An Asynchronous Field-Programmable VLSI Using LEDR/4-Phase-Dual-Rail Protocol Converters

Shota Ishihara, Yoshiya Komatsu, Masanori Harioyama, and Michitaka Kameyama
Graduate School of Information Sciences, Tohoku University
Aoba 6-6-05, Aramaki, Aoba, Sendai, Miyagi, 980-8579, Japan
E-mail: {ishihara@kameyama., ykomatsu@kameyama., hariyama@, kameyama@}`cei.tohoku.ac.jp

Abstract—This paper presents a novel architecture that combines 4-phase dual-rail encoding and Level-Encoded Dual-Rail (LEDR) encoding. 4-phase dual-rail encoding is suitable for function units because of its small area, while LEDR encoding is suitable for programmable interconnection resources because of its high throughput and low power. Area-efficient protocol converters are proposed based on transistor-level optimization. The proposed architecture is designed using a 90nm CMOS process. Compared to the 4-phase dual-rail architecture, the throughput and the power consumption are respectively by 45% higher and by 36% lower with almost the same transistor count. Compared to the LEDR architecture, the transistor count is by 35% lower with almost the same throughput and power consumption.

Keywords—FPGA, Reconfigurable VLSI, Self-timed architecture, Delay-insensitive architecture.

I. INTRODUCTION

Field-programmable gate arrays (FPGAs) are widely used to implement special-purpose processors. FPGAs are cost-effective for small-lot production because functions and interconnections of logic resources can be directly programmed by end users. Despite their design cost advantage, FPGAs impose large power consumption overhead compared to custom silicon alternatives [1]. The overhead increases packaging costs and limits integrations of FPGAs into portable devices.

In FPGAs, the clock distribution power is a serious problem because it has an enormously larger number of registers than custom VLSIs. The most well-known technique to reduce the clock distribution power is clock gating. In ASICs, gated clock which is a kind of clock gating is employed. As shown in Fig. 1, the idea of gated clock is to gate the clock network in the sleep state. The clock-gating can result in saving two kinds of power consumption:

1) Dynamic power consumption of the clock network
2) Dynamic power consumption of registers of a circuit and gates in the fan-out of the registers

On the other hand, clock gating in conventional FPGAs has to be implemented with a different and compromised method. In FPGAs, the customized clock network can be implemented using the programmable interconnects. However, the worst case of clock skew can’t be estimated since FPGA vendors don’t guarantee the worst case of the minimum delay of components. As a result, it is impossible to guarantee that no hold-time violations occur [2]. For this reason, reference the Xilinx ISE manual, the clock-gating method without the customized global clock is recommended. In FPGAs, instead of gated clock, circulation is employed. As shown in Fig. 2, the main idea of circulation is to retain to contents of the flip-flop in the sleep state [3]. Therefore, clock gating in FPGAs only reduces the dynamic power consumption of registers of a circuit and gates in the fan-out of the registers. However, dynamic power consumption of the clock network cannot be reduced.

To cut the clock distribution power, asynchronous FPGAs are proposed. References [4] and [5] propose asynchronous FPGAs based on dual-rail encoding. They use 4-phase dual-rail encoding because of relatively small overheads. However, in the 4-phase dual-rail encoding, a spacer must be inserted between two consecutive data values. This results in the low throughput and high dynamic power consumption of the programmable interconnection resources because of the large number of signal transitions.

Another well-known dual-rail encoding is Level-Encoded Dual-Rail (LEDR) [6] which requires no spacer. Its throughput is double and its dynamic power consumption of programmable interconnection resources is half of the 4-phase dual-rail encoding since the number of signal transitions is half of 4-phase dual-rail encoding. The major drawback of the LEDR encoding is its larger overheads in area. Based on this observation, we have used LEDR encoding and proposed its area-efficient implementation for fine-grain FPGAs [7], [8]. However, LEDR encoding is not suitable for implementing coarse-grain architecture because of the large area of function.
unit.

