International Journal of Engineering and Advanced Technology (IJEAT)
ISSN: 2249-8958 (Online), Volume-7 Issue-3, February 2018

Review of Reconfigurable Finite State Machine

Drissa Traoré, Diouba Sacko, Marie Bernard Sidibé

Abstract: ***This paper surveys a review of reconfigurable finite state machine. It's a powerful methodology for achieving high performance. Minimizing the resource required in the implementation of many applications points out the difference between run time reconfigurable finite state machine and programmable finite state machine. The application of the concept of self-reconfigurable finite state machine to achieve VLSI architecture of dynamically reconfiguration is also discussed.***

Index Terms*—Finite State Machine, VLSI, FPGA, *Reconfiguration.

I. INTRODUCTION

Control circuits are one of the most important components of most digital systems. They are often modeled as finite state machine (FSM) and implemented using hardwired circuit or microprogramming. The used technique to design data path cannot be applied to control circuits which are specific for each project. The hardware implementation is very fast but not flexible while a software implementation can be easily modified.

The specifications of FSMs are based on State Transition Graphs (STG) or Algorithmic State Machine (ASM) char. The design process is dividing in the following steps: the choice of the implementation model (Moore, Mealy.etc), the build of the state diagram or algorithmic, the conversion into the state transition table, encoding of states, logical optimization and finally the implementation of the circuit.

The fast increase of number of reconfigurable FSMs involves the passage of reconfigurable FSMs from the FPGA platform to VLSI architecture.

In this paper, we implement the VLSI architecture for dynamically reconfigurable FSM. Hardware is built in order to reconfigure dynamically the FSM, utilizing some blocks. Other blocks will be utilized for computing. Hardware is called programmable FSM if we can modify any time its functionality without modifying the hardware himself. Run time reconfigurable FSM is utilized in network i.e. packet processing and control circuit for data path.

To ensure the continuity of the services of the data transmission, the great difficulty does not reside at the level of the transmission channel which has a broad frequency band and a very weak attenuation but at the level of emission and
reception modules. Indeed, to carry out a qualitative transmission system between the transmitter and the receiver, it is necessary to overcome the insufficiencies resulting from the optoelectronic components within these modules.

II. STRUCTURE OF PAPER

The work is subdivided in four chapters. The first gives detailed description of a finite state machine. The second chapter reconfigures a FSM in FPGA. The third chapter gives VLSI architecture for programmable FSM. Last chapter gives suggestion for VLSI dynamically reconfigurable.

III. DESCRIPTION OF FSM

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, $A \neq \varnothing$ 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 a 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 qi and q_j with label a/b If $\delta(q_i, a) = q_j$ and $\lambda(q,a) = b$.

Obviously, a reconfigurable FSM allows a FSM to change its output and transition function over time. In this case, the reconfiguration is initiated by the system itself (self reconfiguration). It might be also possible to change output function and transition function autonomously but also in dependence of external events such as reconfiguration events. A reconfigurable FSM is a 10 tuple.

$(I, O, S, S_0, F, G, H_f, H_g, H_i, R)$ in which:

S is a finite set of internal states of the finite state machine
I is a finite set of input states of the finites of the FSM, which either are symbolic or are represented as a binary vector of values of its input signals.
O is a finite set of output states of the FSM, which are either symbolic or are represented as binary vector of values of its output signals.

$F(i,s)$ is a relation from the (input state, present state) pairs, also called total states, to the next state $(i.e..F \subseteq I \times S \times S)$.

Manuscript published on 28 February 2018.
***** Correspondence Author (s)
Dr. Diouba Sacko*, Département Génie Industriel, Ecole Nationale d'Ingénieurs – Abderhamane Baba Touré (ENI-ABT), Bamako, République du Mali. (e-mail:sacko_dioba@hotmail.com).
Dr. Drissa Traoré, Département Génie Industriel, Ecole Normale d'Enseignement Technique et Professionnel (E.N.E.T.P), Bamako, République du Mali. (e-mail: Drissatraore2001@yahoo.fr).
Dr. Marie Bernard, Département Génie Industriel, Ecole Nationale d'Ingénieurs – Abderhamane Baba Touré (ENI-ABT), Bamako, République du Mali.

