Design of 8-Bit RSFQ Based Multiplier for DSP Application

N. Kamakshamma[1], P. Kullaya Swamy [2]

[1]PG Scholar, Electronics and Communication Engineering, Intell Engineering College, AP, India
[2]Assistant Professor, ECE, Intell Engineering College, AP, India
kamakshi.n14@gmail.com

Abstract: *We have developed and experimentally evaluated at high-speed, a complete set of arithmetic circuits (multiply, add, and accumulate) for high performance digital signal processing (DSP). These circuits take advantage of the unique features of the Rapid Single -Flux Quantum (RSFQ) logic/memory family, including fusion of logic and memory functions at the gate level, pulse representation of clock and data, and the ability to maintain inter cell propagation delays using Josephson transmission lines (JTLs). The circuits developed have been successfully used in the implementation of a serial radix 2 butterfly, a decimation digital filter, and of an arithmetic unit for digital beam forming. The 8×8-bit RSFQ multiplier uses a two - level parallel carry - save reduction tree that significantly reduces the multiplier latency. The 80-GHz carry-save reduction is implemented with asynchronous data-driven wave pipelined [4:2] compressors built with toggle flip-flop cells.*

Keywords: *High performance computing, Josephson junctions (JJs), multiplying circuits, super conducting integrated circuits.*

1. INTRODUCTION

SUPERCONDUCTOR rapid single-flux quantum (RSFQ) circuits offer well-known opportunities of building data processing units with frequencies over 20 GHz. However, high frequency alone is no guarantee of high performance on real applications. Other metrics, such as operand length, functionality, latency, throughput, power, and energy efficiency, are equally important in general-purpose processor design.

In this, we discuss the micro architecture, design, andtesting of the first 8×8-bit (by modulo 256) parallel carry-save RSFQ multiplier implemented using the ISTEC 10-kA/cm[2]1.0-µm fabrication technology. The work has been done in collaboration between separately funded teams at Stony Brook University (USA), Yokohama National University, and Nagoya University (Japan). The Stony Brook team has developed the complete logical and physical chip design using the CONNECT cell library and SFQ CAD tools developed at Nagoya and Yokohama.

Since the introduction of superconductor RSFQ logic, there were several attempts to design and fabricate different kinds of RSFQ multipliers. The immaturity of superconductor technology and design tools has limited the functional complexity and data path width of the fabricated multiplier designs to 1–4 bits. The first discussion of conceptual bit-serial and 4×4-bitgeneric carry-save RSFQ multipliers was in [1].

Traditional array-based parallel RSFQ multipliers are suitable for multiplication of small numbers. Their major draw back is that their latency grows linearly and their complexity grows quadratically with the increase in operand length. There are several known and widely-used techniques of reducing multiplication time, such as Booth encoding and parallel partial product reduction [12]. The challenge for RSFQ designers is to develop techniques that could use the full potential of the superconductor logic.

The first successful demonstration of a bit-serial RSFQ multiplier at the frequency of 6.3 GHz was reported in [2]. Several other bit-serial multiplier designs for DSP applications with their configurations from 1 to 4 bits were designed, fabricated, and successfully tested at low frequency [3]–[6].A4×4-bit parallel array-based RSFQ multiplier for a multiply-accumulate unit was designed and demonstrated its correct operation at low frequency [7].A4×4-bit parallel array

multiplier with Booth encoding for FFT applications was implemented using a phase-mode SFQ logic and its low-frequency operation verified experimentally[8], [9].

A bit-serial 16-bit floating-point (FP) multiplier was designed with the CONNECT cell library [10] and demonstrated correct operation at low frequency [11]. This FP multiplier has a systolic-array bit-serial micro architecture without any rounding hardware. It needs 23 clock cycles (920 ps) to calculate a 16-bitFP result.

The goal of our work presented in this paper was to design and demonstrate the first 20-GHz 8-bit parallel wave-pipelined low-latency RSFQ multiplier for high-performance processors.

