150mV Sub-Threshold Asynchronous Multiplier for Low-Power Sensor Applications

Joseph Crop, Scott Fairbanks, Robert Pawlowski, Patrick Chiang
Oregon State University
Corvallis, OR 97331
{cropj, fairbanksc, pawlowsr, pchiang}@eecs.oregonstate.edu

Abstract—A 4-bit Asynchronous multiplier was designed in the sub-threshold regime. The multiplier, using asynchronous completion detection, is more tolerant to process variation that conventional synchronous sub-threshold circuits, and operates with a supply voltage as low as 150mV. The average energy per computation was simulated at 1.13pJ. The minimum energy voltage was simulated at 350mV, with an average micro-pipelined frequency of 11.3 kHz.

I. INTRODUCTION

In wireless sensor applications, power is a scarce commodity. Operating a circuit in the sub-threshold region [1]-[3] enables a designer to implement a circuit's functionality with the minimum energy possible. Unfortunately, circuits operating in sub-threshold exhibit wide variations in delay as supply voltage is scaled [1], especially across process variations. Therefore, conventional synchronous timing schemes exhibit large delay spreads across transistor and process mismatches, resulting in impractical usage of sub-threshold circuits in deeply scaled CMOS technologies.

This paper proposes a self-timed design framework in the implementation a 4-bit multiplier that is robust with respect to the timing variations introduced by deep submicron voltage scaling. A small number of custom logic gates necessary for asynchronous circuits are designed for sub-threshold operation [4], [5] and implemented in a 0.18μm CMOS technology.

II. MOTIVATION FOR ASYNCHRONOUS SUB-THRESHOLD CIRCUITS

A. Advantages of Asynchronous Logic

Asynchronous or self-timed circuits possess a number of properties that make them advantageous for sensor applications:

- They are event driven. They wait indefinitely, burning only leakage power until they are provided with an event. Then they wake to perform the desired computation.
- They relieve a designer from designing a high-fanout, time sensitive clock tree to every block of the design. Power is consumed only when a computation is performed, unlike clocked systems that constantly burn power even if they are not used.
- They are not forced to compute on unused data in globally clocked buses. By construction they provide fine grain clock gating.
- They can exploit the fact that the time required to compute a multiplication varies greatly depending on the operands. If the multiplier is zero, the 'done' signal triggers immediately, allowing the start of the next stage of computation. This bypasses all of unnecessary steps that take place in a typical synchronous multiplier that computes an intermediary sum before then throwing it away. This will be further explained in section III.

The drawback to asynchronous circuits is the overhead of the control circuitry required to detect the completion of an intermediary computation step, as well as in preventing successive calculations from overwriting a result before it has been written back. In addition, there is design complexity in fully understanding asynchronous design. This paper will show that asynchronous design becomes more attractive in sub-threshold sensors.

B. Advantages of Sub-Threshold Operation

A 3-5X improvement in power consumption can be expected for computation using sub-threshold operation [1], [2]. The biggest drawback to sub-threshold design is unpredictable delay, caused by variation in transistor current. In [1], the deleterious effect of lowering the supply voltage on clock frequency variation is shown, where the $3\sigma/\mu$ clock frequency variation can vary as much as 85% as shown in figure 1. In a synchronous system, this poses a significant problem. While in synchronous systems, the slowest sub-block determines the maximum clock frequency, asynchronous systems are tolerant of the maximum delay path and can therefore achieve higher throughput.

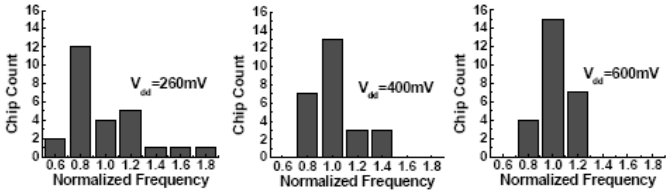
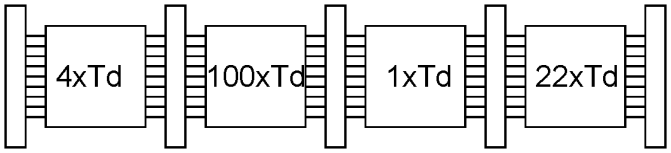


