Degrading Precision Arithmetic for Low Power Signal Processing

Massimo Petricca, Gian Carlo Cardarilli, Alberto Nannarelli(1), Marco Re and Pietro Albicocco

Department of Electronics, University of Rome Tor Vergata, Rome, Italy
(1)DTU Informatics, Technical University of Denmark, Kongens Lyngby, Denmark

***Abstract*— Sometimes reducing the power dissipation of resource constrained electronic systems, such as those built for deep-space probes or for wearable devices is a top priority. In signal processing, it is possible to have an acceptable quality of the signal even introducing some errors. In this work, we analyze two methods to degrade the precision of arithmetic operations in DSP to save power. The first method is based on disabling the lower (least-significant) portion of the datapath by clock-gating and forcing zeros. The second method is based on lowering the supply voltage and re-designing the carry-chains in the datapath to adapt to the increased delays.**

I. Introduction

It might be desirable in some cases to decrease the precision of a Digital Signal Processing (DSP) system to save power/energy when a given level of quality is sufficient for the application.

The power dissipated in a circuit composed of N CMOS cells (gates) is

$$P_{TOT} = \underbrace{\sum_{i=1}^{N} \left(V_{DD}^2 C_{Li} + E_i^{int} \right) a_i f_{clk}}_{\text{dynamic}} + \underbrace{\sum_{i=1}^{N} V_{DD} I_i^{leak}}_{\text{static}} \quad (1)$$

where V_{DD} is the power supply voltage, C_{Li} is the capacitive load connected to the output of each gate, E_i^{int} (internal energy) accounts for the short-circuit currents in the transistors and the power dissipated in the switching of the internal nodes, a_i is the cell's switching activity and f_{clk} is the clock frequency. In addition to the dynamic part, static power dissipation due to device leakage (I^{leak} is device's leakage currents) must be accounted in deep sub-micron CMOS technologies.

Expression (1) suggests a number of ways to reduce the power dissipation. Some of the parameters of (1) are cell library dependent and cannot be modified at design time: E^{int} and I^{leak}. In this work, we focus on reducing the power dissipation by lowering the supply voltage V_{DD}, and by reducing the switching activity a_i in some nodes of the circuit.

The idea to disable part of the logic to reduce the switching activity for low power by having a variable word-length or processing adapted to the characteristics of the signal is not new [1], [2], [3].

On the other hand, by reducing the supply voltage the power dissipation decreases quadratically, but circuits get slower. By carefully designing the carry propagation logic in adders, it is possible to obtain accurate results at lower supply voltages [4], [5].

In our DSP system, we assume fixed-point numbers normalized in $[0, 1.0)$ represented in two's complement by n bits and with dynamic range 2^n. In such systems, degrading the precision means to reduce the dynamic range to 2^{n-k} by introducing an (additional) error in the representation.

This low-power driven degradation can be obtained in two ways:

I **DPA-I** – By disabling the logic in the arithmetic operations producing the lower part (least-significant bits) of the dynamic range.
II **DPA-II** – By lowering the supply voltage, and consequently increasing the circuit delays, and causing a limited carry-propagation (error) into the most significant part (reduced dynamic range).

We illustrate the two methods in the next sections.

II. DPA-I: Disabled logic in least-significant bits

Here we show the tradeoffs between loss of precision and power dissipation for two schemes applied to FIR filters. We can reduce the switching activity by either clock-gating paths starting from a register or by forcing zeros, or ones, in combinational logic.

By considering a FIR filter in transposed form (Fig. 1), we can consider two different approaches:

1) To use clock gating in the registers holding the least-significant (LS) portion of $x(t)$ and $y(t-k)$ (delay line) to *"freeze"* the LS part. The value of coefficients a_k does not change unless a different filter mask is loaded.
2) To force zeros, for example by driving the asynchronous reset of the LS part of the flip-flops.

In both cases the level of degradation can be changed by setting the clock-gating (individual flip-flop resets) control signals.

First, we show the bit detail in the two filter operations (addition and multiplication) for the two cases above.