In order to archive both of low power and high throughput, reference [9] proposed a system-on-chip (SoC) architecture that combines 4-phase dual-rail encoding for function units and LEDR encoding for interconnection resources. However, the circuits are only optimized at the gate level and the overhead is large. In SoCs, the overhead is relatively small. However, it is not suitable for FPGAs since every logic block must have 4-phase dual-rail and LEDR protocol converters.

In this paper, we propose an asynchronous Field-Programmable VLSI (FPVLSI) using area-efficient 4-phase-to-LEDR converters, LEDR-to-4-phase converters and their control circuits. The circuits are optimized at the transistor level. Moreover, to reduce the overhead, the proposed a 4-phase-to-LEDR converter can also used as an output register. The proposed architecture is designed using a 90nm CMOS process. Compared to the 4-phase dual-rail architecture, the throughput and the power consumption are respectively by 45% higher and by 36% lower with almost the same transistor count. Compared to the LEDR architecture, the transistor count is by 35% lower with almost the same throughput and power consumption.

II. ARCHITECTURE

A. Asynchronous protocols

Asynchronous encoding schemes are mainly classified into

- Single-rail encoding (ex. bundled-data encoding)
- Dual-rail encoding
  (ex. 4-phase dual-rail encoding, LEDR encoding)

The bundled-data encoding is most common one in the single-rail encoding. Figure 3 shows a simple bundled-data pipeline. In the bundled-data encoding, request and value are splits into separate wires. The value is encoded as in a synchronous circuit using N wires to denote a N-bit number, and request is encoded using a dedicated request wire denoted by REQ. The bundled-data encoding requires the explicit insertion of delay in REQ to ensure that a request is never received before the bundled value is valid. The bundled-data encoding is the most frequently-used way in ASICs since its hardware overhead is relatively small. This is because the REQ wire is shared among all the N wires. Hence, to transfer an N-bit value, only N + 2 wires are required. The major disadvantage is that it requires the constraint of the delay length. If the data path is fixed in advance, it is relatively easy to meet the constraint by optimizing layouts of wires. On the other hand, in reconfigurable VLSIs such as FPGAs, it is not easy to always meet the constraint since the data path is programmable.

The dual-rail encoding encodes a bit onto two wires. Figure 4 shows a simple dual-rail pipeline. In dual-rail encoding, value is made implicit in the request and no delay insertion is therefore required [10]. Hence, the dual-rail encoding is the ideal one for reconfigurable VLSIs. In the dual-rail encoding, to transfer an N-bit value, 2N + 1 wires are required.

There are two major methods for dual-rail encoding:

- 4-phase dual-rail encoding
- Level encoded 2-phase dual-rail encoding (LEDR encoding)

4-phase dual-rail encoding is the most common one in dual-rail encodings. Table I shows the code table of 4-phase dual-rail encoding. The data value 0 is encoded as (0, 1) and 1 is encoded as (1, 0). Moreover, the spacer is encoded as (0, 0). Figure 5 shows the example where data values 0, 0 and 1 are transferred. The main feature is that the sender sends spacer after a data value. The receiver knows the arrival of a data value by detecting the change of either bit: 0 to 1. The insertion of spacers makes the encoding law simple. This results in a simple hardware of the function unit. Figure 6 and figure 7 show the XOR gates of synchronous (synchronous XOR) and 4-phase dual-rail (4-phase dual-rail XOR) architecture respectively. The circuit for generate out.t of the 4-phase dual-rail XOR gate is similar to the pMOS network of the synchronous XOR gate. On the other hand, the circuit for generate out.f of the 4-phase dual-rail XOR gate is similar to the nMOS network of the synchronous XOR gate. This similarity causes the function unit of 4-phase dual-rail architecture to implement easily. The number of the transistors of the synchronous XOR gate is 12, while the 4-phase dual-rail XOR gate is 16. Accordingly, the hardware overhead of 4-phase dual-rail architecture is smaller than that of other dual-rail architecture. However, the insertion of spacer increases the number of signal transitions. As a result, the throughput of data transfer is low. Moreover, the power consumption of the programmable interconnection resources is large.

