* if it is labeled during this transition. The input burst of each state transition can be represented by an input transition cube (ITC). For example, the ITC for state transition $7 \rightarrow 2$ on Figure 1 is $abc = 220$ (2 means do not care). In this example a and b are trigger signals while c is a context signal (whose value is 0).

Definition 3. Let A and B be a pair of total states in a BM specification and I_b/O_b be the input/output burst for the $A \rightarrow B$ transition. Let E_s be one input signal ($E_s \in I_b$). E_s is an *essential signal* if it is a context signal on all transitions incident on state A and is a trigger signal on the transition $A \rightarrow B$.

For instance (see Figure 1), a, b, c are not essential on transitions $4 \rightarrow 0$, $1 \rightarrow 2$, and $2 \rightarrow 3$ because they are trigger

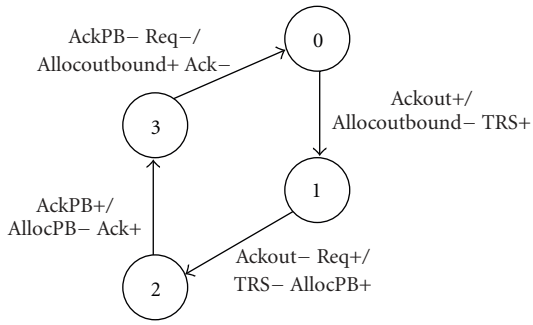

Figure 4: BM-EHF specification.

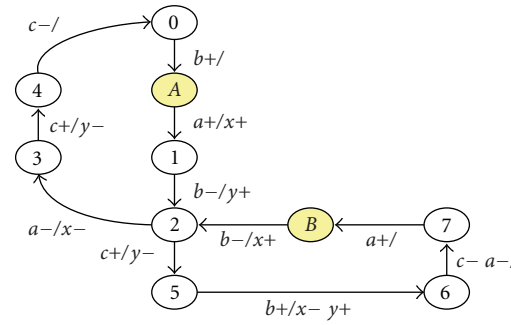

Figure 5: Concurrency reduction.

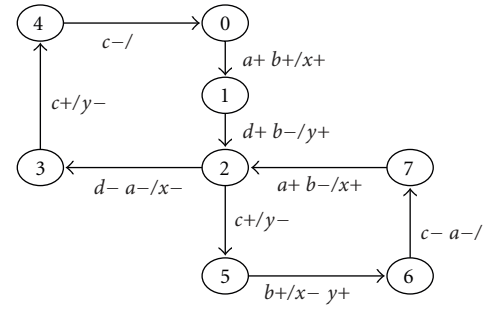

Figure 6: Inserting essential signal.

signals on transitions $3 \rightarrow 4$, $0 \rightarrow 1$, and $7 \rightarrow 2$. Signal b is essential on transition $7 \rightarrow 2$ because it is a context signal on transition $6 \rightarrow 7$. On transition $6 \rightarrow 7$ both a and c are essential signals.

Lemma 1. *A BM specification is essential hazard free (BM-EHF) if and only if for each state transition labeled by I_b/O_b, if $O_b \neq \varnothing$, there must be at least one essential input signal.*

Proof. Let T1$(A \rightarrow B)$ and T2$(B \rightarrow C)$ be two sequential state transitions of a BM-EHF specification. $\mathrm{ITC}_{\mathrm{T1}}$ and $\mathrm{ITC}_{\mathrm{T2}}$ are their respective input transition cubes. Suppose that the transition T2 input burst does not contain as essential signal. Then $\mathrm{ITC}_{\mathrm{T1}} \subseteq \mathrm{ITC}_{\mathrm{T2}}$, which means that the C final total state belongs to a path on $\mathrm{ITC}_{\mathrm{T1}}$. This fact violates Definitions 1 or 2. □

Figure 4 shows the *HP-mp-for-pkt* benchmark [12, 13]. On all transition labels there is at least one essential signal. Therefore, it is a BM-EHF specification.

There are two ways to transform nonessential hazard-free BM specifications into a BM-EHF specification.