FIGURE 1. SYSTEM FREQUENCY VARIATION DUE TO SUB-THRESHOLD OPERATION IN SYNCHRONOUS CIRCUIT [1]

C. Combined Advantages

To illustrate the primary advantage of pairing sub-threshold operation and asynchronicity an example of a simple pipeline is shown in figure 2. Because of process variation coupled with the effects of running in the sub-threshold region each pipeline stage's delay is widely varied. For a typical synchronous system the total pipeline delay is the maximum delay multiplied by each stage because all stages share the same clock. If an asynchronous system is used, the overall pipeline delay will only be the sum delay from all stages. In this particular example the speedup ends up being well over 3X.



$$SyncPipelineDelay = (100 \times 4) \times T_d = 400 \times T_d$$
$$AsyncPipelineDelay = (4 + 100 + 1 + 22) \times T_d = 127 \times T_d$$

FIGURE 2. EXAMPLE PIPELINE WITH VARIABLE BLOCK DELAYS

III. IMPLEMENTATION

The multiplier is designed using a shift-and-add structure due to the simplicity in the implementation of pipelining. The multiplier resolves one bit of resolution in each stage, where each stage consists of an addition block. The addition block is a custom 4-bit adder, including done-detection and asynchronous handshake logic for communicating progress between computational stages.

Figure 3 illustrates the system architecture. Asynchronous logic blocks run along the top of the figure. These async. blocks trigger the computation block to begin computing, gather the occupancy state of adjacent stages, and generate the 'done' signal form the addition block. The addition blocks are responsible for computing the data delivered to them and generating a 'done' signal, while internal latches hold the data locally for the addition blocks that are computing.

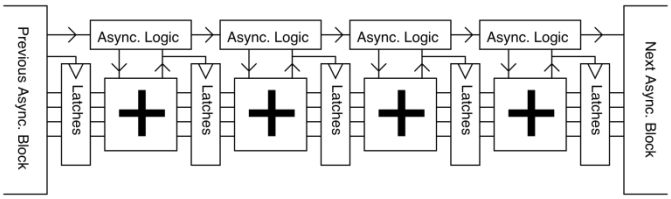


FIGURE 3. SYSTEM SCHEMATIC OF ASYNCHRONOUS MULTIPLIER

A. The Carry-Done Block

The addition logic is completed when it generates its carry bit. The carry bit is communicated using a 1-of-2 wire signaling protocol. When both wires are HI, the logic is indicating that the computation has not fully completed. Either C_o or $\overline{C_o}$ must be pulled LO indicating that the carry has resolved affirmatively or not. The logic for C_o and $\overline{C_o}$ are:

$$C_o = (A \bullet B) + (A \bullet C_i) + (B \bullet C_i)$$
$$\overline{C_o} = (\overline{A} \bullet \overline{B}) + (\overline{A} \bullet \overline{C_i}) + (\overline{B} \bullet \overline{C_i})$$

Using these two equations the carry-out true and false signals are explicitly generated. Pseudo-domino logic implements the equations as shown in figure 4. The single 'done' signal is asserted when either C_o or $\overline{C_o}$ are pulled LO. Next, C_o and its compliment are then latched into the next stage.