LEDR encoding is one of many 2-phase dual-rail encodings. In LEDR encoding, no spacer is required. Table II shows the code table of LEDR encoding. Note that each data value has two types of code words with different phases. For example,
TABLE I
CODE TABLE OF 4-PHASE DUAL-RAIL ENCODING

<table>
<thead>
<tr>
<th>Code word (T, F)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data 0</td>
</tr>
<tr>
<td>Data 1</td>
</tr>
<tr>
<td>Spacer</td>
</tr>
</tbody>
</table>

Fig. 5. Example of 4-phase dual-rail encoding

data value 0 is encoded as (0, 0) in phase 0 and (0, 1) in phase 1. The code word consists of V(Value bit) and R(Redundant bit). The value V is encoded as in a synchronous circuit. The redundant bit R is defined by XOR-ing V and Phase so that R includes the information on phase and consecutive code words get different only by hamming distance 1. Figure 8 shows the example where data values 0, 0 and 1 are transferred. The main feature is that the sender sends data values alternately in phase 0 and phase 1. The receiver knows the arrival of a data value by detecting the change of phase, and data values are continuously transferred between the sender and the receiver without any break. Because no spacer is required, the number of signal transitions is half of 4-phase dual-rail encoding. As a result, throughput of data transfer is high and the power consumption of the programmable interconnection resources is small. However, the hardware cost of the function unit is large since LEDR encoding has two phases for each value. In fine-grain architectures such as the function unit of each logic block is a 2-input-and-1-output LUT, the hardware overhead is small, but in more coarse-grain architectures, the overhead is large.

Fig. 6. XOR gate of synchronous architecture

![Diagram](image)

TABLE II
CODE TABLE OF LEDR ENCODING

<table>
<thead>
<tr>
<th>Code word (V, R)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Phase 0</td>
</tr>
<tr>
<td>Data 0</td>
</tr>
<tr>
<td>Data 1</td>
</tr>
<tr>
<td>Phase 1</td>
</tr>
<tr>
<td>Data 0</td>
</tr>
<tr>
<td>Data 1</td>
</tr>
</tbody>
</table>

B. Overall architecture

Figure 9 shows the overall architecture of the proposed FPVLSI. The FPVLSI consists of a mesh-connected cellular array like conventional FPGAs. As mentioned in the previous section, 4-phase dual-rail encoding is suitable for function units because of its small area, while LEDR encoding is suitable for programmable interconnection resources because of its high throughput and low power. The proposed FPVLSI combines 4-phase dual-rail encoding for function units and LEDR encoding for interconnection resources. Figure 10 shows a simplified logic block structure. The logic block mainly consists of a LEDR-to-4-phase converter, a function unit and a 4-phase-to-LEDR converter. The 4-phase-to-LEDR converter is unified with an output register. 4-phase dual-rail encoding is used for local operation, while LEDR encoding is used for global communication.

Phase 0 Phase 1 Phase 0
Data 0 Data 0 Data 1

![Diagram](image)

Fig. 7. XOR gate of 4-phase dual-rail architecture

Fig. 8. Example of LEDR encoding

![Diagram](image)
C. Logic block structure

Figure 11 shows the detailed structure of the logic block. A converter controller and a data arrival detector are used to control the LEDR-to-4-phase converter and the 4-phase-to-LED converter.

Figure 12 shows the structure of the function unit. The function unit mainly consists of an LUT and a carry logic. It can execute an arbitrary 4-input-and-1-output boolean function or perform as a full adder.

