Logic Design of a 16-bit Bit-Slice Arithmetic Logic Unit for 32-/64-bit RSFQ Microprocessors

Guang-Ming Tang, Pei-Yao Qu, Xiao-Chun Ye, and Dong-Rui Fan

***Abstract*—A 16-bit bit-slice arithmetic logic unit (ALU) is proposed for 32-/64-bit rapid single-flux-quantum microprocessors. It is based on a Ladner–Fischer adder. The ALU covers all of the ALU operations for MIPS32 instructions set. And each of the two 64-bit operands is dividedinto four slices of 16 bits each. The ALU uses synchronous concurrent-flow clocking and consists of 11 pipeline stages. The proposed ALU can be used for any 16*n*-bit processing.**

***Index Terms*—Arithmetic logic unit (ALU), microprocessor, rapid single-flux-quantum (RSFQ), superconducting integrated circuits.**

I. Introduction

THE continued reduction in integrated circuit (IC) feature has made CMOS integrated circuits faster but denser per unit area, well suited to the growing demand for microprocessor performance. However, the traditional semiconductor industry is facing the challenge of narrowing the size of a single device and increasing the integration of transistor per unit area. With the integrated circuit line width close to the atomic diameter, by 2021, if we continue to narrow the semiconductor integrated circuit line width, financially, is no longer desirable. Scientists have faced a series of challenge in terms of material and power consumption for faster computing microprocessors.

The superconductor circuit technology evolves from the rapid single-flux-quantum (RSFQ) [1] into energy-efficient circuit technologies, such as [2], that are expected to constitute a next-generation integrated circuit technology because of its ultra-high-speed computation with ultra-low-power consumption.

Several simple 8-bit RSFQ microprocessors have been designed, including an 8 × 1-bit serial microprocessor (FLUX-1) [3], bit-serial microprocessors (CORE1 series) [4]–[7], a bit-serial asynchronous microprocessor (SCRAM2) [8], and a serial 8-bit microprocessor (CORE e4) [9]. A prototype of a 32-bit superconducting microprocessor based on 4-bit bit-slice architecture has been presented at ASC 2016 [10].

Manuscript received September 19, 2017; accepted January 22, 2018. Date of publication January 31, 2018; date of current version February 15, 2018. This work was supported in part by the Pioneer Hundred Talents Program (C) of Chinese Academy of Sciences, China, under Grant 2017004, in part by the Foundation of Director of State Key Laboratory of Computer Architecture, ICT, CAS under Grant CARCH3101, and in part by the Key Program of the National Natural Science Foundation of China under Grant 61732018. *(Corresponding author: Guang-Ming Tang.)*

The authors are with the State Key Laboratory of Computer Architecture, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China (e-mail: tangguangming@ict.ac.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TASC.2018.2799994

ALU is the most important component of the microprocessor. It is used for the implementation of arithmetic and logic operations, even if the smallest microprocessor also contains ALU as a counting function of the components. In this paper, a logic circuit of the proposed ALU has been designed.

So far, several superconducting RSFQ ALUs have been developed and have been successfully demonstrated. An 8-bit parallel ALU was developed. It was demonstrated successfully at low frequency tests [11] and high frequency tests [12], respectively. An 8-bit parallel ALU with a target frequency of 30 GHz was developed [13]. A serial ALU with a frequency of 80 GHz was developed [14]. A 4-bit bit-slice ALU for a 32-bit RSFQ microprocessor was demonstrated successfully [15].

In this paper, we propose a logic design of 16-bit bit-slice ALU because that: 1) the serial and 2-/4-/8-bit bit-slice processing to calculate 64-bit data is slower than 16-bit bit-slice processing; 2) 32-bit bit-slice and 64-bit parallel processing to calculate 64-bit data needs larger hardware cost than 16-bit bit-slice processing. The ALU can complete all the arithmetic and logic instructions for MIPS32 instruction set. The logic design of the proposed ALU includes a carry feedback loop.

There are the following challenges:

1) try to simplify the superconducting RSFQ logic circuit;
2) try to minimize the delay of the carry feedback loop, because the carry feedback loop limits the clock frequency of the entire circuit;
3) try to simplify the superconducting RSFQ integrated circuit timing design.