3.3. Reduction of Input Burst Concurrency

The transformation consists of decomposing the input burst labeled on a state transition generating two-state transitions. For example, Figure 5 shows a reduced concurrency BM-EHF specification equivalent to the BM specification in Figure 1 in which the concurrency has been reduced. Analyzing the BM specification in Figure 1, we found state transitions $1 \rightarrow 2$ and $2 \rightarrow 3$ without essential signals. Decomposing state transitions $0_{[b+\,a+/x+]} \rightarrow 1$ into $0_{[b+/]} \rightarrow A_{[a+/x+]} \rightarrow 1$ and decomposing state transition $7_{[a+\,b-/x+]} \rightarrow 2$ into $7_{[a+/]} \rightarrow B_{[b-/x+]} \rightarrow 2$, we obtained the BM-EHF specification shown in Figure 5. It is EHF because transitions $4 \rightarrow 0$ and $7 \rightarrow B$ contain empty output bursts while all other transitions contain essential signals.

3.4. Insertion of Essential Signals

This transformation consists of inserting the smallest number of dummy essential signals in all state transitions without essential signal. For example, Figure 6 shows an BM-EHF specification equivalent to the specification in Figure 1 in which adummy essential signal d has been added to state transitions $1 \rightarrow 2$ and $2 \rightarrow 3$. This transformation has a higher cost than the previous one because it increases the number of input signals (I_b), modifying the interaction with the external environment.

If one observes the $2 \rightarrow 3$ state transition in Figure 6, the conclusion is that a is essential on transitions $1 \rightarrow 2 \rightarrow 3$, while d is essential on transitions $7 \rightarrow 2 \rightarrow 3$.

3.5. Super-State Condition

Lemma 1 is a necessary and sufficient condition for an essential hazard free *specification* but not for hazard-free *implementation*. The super-state concept will guarantee the latter condition.

Definition 4 (super-state). Consider an input burst $I_b(a, b, \ldots, n)$ and an output burst $O_b(x, y, \ldots, m)$. We call a *super-state* the set of single total states defined by all 0/1 combinations of a subset S_{Ib} of the input burst signals, keeping fixed the remaining input signals and all the output signals.

Definition 5 (essential super-state). Consider a BM-EHF specification in which a total state F is reached by a set of N incident transitions $\{I_{t\_i}\}$ $i = 1, \ldots, N$. Each incident transition I_{t_i} is activated by an input burst I_{b_i}. Each input burst is labeled with a subset of the input signals set $\{I_s\}$. An *essential super-state* is the super-state defined by the union

	xy \ abc	000	010	110	100	101	111	011	001
$d=0$	00	00 (0)	00	10	00	——	——	——	00 (4)
	01	01 (3)	——	——	01 (3)	——	——	——	00
	11	01	——	11 (2)	11 (2)	——	——	——	——
	10	10 (1)	10 (1)	10 (1)	10 (1)	——	——	——	——
$d=1$	00	——	——	——	——	——	01	——	——
	01	01 (3)	01 (7)	01	11	01 (6)	01 (6)	01	——
	11	11 (2)	11 (2)	11 (2)	11 (2)	10	01	——	——
	10	——	——	10	11	10 (5)	01	——	——

Figure 7: BM flow map with essential super-states (BM-ESS).

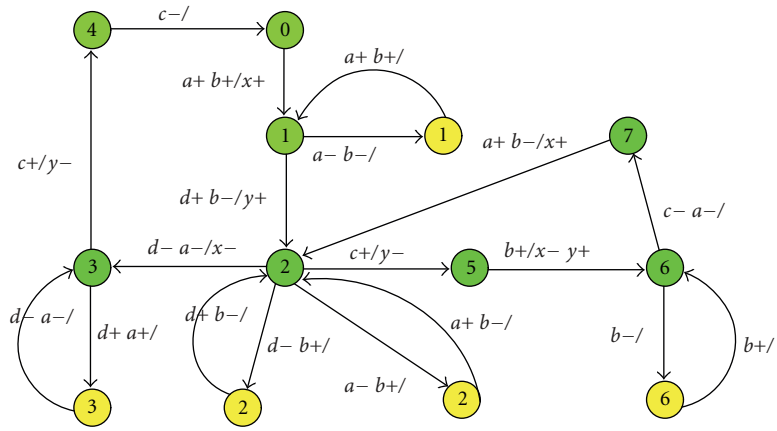

Figure 8: BM-EHF specification with essential super-states (BM-ESS).