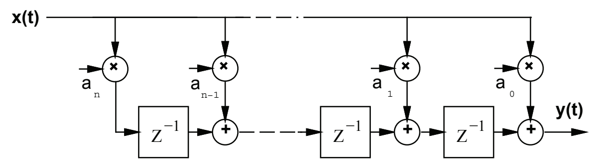


Fig. 1. FIR filter in transposed form.

A. DPA-I Addition

As an example, we use an 8-bit adder in which the dynamic range is reduced to 4 bits. The error-free 8-bit addition $A + B = S$ is reported below

$$\begin{array}{lcccccccccc} A & : & .a & a & a & a & a & a & a & a & + \\ B & : & .b & b & b & b & b & b & b & b & = \\ \hline S & : & .s & s & s & s & s & s & s & s & \end{array}$$

By disabling the logic by forcing zeros we obtain the same effect as truncation:

$$\begin{array}{lcccccccccc} A & : & .a & a & a & a & 0 & 0 & 0 & 0 & + \\ B & : & .b & b & b & b & 0 & 0 & 0 & 0 & = \\ \hline S & : & .s & s & s & \sigma & 0 & 0 & 0 & 0 & \end{array}$$

where σ indicates the sum bit that might be affected by the error[1]. The maximum error is

$$\epsilon_{max} = \left| \frac{2 \cdot (2^k - 1)}{2^n} \right|$$

where k is the number of bits disabled in the addends out of n. In the specific example the maximum error is $|\frac{30}{256}|$.

If we freeze the 4 LSBs, the frozen bits (marked χ in the following) preserve the last value before the freezing. In this case, we obtain

$$\begin{array}{lcccccccccc} A & : & .a & a & a & a & \chi & \chi & \chi & \chi & + \\ B & : & .b & b & b & b & \chi & \chi & \chi & \chi & = \\ \hline S & : & .s & s & s & \sigma & \sigma & \sigma & \sigma & \sigma & \end{array}$$

and the error is constant until the LSBs of both addends are frozen.

B. DPA-I Multiplication

For the multiplication $X \cdot A_k$ we consider a 4×4 multiplier in which two bits of multiplicand X and multiplier A_k are disabled. The error-free scheme is

$$\begin{array}{lccccccccc} X \cdot a_0 & : & & & & & x & x & x & x \\ X \cdot a_1 & : & & & & x & x & x & x & \\ X \cdot a_2 & : & & & x & x & x & x & & \\ X \cdot a_3 & : & & x & x & x & x & & & \\ \hline P & : & p & p & p & p & p & p & p & p \end{array}$$

[1]Clearly a carry which is propagated to the bits marked s might be generated in bit σ.

	freezing		forcing-to-0	
k	P_{save} [%]	$\lvert\epsilon_{mean}\rvert$	P_{save} [%]	$\lvert\epsilon_{mean}\rvert$
1	3	$562 < 2^{10}$	4	$573 < 2^{10}$
2	8	$550 < 2^{10}$	9	$1780 < 2^{11}$
3	13	$1871 < 2^{12}$	14	$4414 < 2^{13}$
4	18	$2419 < 2^{12}$	20	$10414 < 2^{14}$
5	24	$12486 < 2^{14}$	26	$25463 < 2^{15}$
6	30	$34985 < 2^{16}$	32	$67887 < 2^{17}$
7	37	$39819 < 2^{16}$	39	$199108 < 2^{18}$

$|\epsilon_{mean}|$ is divided by 2^{21}.

TABLE I
ERROR/POWER SAVING TRADEOFFS BETWEEN FREEZING AND FORCING-TO-0 METHODS.

By forcing zeros on the 2 LSBs of X and A_k, we have

$$\begin{array}{lccccccccc} X \cdot 0 & : & & & & & 0 & 0 & 0 & 0 \\ X \cdot 0 & : & & & & 0 & 0 & 0 & 0 & \\ X \cdot a_2 & : & & & x & x & 0 & 0 & & \\ X \cdot a_3 & : & & x & x & 0 & 0 & & & \\ \hline P & : & p & \rho & \rho & \rho & 0 & 0 & 0 & 0 \end{array}$$

