Maghrebian Journal of Pure and Applied Science
ISSN : 2458-715X
http://revues.imist.ma/?journal=mjpas&page=index

Simulation and Analysis of the Concepts Self-Reconfigurable FSM

D. Traore

Ecole normale d'enseignement technique et professionnel Bamako Mali
***Corresponding Author; Email:** drissatraore2001@yahoo.fr
Received 14 March 2018, Revised 15 April 2018, Accepted 27 May 2018

***Abstract*:** In this paper we propose extension of self-reconfigurable finite state machine (FSM) with three different Concepts. These hardware concepts are called self-reconfigurable FSM as some reconfiguration sequences of either output function or transition function are initiated by the current state of FSM it self. Compared to Marcus machine only few sequences are exterior events which can contribute to power saving. Experimental results of those different concepts with some FSM benchmarks show compression of reconfigurable sequences data with minimum loss of area compared to Markus machine

Key Words: FSM, self-reconfigurable FSM, Delta transition, reconfiguration sequence, Markus machine.

1. Introduction

Finite state machine are important components in digital electronics such as in control circuit, network processing, testing. But reconfigurable finite state is still in its infancy. With reconfigurable hardware it is possible to change the functionality and wiring of hardware overtime [1-6]. Self-reconfigurable finite state machine uses the Principe of the context of software. In software word, the first generation of microprocessors made use of the idea of self-reconfiguration program code step wise to execute a program that cannot fit into the existing memory. Indeed, self-reconfigurable FSM is circuit that reconfigure it selves gradually. The paper [5] proposes the concept of self-reconfigurable finite states machine with the ability to change output and transition function or both gradually during run operation and algorithm for the overhead of automatic reconfiguration of a given FSM. Ref [7, 8] used the genial idea of reconfigurable finite state machine to build a VLSI statistically and dynamically hardware for finite state machine in which can be mapped some applications such as tries when constrained by slow memory access and difficult to scale when running in processor.

The real application domain for self-reconfigurable finite state machine is communication system. Basically, self-reconfigurable FSM are needed for application such autonomous sequential modules that can be considered as components of more complicated digital system. For example [5] examines a Mealy FSM that reads a sequence of bits and then detects two or more successive ones in the sequence. Reconfiguration considered in [5] permits the behavior of the FSM to be changed in such a way that it will detect two or more successive zeros in the sequence.

Ref [9,10] proposed the concepts of self-reconfigurable Finite machine with exterior delta transition and extension of Marcus machine.

In this paper we will discuss and analyse three different concepts of self-reconfigurable finite state machine.

The basis concept of self-reconfigurable FSM can be founded in [5].

This paper is organized as follows: In the next sections we review definition of FSM, and then we will analyse and propose the different concepts, finally we conclude with using reconfigurable FSM for practical applications and experimental Results.

2. Definition

A finite state machine is defined in the standard way as a tupple $M = (\varepsilon, A, Q, q_0, \delta, \lambda)$ ε is a finite set of input symbols, $Q \neq \varnothing$ is a finite set of states, $q_0 \in Q$ is a reset state, $\delta(q,a): Q \times \varepsilon \to Q$ is the transition function, and $\lambda(q,a): Q \times \varepsilon \to A$ is the output symbol in the specification of the state transition graph of the FSM, each state correspond to one node in the graph, and there exists an edge e_{ij} between two states $q_i and q_j$ with label a/b If $\delta(q_i, a) = q_j$ and $\lambda(q,a) = b$.

3. Self Reconfigurable FSM with Exterior Delta Transition

3.1. Analysis

In self-reconfigurable FSM with exterior delta transition analysis, it is assumed that only the delta transitions are exteriors event.

In the basis case [5], Suppose that we want to reconfigure a machine M to a machine M'

Let Z1, Z_2 ...Z_n denotes all the delta transitions and Z_t the total set of all the delta transitions.

$Z_t = Z_1+Z_2+...+Z_n$ is a total set of delta reconfiguration sequences we need to reconfigure M to M'.

Each delta transition being a set of *n* transition sequences, thereby we can write:

$Z_1 = Z_{11}+Z_{12}+\ldots\ldots+Z_{1n}$

$Z_2 = Z_{21}+Z_{22}+\ldots\ldots+Z_{2n}$

And

$Z_n = Z_{n1}+Z_{n2}+\ldots\ldots+Z_{nn}$

And the total sequence can be write

$Z_t = Z_{11}+Z_{12}+\ldots\ldots+Z_{1n} + Z_{21}+Z_{22}+\ldots\ldots+Z_{2n}+\ldots\ldots+Z_{n1}+Z_{n2}+\ldots\ldots+Z_{nn}$

Let denotes *N* in number denotes the total sequences to reconfigure M to M'.

$N=(1+2+...+n) + (1+2+\ldots\ldots+n) + \ldots\ldots\ldots+(1+2+...+n)$

$N = n^2$

So, we need n^2 sequences to reconfigure M to M' [5].