Our 8 × 8-bit unsigned integer multiplier performs multiplication by modulo 256, calculating the eight least significant bits of the product (see Fig. 1).
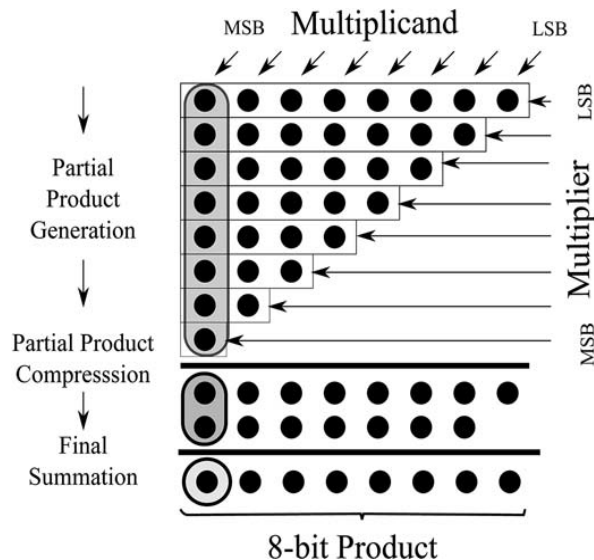

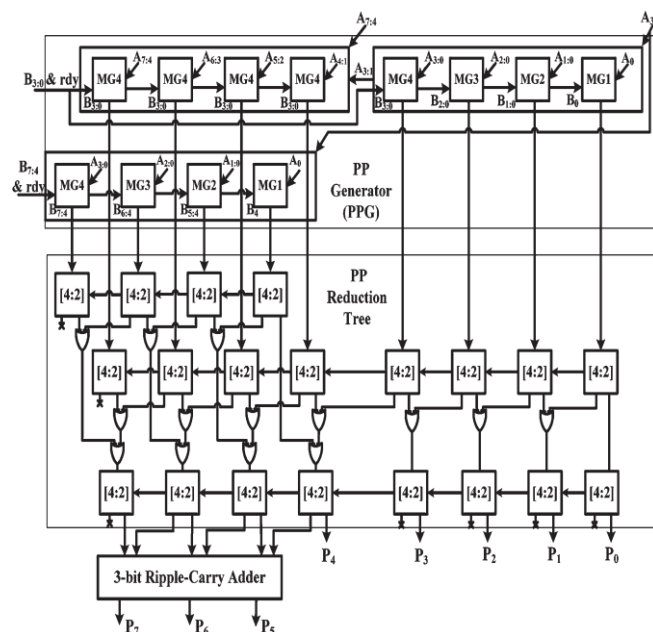**Fig1.** *Three steps of the 8 × 8 (by modulo 256) multiplication*

Fig2. *8 × 8-bit multiplier structural block diagram.*

Fig. 1 shows three major steps of multiplication of two unsigned integer operands: partial product generation, partial product compression (reduction), and final summation. When designing our 8 × 8-bit parallel integer multiplier, we had four major targets: high operation frequency of 20 GHz, multiplication time below 500 ps, complexity around 6000 Josephson junctions (JJs), and mostly regular layout employing both local and global connections.

To achieve these challenging goals, we used several advanced techniques, such as wave-pipelining [13]–[15], parallel partial product generation, and partial product compression with two

level carry-save reduction tree built with [4:2] asynchronous wave-pipelined compressors operating at the internal "hardwired" rate of 80 GHz.

Those techniques have been developed and verified by simulationduring the recent work on 32×32 multipliers done at Stony Brook University (SBU) with a use of the SBU VHDLRSFQ cell library [16]. We will discuss them in detail in Section III of this paper.

2. MICRO ARCHITECTURE AND DESIGN

Fig. 2 shows the block diagram of our 8×8-bit multiplier. The multiplier consists of three major blocks: a partial product generator, a parallel carry-save partial reduction (compression) tree, and a ripple-carry adder for final summation of carry-sum operands