Example: $15 \times 15 = 225 = (1110\ 0001)_2$
$\Rightarrow \quad [12 \times 4 + 12 \times 8] = 144 = (1001\ 0000)_2$
$\Rightarrow \quad \epsilon_{max} = 81 = (0101\ 0001)_2$

For this case the error is

$$\epsilon_{max} = \left| \frac{(2^k - 1) \cdot [2(2^n - 1) - (2^k - 1)]}{2^{2n}} \right|$$

On the other hand, if the 2 LSBs of X are frozen (all bits of A_k are constant - frozen - during filtering), we have

$$\begin{array}{lccccccccc} X \cdot a_0 & : & & & & & x & x & \chi & \chi \\ X \cdot a_1 & : & & & & x & x & \chi & \chi & \\ X \cdot a_2 & : & & & x & x & \chi & \chi & & \\ X \cdot a_3 & : & & x & x & \chi & \chi & & & \\ \hline P & : & p & \rho & \rho & \rho & \rho & \rho & \rho & \rho \end{array}$$

where bits marked ρ show the positions affected by errors.

Example: $4 \times 15 = 60 = (0011\ 1100)_2$.
Last X before freezing $X = 15$
$\Rightarrow \quad [4 + (11)_2] \times 15 = 105 = (0110\ 1001)_2$
$\Rightarrow \quad \epsilon = 45 = (0100\ 0101)_2$

C. DPA-I Filter Experiment

To compare the two DPA-I methods (freezing, and forcing-to-0), we implemented a 16-tap low-pass FIR filter with 10-bit input x and coefficients a_k, and output dynamic range of 20 bits which ensures error-free processing. For the two methods, we disable bits with granularity k for x and a_k, and $2k$ for the y delay line.

The results in terms of power saving and mean error with respect to the error-free filter are reported in Table I. The power savings are similar for the two methods and the error is better for the bit-freezing solution.

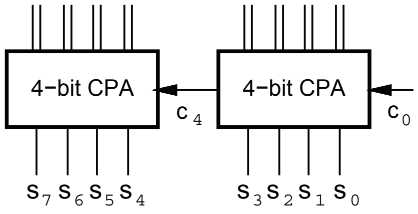


Fig. 2. Example of 8-bit adder.

III. DPA-II: LOWER SUPPLY VOLTAGE

By reducing the supply voltage the power dissipation decreases quadratically, but the delay increases. By keeping the system clocked at the nominal rate and by lowering the supply voltage errors might appear in the datapath degrading the precision.

A. Adder Example

We illustrate the method with an example of a 8-bit carry-propagate adder $A + B = S$. By splitting the 8-bit adder into two 4-bit portions as shown in Fig. 2, we can represent the bit-level addition algorithm as

$$\begin{array}{ccccccc|cccccc}
A & : & a_7 & a_6 & a_5 & a_4 & & a_3 & a_2 & a_1 & a_0 & + \\
B & : & b_7 & b_6 & b_5 & b_4 & & b_3 & b_2 & b_1 & b_0 & + \\
 & & & & & c_4 & & & & & c_0 & = \\
\hline
S & : & s_7 & s_6 & s_5 & s_4 & & s_3 & s_2 & s_1 & s_0 &
\end{array}$$

where c_4 is the carry-out of the least-significant portion on the adder. By disconnecting c_4, for S we get an error 2^4 for the combinations of input which generate $c_4 = 1$. Generalizing, if the n-bit adder (n even) is split into two equal parts, by disconnecting $c_{\frac{n}{2}}$ the error is either 0 or $2^{\frac{n}{2}}$.

By considering the propagation delays of the adders in Fig. 2, the delay of c_4 is close to the delay of the most-significant bit of the 4-bit adder s_4. We assume that the maximum delay of the 4-bit adder is the delay of c_4: t_{CPA4b}. Because the two 4-bit adders are identical, the maximum delay of the 8-bit adder is

$$t_{CPA8b} = t_{CPA4b} + t_{CPA4b} \simeq 2 \cdot t_{CPA4b} \ .$$

If the supply voltage is reduced from V_{DD} to V_2 such that the delay doubles

$$t^{V_2}_{CPA4b} = 2 \cdot t^{V_{DD}}_{CPA4b}$$

in the 8-bit adder of Fig. 2 there is not enough time to propagate c_4 through the 4-bit adder of the most significant portion. In other words, there is time only to propagate the carry inside a 4-bit adder. This is equivalent to disconnecting c_4 in Fig. 2. A timing diagram is shown in Fig. 3.