$S_{\cup Ib} = \cup\{I_{b\_i}\}$ of all input signals active on the incident transitions set $\{I_{t\_i}\}$, labeled *with nonempty output bursts*.

An essential super-state BM flow map is derived from a BM-EHF specification by applying Definition 5 to all total states. Figure 7 is such a map for the specification in Figure 6. Cells in red are used to compose essential super-states. For example, the $0_{[a+b+/x+]} \rightarrow 1$ transition creates essential super-state 1 composed of four total states: $abcdxy = [000010, 010010, 100010, 110010]$. State 110010 is the final total state. Total state 2 may be reached from either state 1 or state 7. Applying Definition 5, we find that it must be composed of six total states (essential super-state 2): $abcdxy = [000111, 010111, 100111, 110111, 100011, 110011]$. This set of total states can be described by a cube (super-state transition cube—SSTC). Figure 8 shows the description in states transition diagram of the BM-ESS flow map.

Proposition 1. *If a total state F in a BM-EHF specification is reached by one or more incident transitions labeled with empty output bursts, then F is an essential super-state.*

Proof. Let $T1(A \rightarrow F)$ be a state transition with an empty output burst. $SSTC_A$ and $SSTC_F$ are super-state transition cubes for final total states A and F. As A must be essential, and as $SSTC_A[\text{output}] = SSTC_F[\text{output}]$ because both output bursts are empty, then F is also an essential super-state. □

Lemma 2. *The BM-EHF specification has an EHF implementation if and only if for ∀ total state $A \in$ BM-EHF it is an essential super-state.*

Proof. Let $T1(B \rightarrow A)$ and $T2(A \rightarrow C)$ be state transitions with output bursts. $SSTC_A$ and $SSTC_C$ are the super-state transition cubes of final total states A and C. Suppose that the $T2(A \rightarrow C)$ input burst does not contain an essential signal. Then $SSTC_A[\text{input}] \subseteq SSTC_C[\text{input}]$ hence $SSTC_A[\text{output}] \neq SSTC_C[\text{output}]$. This means that A cannot be an essential super-state because this would violate Definition 5. □

4. Metodolology

Our method begins from the BM specification and implements the asynchronous controllers in the architecture of Huffman with feedback output. The synthesis procedure has five steps.

(1) If the BM specification satisfies Lemma 1 to go for Step (3), otherwise, Step (2).

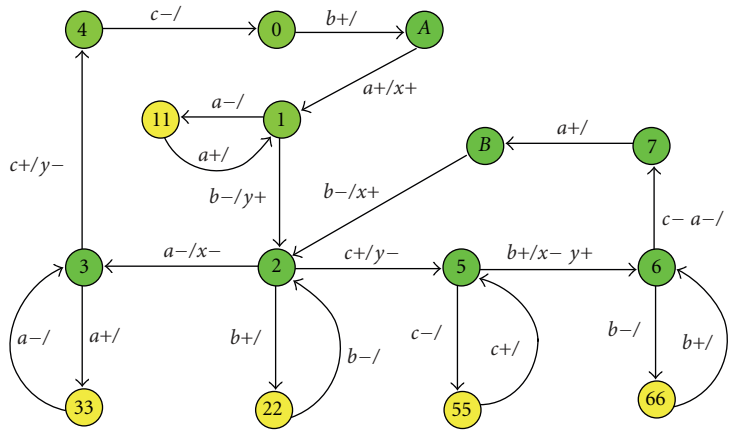

Figure 9: BM-ESS specification of Figure 5.