Self-reconfiguration FSM with exterior delta transition meaning to reduce *n* sequences for each delta transition to 1 sequence. Let Z_{td} the new total set of all delta transitions and N_d denotes in number the total sequences. We can write:

$Z_{td} = Z_1+Z_2+...+Z_n$

And

$N_d=(1+2+...+n)$

$N_d = n$

So, we need *n* sequences from exterior to Reconfigure M into M' in case of self-reconfiguration with exterior Delta Transitions.

Let we denote *D* the difference in exterior sequences or gain in sequences between the two cases.

$D = N_n - N_d$

$D = n^2 - n$

For $n \to \infty$ we can write

$D;\ n^2$

Thereby we gain n^2 Reconfiguration sequences from exterior.

Gain in reconfiguration bits.

For $n \to \infty$ the number of bits for the last sequence are n bits

So, we can save $n \times n \times n$ bits from Exterior.

We can write that the gain in reconfiguration bits is $O(n^3)$

Reconfigurator Hardware saving:
We can analyze the saving in Hardware in terms of number of sequences to the reconfigurator.
For the Ref [5]:
Number of reconfiguration sequences to reconfigurator is:

$$N_R = n^2$$

For self-reconfiguration with exterior delta transition this number can be write as the sum of *n* delta transition and *n* states of the FSM because all sequences for each delta transitions are generated by the current state of the FSMs registers.

$$N_{Rd} = n + n = 2n$$

The Hardware saving can be expressed as the difference between N_R and N_{Rd}

$$H_S = N_R - N_{Rd} = n^2 - 2n$$

For $n \to \infty$ the saving in the Reconfigurator Hardware is also $O(n^2)$

In this section we have gave the benefit of self-reconfiguration with exterior delta transition in terms of hardware cost in the FSMs itself and exterior to it. Now we are going to propose the Concept of self-reconfiguration with exterior delta transition.

3.2. Concepts and Example

A self reconfigurable FSM is a 9 tupple: $(\varepsilon, R, A, Q, q_0, \delta, \lambda, M, D)$ In which:

* $(\varepsilon, A, Q, q_0, \delta, \lambda)$ describes FSM according to finite state machine definition.

* R is a set of reconfiguration sequences called reconfigurator and generates the vectors R_λ, R_δ, I_R during reconfiguration to update the output function and transition function. And the output of the reconfiguration function is a mapping from the current state and the delta transition

$$R_\delta(i) := R_\lambda(i) := I_R(i) := R(S_i, D_i)$$

* D is a set of delta transitions, it determines toward which delta transition the machine has to traverse, and it is an exterior event.

* $M(I_R, \varepsilon)$ Is a mapping from the input state, reconfiguration state, to the internal input *I'*.

The concept is show in the block diagram of Fig.1. In this model, the transition function is updated by R_δ and the output function is updated by R_λ. R_λ And R_δ depend on the current state and the delta transition, whereas M depends on the input state ε and the reconfigurator input I_R. The function M can be a multiplexer and is defined such that during normal operation of the hardware $I' = \varepsilon$ and during reconfiguration process *I'* depend on I_R only.

Example
Considers two FSMs M and M' with their state transition diagram show in Fig.2. Supposes that the machine is in M and we want to migrate to M'. Let assume the machine is in state S_0 and D = {D_0, D_1, D_2, D_3} is the set of delta transitions then a reconfiguration sequences take place for each delta transition. Table.1 show the reconfiguration sequences.
Notes that D_1 and D_2 have same reconfiguration sequences and D_0 is let for normal operation in which the transition function and output function will keep their values.

3.3 Definition and Theorem

3.3.1 Definition

Given a finite state machine M and a finite state machine M' and a set of states including the states of M and M', a set of inputs including the inputs of M and M', a set of outputs including the outputs of M and M'.

The problem of feasible self-reconfigurable FSM with exterior delta transition denotes the decision problem whether the current state can generate all sequences for a given delta transition.

3.3.2 Theorem

According to definition 4.2.1 given the specifications of a state machine M and a state M' it is always possible to migrate from M into M' if in the sequences for any delta transition there is non self loop.