Summarizing, if the circuit is clocked at $T_{clk} = t^{V_{DD}}_{CPAnb}$, the adder is error-free when operating at V_{DD}, while it might have an error $\epsilon \approx 2^{\frac{n}{2}}$ when operating at $V_2 < V_{DD}$.

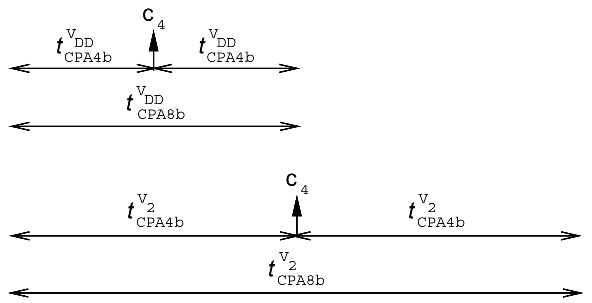


Fig. 3. Delay of adder for supply voltage at V_{DD} (top) and V_2 (bottom).

B. SPICE Characterization

To test the DPA-II method, we consider a 20-bit adder implemented as

- A regular 20-bit carry-propagate adder, called *CPA20b* in the following, which is synthesized to obtain the maximum speed.
- A scheme with two cascaded 10-bit adders, similar to the adder of Fig. 2, called *CPA2×10b*, in which the two identical 10-bit adders (*CPA10b*) are also synthesized to obtain the maximum speed.

The adders are synthesized by Synopsys Design Compiler in a 90 nm library of standard cells with nominal supply voltage $V_{DD} = 1.0\ V$. The synthesized netlist is then converted into a SPICE netlist and simulated with Synopsys Nanosim. Data in Table II show the dependency V_{DD}-delay for the adders.

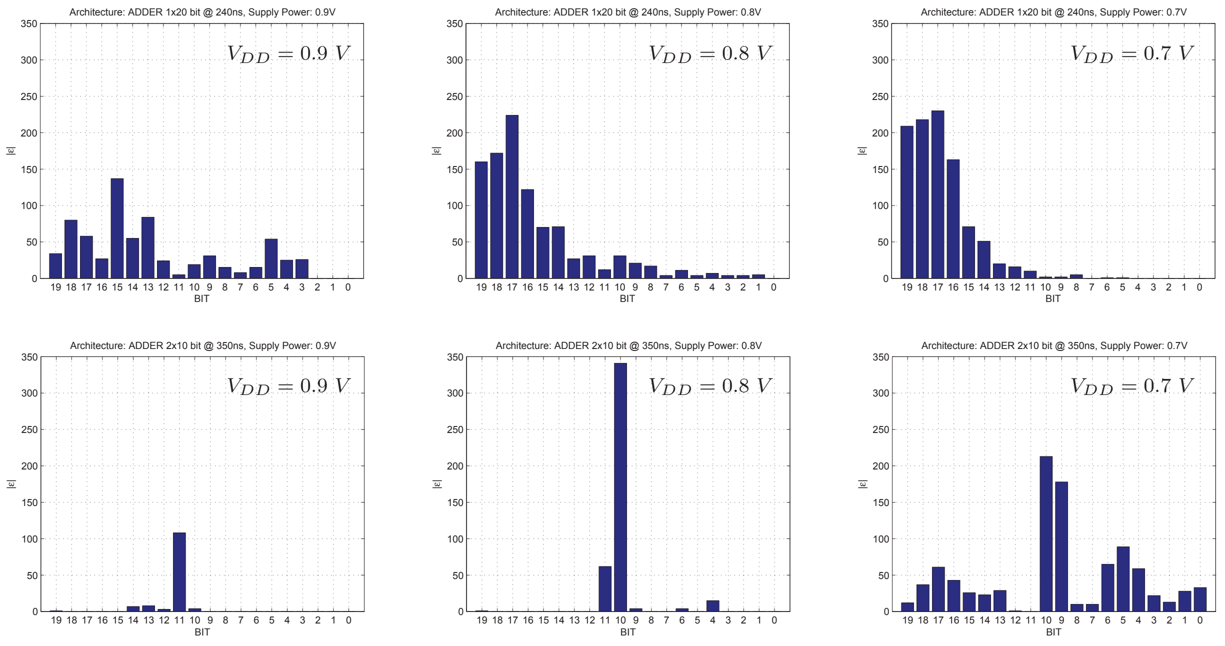
	CPA20b		*CPA2×10b*		*CPA10b*	
V_{DD} $[V]$	t_{MAX} $[ps]$	ratio	t_{MAX} $[ps]$	ratio	t_{MAX} $[ps]$	ratio
1.0	240	1.00	350	1.00	195	1.00
0.9	295	0.81	435	0.80	245	0.80
0.8	380	0.63	540	0.65	300	0.65
0.7	495	0.48	725	0.48	400	0.49

