FPGA IMPLEMENTATION OF 32-BIT WAVE-PIPELINED SPARSE-TREE ADDER

Kasharaboina Thrisandhya*1, LathaSahukar*2

*1Post graduate (M.Tech) in ATRI, JNTUH University, Telangana, India.
*2Associate Professor in ATRI, JNTUH University, Telangana, India.

ABSTRACT

In this novel presentation include the design, testing and architecture of the 32-bit asynchronous wave pipelined sparse-tree superconductor rapid single-flux quantum adder implemented. Compared to the Kogge Stone adder, our prefix parallel sparse-tree adder has better efficiency on energy with significantly decreased complexity and almost no reduced operation frequency. The 32-bit adder core has 9941 Josephson junctions occupying an area of 8.5 mm2. It is designed operation frequency targeted as 30 GHz with the expected latency of 352ps at bias voltage of 2.5 mV. The adder chip was fabricated and tested successfully at low frequency for all test patterns with measured bias margins of +9.8% / −10.7%.

Index-terms: Adders, digital arithmetic, superconducting integrated circuits, superconducting logic circuits, sparse tree.

I. INTRODUCTION

In the universal digital circuits for almost any application is an adder. It is the fundamental building block of Arithmetic Logic Units (ALUs) in general-purpose and special-purpose digital signal microprocessors. Currently, in the CMOS domain, the design space of adder structures has been nearly exhausted, with only minimal improvements shown over previous designs. In contrast, emerging digital circuit technologies such as superconducting Rapid Single Flux Quantum (RSFQ) logic opens a way for researchers to explore new design methodologies for extremely fast, energy-efficient adders. In RSFQ logic, most adder designs demonstrated to date are bit-serial or digit-serial architectures which operate on a single bit or a small group of bits sequentially at a very high processing rate [1]–[2]. Such designs allow for simple clocking and compact structures. However, the latency of serial adders scales O(n), where n is the number of bits per operand, which leads to long latencies for 32-/64-bit operations in general purpose processors. In the past, parallel architectures in RSFQ have been limited to small data widths or relatively long latency ripple-carry adders [3]. One study evaluated 32-/64-bit parallel Kogge-Stone RSFQ adders using co-flow clocking [4].

In the effort of realizing scalable, high-performance, fully parallel designs, a new technique of asynchronous hybrid wave-pipelining for RSFQ circuits has been developed at Stony Brook University (SBU) [5], [6]. Later, as a result of the collaboration between the SBU and HYPER designers, an 8-bit wave-pipelined ALU was successfully designed, fabricated, and demonstrated correct operation at the rate of 20 GHz[7], [8]. In this paper, we present the design of the first 32-bit asynchronous parallel adder implemented in RSFQ logic. It builds upon the proven hybrid wave-pipelining techniques to provide 32-bit wide processing and synchronization. It incorporates an energy efficient, low complexity sparse-tree structure with very high processing rate. The work is based on a design study for a scalable 32-bit wave-pipelined sparse-tree adder conducted at SBU.

II. 32-BIT SPARSE TREE RSFQ ADDER

A. Sparse-tree RSFQ Adder

High-performance parallel adders typically use prefix trees which generate carries in log2(n) time, where n is the number of bits of the data path. The Kogge-Stone adder (KSA) is considered to be the fastest among parallel-prefix adders. Further enhancements to the KSA prefix structure such as the sparse-tree configuration have been proposed and used in high-
performance Intel processors. In our 32-bit RSFQ adder design, we chose the sparse-tree structure to reduce the number of wiring junctions needed for its implementation without any significant effect on its processing rate. As a side effect, this will also lead to a more energy-efficient design by reducing the total bias current and power consumption. It consists of the following three stages: Initialization, Prefix-Tree and Summation.

The Initialization stage receives two 32-bit data operands A and B to create bitwise Generate (G) and Propagate (P) signals which will be merged in a logarithmic manner in the Prefix-Tree stage. The Initialization stage consists of GPR_INIT logic blocks, one for each bit. The GPR_INIT creates the bitwise prefix functions described as $G_i = A_i \cdot B_i$ and $P_i = A_i \oplus B_i$ where i is the bit index column ranging from 31 down to 0 in the 32-bit adder. These functions are easily realized through clocked AND and XOR gates in a co-flow clocking arrangement. The clock is the Rdy signal provided to all bits. Additionally, it is necessary to create the trailing reset signal R which will be used to reset the asynchronous elements in the Prefix-Tree. Signal R is a copy of the Rdy signal for each bit with $\omega$-based delay lines to ensure data signals are processed before reset follows in the asynchronously Prefix-Tree.

The Prefix-Tree stage consists of Carry-Merge (CM) blocks to merge the prefix signals and provide a group carry to each 4-bit summation block. In contrast, the Kogge-Stone prefix tree provides a carry to every individual bit of the adder. DFF (D flip-flop) buffers appropriately delay prefix and bitwise P signals until they are ready to be merged or processed at the Summation stage, respectively. The first three levels of the Prefix-Tree also perform the ripple-carry addition within each 4-bit group before data arrive at the Summation stage. Merging of the prefix signals is described in [10]. It is implemented with CFFs (resettable Muller C-flip-flop gates based on the Muller C-element and confluence buffers used as asynchronous OR gates without any danger of violating the time separation requirement of their input pulses.

**B. Parallel prefix adders**

The parallel prefix adders are more flexible and are used to speed up the binary additions. Parallel prefix adders are obtained from Carry Look Ahead (CLA) structure. We use tree structure form to increase the speed of arithmetic operation. Parallel prefix adders are fastest adders and these are used for high performance arithmetic circuits in industries. The construction of parallel prefix adder [10] involves three stages.

**Prepossessing stage:** In this stage we compute, generate and Propagate signals to each pair of inputs A and B. These signals are given by the logic equations 1&2:

$P_i = A_i \oplus B_i$ ................................ (1)

$G_i = A_i \cdot B_i$ ............................ (2)

**Carry generation network:** In this stage we compute carries corresponding to each bit. Execution of these operations is carried out in parallel [9]. After the computation of carries in parallel they are segmented into smaller pieces. It uses carry propagate generate as intermediate signals which are given by the logic equations 3&4:

$C_{Pi:j} = P_{i:k+1}$ and $P_{k:j}$ ......(3) $C_{Gi:j} = G_{i:k+1}$ or $(P_{i:k+1}$ and $G_{k:j})$ ......(4)

**Post processing:** This is the final step to compute the summation of input bits. It is common for all adders and the sum bits are computed by logic equation 4&5:

$C_{i-1} = (P_i \text{ and } C_{in})$ or $G_i$ ............... (4)

$S_i = P_i \oplus C_{i-1}$ .................................. (5)
To reduce the computation time, engineers devised faster ways to add two binary numbers by using carry-look ahead adders. They work by creating two signals (P and G) for each bit position, based on if a carry is propagated through from a less significant bit position (at least one input is a '1'), a carry is generated in that bit position (both inputs are '1'), or if a carry is killed in that bit position (both inputs are '0'). In most cases, P is simply the sum output of a half-adder and G is the carry output of the same adder. After P and G are generated the carries for every bit position are created. Some advanced carry-lookahead architectures are the Manchester carry chain, Brent–Kung adder, and the Kogge–Stone adder. Some other multi-bit adder architectures break the adder into blocks. It is possible to vary the length of these blocks based on the propagation Delay of the circuits to optimize computation time. These block based adders include the carry by pass adder which will determine P and G values for each block rather than each bit, and the carry select adder which pre-generates sum and carry values for either possible carry input to the block.

A carry-look ahead adder (CLA) is a type of adder used in digital logic. A carry-look ahead adder improves speed by reducing the amount of time required to determine carry bits. It can be contrasted with the simpler, but usually slower, ripple carry adder for which the carry bit is calculated alongside the sum bit, and each bit must wait until the previous carry has been calculated to begin calculating its own result and carry bits (see adder for detail on ripple carry adders). The carry-look ahead adder calculates one or more carry bits before the sum, which reduces the wait time to calculate the result of the larger value bits. The Kogge-Stone adder and Brent-Kung adder are examples of this type of adder.

### III. SIMULATION RESULTS

Various adders were designed using Verilog language in Xilinx ISE Navigator and all the simulations are performed using Model sim 6.5e.
The performance of proposed adders are analyzed and compared. In this proposed architecture, the implementation code for modified 32-bit sparse-tree RSFQ adder carry look ahead adders were developed and corresponding values of delay and area were observed. Table 1 shows the comparison of adders. The simulated outputs of 32-bit proposed adders are shown in figure.

Figure 2: Simulation waveform for Sparse Tree Adder

Figure 3: RTL diagram

IV. CONCLUSION

We have designed, fabricated, and tested the first 32-bit wave-pipelined sparse-tree RSFQ adder chip with the core complexity of 9941 JJs and the target operation rate of 30 GHz. We have successfully demonstrated the correct operation of the chip at low frequency, passing all carefully chosen test vector with a measured bias margin of +9.8% – 10.7%. Another adder chip consisting of 12785 junctions with additional on-chip circuits for 30 GHz testing was also fabricated but its testing showed the need for another fabrication run.

REFERENCES


Kasharaboina Thrisandhya received B.Tech degree in ECE from Vanjari Seethaiah Memorial Engineering College in 2012, pursuing M.Tech (2012-2014) in the stream of VLSI at Aurora’s Technological and Research Institute, (Affiliated to JNTUH) Hyderabad. Her interest area is VLSI Design.

Latha Sahukar, Presently working as Associate professor in ATRI, Hyderabad. Her’s area of interest is VLSI Design, Communication Systems.