$G(i,s)$ is a relation from the input (input state, present state) pairs to the output states $(i,e..G \subseteq I \times S \times O)$.

$S_0 \subseteq S$ Is a set of initial (or reset) states?

R is a finite set of reconfiguration states of the FSM, which are symbolic or represented as a binary vector of values of its reconfiguration input signals.

The transition reconfiguration function $H_f(r), r \in R$ is mapping from the reconfiguration state to the total state $F(i',s), i' \in I, s \in S, i.e..F(i',s)$ may reconfigured by an update as follows:

$F(i',s) := H_f(r)$.

The output reconfiguration function $H_g(r), r \in R$ is a mapping from the reconfiguration state to the output state $G(i',s), i' \in I, s \in S, i.e..\ G(i',s)$ may be reconfigured by an update as follows:

$G(i',s) := H_g(r)$.

$H_i(i,r), i' \in I, r \in R$ is a mapping from the (input state, reconfiguration state) tuple to the internal input state $i' \in I$.

The concept is shown in the schematic of figure 1 .in this schematic the transition function F may be updated by H_f and the output function G may be updated by H_g. H_f and Hg depend on the reconfiguration state, whereas H_i depends on the input state I and the reconfiguration state R. The function H_i is defined such that during normal operation of the finite state machine, I' = I and during the reconfiguration process, I' is depending on r only.

Reconfiguration program

Given an FSM M and a FSM M' it's possible to specify a finite sequence of reconfiguration transition in order to transform M into M'.

Consider a given state machine M we want to reconfigure into a target state machine M'. First we need to know what transitions have to be reconfigured when migrating M to M'.

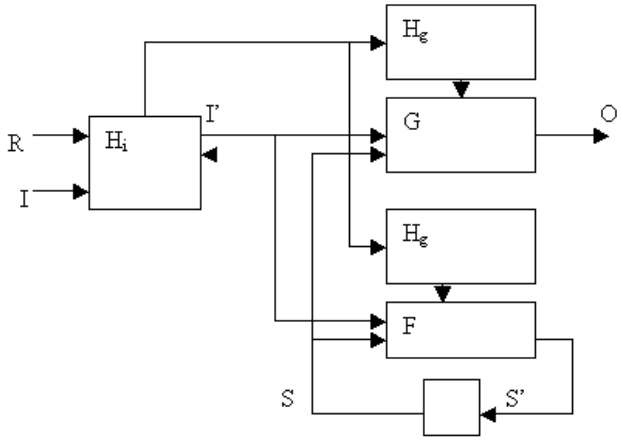
Given a given finite state machine M= (I,O,S,S_0,F,G) and a target finite state machine M'= (I',O',S',S'_0,F',G') . Let T'= $\{(i,s_x,s_y,o): i \in I', s_x \in S, s_y = F(i,s_x)$, $o = G'(I,S_x)\}$ denote the total set of transition of M', then a transition $t_d = (i,s_x,s_y,o) \in T$ is called delta transition and needs to be reconfigured in order to mutate M into M' if at least one of the following conditions hold:

$-i \notin I$ Or $s_x \notin S$ or $s_y \notin S$, or $o \notin O$ or

$-s_y \neq F(i,s_x) \wedge i \in (I \cap I') \wedge s_x \in (S \cap S')$, Or

$-o \neq G(i,s_x) \wedge i \in (I \cap I') \wedge s_x \in (S \cap S')$.

Let $T_d \subseteq T'$ be the set of all delta transition.



n

Figure 1: Schematic of reconfigurable FSM

IV. RECONFIGURATION FINITE STATE MACHINES IN FPGA

Reconfigurable finite state machine are used in some applications. In FPGA based design for control circuit, the reconfigurable hardware approach is like virtual memory management approach and the control circuit is modeled in form of hierarchical finite state machine (HFSM).