TABLE II
MAXIMUM DELAYS FOR DIFFERENT V_{DD}.

Table II shows that at $V_{DD} = 0.7\ V$ the delay is more than doubled for all adder schemes. By comparing the delays of *CPA2×10b* and *CPA10b*, we notice that the total delay in *CPA2×10b* (two cascaded *CPA10b* stages as in Fig. 2) is about 10% shorter than two times the delay of *CPA10b*.

C. Power-Precision Trade-off

The next step is to characterize the error when V_{DD} is lowered to 0.9, 0.8 and 0.7 V. This is done by operating the two adder schemes at the maximum frequency ($1/t_{MAX}$) under reduced V_{DD} and evaluating the error at the output. The histograms of the error distribution are shown in Fig. 4. In the histograms, the bar in position k indicates the occurrences of errors with magnitude $2^{k-1} < \epsilon < 2^k$. For example, $\epsilon = 2000 < 2^{11}$ will contribute to bar '11'.



a)

b)

Fig. 4. Trade-off Error-V_{DD} (at max speed). a) *CPA20b*. b) *CPA2×10b*.

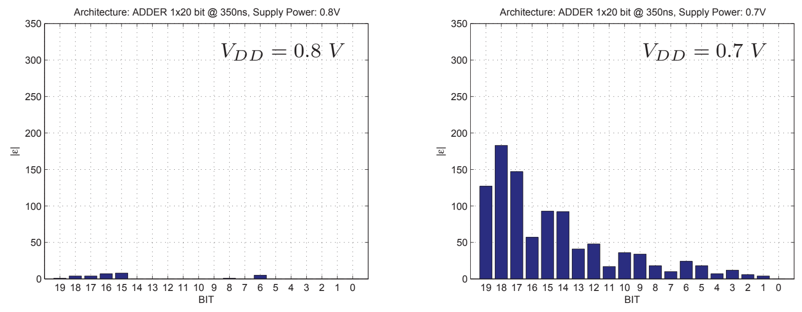


Fig. 5. Trade-off Error-V_{DD} for *CPA20b* at rate of 350 ps per operation.

For implementation *CPA20b* from Fig. 4.a, it is clear (bars are clustered in most-significant bits) that even for a small reduction of V_{DD} the error becomes unacceptable. From Table II for $V_{DD} = 0.9\ V$, the delay is incresed by 55 ps (22%) and 240 ps is not enough time to propagate the carry in the most-significant part.

On the other hand, for *CPA2×10b* (Fig. 4.b) most of the errors occur because the carry between the two cascaded 10-bit adders c_{10} is not propagated in the upper 10-bit adder for $V_{DD} = 0.9 - 0.8\ V$. For $V_{DD} = 0.7$, the carry propagation is limited within both portions of *CPA2×10b*. This experimental result confirms the theoretical analysis of Section III-A.

However, because *CPA20b* is faster than *CPA2×10b*, we can compare the error distribution for both schemes when clocked at the same rate (the delay of *CPA2×10b*, 350 ps). The histograms for *CPA20b* clocked at 350 ps are shown in Fig. 5. The histogram for $V_{DD} = 0.9\ V$ produced no errors and it has been omitted in the figure.