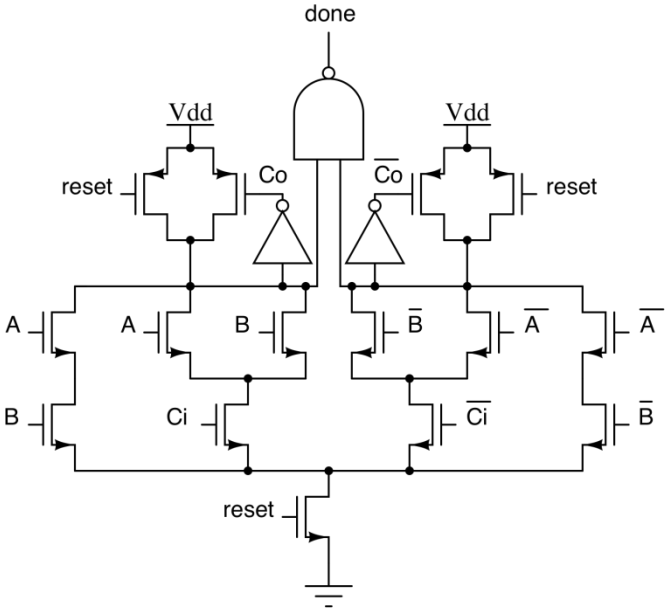


FIGURE 4. CIRCUIT SCHEMATIC OF CARRY-DONE BLOCK

B. The Asynchronous Communication Block

The asynchronous communication block determines the progress of the multiplication. A block level schematic of the circuit that performs this function can be observed in figure 5. This block uses three input control signals: one from each adjacent stage reports occupancy, while one from the adder indicates completion. This control block has two outputs: a signal named 'stage_done' which communicates its occupancy to adjacent stages and the signal that enables the adder to compute.

When the next stage signals that it is ready to receive data and the previous stage signals that its data is ready to be passed, the C-element [9] asserts. This enables adder operation and shifts in new data from the last stage. The adder signals its completion by lowering the signal 'adder_done', and then drives a toggle gate. Each transition received by a toggle gate alternatively places a transition on one of its two inputs. The adder's completion sends a transition form the top output of the toggle, there by disabling the adder. The adder responds by resetting and raising the signal 'adder_done'. This causes a transition on its lower output and communicates to the predecessor and successor stages that it has completed. Finally, this triggers the next stage to begin computing on the passed signals and tells the previous stage that is ready for more data.

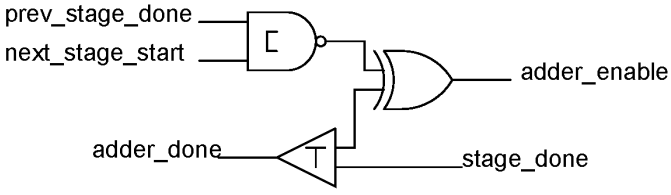


FIGURE 5. CIRCUIT SCHEMATIC OF ASYNCHRONOUS COMMUNICATION BLOCK.

C. Adder Done / Stage Skip Block

Each of the four carry bits generated form each 4-bit adder are sent to a simple block that waits for all four 'done' signals asserting a master 'done' signal to the next stage to begin computation. The block's operation is illustrated in figure 6.

In order to improve the performance of each adder a special circuit

was used to bypass each stage that doesn't require computation. For example, if the active bit of the multiplier is zero there is no need to wait for the sum to compute and hence the stage can be skipped. This allows for faster computation times, which will directly result in lower energy per computation.

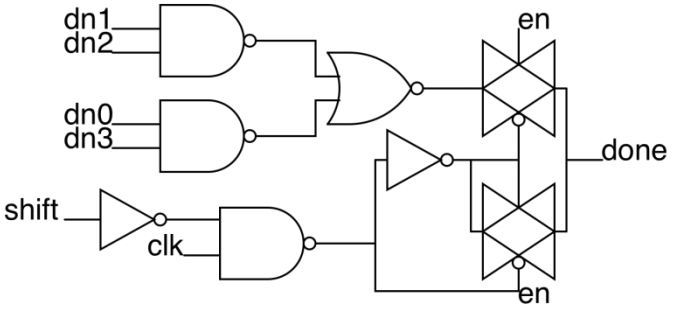


FIGURE 6. CIRCUIT SCHEMATIC OF STAGE SKIP BLOCK

IV. RESULTS

The multiplier was designed and the layout was completed in a 0.18µm process. Figure 7 is a picture of the completed layout. Table 1 shows the sizes of each block after layout and the overall multiplier dimensions. The asynchronous multiplier was simulated successfully at a minimum sub-threshold voltage of 150mV at best process corners. Across typical process corners, a $VDD_{(min)}$=230mV was achieved for an average case of input vectors. Under typical process corners, at VDD=350mV using an average-case multiplicand and multiplier, the multiplier exercised its lowest energy per computation of 1.17pJ/computation.