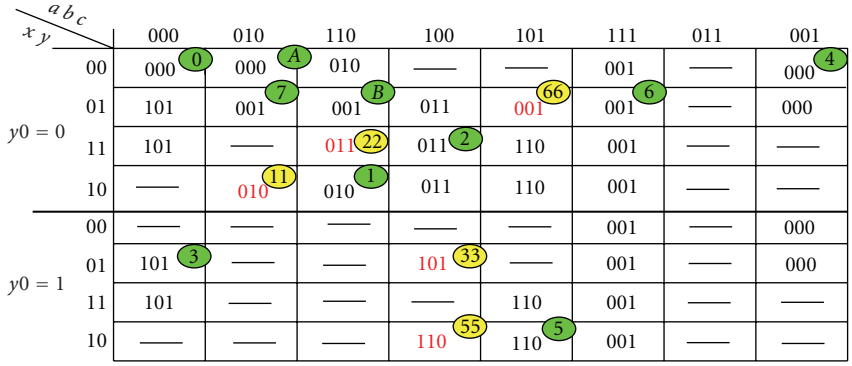

Figure 10: BM flow map with essential super-states (BM-ESS).

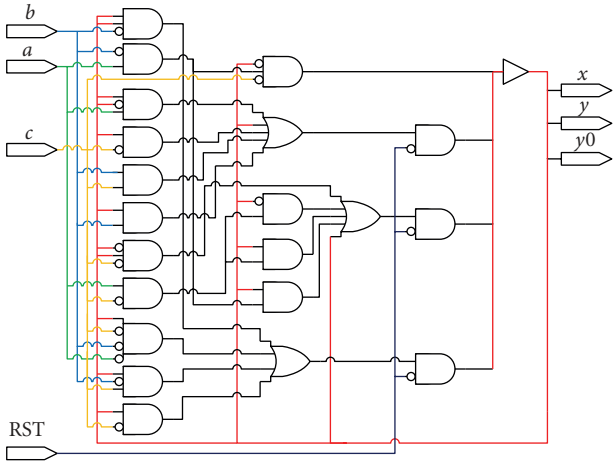

Figure 11: Logic circuit: RTL view—Altera.

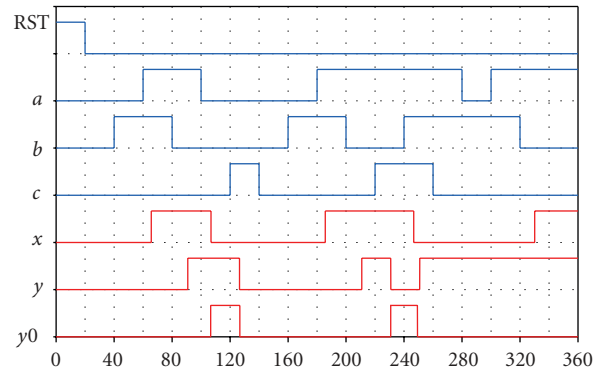

Figure 12: Simulation of the logic circuit of Figure 5.

(2) Apply in the BM specification the functional transformations that satisfy Lemma 1 (Sections 3.3 and 3.4).

(3) Generate the BM-EHF specification with essential super-states (BM-ESS) according to Section 3.5 (applying Definition 5).

(4) Use the Minimalist tool that starts from the BM-ESS specification and produces the equations of next-state hazard-free (sum of products—*netlist*).

(5) Use the Quartus tool [31] that starts from the *netlist* in structural VHDL.

The BM specification shown in Figure 1 has been used to illustrate our method. Figure 5 shows BM-EHF specification (Steps (1) and (2)). Figure 9 shows the BM-ESS specification (Step (3)). Steps (4) and (5) accomplish the automatic

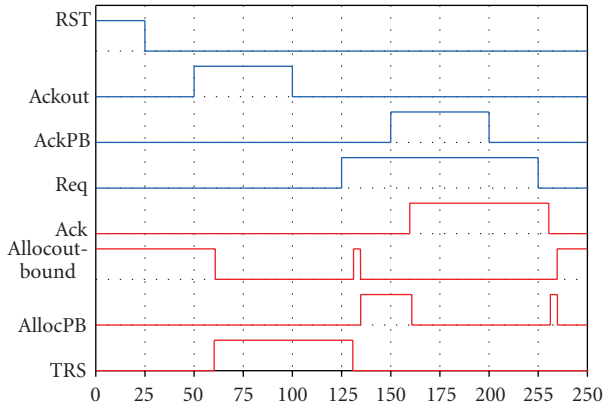

Figure 13: Simulation of the mp-for-pkt: with transient essential hazard.