In order to design the proposed 16-bit bit-slice ALU, the following methods have been adopted: 1) Algorithm level: Design an algorithm to reduce the number of feedback loop, 2) Logic level: Minimize the delay of the feedback loop.

The remainder of this paper is organized as follows. Section II describes the functions, architecture, and RSFQ logic design details of the ALU. Section III summarizes our findings and concludes the paper.

II. 16-Bit Bit-Slice ALU

For bit-slice adders, the carry-out of the most significant bit (MSB) of a slice must be the carry-in of the least significant bit of the next slice, so there must be a carry feedback loop. If the ripple-carry adder is used to implement the bit-slice superconducting RSFQ ALU, it takes a long time for the next slice to be fed by the carry feedback from the slice. And the pipeline stage

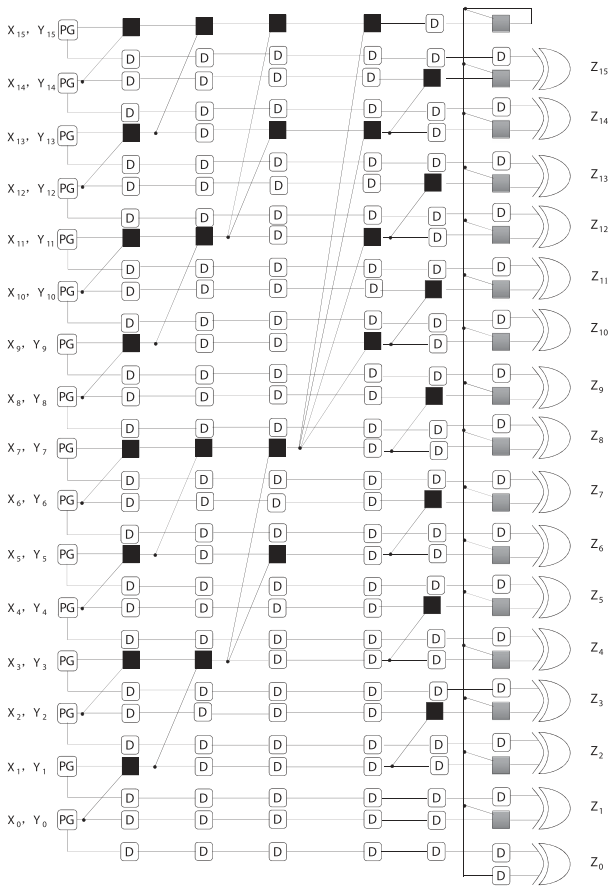

Fig. 1. The architecture of the 16-bit bit-slice Ladner-Fischer adder using RSFQ logic.

is large, resulting in much hardware cost. Therefore, we adopt a carry look-ahead (or parallel prefix) adder. Unlike the previously developed RSFQ 4-bit bit-slice ALU [15], the proposed bit-slice ALU continuously processes bit-sliced 32-/64-bit data divided into 2-/4-slice of 16 bits each in order to increase the throughput of microprocessors. We adopt Ladner-Fischer adder that has fewer pipeline stages than Brent-Kung adder, a lesser fan-out than Sklansky adder (from the fifth to sixth stage), and a smaller wiring than Kogge-Stone adder.

This circuit structure is advantageous for constructing an ALU of processing a 32-/64-bit data.

The 64-bit operand is divided into 4 slices of 16-bit each. The four pairs of operand slices are input one by one starting from the least significant bit (LSB). The ALU implements addition (ADD), subtraction (SUB), set on less than (SLT), AND, OR, NOR, exclusive OR operation (XOR), as well as comparison of equality of operands (EQ) in order to implement branch and jump instructions. The logic circuits of the proposed ALU include control signals, inversion operand, calculation of propagate (P) and generate (G), PG carry network, an ALU operation selection (a type of operation for selecting a logic operation or an arithmetic operation), calculation carry, and calculation result output.

Fig. 1 shows the architecture of the 16-bit bit-slice Ladner-Fischer adder using RSFQ logic. In order to ensure the synchronization of each slice, D flip-flops (DFFs) are required because the clocked logic gates have latch function in RSFQ circuits. There are 8 steps in this PG carry network.