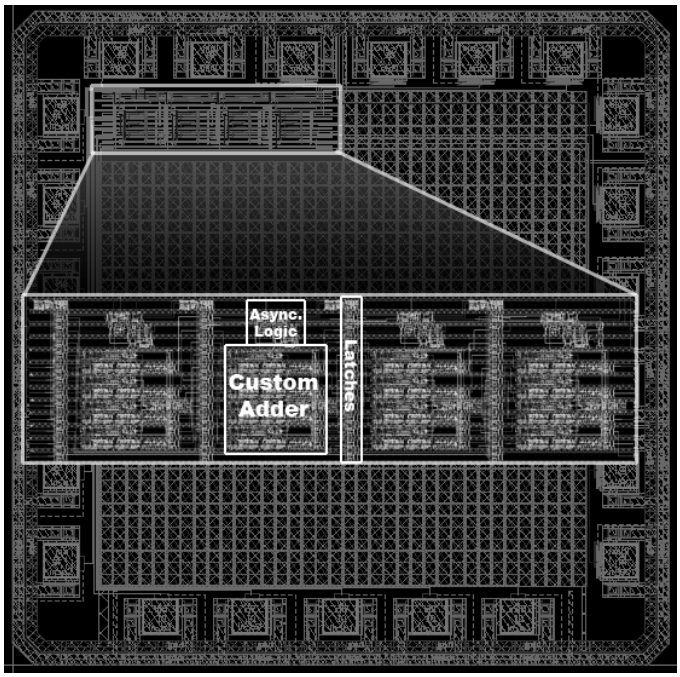


FIGURE 7. LAYOUT OF FINAL 4-BIT ASYNCHRONOUS MULTIPLIER

Block	*Size*
Asynchronous Logic	540 µm^2
Custom Adder	4356 µm^2
Latches	930 µm^2
Total Dimensions	366 µm x 96 µm

TABLE 1. BLOCK SIZES FOR MULTIPLIER LAYOUT

A. Delay Insensitivity

A synchronous multiplier must always operate at the worst-case frequency (for a worst-case input stimulus). However, an asynchronous multiplier operates at an average-case frequency. Its speed automatically adjusts to operate across significant temperature gradients, and unexpected delays are seamlessly time borrowed.

To verify its operation across process variation, the asynchronous multiplier was simulated across three different process corners: TT, SS, and FF with worst-case input vectors. Figure 8 shows the delay variability across these corners. Such process variations can result in delays that can differ by over 30X. While the proposed, on-chip asynchronous circuit can adapt to the best possible performance and suffer the worst; its synchronous counterpart is limited in throughput by the worst-case performance.

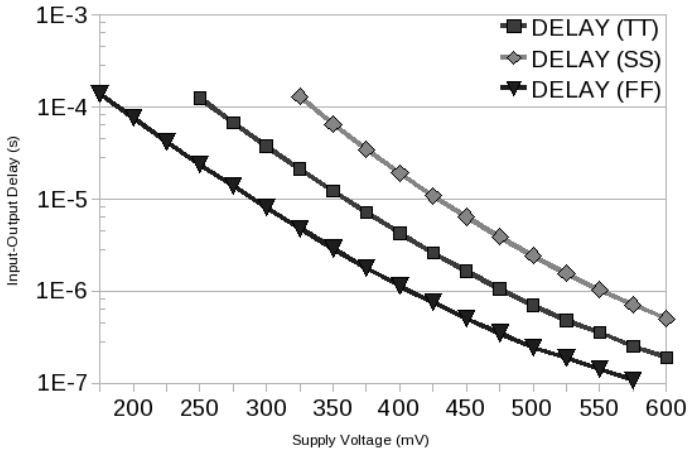


FIGURE 8. DELAY VARIATION ACROSS THREE DIFFERENT PROCESS CORNERS.