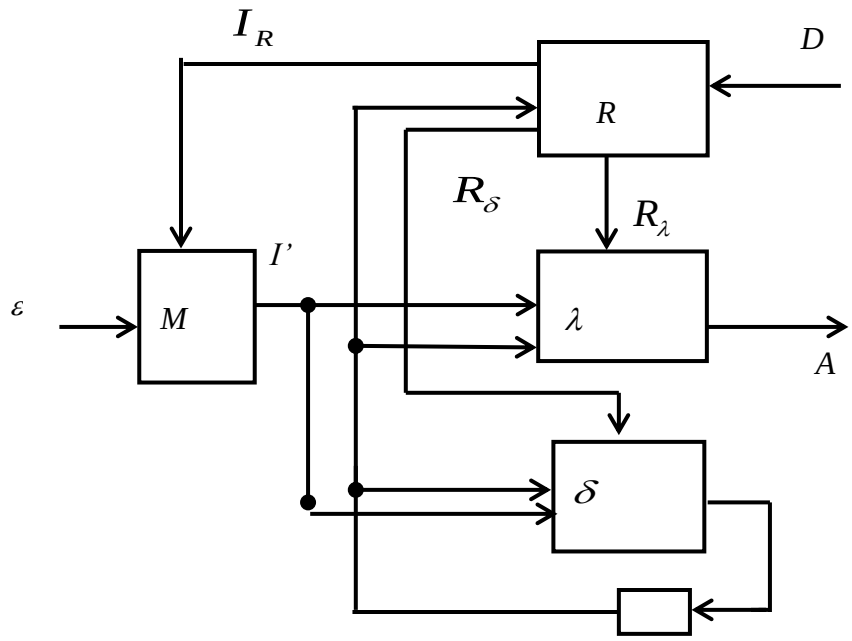


Figure 1: Schematic of reconfigurable FSM with exterior delta transition

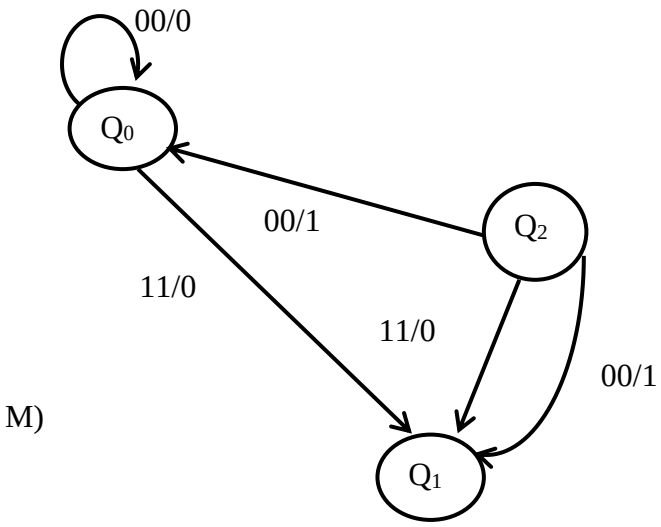
Table 1. Reconfiguration sequences for Fig.1

D	SEQUENCES	CURRENT STATE	INPUT	OUTPUT	NEXT STATE
$D_1=D_2$	1st sequence	Q_0	11	0	Q_1
$D_1=D_2$	sequence 2	Q_1	00	1	Q_2
$D_1=D_2$	sequence 3	Q_2	10	0	Q_3
$D_1=D_2$	sequence 4	Q_3	01	1	Q_0
D_3	1st sequence	Q_0	11	0	Q_1
D_3	sequence 2	Q_1	01	1	Q_0
D_0	Normal operation				

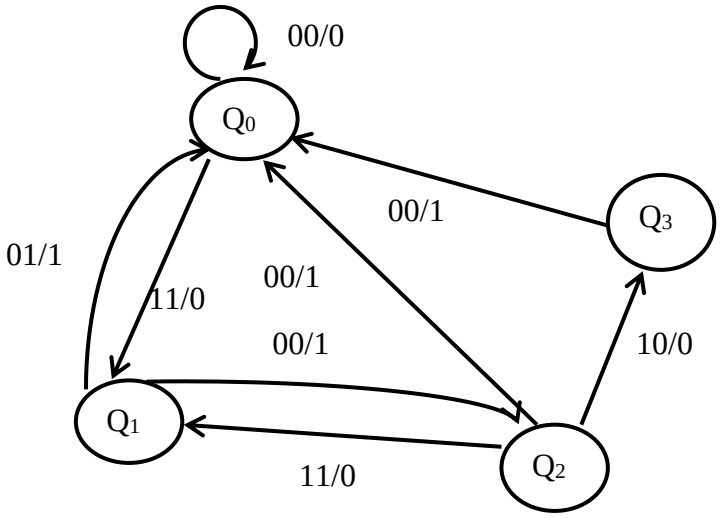
4. Self-Reconfigurable Hardware for FSMs

4.1 Analysis

In self reconfiguration FSM concept if all the delta transition need only one sequence to be traversed to reconfigure M into M' we can extend the concept to self reconfigurable hardware for FSMs. In That case all the sequences including the delta transitions to reconfigure one machine to another are generated by the current state of the machines. And only one sequence from exterior is need.



M)



M')

Figure 2: STGs of two FSMs

In that case:

$N = n$ Number of sequence from exterior in [5]

$N_d = 1$ Number of sequences in that reconfigurable hardware for FSMs.

$D = n - 1$ The difference in reconfiguration sequences from exterior.

And we can gain *n*-1 sequences from exterior.

We can write that the gain in reconfiguration bits is $O(n^2)$

And the hardware cost is

$H_s = n - n - 1 = -1$

For any *n* losing in the Reconfigurator Hardware is also *O (*1)

4.2. Concept

A self-reconfigurable hardware for FSMs is a 9 tuple: $(\varepsilon, R, A, Q, q_0, \delta, \lambda, M, C)$ In which:

* $(\varepsilon, A, Q, q_0, \delta, \lambda)$ describe an FSM according to finite state machine definition.

* R is a set of reconfiguration program called reconfiguration function and generates the vectors R_λ, R_δ, I_R during reconfiguration to update the output function and transition function. And the output of the reconfiguration function is a mapping from the current state and the context index.