Table 1: BM specifications show data.

	BM specification	
Design	States/trans.	In/out
Call-proc	12/15	3/3
Chu133	4/4	3/3
Chu150	5/5	3/3
Diff-Alu1	7/9	3/5
Dram-ctrl	12/14	7/6
Figure 1	8/9	3/2
Hp-ir-if	7/7	5/5
Mp-for-pkt	4/4	3/4
QR42	4/4	2/2
Rcv-setup	6/7	3/2

Table 2: Experimental Results to Huffman Machine The column State vars shows the variables of state.

	Huffman machine		
Design	State vars	Total of LUTs	Latency (ns)
Call-proc	0	8	10,888
Chu133	2	5	10,650
Chu150	0	3	10,017
Diff-Alu1	3	17	11,081
Dram-ctrl	0	10	11,633
Figure 1	0	4	10,675
Hp-ir-if	0	7	10,900
Mp-for-pkt	0	5	10,719
QR42	1	6	9,699
Rcv-setup	0	4	11,104

synthesis. One-state variable $y0$ was required to solve the existing conflicts (see Figure 10) [28]. Figure 11 shows logic circuit (RTL view—Altera). Figure 12 shows result of simulation of the circuit that was obtained by our method (hazard-free waveforms).

Table 3: BM-ESS specifications lead to the following data.

	BM-ESS specification		
Design	States/trans.	In/out	Dummy signals
Call-proc	22/40	3/3	0
Chu133	6/8	3/3	0
Chu150	10/15	3/3	0
Diff-Alu1	14/23	3/5	1
Dram-ctrl	22/36	7/6	0
Figure 5	14/19	3/2	0
Hp-ir-if	7/7	5/5	0
Mp-for-pkt	6/8	3/4	0
QR42	8/12	2/2	1
Rcv-setup	9/13	3/2	1

Table 4: Experimental results show Huffman machine—EHF (*Minimalist Tool did not complete the synthesis).

	Huffman machine—EHF		
Design	State vars	Total of LUTs	Latency (ns)
Call-proc	0	6	10,898
Chu133*	—	—	—
Chu150	0	7	11,606
Diff-Alu1	3	24	11,879
Dram-ctrl	0	16	12,271
Figure 5	1	11	10,855
Hp-ir-if	1	14	11,454
Mp-for-pkt	0	8	10,948
QR42	1	7	10,769
Rcv-setup	1	6	10,263

5. Discussion & Results

5.1. Discussion

Figures 13 and 14 show, respectively, the simulation results and the logic circuit of the mp-for-pkt benchmark whose specification is shown in Figure 4. The synthesis was performed using the Minimalist tool followed by the Quartus tool. Figure 13 shows two glitches, one on the *Allocoutbound* output and one on the *AllocPB* output. For example, the glitch on signal *Allocoutbound* occurs on state transition $1_{[\text{Ackout}-\text{Req}+/\text{TRS}-\text{AllocPB}+]} \rightarrow 2$. Figure 14 shows the behavior in the logic circuit of the state transition $1 \rightarrow 2$. The reason of the glitch: input signal *Req*+ acts in the paths 1 and 2, where the change in the path 1 arrives first in LUT-5 (see Figure 14). This glitche can also be identified in the BM flow map. Thespecification is EHF (Lemma 1 is satisfied) but the implementation is not (Lemma 2 is not satisfied), causing a transient essential hazard shown in Figure 15 (to apply GUR rule—T2: $2_{[\text{Req}-\text{Ackout}+]} \rightarrow 1 - \textit{Allocoutbound}\ 0 \rightarrow 1 \rightarrow 0$).

The result of simulation of the circuit that was obtained by our method shows that the glitches have been eliminated (see Figure 16). The area penalty was 8 LUTs against 5 LUTs in the first solution. The latency penalty was 2,2%.