2.1. Partial Product Generator with 80-Ghz Output Streams

The multiplier partial product generator (PPG) consists of 36 partial product (PP) bit generators built with clocked AND gates operating on their multiplicand and multiplier bits. These circuits are organized into three PPG groups, one (top left) with 16 and two other with 10 PP generators each. PPs in each PPG group are calculated in parallel, significantly reducing the partial product generation time.
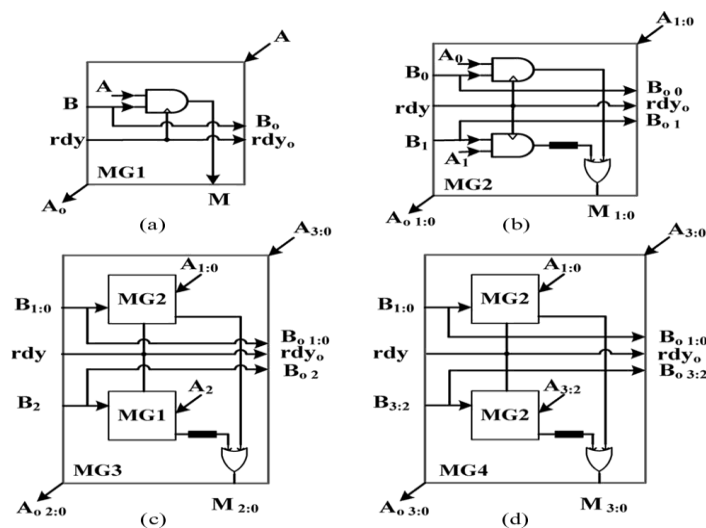

Fig3. *PP generation modules:* **(a)** *MG1;* **(b)** *MG2;* **(c)** *MG3;* **(d)** *MG4.*

Dark rectangles represent JJ-based delay lines used to create 12.5 ps time intervals between output PP signals called Mi.

The PPG groups are implemented with four different types of modules MG1–MG4 with their indexes corresponding to the number of PPs generated by the modules (see Fig. 3). When generated, PPs within each MG are merged together with confluence buffers (implementing asynchronous OR operations) and sent ~12.5 ps apart over a single passive transmission line (PTL) to their first-level [4:2] compressor. The minimum time gap of 11–12 ps between PP signal pulses is necessary to meet timing constraints and provide some DC bias margins of the confluence buffers and [4:2] compressors. The required time separation between PPs is achieved with a use of carefully designed operand and control distribution networks utilizing JJ-based delay lines, parallel and serial signal splitting. Working in parallel, the 12 MG blocks synchronously generate and send PPs (36 total) to the [4:2] compressors at the "hardwired" rate of 80 GHz.

2.2. 80-GHZ Partial Product Reduction and Final Summation

To reduce (compress) partial products in each column, we use a two-level binary carry-save reduction tree built with[4:2] compressors (see Fig. 4). First, up to 8 PPs in each column are reduced to 4 by two [4:2] compressors workingin parallel, each producing 2 PPs. The 4 PPs from the two first-level compressors are merged together with asynchronous confluence buffers and sent ~12.5 ps apart over a single PTL to a second-level [4:2] compressor for that column. Then, the second-level [4:2] compressor will reduce those 4 PPs to 2. The benefits of using this approach are as follows: 1) the $O(log2n)$ PP reduction time, where n is the operand length, and 2)

a regular layout. The latter is very important for our 80-GHz asynchronous data-driven wave-pipelined implementation of the [4:2] compressors.