Fig. 2 shows an RSFQ logic design of the proposed 16-bit bit-slice ALU for processing 64-bit data in an 11-stage pipeline. It is based on Ladner-Fischer adder. Concurrent-flow clocking is employed.

The ALU has 32 input ports for a pair of operand slices, i.e., $X_0, X_1, \ldots X_{15}, Y_0, Y_1, \ldots Y_{15}$, and seven ports for control signals (*Op_ARITH*, *Op_AND*, *Op_XOR*, *Cmpl_X*, *Cmpl_Y*, *Carry_in*, and *End_bar*). And it has 16 output ports for a resultant slice, i.e., $Z_0, Z_1, \ldots Z_{15}$ and a port called *Carry_out* for the result of unsigned SLT and EQ.

To simplify the feedback loop, the circuit for producing the carry from the MSB of a slice is duplicated. Computation with *Carry_in*, and *End_bar* is performed in advance, before entering the feedback loop.

Here, the logical expressions $\wedge$, $\vee$, $\oplus$ and $^-$ represent logical AND, OR, XOR and NOT respectively. For ADD operation, the processing for a slice pair in a 16-bit bit-slice Ladner-Fischer adder can be divided into the following ten steps:

1) The slice pair is input to the adder from the input ports (i.e., $X_0 - X_{15}$ and $Y_0 - Y_{15}$) and the PG signals (i.e., $P_{0:0}, P_{1:1}, P_{2:2}, P_{3:3}, P_{4:4}, P_{5:5}, P_{6:6}, P_{7:7}, P_{8:8}, P_{9:9}$, $P_{10:10}$, $P_{11:11}$, $P_{12:12}, P_{13:13}$, $P_{14:14}$, $P_{15:15}, G_{0:0}$, $G_{1:1}$, $G_{2:2}$, $G_{3:3}$, $G_{4:4}$, $G_{5:5}$, $G_{6:6}$, $G_{7:7}$, $G_{8:8}$, $G_{9:9}$, $G_{10:10}$, $G_{11:11}, G_{12:12}, G_{13:13}, G_{14:14}, G_{15:15}$) of this slice pair are calculated in the first and second stage.
2) In the third stage, the *Op_ARITH* signal is used to distinguish between arithmetic operations and logical operations. The code combination of *Op_AND* and *Op_XOR* signals are used to execute an arithmetic or logic instruction. See Table I for other corresponding ALU instruction codes.
3) $P_{1:0}$, $P_{3:2}$, $P_{5:4}$, $P_{7:6}$, $P_{9:8}$, $P_{11:10}$, $P_{13:12}$, $P_{15:14}$, $G_{1:0}$, $G_{3:2}, G_{5:4}, G_{7:6}, G_{9:8}, G_{11:10}, G_{13:12}$ and $G_{15:14}$ signals of the slice pair are calculated in the fourth stage.
4) $P_{3:0}, P_{7:4}, P_{11:8}, P_{15:12}, G_{3:0}, G_{7:4}, G_{11:8}$, and $G_{15:12}$ signals of the slice pair are calculated in the fifth stage.
5) $P_{5:0}, P_{7:0}, P_{13:8}, P_{15:8}, G_{5:0}, G_{7:0}, G_{13:8}$, and $G_{15:8}$ signals of the slice pair are calculated in the sixth stage.
6) $P_{9:0}, P_{11:0}, P_{13:0}, P_{15:0}, G_{9:0}, G_{11:0}, G_{13:0}$, and $G_{15:0}$ signals of the slice pair are calculated in the seventh stage.
7) $P_{2:0}, P_{4:0}, P_{6:0}, P_{8:0}$, $P_{10:0}, P_{12:0}, P_{14:0}, G_{2:0}, G_{4:0}, G_{6:0}$, $G_{8:0}, G_{10:0}, G_{12:0}$, and $G_{14:0}$ signals of the slice pair are calculated in the eighth stage.
8) The control signal *End_bar* determines whether or not the slice is the last one before the ninth stage, that is, before entering the feedback loop, and whether or not the "+1" operation is performed based on the carry signal *Carry_in.*
9) The carries (i.e., $C_0, C_1, \ldots C_{15}$) are calculated using the carry from the proceeding slice in the tenth stage, and $P_{0:0}, P_{1:1}, P_{2:2}, P_{3:3}, P_{4:4}, P_{5:5}, P_{6:6}, P_{7:7}, P_{8:8}, P_{9:9}$, $P_{10:10}, P_{11:11}, P_{12:12}, P_{13:13}, P_{14:14}$ and $P_{15:15}$ signals of the slice pair are moved into the DFFs. The carry C_{16} will be fed back to this stage for the next stage.