$R_\delta(i) := R_\lambda(i) := I_R(i) := R(S_i, C_i)$

* C is a reconfiguration indexer, it determines toward which FSM the machine must migrate, it is an exterior event.

* $M(I_R, \varepsilon)$ Is a mapping from the input state, reconfiguration state, to the internal input *I'*.

The concept is show in the block diagram of Fig.3

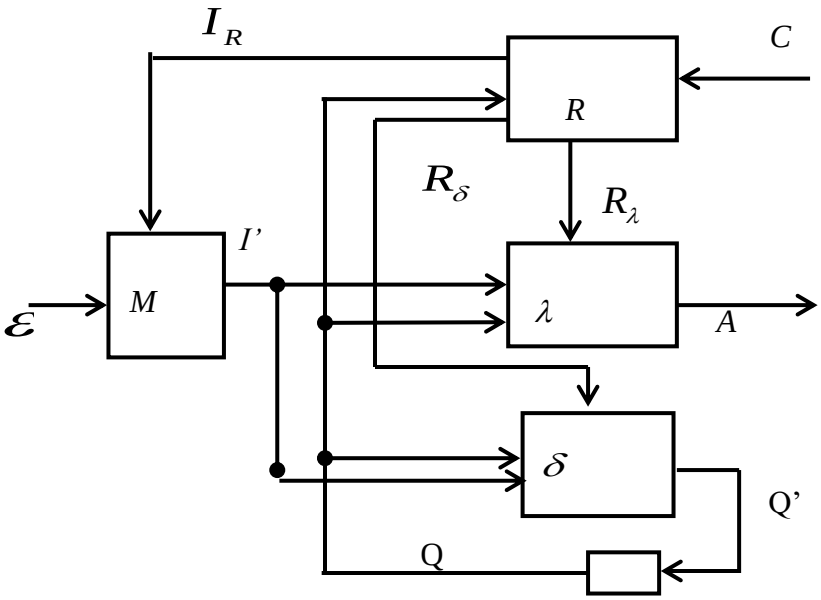


Figure 3: Block diagram of self-reconfigurable hardware for n FSMs

4.3. Feasibility of Self-Reconfigurable Hardware for n FSMs

Given *n* FSMs, it is only possible to make a self reconfigurable hardware for those n FSMs if only if there exist a unique reconfiguration program for all delta transition and a state is traversed only once during reconfiguration and two Sequences may have same state, so the reconfiguration program is a function of the number of the states and during reconfiguration a state is traversed only once. So, this is possible if only if $Z=Z_1=Z_2=\ \ldots=Z_n$ where $Z_1, Z_2, \ldots Z_n$ Are reconfiguration program for all delta transitions starting at idle state.

Example Set of FSMs for self-reconfigurable hardware:

Fig.4 are some examples of FSMs that can be map in self-reconfigurable hardware. In the two cases the machine may migrate from M_1 into M_2 and from M_2 into M_3 in1) for migrating from M_2 into M_3 there are two delta

transitions (01, Q_3, Q_0, 1) and (10, Q_2, Q_3, 0) and a reconfiguration program ((11, Q_0, Q_1, 0), (00, Q_1, Q_2, 1), (10, Q_2, Q_3, 0), (01, Q_3, Q_0, 1)) is valuable for the two delta transition.