The [4:2] carry-save compressors shown in Fig. 5 are implemented with (4,3) and (3,2) counters playing a role of carry-save adders. Each (4,3) counter can count up to 4 PPs arriving at its input and produce two inter-column carries and one intermediate sum bit. These intermediate sum and the two carries coming out from the previous bit column are then added by a (3,2) counter producing one carry (to the next bit column) and one sum bits. The inter-column carries from the previous bit column are processed by the (3,2) counters, not affecting the inter-column carry signals from the (4,3) counters to the next bit column.
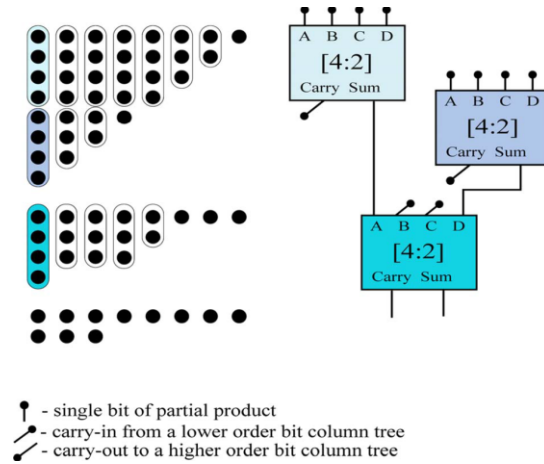
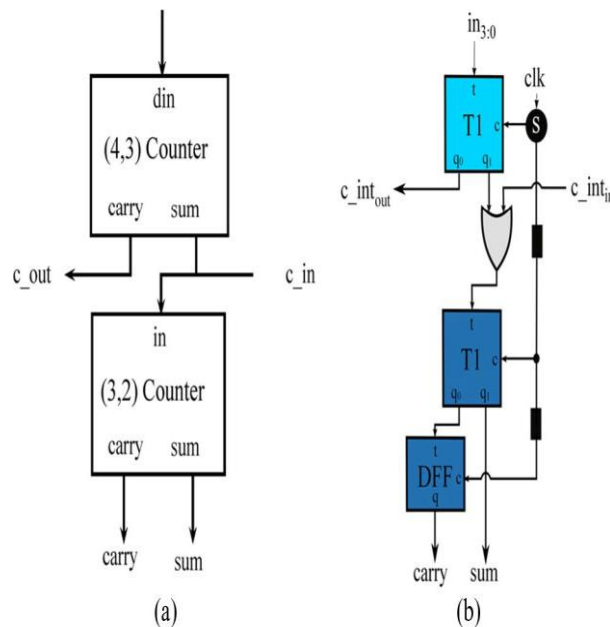

Fig4. *Two-level partial product reduction tree*

Fig5. *[4:2] compressor:* **(a)** *conceptual diagram;* **(b)** *cell-level RSFQ implementation. Dark rectangles represent JJ-based delay lines.*

Both counters have very efficient hardware implementation and small PP reduction time. They are implemented with T1 (toggle flip-flop) cells [17] that asynchronously generate up to two carry-out (one per every two input PPs received) and one clocked sum (the XOR sum of all PPs received before the clock signal arrival) output signals. The propagation of the carry signals and clocking of the T1 cells is properly tuned using JJ-based delay lines. Additional D flip-flops (DFFs) [18], one per [4:2] compressor, are used to buffer carries from the (3,2) counters.

The PP reduction pipeline diagram is shown in Fig. 6. The operation of each [4:2] compressor is asynchronously wavepipelined and data-driven by PPs coming ~12.5 ps apart at the internal rate of 80 GHz. It takes six 12.5-ps micro-steps (75 ps) to complete the 4-to-2 reduction operation. In each [4:2] compressor, the execution of the last two microsteps of one multiply operation is done in parallel with the execution of the first two micro-steps of the next multiply operation. As a

result, 8×8-bit multiply operations can start and produce results every 50 ps at the rate of 20 GHz.

The five least significant bits of the product are calculated during the PP reduction by the [4:2] compressors. The partial products in the three most significant bit columns are reduced to carry-sum pairs and then go through the final summation done by a wave-pipelined ripple-carry adder (see Fig. 7).
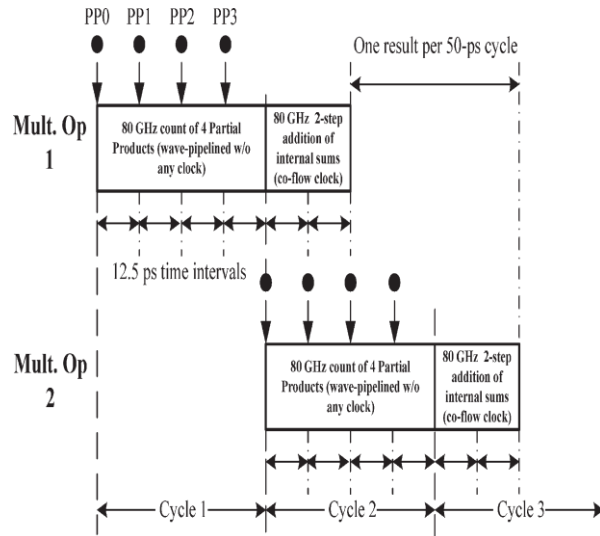