To achieve reconfigurable FSM hardware, one of the first utilized methods consists to model the hardware in a form of virtual memory management (figure 5). These hardware templates are reprogrammable blocks (RB), which can be reloaded if necessary during run time. The template design which provides a set of capabilities, who has simplified the design of dynamically reconfigurable HFSM , must be composed of synthesizable and parametrable parts.

Templates for design have the following components:

1) The reconfigurable area defines windows in the FPGA configuration address space, which has been preliminarily allocated for implementation replaceable subalgorithms.
2) The synchronization circuit ensures the correct behavior of the control circuit; it can generate interruption to the host.
3) The stack replaces the traditional state registers of ordinary FSM. It supports the hierarchy.
4) Coder/Decoder: The coder allows a smaller word stack to be used through the binary encoding of one hot state within any hierarchical transition. The coder is active only during a call of a sub algorithm. The decoder performs reverse operation.

Registers

Have mapping registers (algorithm register and identification register). They propose an implementation of self-reconfigurable state machine in FPGA. Figure -3 shows the schematic of the hardware implementation of self reconfigurable FSM. The reconfiguration generates two signals rst and rst state. F-RAM and G-RAM are memory devices, which implement the reconfigurable transition and output function. The state register ST-REG output the current states. It is updated on every rising clock edge by the next state.

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

The reset multiplexer RST-MUX is used to force the next state to the reset state in independence of the reset state. In fact, it is always possible to enter the reset state on the next clock edge.

When FSM works in normal mode, the multiplexer IN-MUX selects the external input i. So the address of the memory blocks F-RAM and G-RAM depend on the external input the current state and I. G-RAM generates the output whereas F-RAM generates the next state if the reset state is not forced. In the case of FSM in the reconfiguration mode, the multiplexer IN-MUX selects the signal ir generated by the block reconfiguration. Then, the address of the memory block F-RAM and G-RAM depends on ir and the current state, so the external input does not affect the machine. Depending on the reconfiguration state r, the reconfiguration generates new values for ir, Hf, Hg to update the corresponding memory location in F-RAM and G-RAM .The block generator is realized using logic block whereas F-RAM and G-RAM have been realized in embedded memory block.

This implementation has some advantage because the reconfiguration function is independent on the placement and routing of the hardware on the FPGA. This is in contrary, where the placed circuit has to generate reconfiguration sequences in the form of technology dependent bit streams.