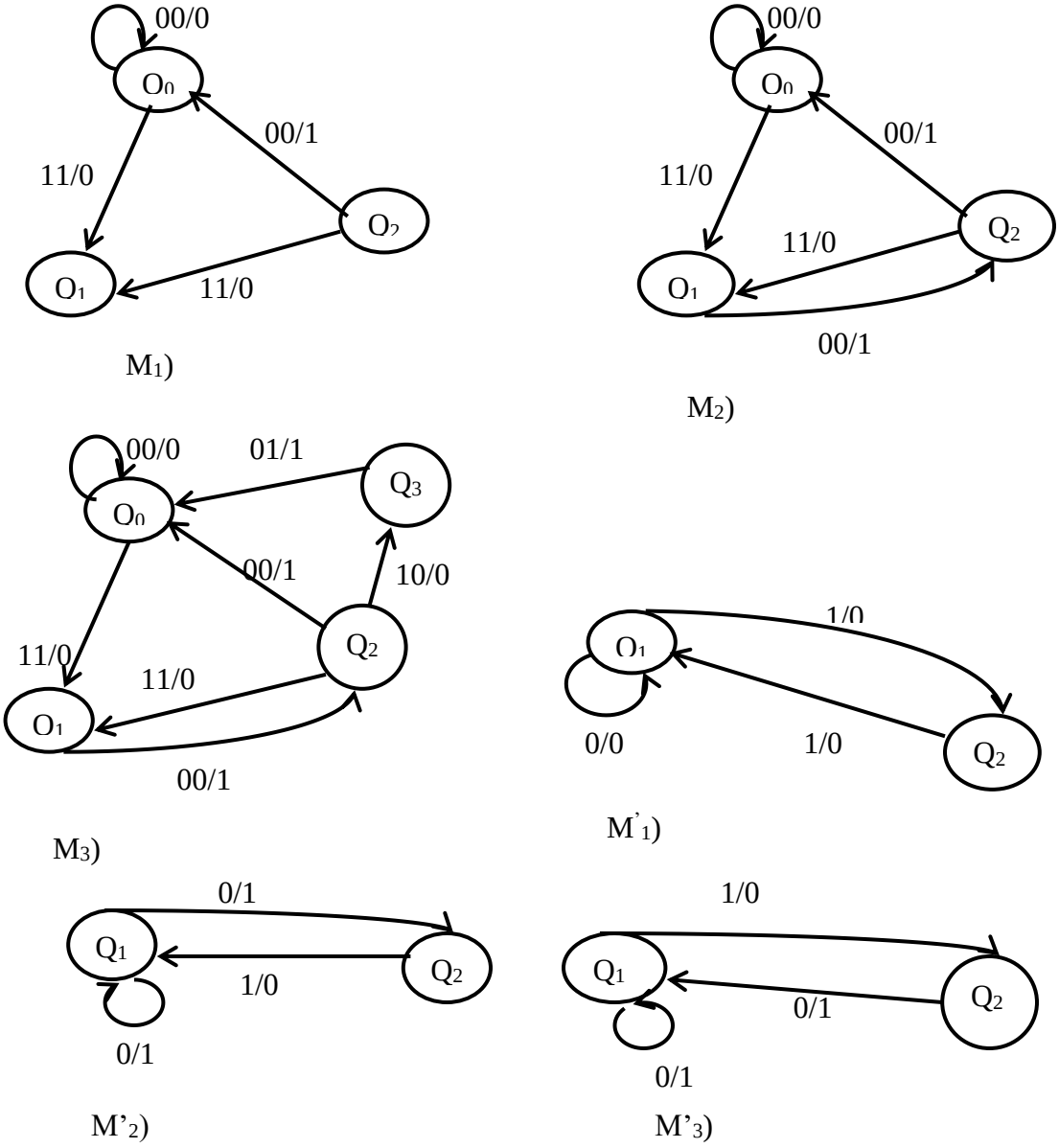


Figure 4: FSMs for self-reconfigurable hardware

5. Extension of Marcus Machine: Self Reconfigurable FSM with Reduces Reconfiguration Sequences Concept

5.1. Analysis

The same analysis in the last section is valuable here. Here we intend to generate the sequences of the longest delta transition by the current state of the machine and all other sequences from exterior.

Self-reconfiguration with reduced reconfiguration sequences being mean that the sequences of the longest delta transition is generated by the current state of the machine. So, let Z_L be that delta transition. We can write the new Z_t as:

$Z_t = Z_1+Z_2+\ldots+Z_{n-1}$ the sum of delta transition from exterior

Assume each of theses delta transitions hold n Sequences the total sequences N_E needed from exterior will be:

$$N_E = n^2 - n$$

Let we denote D the difference in exterior sequences or gain in sequences between the two cases.

$$D = N - N_E$$

$$D = n^2 - n^2 + n = n$$

Thereby we gain n Reconfiguration sequences from exterior

Gain in reconfiguration bits.

For $n \to \infty$ the number of bits for the last sequence is n bits

So, we can save $n \times n$ bits from Exterior.

We can write that the gain in reconfiguration bits is $O(n^2)$

Reconfigurator Hardware saving:

We can analyze the saving in Hardware in terms of number of sequences to the reconfigurator.

For the Ref [5]:

Number of reconfiguration sequences to reconfigurator is:

$$N_R = n^2$$

And the number of sequences from the current state to transverse the delta transition Z_L is:

$$Z_L = n$$

For self-reconfiguration with reduced reconfiguration sequences the number of sequences to the reconfigurator can be write as:

$$N_{RR} = N_L + N_E = n + n^2 - n = n^2$$

The hardware saving can be expressed as the difference between N_R and N_{RR}

$$H_S = N_R - N_{RR}$$

$$H_S = N_R - N_{RR} = n^2 - n^2 = 0$$

For any n there is no hardware saving.

The advantage of this concept is being reconfiguration data saving.

5.2. Concept

A self reconfigurable FSM with a delta transition initialization is a 9 tuple: $(\varepsilon, R, A, Q, q_0, \delta, \lambda, M, C, E_E)$ In which:

* $(\varepsilon, A, Q, q_0, \delta, \lambda)$ describes an FSM according to finite state machine definition.

* R is a set of reconfiguration sequences called reconfigurator and generates the vectors R_λ, R_δ, I_R during reconfiguration initialization and $R_{\lambda I}, R_{\delta I}$ during exterior reconfiguration event to update the output function and transition function. And the output of the reconfiguration function is a mapping from the current state and the C during initialization and mapping from E_E during exterior event.

$R_{\delta I}(i) := R_{\lambda I}(i) := I_R(i) := R(S_i, C)$ during initialization

$R_\delta(i) := R_\lambda(i) := R(E_E)$ during exterior event

* C is a delta transition, it determines toward which delta transition the machine has to traverse for initialization, and it is an exterior event.

* E_E is a set of transitions, it determines toward which transition the machine has to traverse during exterior event, and it may an exterior event.