Fig6. *[4:2] compressor pipeline diagram. The 50 ps clock cycle time of the multiplier is determined by the time to complete four 12.5-ps micro-steps.*
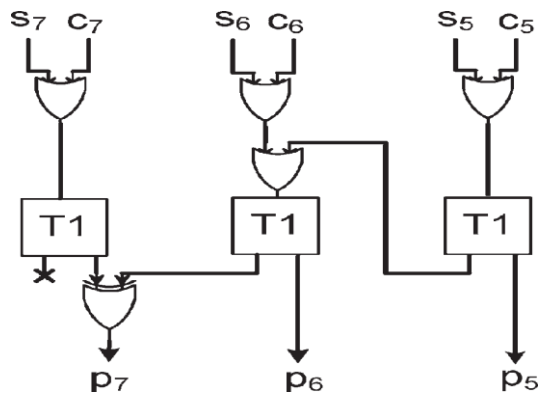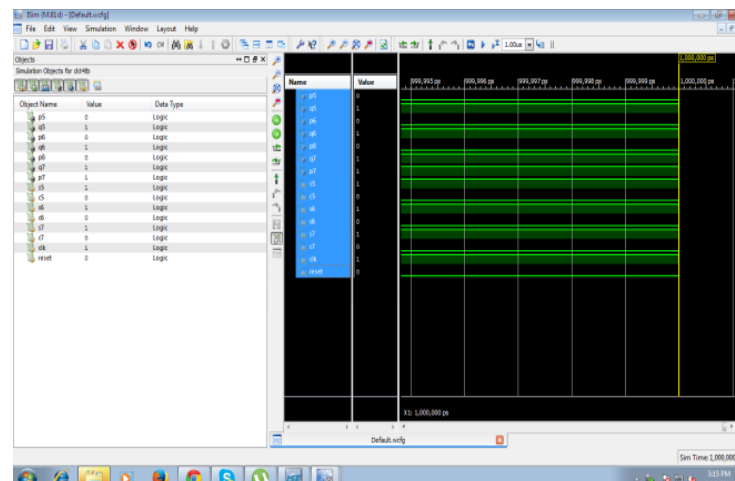

Fig7. *Ripple-carry adder for final summation. Clock signal distribution lines for T1 and XOR gates are not shown.*

3. IMPLEMENTATION AND RESULTS

The proposed RSFQ multiplier designed using Verilog hardware description language and structural form of coding. The proposed system simulation results are as follows

4. CONCLUSION

We have successfully designed, and tested the first8×8-bit (by modulo 256) parallel carry-save superconductor RSFQ multiplier with the target frequency of 20 GHz. The multiplier employs an efficient RSFQ logic specific technique of the 80-GHz asynchronous wave-pipelined partial product generation and reduction.

REFERENCES

[1] O. A. Mukhanov, S. V. Rylov, V. K. Semenov, and S. V. Vyshenskii,"RSFQ logic arithmetic,"IEEE Trans. Magn., vol. 25, no. 2, pp. 857–860,Mar. 1989.

[2] O. A. Mukhanov and A. F. Kirichenko, "Implementation of a FFT radix 2butterfly using serial RSFQ multiplier-adders,"IEEE Trans. Appl. Supercond., vol. 5, pp. 2461–2464, Jun. 1995.

[3] S. V. Polonsky, J. C. Lin, and A. V. Rylyakov, "RSFQ arithmetic blocksfor DSP applications,"IEEE Trans. Appl. Supercond., vol. 5, pp. 2823–2826, Jun. 1995.

[4] Q. P. Herr, N. Vukovic, C. A. Mancini, K. Gaj, K. Qing, V. Adler, E. G.Friedman, A. Krasniewski, M. F. Bocko, and M. J. Feldman, "Designand low speed testing of a four-bit RSFQ multiplier-accumulator,"IEEETrans. Appl. Supercond., vol. 7, no. 2, pp. 3168–3171, Jun. 1997.

[5] A. Akahori, M. Tanaka, A. Sekiya, A. Fujimaki, and H. Hayakawa, "Design and demonstration of SFQ pipelined multiplier,"IEEE Trans. Appl.Supercond., vol. 13, no. 2, pp. 559–562, Jun. 2003.