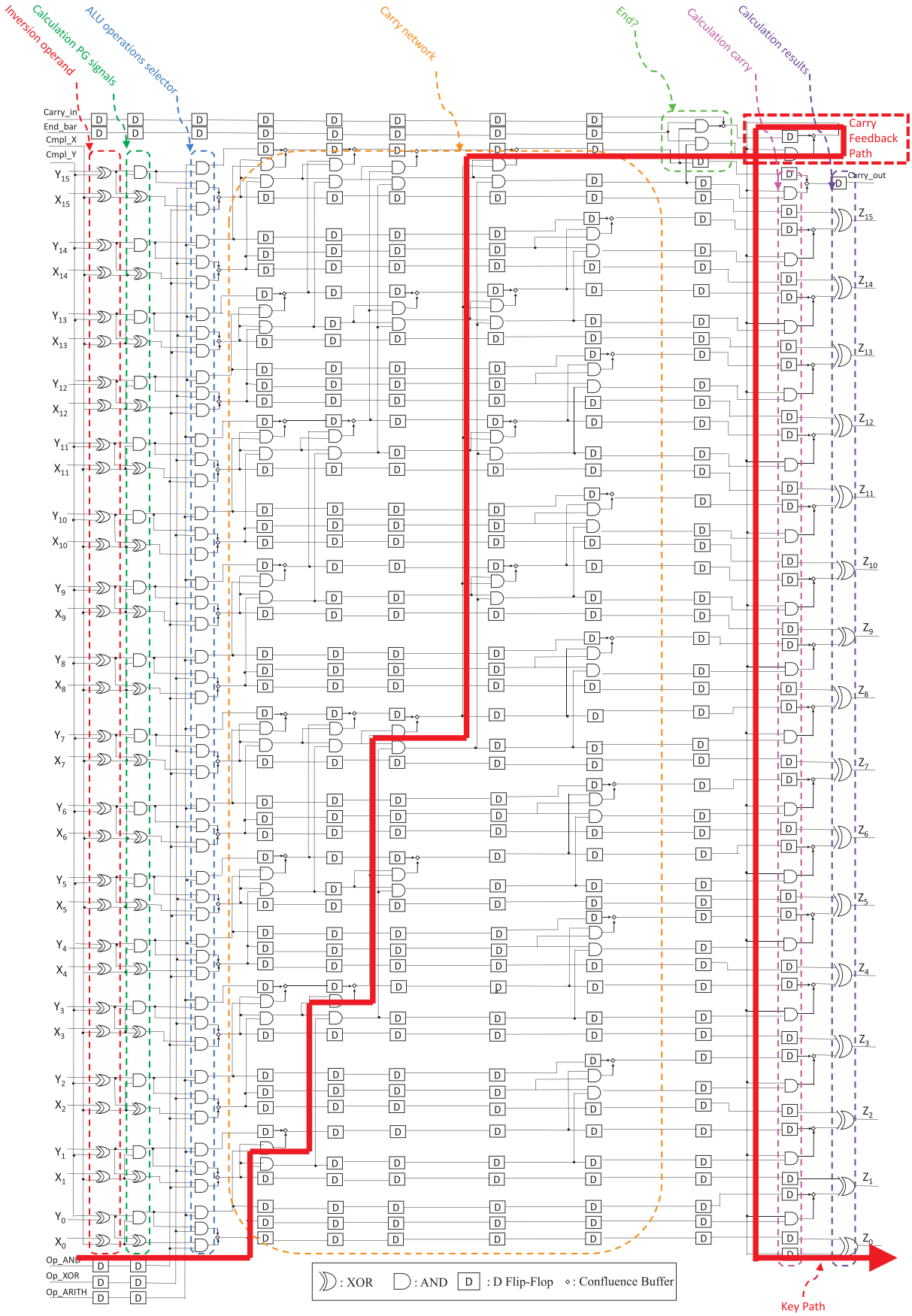

Fig. 2. An RSFQ logic design of the proposed 16-bit bit-slice ALU.