* $M(I_R, \varepsilon, E_E)$ Is a mapping from the input state, reconfiguration state I_R, to the internal input *I'* during initialization and mapping from the input state, Reconfiguration state E_E to the internal input *I'* after initialization in normal reconfiguration process.

The concept is show in the block diagram of Fig.3.9 In this model, during initialization process the transition function is updated by R_δ and the output function is updated by R_λ. R_λ And R_δ depend on the current state, whereas M depends on the input state ε and the reconfigurator input I_R. During exterior reconfiguration process the transition function and output function are updated respectively by $R_{\lambda I}$ and $R_{\delta I}$ depending on E_E. The function M can be a multiplexer and is defined such that during normal operation of the hardware $I' = \varepsilon$ and during initializations reconfiguration process I' depend on I_R only and depend on E_E When exterior reconfiguration process

Example

considers two FSMs M and M' with theirs state transition diagram show in Fig.2.

Supposes that the machine is in M and we want to migrate to M'. Let assume the machine is in state S_0 and

D = {D_1, D_2, D_3} is the set of delta transitions then a reconfiguration sequence take place for each delta transition.

$D_1 = D_2 = (01, Q_1, Q_0, 1)\}, (00, Q_1, Q_2, 1), (10, Q_2, Q_3, 0), (01, Q_3, Q_0, 1)\}$

$D_3 = \{(11, Q_0, Q_1, 0), (01, Q_1, Q_0, 1)\}$

We can see that D_1 and D_2 have same reconfiguration program and longest, so we can attribute these delta transitions for initialization and D_3 as exterior reconfiguration event.

We can gain four reconfiguration sequences. For this case the reconfiguration sequences are reduced about 60%.

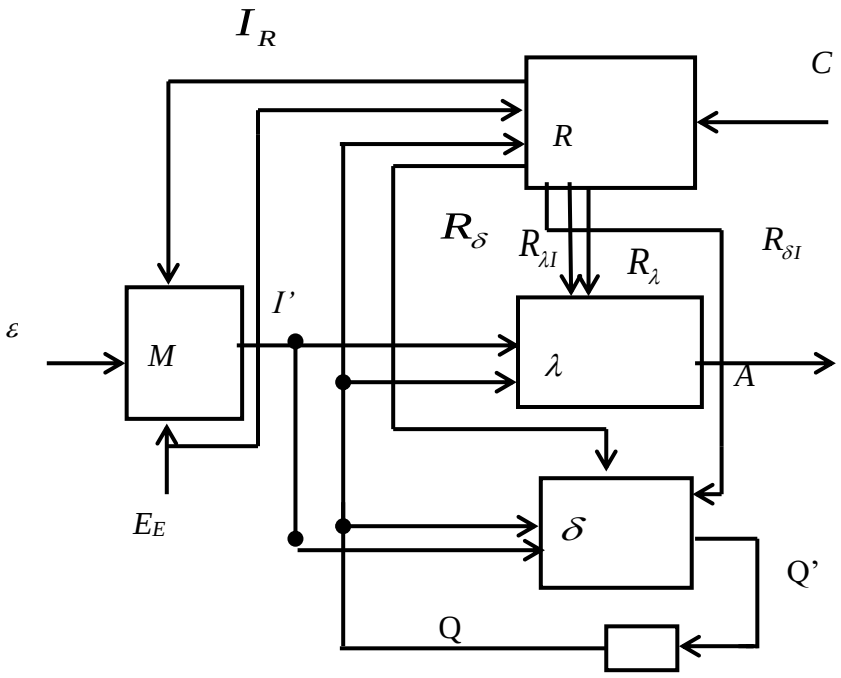


Figure 5: Schematic of the extended self reconfigurable FSM with reduced sequences

6. Reconfigurable FSMs for Practical Applications

It is known that digital systems can be decomposed into a reusable execution unit (which is extendable in the general case) and many control units. The later can be modeled by a reprogrammable FSM. Let us consider an example. There are many practical applications that require the solution of various combinational problems. As a rule, these problems are time consuming. That is why many different hardware accelerators have been designed. It is known that the majority of combinatorial problems can be formulated over Boolean. Thus, we can construct a reusable hardware circuit that permit the realization of various operations over Boolean. This circuit can be employed for solving different combinatorial problems by altering the control sequence. To be able to change the behavior of the control circuit that are used at different levels. This can be done based on reusable hardware.
Reconfigurable FSMs consists of configurables parts, multiplexers, combinational, registers and controller, which can be used in communication systems, in a combinatorial coprocessor, in application specific circuits, in embedded controllers. Those models can serve in autonomous sequential modules that can be considered as components of more complicated digital systems.

7. Implementation of the Self-Reconfigurable FSM

In this implementation the transition function and output function are memory devices. The two multiplexers are used to force the reset state and the reconfiguration input during reconfiguration. And the input and current state are memory address and the locations of the memory are output and next state for the FSM. For the reconfiguration program generator there are many concepts to build it, the one show in Fig.6 may not the optimal one, which is an array of reconfiguration program. The context index or (Delta transition) chooses the reconfiguration program and then the current state generates the different sequences. Each reconfiguration program is a set of registers, which contains the different data, and different signals and each register contains information for one sequence addressed by the current state.