[6] M. Obata, M. Tanaka, Y. Tashiro, Y. Kamiya, N. Irie, K. Takagi,N. Takagi, A. Fujimaki, N. Yoshikawa, H. Terai, and S. Yorozu, "Singleflux-quantum integer multiplier with systolic array structure,"Phys. C,vol. 445–448, pp. 1014–1019, 2006.

[7] I. Kataeva, H. Engseth, and A. Kidiyarova-Shevchenko, "New design ofan RSFQ parallel multiply accumulate unit,"Supercond. Sci. Technol.,vol. 19, pp. 381–387, May 2006.

[8] Y. Horima, T. Onomi, M. Kobori, I. Shimizu, and K. Nakajima, "Improveddesign for parallel multiplier based on phase-mode logic,"IEEE Trans.Appl. Supercond., vol. 13, no. 2, pp. 527–530, Jun. 2003.

[9] R. Nakamoto, S. Sakuraba, T. Onomi, S. Sato, and K. Nakajima, "4-bitSFQ multiplier based on Booth encoder,"IEEE Trans. Appl. Supercond.,vol. 21, no. 3, pp. 852–855, Jun. 2011.

[10] S. Yorozu, Y. Kameda, H. Terai, A. Fujimaki, T. Yamada, and S. Tahara,"A single flux quantum standard logic cell library," Phys. C, vol. 378,pp. 1471–1474, 2002.

[11] H. Hara, K. Obata, H. Park, Y. Yamanashi, K. Taketomi, N. Yoshikawa,M. Tanaka, A. Fujimaki, N. Takagi, K. Takagi, and S. Nagasawa, "Design, implementation and on-chip high-speed test of SFQ half-precisionfloating-point multiplier,"IEEE Trans. Appl. Supercond., vol. 19, pt. 1,no. 3, pp. 657–660, Jun. 2009.

[12] J. L. Hennessy and D. A. Patterson,Computer Architecture: A Quantitative Approach, 5th ed. Elsevier, 2012.

[13] W. P. Burleson, M. Ciesielski, F. Klass, and W. Liu, "Wave-pipelining:A tutorial and research survey,"IEEE Trans. Very Large Scale Integr.(VLSI) Syst., vol. 6, no. 3, pp. 464–474, Sep. 1998.

[14] M. Dorojevets, C. Ayala, and A. Kasperek, "Data-flow microarchitecturefor wide datapath RSFQ processors: Design study," IEEE Trans. Appl.Supercond., vol. 21, no. 3, pp. 787–791, Jun. 2011.

[15] M. Dorojevets, C. Ayala, and A. Kasperek, "Development and evaluationof design techniques for high-performance wave-pipelined wide datapathRSFQ processors," inProc. ISEC, Fukuoka, Japan, Jun. 2009, p. 46.

[16] M. Dorojevets and A. Kasperek, "Design and evaluation of a 32-bit integermultiplier," Ultra-High-Speed Comput.Lab. Tech. Rep., Stony BrookUniversity, Stony Brook, NY, Jul. 2011, (unpublished).

[17] S. Polonsky, V. K. Semenov, and A. F. Kirichenko, "Single flux, quantumB flip-flop and its possible applications,"IEEE Trans. Appl. Supercond.,vol. 4, no. 1, pp. 9–18, Mar. 1994.

[18] SUNY RSFQ Cell Library.[Online]. Available: pavel.physics.sunysb.edu/RSFQ/Lib/AR/dff.html

AUTHORS' BIOGRAPHY

N. Kamakshamma received the B.Tech degree in Electronics and Communication Engineering in the year 2012 and pursuing M.Tech degree in VLSI System design from Intel engineering college. Her area of interests includes in VLSI Design and Wireless Communication.

P. Kullaya Swamy received the M.Tech degree in VLSI system design. Currently, he is working as Assistant Professor in the Department of Electronics and Communication Engineering, Intell Engg. College, Ananthapuramu. Having 10 years of Research and Teaching experience. Area of interest includes VLSI Testing and Digital Design (VHDL, Verilog), Microprocessor & micro controller, Embedded systems and Mobile Applications.