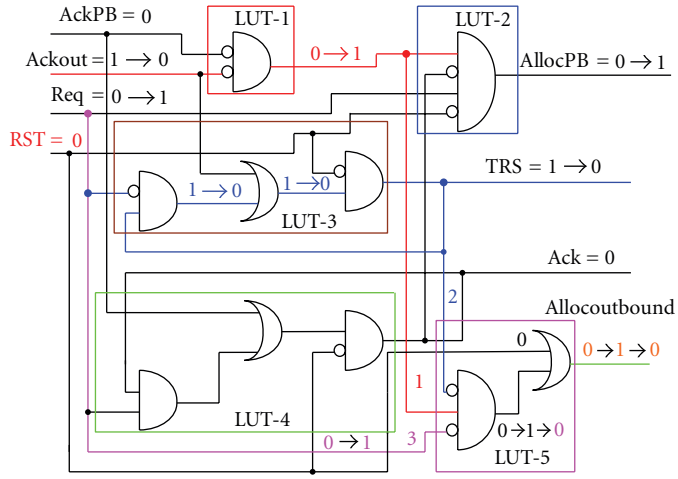

Figure 14: Logic circuit of the mp-for-pkt: map view—Altera (behavior 1 → 2).

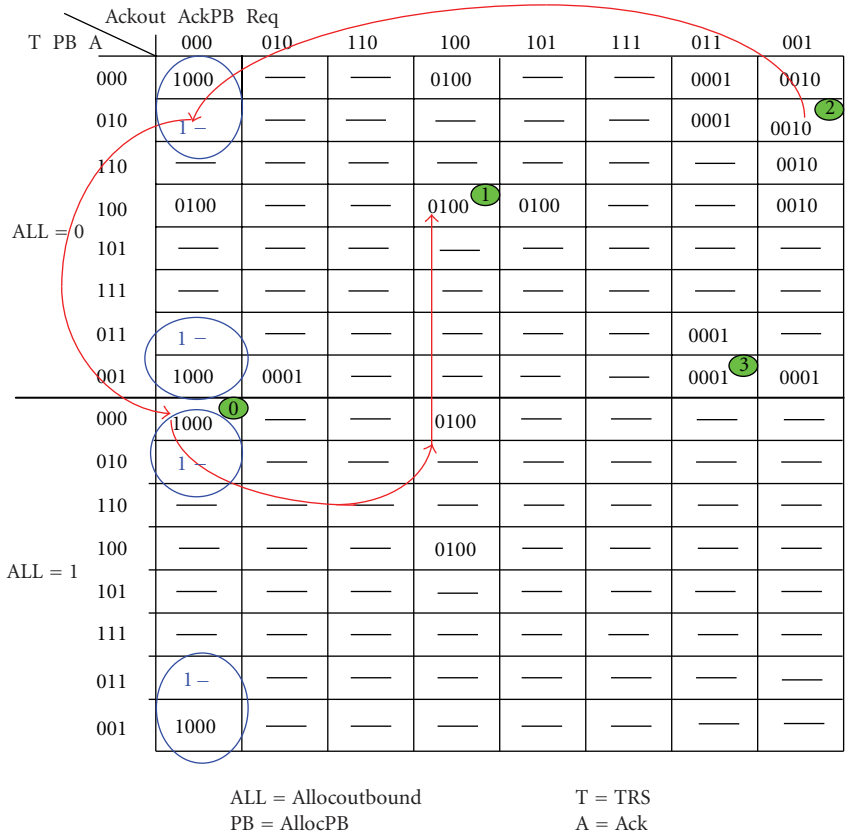

Figure 15: BM flow map showing a transient essential hazard.

5.2. Results

We applied our theory to 9 known [8, 9, 12, 13] and one homemade benchmark. Table 1 presents the number of input and output signals, states, and transitions for each benchmark. Table 2 presents the area and timing results for these benchmarks synthesized as Huffman machines (with feedback output) before applying our theory. Syntheses performed using Minimalist followed by Quartus. The area was measured in terms of the number of LUTs while the latency was derived from simulations of the circuits already fitted on an EP2C35F672C7 device from Altera (Cyclone II family).