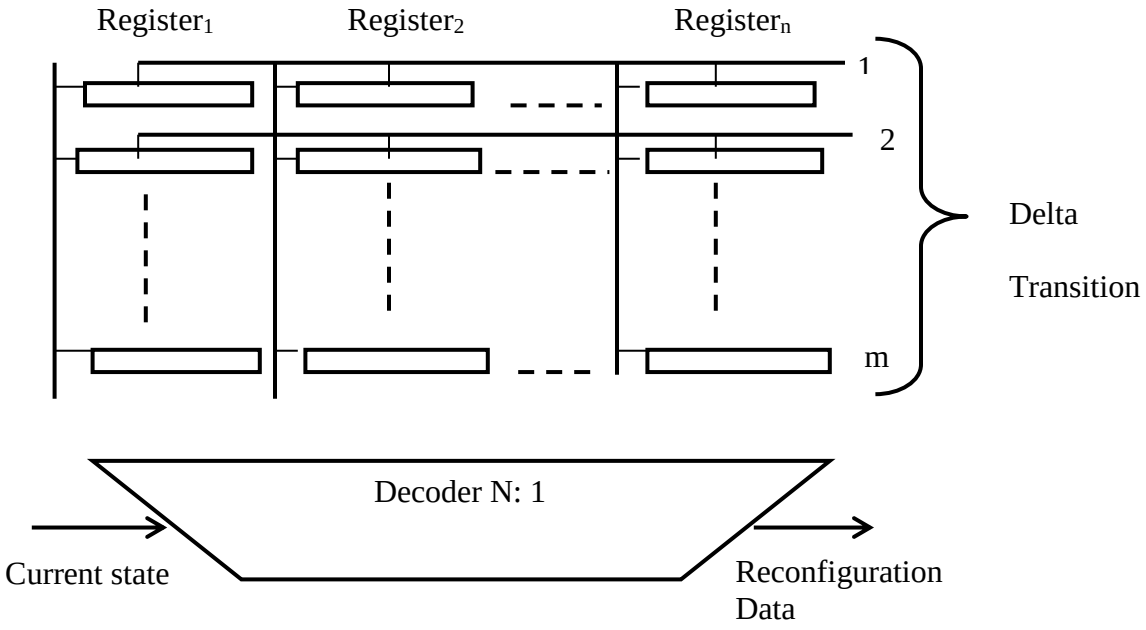


Figure 6: Reconfiguration mapping

8. Experimental Results

To simulate our concepts for area we have synthesized self-reconfigurable hardware with dk27, mc, bbtas benchmark FSMs with the different concepts. For synthesis we have used Synopsys tools, the optimization was done under same conditions and same constraints. The experimental results in the following tables show compression of reconfigurable sequences data with minimum lost of area compared to Markus machine.