TABLE I
ALU INSTRUCTION SET

ALU Operations	Op_ ARITH	Op_ AND	Op_ XOR	Cmpl_ X	Cmpl_ Y	Carry_ in
ADD	1	0	1	0	0	0
SUB	1	0	1	0	1	1
SLT	1	0	1	0	1	1
EQ	0	0	1	0	1	1
AND	0	1	0	0	0	0
OR	0	1	1	0	0	0
XOR	0	0	1	0	0	0
NOR	0	1	0	1	1	0

TABLE II
COMPARISON OF ALUS BASED ON LADNER-FISCHER ADDER

Bit-slice size	#Logic and DFFs	#SPLs and CBs	pipeline stages	JJs	Clock delay (*ps*)	Latency @10 GHz (*ps*)
Bit-serial	33	15	6	356	>68	>6968
2-bit	55	29	7	620	>85	>3885
4-bit	106	65	8	1246	>111	>2411
8-bit	220	148	9	2647	>154	>1854
16-bit	517	301	10	5784	>224	>1524
32-bit	1124	759	11	12932	>377	>1571
Parallel	2239	1451	11	26243	>660	>1760

10) The sum bits (i.e., $Z_0, Z_1, \ldots Z_{15}$) are calculated in the eleventh stage.

The above procedure is the processing of the addition. The subtraction is implemented by attaching a path of reversing operand. Therefore, the control signal *Cmpl_Y* controls the operation of reversing the Y_i, and the control signal *Carry_in* performs the operation of "+1". The subtraction operation is similar to the addition operation except that the *Carry_in* signal is "1" at the first clock cycle.

For SLT operation, if and only if $X - Y < 0$, its result equals to 1. So, subtraction must be carried out, and its result is the MSB of the subtraction result. For signed SLT operation, the result is Z_{15} of the last slice; for unsigned SLT operation, the result is the inverse of the carry out i.e., C_{16} of the last slice. The output interface *Carry_out* is used to output the result C_{16}.

For EQ operation, if and only if $X = Y$, the result of EQ equals to 1. The result of this operation can be obtained by setting $G_{i:i}$ of all positions i (0–15) in each slice of the adder to 0, and $P_{i:i}$ is set to $X_i \oplus \overline{Y}_i$ to obtain the result, and the result of the operation is C_{16} of the last slice. Note that when $X = Y$, all $P_{i:i}$s are equal to 1.

In the logic operations, the ALU provides the control signals *Op_AND* and *Op_XOR* to select the AND operation or the XOR operation, respectively. For AND operation, we use $G_{i:i}$ for each location i (0–15) to implement it. For XOR operation, we use $P_{i:i}$ for each location i (0–15) to implement it. For the OR operation, it can be implemented by $(X_i \wedge Y_i) \vee (X_i \oplus Y_i)$, and the confluence buffer (CB) is used to achieve the $\vee$ operation, reducing the pipeline number of the ALU. For NOR operation, it can be calculated by $\overline{X}_i \wedge \overline{Y}_i$, so here we provide the control signal *Cmpl_X* to reverse the X_i. To propagate the result of a logic operation to the output port and reduce the number of the D flip-flops (DFF) and wirings, we use the path to propagate the $P_{i:i}$ to Z_i at each position i (0–15).

