Chapter 1

Introduction

1.1 Motivation

In 1992, the automatic design of electronic circuits was introduced at Electrotechnical Laboratory (ETL), Tsukuba, Japan (Higuchi, 1992). The genetic algorithms (GA), which is an evolutionary algorithm, is applied to automated circuit design. The complexity of the design process is hidden behind the evolutionary process – circuit designer specifies "what is to be done" rather than "how to do it". The design constraints (e.g. circuit function, silicon area, and power consumption) are given to a computer program returning a desired solution. Following that, the final solution could be further implemented in the actual circuit. It can be seen that the final circuit cannot change itself further, and thus this approach is named as *off-line evolvable hardware (off-line EHW)* or sometimes called *evolutionary circuit design*.

In conventional hardware design, the hardware performs a permanent function. This is contrary to the *field-programmable gate array (FGPA)* which enables the ability to change its structure (Villasenor, 1997). The FPGA consists of an array of configurable logic blocks (CLBs) and the configurable interconnections of CLBs. The CLB functions and the interconnections are determined by the configuration bits stored in a static RAM. Therefore the FPGA is a run-time reconfigurable device. The current FPGA technology allows the final solution can be implemented immediately since the reconfiguration time is about milliseconds. The FPGA is very helpful to build a system which can automatically adapt itself.

The ability of the circuit to change its structure is necessary for a task of which the function cannot be predefined or changes during its lifetime. In many scenarios, the operational environment is unpredictable, for example, space explorations, noisy domains, and human-interactive systems. As a result, those operations highly require the adaptability. In addition, the adaptability enables the fault tolerance by means of partially reconfiguring

itself to avoid the bad areas on a chip. It is more economic than the traditional approach using redundancy technique and spare parts. Because the adaptation takes place in the actual hardware, this approach is named as *on-line evolvable hardware (on-line EHW).*

The on-line EHW is becoming more approachable for applications requiring flexibility, compactness, and low power-consumption. The ability to be reconfigured allows the circuit to be more flexible to cope with the uncertain environment. For a particular task, all modules in the circuit are not used simultaneously. The task can be accomplished by reconfiguring the modules demanded for an instant use. By this way, the reconfigurable circuit would be more compact than the traditional circuit composed of all necessary modules. Consequently, the power consumption of a smaller circuit would be lower than a bigger one.

In recent years, the promising applications of EHW have been put forward. (Mizoguchi, 1994; Sakanashi, 1996; Higuchi, 1997; Sipper, 1997a; Sipper, 1997b; Koza, 1997; Thompson, 1999; Higuchi, 1999, Lohn, 1999; Kasai, 2000). This has great potential for the real-world applications.

1.2 The Purpose of the Study

The thesis aims to demonstrate an on-line EHW, called *mimetic EHW*. The task is set to mimic a sequential circuit from its partial input/output sequences. Given partial input/output sequences, the mimetic EHW can resemble the behavior of the target circuit. Note that only partial input/output sequences can be observed, and the internal states are hidden. This problem is now well established to be a hard computational problem, see for example (Rivest, 1994).

The task of mimicking sequential circuits was studied in software. The genetic algorithms (GA) (Goldberg, 1989) was used to search for the circuit satisfying the input/output sequences collected from the target circuit (Manovit, 1998; Chongstitvatana, 1999). The software version is not practical for on-line applications due to several reasons.

- The software is an *off-line* process, and therefore the evolved solution cannot be used immediately.
- The GA is time-consuming. About 90% of execution time is spent to evaluate the candidate solutions. Moreover, the evaluation time increases rapidly with the circuit size.
- The software running on a personal computer (PC) is impractical for portable applications in which the compactness is essential.
- The PC, consisting of a high-performance CPU and peripheral devices, consumes a lot of energy. The normal-size battery cannot supply the PC for day and night.

As a result, the design of mimetic EHW is emphasised on on-line, efficiency, compactness, and power consumption.

The mimetic EHW is implemented on the FPGA prototyping boards consisting of a custom microprocessor and a fitness evaluator. The microprocessor and the fitness evaluator are designed using *Verilog* hardware description language (Verilog HDL). The microprocessor is particularly designed for an execution of GA. The fitness evaluator acts as a coprocessor, accelerating the fitness evaluation which is a bottleneck of GA. Following that, the designs are realised on Xilinx XC4010 FPGAs.

1.3 The Scope of the Study

This work concentrates on the demonstration of mimetic EHW. The size of the target circuit is limited at 16-state finite-state machine (FSM) which has 4-bit inputs and 4-bit outputs. The scope does not include the theoretical aspects such as the complexity of the problem or the circuit verification in case the target circuit is unknown.

1.4 Thesis Outline

The remainder of this thesis is organised as follows. In chapter 2, the simple GA is presented as most applications of EHW heavily rely on this algorithm. Chapter 3 presents

the literature on EHW and a review of hardware-based GA. Chapter 4 describes my initial work which is an inspiration of the mimetic EHW. Chapter 5 describes the GA conducted in software. The software version, presented in this chapter, is migrated to the FPGA prototyping boards composed of the custom icroprocessor and the fitness evaluator. Chapter 6 presents the hardware organisation. The microprocessor and the fitness evaluator are described in this chapter. Chapter 7 presents the performance analysis of the mimtic EHW. Chapter 8 concludes the thesis.