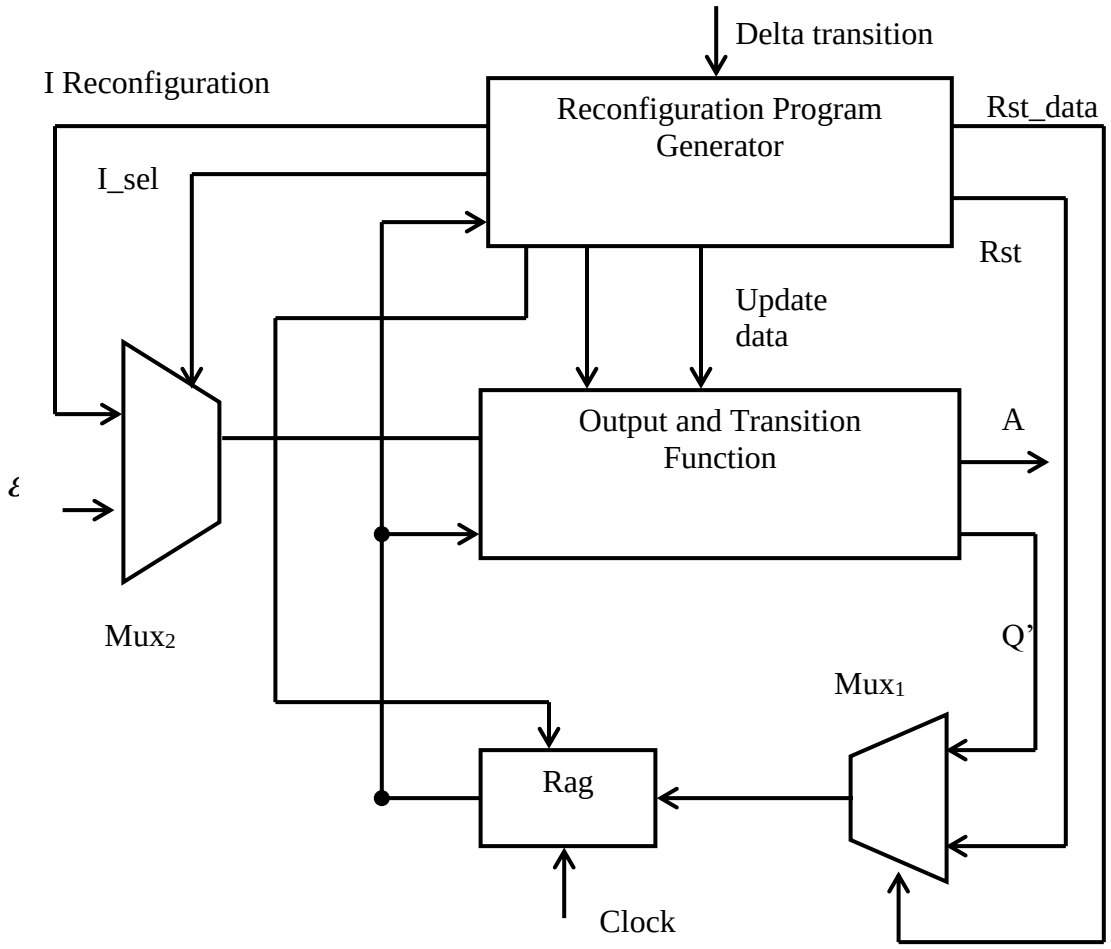


Figure 7: Implementation of self-reconfigurable FSM

Table 2. Markus machine

FSM	Sequences	Reconfigurable data	Reconfigurator cells
bbtas to dk27	24 sequences	120 bits	70 cells
dk27 to bbtas	27	135 bits	87
bbtas to mc	17	85 bits	68
mc to bbtas	25	125	81
dk27 to mc	16	80	68
mc to dk27	21	105	73

Table 3. Reconfigurations with exterior delta transition

FSM	Sequences	Reconfigurable data	Reconfigurator cells
bbtas to dk27	6 sequences	18bits	68 cells
dk27 to bbtas	7	21 bits	67
bbtas to mc	6	18	64
mc to bbtas	7	21	76
dk27 to mc	6	18	61
mc to dk27	7	21	56

Table 4. Reconfiguration with initialization

FSM	Sequences	Reconfigurable data	Reconfigurator cells
bbtas to dk27	18 sequences	90bits	85 cells
dk27 to bbtas	21	110	96
bbtas to mc	14	56	77
mc to bbtas	20	100	90
dk27 to mc	13	39	74
mc to dk27	14	42	73

9. Conclusion

In this paper, we introduced extension of self reconfigurable FSMs with three different concepts. Experimental results shown compression of reconfigurable data compared to Markus Machine**.** By implementing those models, a programmable control unit can be designed for a combinatorial processor. This permits the processor to be optimized by customizing the reusable hardware for a combinational problem that requires specific subsets of operations.

By working in those models, we obtained the following results:

- ✓ More self-reconfiguration
- ✓ Reduction of reconfiguration sequences that can decreases the switching activities therefore power saving.
- ✓ Reduction of configuration data leading to speed up Reconfiguration time.

References

1. V. Tapatho, N. Rayapati, B. K. Aminska. Dynamic reconfigurable schemes for megabit BiCMOS SRAMs and performance evaluation. **Microelectronic Reliab, 37(5): 785~794 (1997)**
2. Y. Mitsuyama, Z. Andales, T. Onoye, I. Shirakawa. VLSI 1mplementation of dynamically reconfigurable hardware –based cryptosystem.2000 Symposium on VLSI Circuit Digest of Technical Papers. **Hawaii, USA, 204~205 (2000)**
3. L. Kurian John, E. John. A dynamically reconfigurable interconnect for array processors. **IEEE Transaction on VLSI System 1998. 6(1): 150~155 (1998)**
4. T. Fujii. A dynamically reconfigurable logic engine with a multicontex/multi –mode unified cell architecture. **In ISSCC Digest of technical. 364~365 (1999)**
5. M. Koster, J. Teich. Self-reconfiguration finite state machine theory and implementation "Proc of the 2002 design, Automation and Test, **Europe Conference and Exhibition IEEE, 550~566 (2002)**
6. A. Oliveiro, V. Sklyarov. Implementation of virtual control circuit in dynamically reconfigurable FPGAs. **The 6th IEEE International Conference on Electronics, Circuit and System. Cyprus, 217~220 (1999)**
7. M. Desai, R. Gupta. Reconfigurable finite state machine-based IP lookup engine for high-speed router. **IEEE Journal on Selected Area in Commun, 21(4): 501~512 (2003)**

8. D. Traore, M. Zhi. Gang "VLSI dynamically reconfigurable hardware for finite state machine design and analysis" 6 Th int on ASIC, **Shanghai China. 24-27, ISBN 0-7803-9210-8, IEEE. Press, 734-738**
9. D. Traore, M. Zhi. Gang. An extension of self-reconfigurable FSM with reduced reconfigurable sequences concept. 5th WSEAS Int Conf on Circuits and System**. Hangzhou, China.16~18,180~185 (2006)**
10. D. Traore, M. Zhi. Gang. Self-reconfigurable FSM with exterior delta transitions concept and analysis. **WSEAS Transaction on Circuits and Systems (Included EI compendex), 5(5): 640~646 (2006)**

(2018) ;http://revues.imist.ma/?journal=mjpas&page=index