Figure 13 shows the structure of the converter controller. The converter controller is similar to a C-element. The behavior of converter controller is as follows. When the operation is complete, the first transmission gate (TG1) turns on. As a result, if the phases of the inputs (Phase_in) matches the required phase from next logic block (Ack_in) and the operation is completed, Phase_LED turns to the same as Phase_in. To retain Phase_LED when the operation is not completed, an SRAM cell (Master) is used. Signal Phase_LED is used for generating an acknowledge signal to the previous logic block, controlling the 4-phase-to-LED converter and detecting the data arrival. After the operation completion, a spacer phase for 4-phase dual-rail encoding is generated for the function unit. To guarantee the new data don’t input to the function unit before the spacer phase complete, Phase_interim is used. Signal Phase_interim is used to generate the phase of 4-phase dual-rail encoding and as an input of the data arrival detector. Signal Phase_interim turns to the same as Phase_LED when the spacer phase is completed. To retain Phase_interim when the spacer phase is not completed, an SRAM cell (Slave) is used.

Figure 14 shows the structure of the data arrival detector. Signal Phase_in is used as an input of the converter controller. It changes when the phases of all inputs match. It remains in this state until the phases of all inputs transit to the other state. Signal Phase_4-phase is the phase of 4-phase dual-rail encoding. It is used as a control signal of the LEDR-to-4-phase converter and as a pre-charge signal of the function unit.

Figure 15 shows the structure of the LEDR-to-4-phase converter. If the Phase_4-phase is “0”, all of the outputs turn to “0”. That is, the inputs of the function unit are spacers. If the Phase_4-phase is “1”, T is the same as V, while F is the
same as $\overline{V}$. As a result, all of the inputs which are encoded in LEDR encoding are converted to 4-phase dual-rail encoding.

Figure 16 shows the 4-phase-to-LEDR converter. The 4-phase-to-LEDR converter is also used as an output register. As the LEDR encoding is a dual-rail encoding, two double-edge-triggered flip-flops are required as a register to store a data value in the typical manner. To reduce this overhead, we have proposed a register for LEDR encoding only requires a single flip-flop [8]. When the operation is completed, Phase-LEDR changes. Then $Out_T$ is stored in an SRAM cell and $Out_V$ turns to the same as $Out_T$. The upper SRAM cell retains $Out_V$ if Phase-LEDR is “0”, and the lower SRAM cell retains $Out_V$ if Phase-LEDR is “1”. When Phase-LEDR is “0”, $Out_R$ is the same as $Out_V$. When Phase-LEDR is “1”, $Out_R$ is the same as $Out_V$. Therefore, $Out_R$ can be generated by the outputs of the two SRAM cells and Phase-LEDR. This register is used to convert the inputs encoded in 4-phase encoding to LEDR encoding.

III. EVALUATION

The proposed FPVLSI is designed using 90nm CMOS process with 1.0V supply. We compare the proposed FPVLSI to the FPVLSIs based on 4-phase dual-rail encoding and LEDR encoding. The circuits are simulated by HSPICE.

Figure 17 shows the comparison of transistor counts of a single cell. Compared to the 4-phase dual-rail architecture, the transistor count is increased to 145% because of the high throughput of the programmable interconnection resources. Compared to the LEDR architecture, the throughput is decreased to 78%.

Figure 19 shows the comparison of energy consumptions per data set of a single cell. Compared to the 4-phase dual architecture, the energy consumption is reduced to 64% because of the low power of the programmable interconnection resources.

![Fig. 14. Data arrival detector.](image1)

![Fig. 15. LEDR-to-4-phase converter.](image2)

![Fig. 16. 4-phase-to-LEDR converter.](image3)

![Fig. 17. Comparison of transistor counts.](image4)

![Fig. 18. Comparison of throughputs.](image5)
Compared to the LEDR architecture, the energy consumption is increased to 102%.

IV. CONCLUSION

This paper proposes an asynchronous FPVLSI that combines 4-phase dual-rail and LEDR encoding. 4-phase dual-rail encoding is suitable for function units because of its small area, while LEDR encoding is suitable for programmable interconnection resources because of its high throughput and low power. The key to success is the area-and-power-efficient 4-phase-to-LED converter which can also be used as an output register. As a future work, the development of the CAD tool for the asynchronous FPVLSI is now undergoing.

ACKNOWLEDGMENT

This work is supported by VLSI Design and Education Center (VDEC) with Cadence Design Systems Inc and Synopsys Inc.

REFERENCES