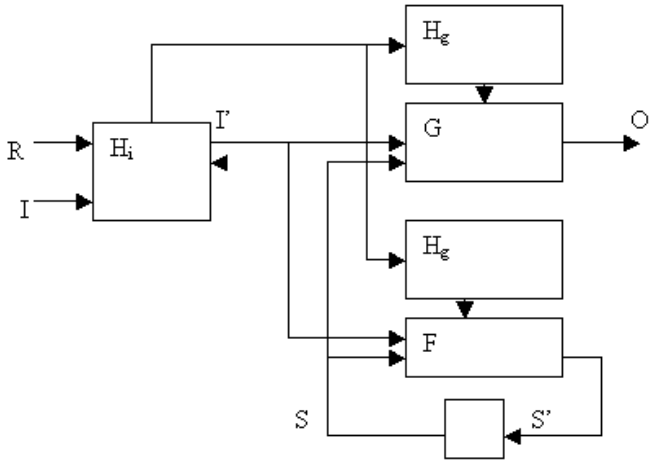


Figure 2: Template structure

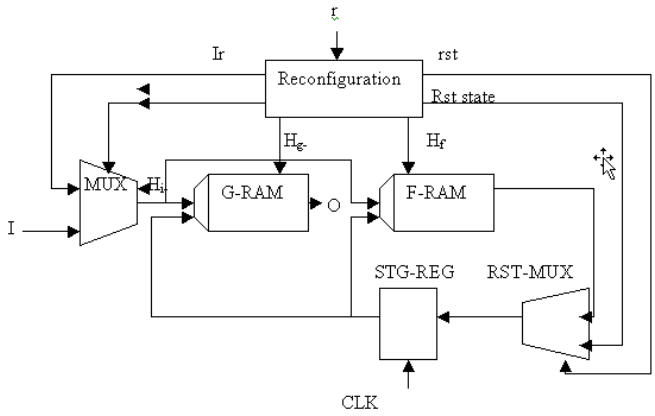


Figure 3: implementation of reconfigurable FSM

V. VLSI ARCHITECTURE FOR PROGAMMABLE FSM

Presents architecture of an IP address of an lookup engine based on programmable finite state machines where the lookup problem is translated into the implementation of a large FSM. This large FSM is broken down into a set of smaller FSMs, which are then mapped into reconfigurable hardware blocks (see Figure 4)

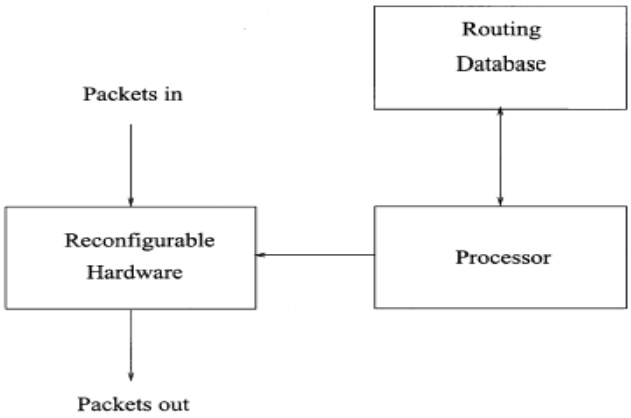


Figure 4: lookup Engine architecture

A reconfigurable hardware is essentially a circuit whose behavior can be modified on the fly. The hardware implementation is in the form of a programmable FSM. A processor can load the state transition table onto it. The reconfigurable hardware has been implemented with a double PLA structure. The programmability of the architecture is obtained by introducing memory elements.

VI. SUGGESTION FOR VLSI DYNAMICALLY RECONFIGURABLE

Static RAM programming technology is achieved with the help of pass transistors, pass gates, and multiplexers. A pass transistor can be used in order to realize a programmable interconnection between two wires. The pass transistor turns on or off depending on the control value. When it turns off, it presents a very high resistance and thereby an open connection between the two wires and when turned on, it forms a relatively low resistance and thereby a closed connection between the two wires. The values to control the programmable points are stored in a static RAM, which can be altered according to specific requirements. This basic principle being used in field programmable gate array being manufactured by xilinx and others. Since the static RAM cell contents can be altered several times, the configuration of the interconnect can be varied any number of times. RAM cell could be configured very quickly while in circuit it can be produced using the standard CMOS process technology. Current FPGA technology does not place a strong emphasis on high reconfiguration speed. For complex application the required time for configuration of current FPGA is in order of (ms). Designers continue to develop systems that demand high speed device configuration and that work at wire speed. FPGA vendors will need faster reconfiguration mechanisms. So for high performance reconfigurable finite state machine, it will be more efficient if the functional blocks are mapped with programmable PLA (with memory cells) and the reconfigurator with reconfigurable logic. In that case, the reconfigurable logic is composed of an array of basic logic cells.

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

A basic cell is an input lookup table with a multiplexer. Decomposed of a large finite state machine into many smaller; each will be mapped with self-reconfiguration

VII. CONCLUSION

Many research projects have been carried out, however reconfigurable FSM applications are still in its infancy. There are a number of area where reconfigurable finites must continue to develop such as networking in packet processing. In this paper, we have make a survey of reconfiguration FSM and shown the future direction which can be of great importance i.e. VLSI architecture for dynamically reconfiguration. The key technical difficulties that need to be addressed here are the reconfiguration mechanism and the speed of the reconfiguration.