Table 3 presents the number of inputs and output signals, states, transitions, and dummy signals for the same benchmarks after applying the functional transformations

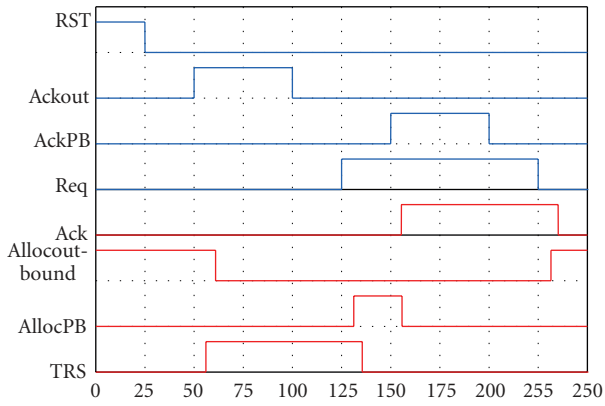

Figure 16: Simulation of the EHF version of the mp-for-pkt.

required to satisfy Lemmas 1 and 2. Table 4 shows the same results for the benchmarks after adhering to Lemmas 1 and 2.

As expected we found an area penalty (average of 54%), a latency penalty (average of 4,8%), and a state variables penalty (average of 75%). The *call-proc* benchmark showed a smaller area (less LUTs) and the *rev-setup* benchmark showed a reduced latency time. However, the area penalties did not impact significantly the FPGA usage (≅1%) still leaving enough free space for a datapath and other components that could be placed on the same device.

6. Conclusions

This work presented two conditions that, if satisfied, guarantee that burst-mode asynchronous controllers can be mapped on any commercial LUT-based FPGA without incurring in essential hazards.

When these conditions are not satisfied, we presented functional transformations that may be used to solve the problem. In this case, there is an area (mainly are added state variables—75%) and a latency penalty. However, our experimental results on a set of known benchmark showed low latency penalty (4,8%) and low FPGA occupation overhead (≅1%). This type of burst-mode controllers may be combined with a self-timed datapath that have already been successfully synthesized on commercial FPGAs, in order to create fully asynchronous processor on FPGAs.

References

[1] C. J. Myers, *Asynchronous Circuit Design*, John Wiley & Sons, New York, NY, USA, 2001.

[2] S. Hauck, "Asynchronous design methodologies: an overview," *Proceedings of the IEEE*, vol. 83, no. 1, pp. 69–93, 1995.

[3] S. H. Unger, "Hazards, critical races, and metastability," *IEEE Transactions on Computers*, vol. 44, no. 6, pp. 754–768, 1995.

[4] S. Hauck, S. Burns, G. Borriello, and C. Ebeling, "An FPGA for implementing asynchronous circuits," *IEEE Design & Test of Computers*, vol. 11, no. 3, pp. 60–69, 1994.

[5] D. M. Chapiro, *Globally-asynchronous locally-synchronous systems*, Ph.D. thesis, Stanford University, Stanford, Calif, USA, 1984.

[6] S. Burns and A. Martin, "Syntax-directed translation of concurrentprograms into self-timed circuits," in *Proceedings of the 5th MIT Conference on Advanced Research in VLSI*, J. Allen and T. Leighton, Eds., pp. 35–50, MIT Press, Cambridge, Mass, USA, March 1988.

[7] I. E. Sutherland, "Micropipelines," *Communications of ACM*, vol. 32, no. 6, pp. 720–738, 1989.

[8] T.-A. Chu, *Synthesis of self-timed VLSI circuits from graph-theoretic specifications*, Ph.D. thesis, Department of EECS, MIT, Cambridge, Mass, USA, June 1987.

[9] S. M. Nowick, K. Y. Yun, and D. L. Dill, "Practical asynchronous controller design," in *Proceedings of the IEEE International Conference on Computer Design (ICCD '92)*, pp. 341–345, Cambridge, Mass, USA, October 1992.

[10] S. M. Nowick and B. Coates, "UCLOCK: automated design of high-performance unclocked state machines," in *Proceedings of the IEEE International Conference on Computer Design (ICCD '94)*, pp. 434–441, Cambridge, Mass, USA, October 1994.

[11] S. M. Nowick, M. E. Dean, D. L. Dill, and M. Horowitz, "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, 1993.