As shown in Fig. 2, the proposed ALU consists of six parts: inversion operand, calculation PG signals, carry network, calculation carry, and calculation results. It contains the basic RSFQ logic gate (AND gate, XOR gate and D flip-flop (DFF)) and wiring elements (Josephson transmission line (JTL), passive transmission line (PTL), and confluence buffer (CB)) from the RSFQ cell library for AIST ADP2 [16]. The ALU contains 135 AND gates, 64 XOR gates, 240 splitters, 318 DFFs and 61 CBs, and it consists of 5784 Josephson junctions (except for Josephson transmission lines and passive transmission lines). Without considering the delay of wirings in the cell library for AIST ADP2, the maximum clock speed is less than 77 GHz because the delay of carry feedback path is greater than 13 ps, and the latency is greater than 1524 ps by computing the latency of the key path at 10 GHz.

Pairs of a 64-bit operand slice are input one by one from the first clock cycle to the fourth clock cycle. The control signal *End_bar* is equal to 0 for the fourth clock cycle and 1 for the first to third clock cycles. The control signal *Carry_in* enters in the first clock cycle, and if the "+1" operation is performed, the control signal *Carry_in* is equal to 1 at the first cycle. The other five control signals, *Op_ARITH, Op_AND, Op_XOR, Cmpl_X and Cmpl_Y*, remain the same values during four clock cycles.

The total number of pipeline stages for the proposed ALU is 11. We used a merger (CB) as an OR gate for implementing OR operation, thus, reducing the number of clocked logic gates and pipeline stages and overall the hardware cost. As soon as input the most significant position of the previous ALU operation, the next ALU operation can begin. That is, the ALU can perform a 64-bit operation every four clock cycles. The first, i.e., the least significant, slice of the result is output after the 11th cycle, and the last slice of the result is output after the 14th cycle.

In order to minimize the delay of the carry feedback loop, the control signal *End_bar* judges whether or not the slice is the last one before entering the feedback loop, and the carry signal *Carry_in* determines whether or not the "+1" operation is performed. For the same purpose, the carry signal is divided into two directions: one immediately enters the AND gate of the carry feedback loop and the other is fed back to each bit of the next slice to calculate $Z_0 - Z_{15}$ and *Carry_out*.

III. DISCUSSION

We have designed logic circuits of 2-/4-/8-/32-bit bit-slice as well as bit-serial and parallel ALUs. Comparison of bit-slice ALUs is summarized in Table II. Without considering the delay of wirings (JTLs and PTLs), the latency of the ALUs is estimated by the following formula:

$$\text{Latency} = \text{clock delay} + \left(\text{pipeline stages} + \frac{\text{word length}}{\text{slice size}} - 1\right) \times \text{clock period}$$

Here, "clock delay" is defined as the total delay of the cells from clock-in to clock-out.

As shown in Table II, 16-bit bit-slice processing has the least latency to calculate 64-bit data, and can achieve the tradeoff between latency and hardware cost at 10 GHz.

IV. Conclusion

We have designed the RSFQ logic circuit for an ALU based on 16-bit bit-slice Ladner-Fischer adder. It implements the corresponding ALU instruction for MIPS32 instruction set with seven control signals and is capable of completing a 64-bit operand operation in four clock cycles. It contains a total of 135 AND gates, 64 XOR gates, 240 splitters, 318 D flip-flops (DFFs) and 61 confluence buffers (CBs).

With the development of fabrication process, the parallel architecture will be used for 32-/64-bit RSFQ microprocessors. Therefore, the ALU based on Ladner-Fischer adder has obvious advantages: fewer pipeline stages than Brent-Kung adder, a lesser fan-out than Sklansky adder (There is not feedback in a parallel architecture.), and a smaller wiring than Kogge-Stone adder.

In this paper, we only design the logic circuits of the proposed ALU based on Ladner-Fischer adder, but not do physical realization and simulation, which can be studied in the future research.

Acknowledgment

The authors would like to thank Takagi Laboratory of Kyoto University in Japan for providing the number of Josephson junctions in the cell library.

References

[1] K. K. Likharev and V. K. Semenov, "RSFQ logic/memory family: A new Josephson-junction technology for sub-terahertz-clock-frequency digital systems," *IEEE Trans. Appl. Supercond.*, vol. 1, no. 1, pp. 3–28, Mar. 1991.

[2] D. E. Kirichenko, S. Sarwana, and A. F. Kirichenko, "Zero static power dissipation biasing of RSFQ circuits," *IEEE Trans. Appl. Supercond.*, vol. 21, no. 3, pp. 776–779, Jun. 2011.