REFERENCES

1. Venla Tapatho N. Rayapati and Bozena K Aminska 'dynamic reconfigurable schemes for megabit BiCMOS SRAMs and performance evaluation" microelectronic reliab vol 37 No 5pp
2. Yukio MITSUYAMA, Zaldy andales, tAKAO onoye, and Isao SHIRAKAWA"VLSI 1mplementation of dynamically reconfigurable hardware –based cryptosystem"2000 symposium on VLSI circuit digest of technical papers.
3. Jurgen Becker, THILO Pionteck, Christian Haberman, Manfred Glesner" design and implementation of a coarse grained dynamically reconfigurable Hardware architecture" 2001 IEEE
4. H.ITO R.Konishi, H.Nakada K.Oguri A.Nagoya "Dynamically reconfigurable Logic LSI- PCA-1"2001 symposium on VLSI Circuits Digest of technical papers.
5. Sayfe Kiaei, Jaisimha K.Durgam "VLSI design of dynamically reconfigurable ARRAY"1989 IEEE.
6. Lizy Kurian John, Eugence John" a dynamically reconfigurable Interconnects for array processors."IEEE Transaction on VLSI system 1998.
7. T.Fujii et al " a dynamically reconfigurable logic engine with a multicontex/multi –mode unified cell architecture," in ISSCC Digest of technical papers, pp364-365, Feb 1999.
8. V.Sklyarov "reconfigurable models of finite state machines and their implementation in FPGAs"journal of systems architectures 47(2002) 1043-1064
9. Markus Koster, Jurgen Teich" selfreconfiguration finite state machine theory and implementation " proc of the 2002 design, automation and test, Europe conference and exhibition IEEE.
10. Armaldo Oliveiro Valery Sklyarov" implementation of virtual control circuit in dynamically reconfigurable FPGAs' IEEE
11. Madhav Desai Ritu Gupta, Abhay Karandikar Kshitiz Saxena and Vinayak Samant."Reconfigurable finite state machine based IP lookup engine for high speed router"

AUTHOR PROFILE

Diouba Sacko PhD in Communication and Information Systems / MANETs, publications : " Overview of Reference Region Group Mobility Model for Ad Hoc Networks", "A Survey of Group Merge and Split Mobility Models", "Dynamic Group Mobility Model for Hybrid Networks", "Group Partitioning and Merging Mobility model for MANETs", "Autonomic Group Mobility for Mobile Ad Hoc Networks", "Study on Indoor and Outdoor environment for Mobile Ad Hoc Network: Random Waypoint Mobility Model and Manhattan Mobility Models ", "A Survey of Geographic Restriction Mobility Models", "A Survey of Geographic Restriction Mobility Models", "Link analysis on Indoor and Outdoor environment for Mobile AdHoc Network using Random Waypoint Mobility Model and Manhattan Mobility Model", "Opportunistic Virtual Antenna Arrays with Optimal Relay Mobile Terminals Selection", research work: "Group Partitioning and Merging Mobility model for MANETs".

Drissa Traoré is graduated from Microelectronics of Harbin Institute of Technology, People's Republic of China, Research work: VLSI Implementation of Reconfigurable Finite State Machine. Currently, he is Teacher-Researcher at the Normal School at Bamako, Republic of Mali (West Africa).). In addition, he is head of the Department of industrial engineering. Publications: "ASICON IEEE 2005, 734-738", "HIT journal 2006 13(3) 251-254", "WSEAS Transanction on Circuits and System 2006 5(3) 640-646".

Marie Bernard Sidibé is graduated Doctor in Robotic from Mecatronics of Harbin Institute of Technology, People's Republic of China, Research work: No inverted pendulum robot; Nano robots. Publications: "Kinematics of wheeled mobile robots"; "New steering mechanism for wheeled mobile robots" ; "Maneuver algorithm and kinematical energy analysis of a robot based on instantaneous center of rotation";"General maneuver algorithm description and typical mode analysis for a four-steer-four-drive mobile robot".

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP)