To appear in *IEEE International Solid-State Circuits Conference (ISSCC) Digest of Technical Papers,* San Francisco, February 2002.

3.5. A 1.3 GSample/s 10-tap Full-rate Variable-latency Self-timed FIR filter with Clocked Interfaces

Jose Tierno, Alexander Rylyakov, Sergey Rylov, Montek Singh (1),
Paul Ampadu (2), Steven Nowick (1), Michael Immediato, Sudhir Gowda

IBM T. J. Watson Research Center, Yorktown Heights, NY
(1) Columbia University, New York, NY
(2) Cornell University, Ithaca, NY

This work reports a 6-bit, 10-tap, full-rate distributed arithmetic digital finite impulse response filter (DFIR), using dynamic logic datapath with independent precharge and compute signals per domino stage. These precharge and compute signals are controlled by self-timed control circuits that ensure proper sequencing of actions between adjacent logic stages. Latches at the input and output of the datapath resynchronize the data to the clock, making it possible for this circuit to be dropped into a clocked environment. A key feature of our design is that the filter latency can be varied, from 4 to 12 clock cycles, as the clock speed is increased. This feature allows the self-timed datapath to appear, to the clocked part, as a pipeline with variable depth, allowing a trade-off between latency and throughput.

The basic architecture of the filter, shown in **Figure 3.5.1**, is full rate, distributed arithmetic with signed-digit offset binary (SDOB) number representation [1,2]. Two 8-bit, 16-entry loadable tables contain the precomputed partial sums (for odd-index and even-index coefficients respectively) used by the distributed arithmetic algorithm. Six bit-slices are used to generate six 4-bit addresses, plus six extra sign bits. These addresses are fed to a decoder that selects one 8-bit word from the even-coefficients table. A delayed version of this address selects an 8-bit word from the odd coefficients table.

The shift register, decoder, and 16:1 mux in the bit slice were designed using standard cells. At the output of the mux, a latch with a domino output stage generates dual-rail RZ

data. The first self-timed stage is a layer of XOR gates that restores the sign to the output of the partial sum tables. The shift register, decoder, mux, latch, and XOR gate form one bit slice.

The next eight stages of the self-timed datapath correspond to five layers of carry-save and three layers of carry look-ahead adders. The output 15-bit word is then latched back into the clock domain. **Figure 3.5.2** shows a detail of the self-timed datapath, together with the self-timed control and the clocked to self-timed interfaces. On the input side, data is launched into the self-timed datapath using a dynamic buffer. The pulse generator creates a timed pulse for the dynamic buffer, as well as the initial handshake request for the first stage of the self-timed control. The request pulse is long enough for the first asynchronous stage to respond before the pulse resets. Also, the clock cycle time is kept at least as long as the cycle time of the self-timed pipeline.

The self-timed datapath uses the *high-capacity* pipeline style of [3]. Each stage's control uses a 4-phase request-acknowledge protocol to sequence actions with its left and right neighbors. It also generates separately controllable precharge and compute signals for the datapath. The assertion of precharge or compute causes the corresponding phase to occur; while both are deasserted, the stage holds its value. This approach allows a distinct piece of data to be stored in every pipeline stage; in contrast, other dynamic logic approaches allow only one data item in every other stage. When full, the self-timed datapath can hold 9 different tokens, and can have a maximum latency of 9 clock cycles; the minimum latency is one clock cycle.

The rest of the clocked circuit takes another three clock cycles, giving a total latency of four to twelve cycles. The latency is determined by programming the synchronous delay line shown at the top of **Figure 3.5.2**. After reset, the delay line will wait from one to four clock cycles before allowing the output latch to remove a token. The tokens will be held on the last stages of the datapath, potentially by de-asserting both precharge and compute. After the initial wait, a token is removed on each clock cycle, and the number of tokens in

the datapath stays constant. Other latency amounts, from five to nine, were obtained during testing by direct manipulation of the control signals.

On the output side of the pipeline, another pulse generator generates an acknowledge pulse for the last stage. The request output of the self-timed control is ignored; instead, the output of the clocked delay line indicates data arrival. For correct operation, the latency of the delay line is programmed to be greater than the latency through the self-timed datapath. At higher clock rates, we need more clock cycles to go through the self-timed datapath.

We designed two different pipeline controls, shown in **Figure 3.5.3**. One is faster, but has stronger timing assumptions. The other is more robust, at the cost of some throughput. The two versions of the filter were fabricated side-by-side on the same chip, and the measured difference in performance was about 20%, as expected.

The chip was fabricated in a 0.18μm process and 1.8V nominal Vdd. **Figure 3.5.4** shows the chip micrograph. The DFIR cell occupies an area of 1.3x0.35mm^2 in its center. The chip was programmed and tested for correctness at low speed. A test-enable signal converts the self-timed datapath into a pseudo-NMOS circuit, for LSSD testing. The clock speed was then increased until a failure was detected in the output data. **Figure 3.5.5** shows plots of the measured maximum achievable frequency, and the corresponding power dissipation, for various Vdd levels, against the number of tokens present in the self-timed datapath. Throughput increases from one to four tokens, but decreases afterwards, as the pipeline becomes congested. The best observed performance was 1.3 GSamples/s with three tokens and 2.1V power supply.

Acknowledgements

The authors wish to thank Brian Worth and IBM Microelectronics for chip fabrication support, and Dave Heidel for test support. This work was partially supported by NSF ITR Award No. CCR-00-86007.

References

[1] Pearson, D. et al., "Digital FIR filters for High Speed PRML Disk Read Channels," IEEE JSSC, vol. 30, pp. 1517-1523, Dec. 1995

[2] Rylov, S. et al. "A 2.3 GSample/s 10-tap Digital FIR Filter for Magnetic Recording Read Channels" In Proc. ISSCC, vol. 44, pp. 190-191, Feb. 2001

[3] Singh, M. and Nowick, S. "Fine-Grain Pipelined Asynchronous Adders for High-Speed DSP Applications", In Proc. of the IEEE Computer Society Annual Workshop on VLSI, pp 111-118, Apr. 2000

Figure 3.5.1: Top-level architecture of the self-timed DFIR.

Figure 3.5.2: Self-timed datapath and control with clocked interfaces.

Figure 3.5.3: Self-timed pipeline stage controls, two versions using the high-capacity style.

Figure 3.5.4: Micrograph of the DFIR.

Figure 3.5.5: Measured throughput and power graphs against Vdd and number of tokens.

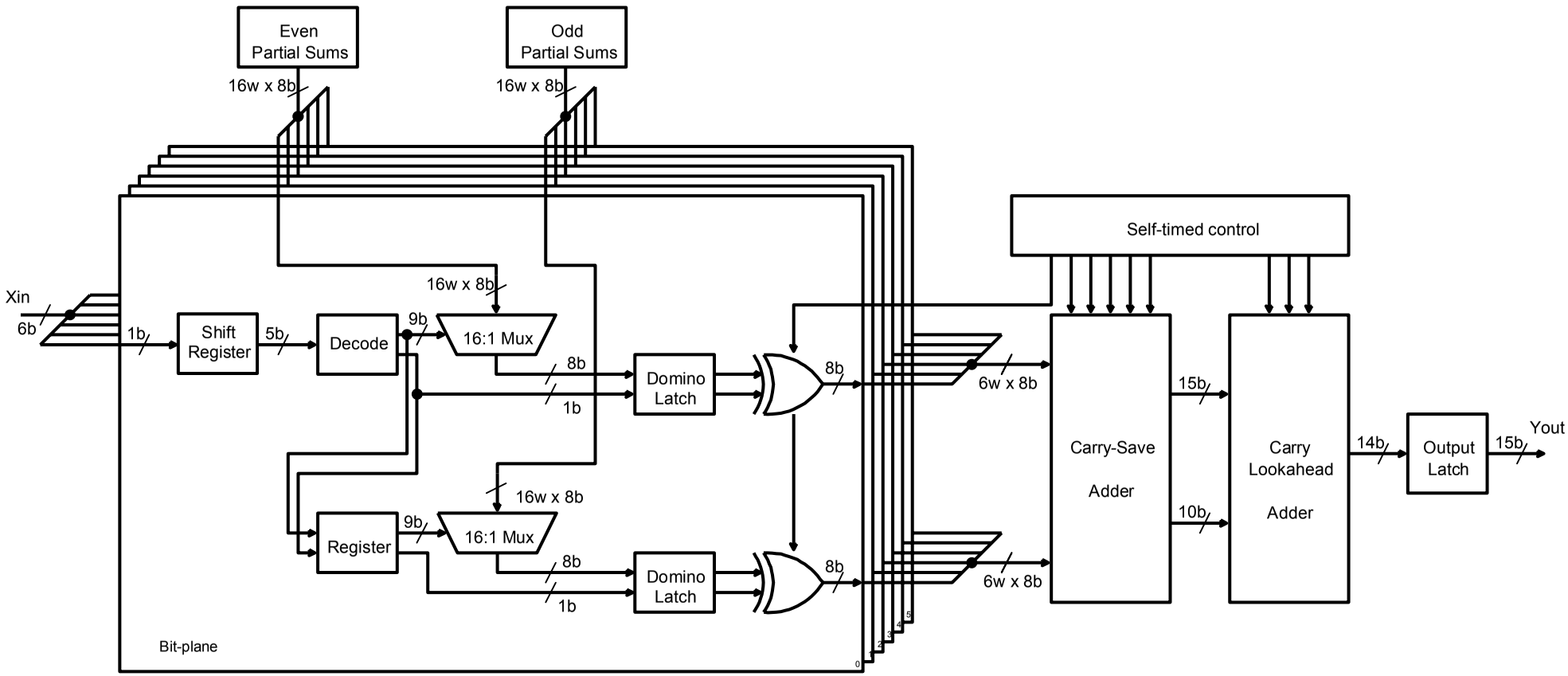

Figure 3.5.1: Top-level architecture of the self-timed FIR filter.

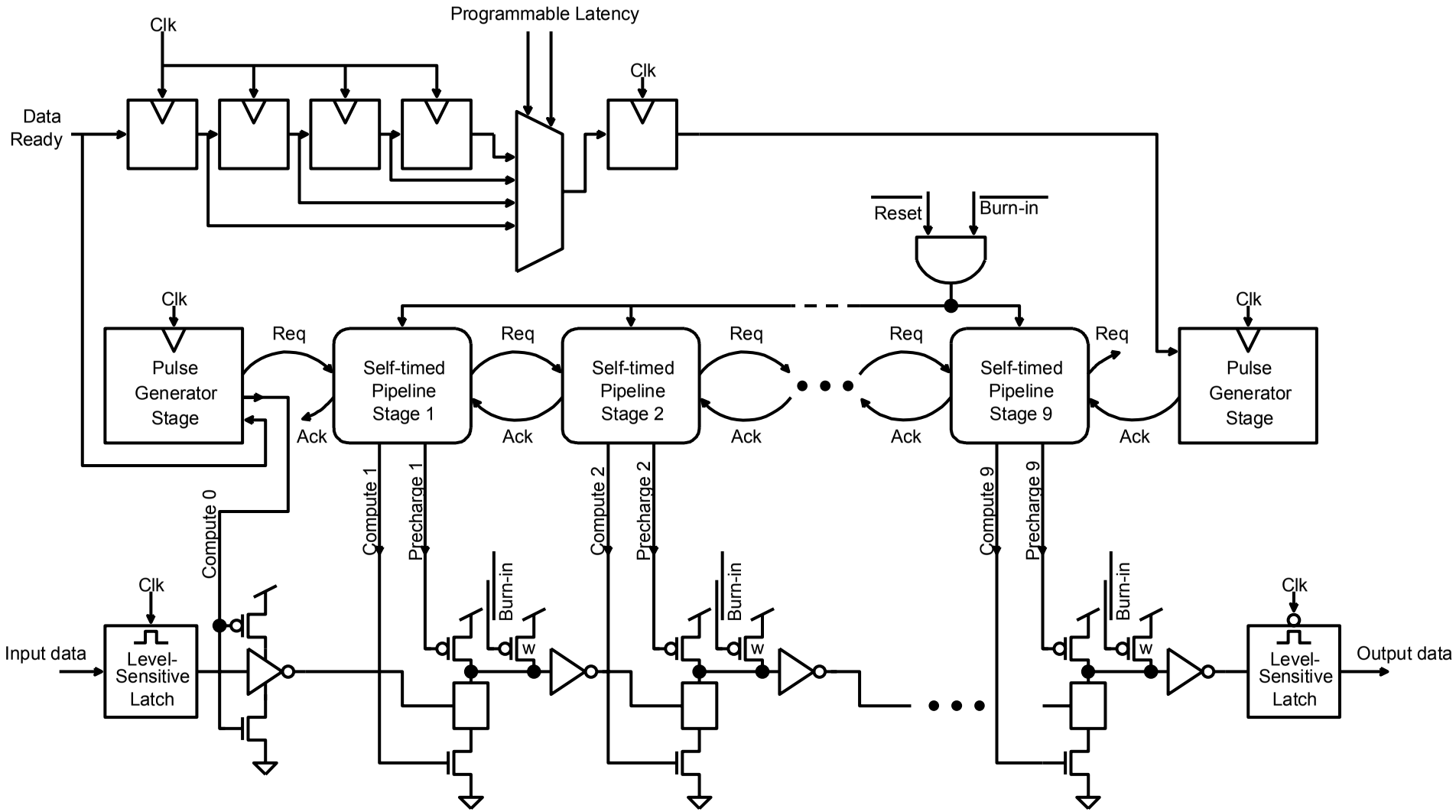

Figure 3.5.2: Self-timed datapath and control with clocked interface.

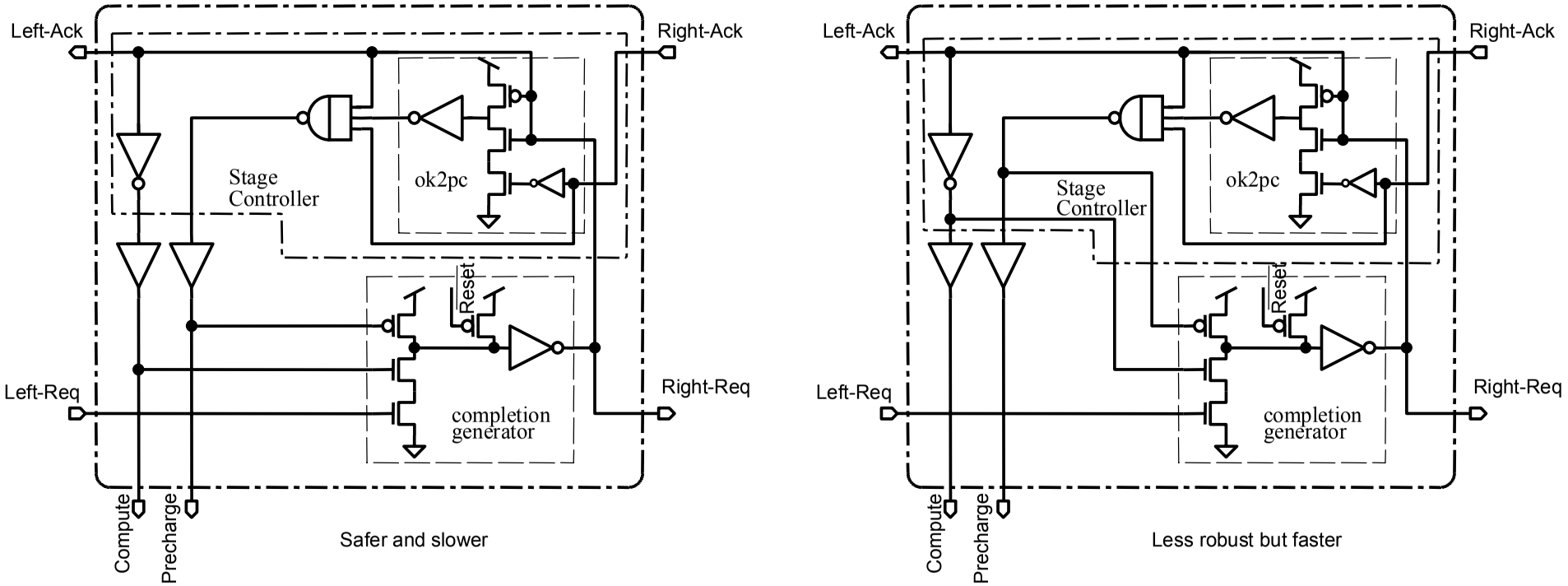

Figure 3.5.3: Self-timed pipeline stage controls, two versions using the high-capacity style.

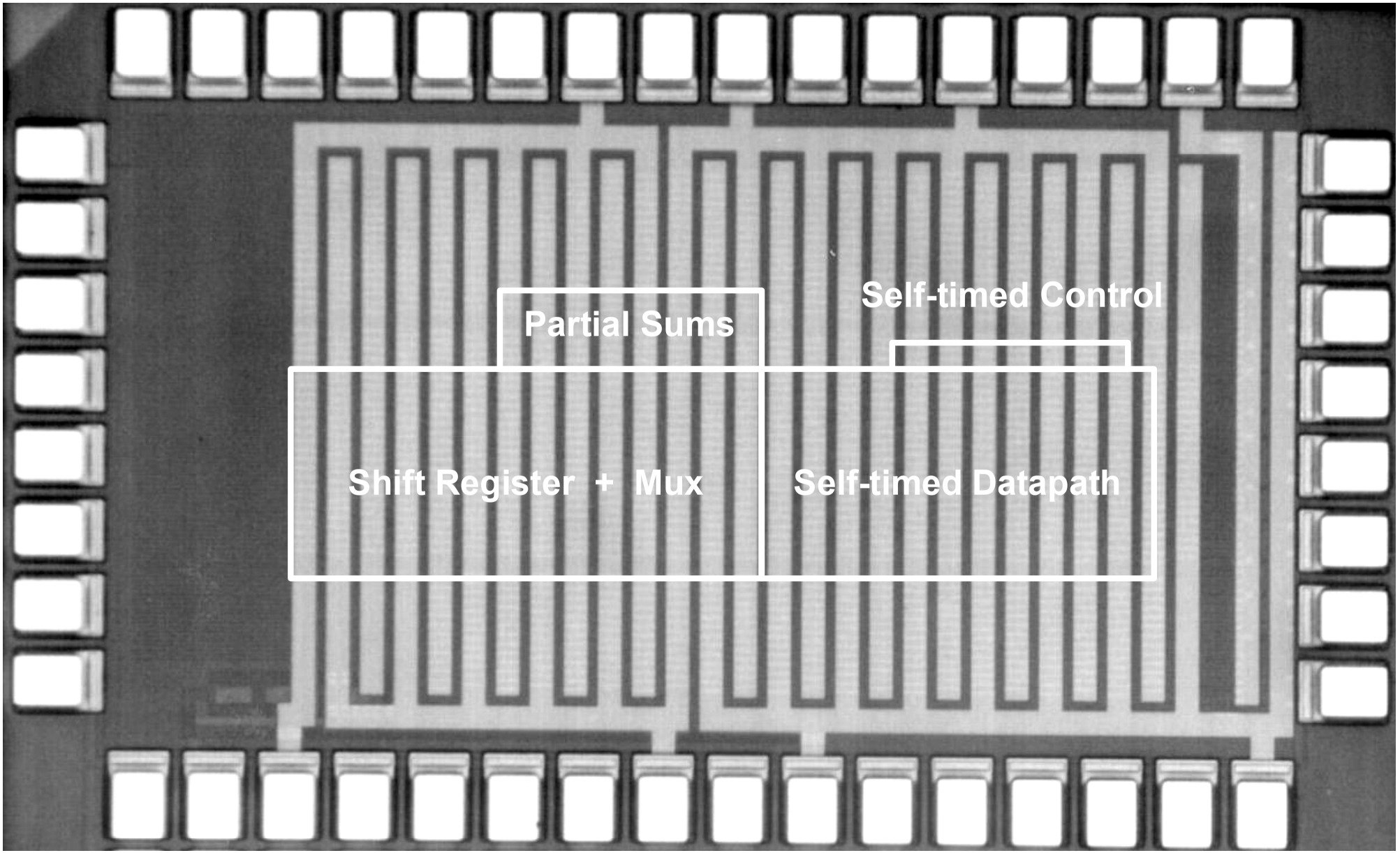

Figure 3.5.4: Micrograph of the DFIR.

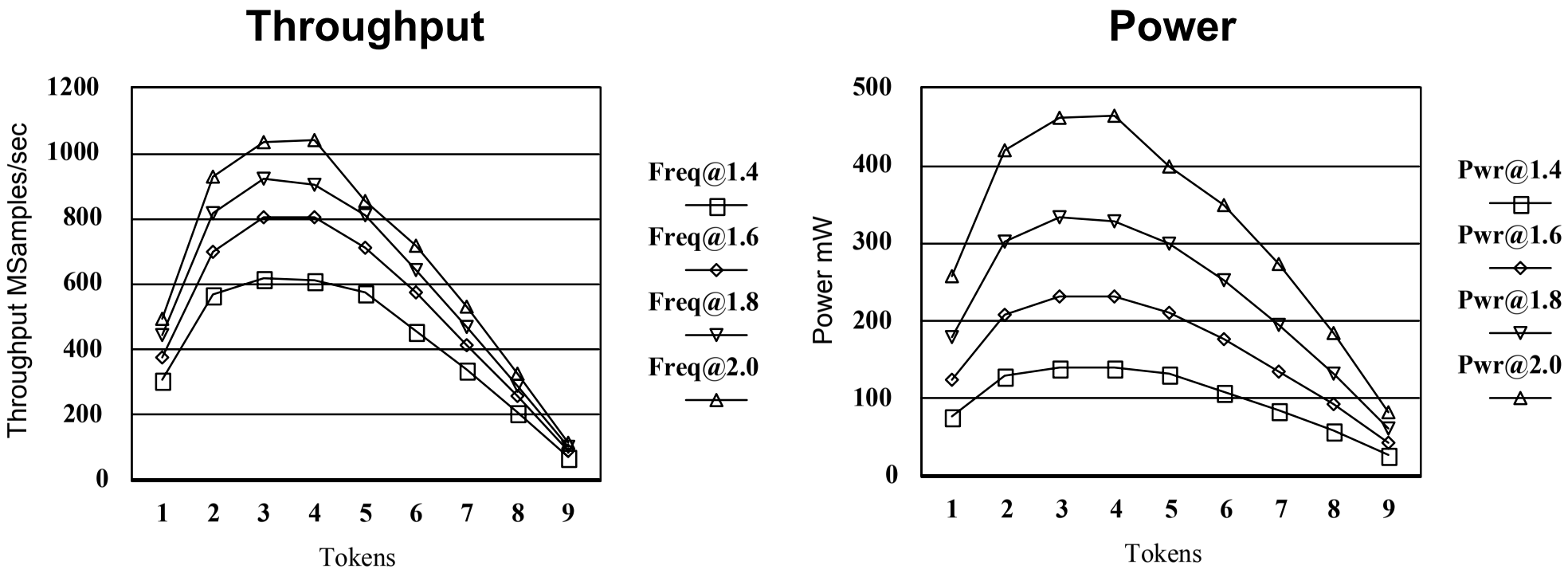

Figure 3.5.5: Measured throughput and power graphs against Vdd and number of tokens.