[3] M. Dorojevets, P. Bunyk, and D. Zinoviev, "FLUX chip: Design of a 20-GHz 16-bit ultrapipelined RSFQ processor prototype based on 1.75-μm LTS technology," *IEEE Trans. Appl. Supercond.*, vol. 11, no. 1, pp. 326–332, Mar. 2001.

[4] M. Tanaka, F. Matsuzaki, T. Kondo, and N. Nakajima, "A single-flux-quantum logic prototype microprocessor," in *Proc. IEEE ISSCC Dig. Tech. Papers*, Feb. 2004, vol. 1, pp. 298–529.

[5] Y. Yamanashi *et al.*, "Design and implementation of a pipelined bit-serial SFQ microprocessor, CORE1β," *IEEE Trans. Appl. Supercond.*, vol. 17, no. 2, pp. 474–477, Jun. 2007.

[6] A. Fujimaki *et al.*, "Bit-serial single flux quantum microprocessor CORE," *IEICE Trans. Electron.*, vol. E91-C, pp. 342–349, Mar. 2008.

[7] M. Tanaka *et al.*, "Design and implementation of a pipelined 8 bit-serial single-flux-quantum micro- processor with cache memories," *Supercond. Sci. Technol.*, vol. 20, no. 11, pp. S305–S309, Nov. 2007.

[8] Y. Nobumori, T. Nishigai, K. Nakamiya, and N. Yoshikawa, "Design and implementation of a fully asynchronous SFQ microprocessor: SCRAM2," *IEEE Trans. Appl. Supercond.*, vol. 17, no. 12, pp. 478–481, Jun. 2007.

[9] Y. Ando, R Sato, M. Tanaka, K. Takagi, N. Takagi, and A. Fujimaki, "Design and demonstration of an 8-bit bit-serial RSFQ microprocessor: CORE e4," *IEEE Trans. Appl. Supercond.*, vol. 26, no. 5, Aug. 2016, Art. no. 1301205.

[10] G. Tang, Y. Ohmomo, K. Takagi, and N. Takagi, "Conceptual design of a 4-bit bit-slice 32-bit RSFQ microprocessor," in *Proc. ASC 2016*, Denver, CO, USA, Sep. 4–9, 2016, Paper #4EOr2B-03.

[11] T. Filippov, M. Dorojevets, A. Sahu, and A. Kirichenko, "8-bit asynchronous wave- pipelined RSFQ arithmetic-logic unit," *IEEE Trans. Appl. Supercond.*, vol. 21, no. 3, pp. 847–851, Jun. 2011.

[12] T. Filippov *et al.*, "20 GHz operation of an asynchronous wave-pipelined RSFQ arithmetic logic unit," *Phys. Procedia*, vol. 36, pp. 59–65, Jun. 2012.

[13] M. Dorojevets, C. L. Ayala, N. Yoshikawa, and A. Fujimaki, "8-bit asynchronous sparse-tree superconductor RSFQ arithmetic-logic unit with a rich set of operations," *IEEE Trans. Appl. Supercond.*, vol. 23, no. 3, Jun. 2013, Art. no. 1700104.

[14] Y. Ando, R. Sato, M. Tanaka, and K. Takagi, "80-GHz Operation of an 8-Bit RSFQ arithmetic logic unit," in *Proc. 15th Int. Supercond. Electron. Conf.*, Jul. 6–9, 2015, Paper DS-P17.

[15] G. Tang, K. Takata, M. Tanaka, A. Fujimaki, K. Takagi and N. Takagi, "4-bit bit-slice arithmetic logic unit for 32-bit RSFQ microprocessors," *IEEE Trans. Appl. Supercond.*, vol. 26, no. 1, Jan 2016, Art. no. 1300106.

[16] Y. Yamanashi *et al.*, "100 GHz demonstrations based on the single-flux-quantum cell library for the 10 kA/cm$_2$ Nb multi-layer process," *IEICE Trans. Electron.*, vol. E93-C, no. 4, pp. 440–444, Apr. 2010.