[12] A. Marshall, B. Coates, and F. Siegel, "Designing an asynchronous communications chip," *IEEE Design & Test of Computers*, vol. 11, no. 2, pp. 8–21, 1994.

[13] A. Davis, B. Coates, and K. Stevens, "Automatic synthesis of fast compact self-timed control circuits," in *Proceedings of the IFIP Working Conference on Asynchronous Design Methodologies*, Manchester, UK, April 1993.

[14] E. Brunvand, "Using FPGAs to implement self-timed systems," *The Journal of VLSI Signal Processing*, vol. 6, no. 2, pp. 173–190, 1993.

[15] K. Maheswaran and V. Akella, "Hazard-free implementation of the self-timed cell set in a Xilinx FPGA," Tech. Rep., University of California, Davis, Calif, USA, 1994.

[16] Y. Zafar and M. M. Ahmed, "A novel FPGA compliant micropipeline," *IEEE Transactions on Circuits and Systems II*, vol. 52, no. 9, pp. 611–615, 2005.

[17] A. Ejnioui, "FPGA prototyping of a two-phase self-oscillating micropipeline," in *Proceedings of the IEEE Computer Society Annual Symposium on VLSI (ISVLSI '07)*, pp. 437–438, Porto Alegre, Brazil, March 2007.

[18] N. Huot, H. Dubreuil, L. Fesquet, and M. Renaudin, "FPGA architecture for multi-style asynchronous logic," in *Proceedings of the Conference on Design, Automation and Test in Europe (DATE '05)*, vol. 1, pp. 32–33, Munich, Germany, March 2005.

[19] X. Jia and R. Vemuri, "The GAPLA: a globally asynchronous locally synchronous FPGA architecture," in *Proceedings of the 13th Annual IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM '05)*, pp. 291–292, Napa, Calif, USA, April 2005.

[20] S. C. Smith, "Design of an FPGA logic element for implementing asynchronous NULL convention logic circuits," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 15, no. 6, pp. 672–683, 2007.

[21] D. Sokolov and A. Yakovlev, "Clockless circuits and system synthesis," *IEE Proceedings: Computers and Digital Techniques*, vol. 152, no. 3, pp. 298–316, 2005.

[22] H. Jacobson, *Asynchronous circuit design: a case study of a framework called ACK*, M.S. thesis, Luleå University of Technology, Luleå, Sweden, 1996.

[23] S. M. Nowick, "Automatic synthesis of burst-mode asynchronous controllers," Tech. Rep. CSL-TR-95-686, Stanford University, Stanford, Calif, USA, 1995.

[24] S. M. Nowick and D. L. Dill, "Exact two-level minimization of hazard-free logic with multiple-input changes," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 14, no. 8, pp. 986–997, 1995.

[25] P. Siegel, G. De Micheli, and D. Dill, "Automatic technology mapping for generalized fundamental-mode asynchronous designs," in *Proceedings of the 30th ACM/IEEE Design Automation Conference (DAC '93)*, pp. 61–67, Dallas, Tex, USA, June 1993.

[26] B. Lin and S. Devadas, "Synthesis of hazard-free multilevel logic under multiple-input changes from binary decision diagrams," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 14, no. 8, pp. 974–985, 1995.

[27] K. Y. Yun and D. L. Dill, "Automatic synthesis of extended burst-mode circuits. I. (Specification and hazard-free implementations)," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 18, no. 2, pp. 101–117, 1999.

[28] R. M. Fuhrer, S. M. Nowick, M. Theobald, N. K. Jha, and L. A. Plana, "Minimalist: an environment for the synthesis and verification of burst-mode asynchronous machines," in *Proceedings of the International Workshop on Logic Synthesis (IWLS '98)*, Lake Tahoe, Calif, USA, June 1998.

[29] H. M. Jacobson and C. J. Myers, "Efficient algorithms for exact two-level hazard-free logic minimization," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 21, no. 11, pp. 1269–1283, 2002.

[30] S. H. Unger, "Hazards and delays in asynchronous sequential switching circuits," *IRE Transactions on Circuit Theory*, vol. 6, no. 1, pp. 12–25, 1959.

[31] Altera Corporation, http://www.altera.com/.