B. Asynchronous Speedup

One of the key features of this asynchronous multiplier is the speedup the circuit when the input conditions are favorable. For example, if this multiplier were to multiply '0' with '0', no computation needed. Because the architecture of the multiplier was designed to ignore a zero in the multiplier input of '0', a speedup can be observed. Figure 9 illustrates a speedup of 6X, consistently across multiple sub-threshold voltages, from worst to best-case input vectors.

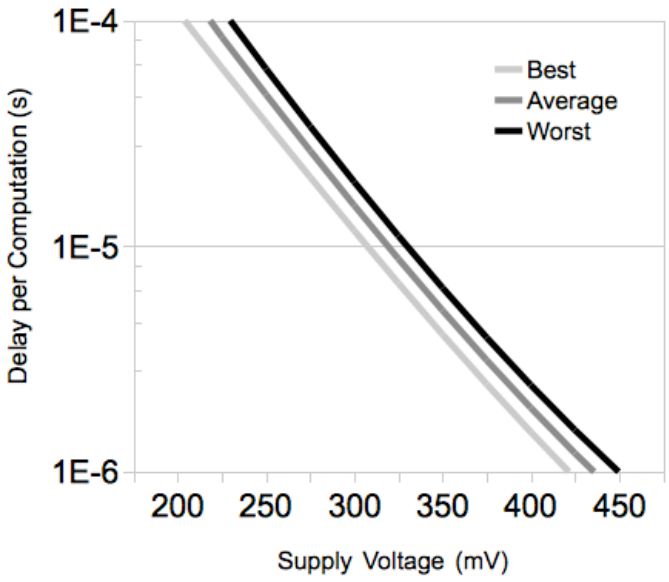


FIGURE 9. LOGARITHMIC PLOT OF DELAY VARIATION ACROSS THREE DIFFERENT INPUT CONDITIONS.

V. CONCLUSIONS

Asynchronous logic shows compelling advantages when operating in environments with extreme timing variations. Continued technology integration and scaling makes these environments more prevalent, especially for future energy-constrained, sensor network applications. This paper demonstrates that self-timed asynchronous design, while unconventional and more complex, can meet the looming timing and power dissipation challenges in future VLSI design. The results outlined in this paper further qualify the need for exploration of asynchronously controlled circuits in the sub-threshold region.

VI. REFERENCES

[1] B. Zhai, et al., “A 2.60pJ/Inst. Subthreshold Sensor Processor for Optimal Energy Effi- ciency,” IEEE Symposium on VLSI Circuits (VLSI-Symp), June 2006

[2] A. Wang and A. Chandrakasan, "A 180-mv subthreshold fft processor using a minimum energy design methodology," *Solid-State Circuits, IEEE Journal of*, vol. 40, no. 1, pp. 310-319, 2005.

[3] S. Hanson, et al., “Ultralow-voltage minimum-energy CMOS,” *IBM Journal of Research and Development*, Vol. 50, pp. 469-90, 2006.

[4] B.H. Calhoun, A. Wang and A. Chandrakasan, "Device sizing for minimum energy operation in subthreshold circuits," Custom Integrated Circuits Conf., 2004.

[5] B. H. Calhoun, A. Wang, and A. P. Chandrakasan, “Modeling and sizing for minimum energy operation in subthreshold circuits,” IEEE Journal of Solid-State Circuits, vol. 40, no. 9, pp. 1778–1786, 2005.

[6] B. Luderman, A. Albicki, “An Asunchronous Multiplier,” Proceedings of the Second Great Lakes Symposium on VLSI, 1992.

[7] Y. Liu, S. Furber, “The Design of a Low Power Asynchronous Multiplier,” ISLPED, 2004

[8] A. Martin, et al., “Asynchronous Techniques for System-on-Chip Design,” Proceedings of the IEEE, Vol. 94, No. 6, pp. 1089-1120, June 2006.

[9] J. Sparsø, S. Furber, “Principles of asynchronous circuit design: a systems perspective,” Springer, pp. 14-28, 2001