The data of Fig. 4.b and Fig. 5 together with the respective power dissipation figures are reported in Table III.

Table III shows the average power dissipation for the given V_{DD} values, the number of vectors with errors (# ϵ) for each experiment out of the 1,000 tested input combinations (randomly distributed), the maximum error (ϵ_{max}), and the average error ϵ_{mean}) for the 1,000 combinations.

From the data in Table III, we can derive the following conclusions:

1) Scheme *CPA20b* is error-free at $V_{DD} = 0.9\ V$ with power savings of about 15%.
2) For $V_{DD} = 0.8\ V$ the average error in *CPA2×10b* is significantly smaller although the power dissipated by *CPA20b* is slightly lower.
3) For $V_{DD} = 0.7$ both schemes give high average error and both are practically unusable at the rate of one addition per 350 ps.

		CPA20b						*CPA2×10b*					
T_C	V_{DD}	Power		Error				Power		Error			
$[ps]$	V	$[mW]$	ratio	# ϵ	ϵ_{max}	ϵ_{mean}		$[mW]$	ratio	# ϵ	ϵ_{max}	ϵ_{mean}	
350	1.0	2.94	1.00	0	-	-	-	3.08	1.00	0	-	-	-
350	0.9	2.43	0.83	0	-	-	-	2.56	0.83	131	22,528	454	$< 2^9$
350	0.8	1.84	0.63	30	524,288	2,851	$< 2^{12}$	1.95	0.63	427	2,012	481	$< 2^9$
350	0.7	1.35	0.46	974	786,428	157,676	$< 2^{18}$	1.42	0.46	952	525,424	32,926	$< 2^{16}$

Total number of vectors is 1,000

Note: ratio $\simeq V_{DD}^2$

TABLE III

SUMMARY OF TRADE-OFFS FOR *CPA20b* AND *CPA2×10b* SCHEMES.

Moreover, for the *CPA2×10b* scheme, the average error for $V_{DD} = 0.8 - 0.9\ V$ is the same although the number of errors is much higher (three times) in the 0.8 case. This result can be explained as at $V_{DD} = 0.8\ V$ most of the input combinations fail in propagating the carry in the upper 10-bit adder and the errors are clustered around the least-significant bits of the upper part $(2^{10}, 2^{11})$ as shown in Fig. 4.b. At $V_{DD} = 0.9\ V$ on the other hand, the circuit is faster (about 25%) and there is time to propagate the carry beyond the least-significant bits of the upper adder.

This last result shows that by designing the carry-propagation chain according to the available time slack (due to the reduction of V_{DD}) we can control the error propagation in the bits of different weight. This result also confirm, that having a faster adder under reduced supply voltage does not guarantee the best precision when compared to an adder which is slower, but has a carry-chain designed to tolerate some extra latency.

IV. CONCLUSIONS AND FUTURE WORK

It is possible to tradeoff precision with power dissipation in DSP systems. The first method proposed is based on the reduction of the circuit switching activity by selectively disabling the least-significant bits. The application of bit-freezing or forcing-to-0 to FIR filters leads to power savings of about 30% with an error roughly half-way the maximum precision.

Similar power consumption reductions can be obtained by lowering the power supply voltage and making sure that the carry-chains are designed to obtain a target error distribution.

Furthermore, the results obtained for DPA-II suggest to investigate further in the design of adders with configurable carry-chains to adapt to the extra latency introduced by reduced supply voltage.

REFERENCES

[1] P. Larsson and C. J. Nicol, "Self-adjusting bit precision for low-power digital filters," *Proc. of IEEE Symposium on VLSI Circuits*, p. 123124, June 1997.

[2] S. Yoshizawa and Y. Miyanaga, "Use of a Variable Wordlength Technique in an OFDM Receiver to Reduce Energy Dissipation," *IEEE Transactions on Circuits and Systems-I*, vol. 55, no. 9, pp. 2848–2859, Oct. 2008.

[3] A. Bonanno, A. Bocca, A. Macii, E. Macii, and M. Poncino, "Data-Driven Clock Gating for Digital Filters," *Proc. of 19th International Workshop on Power And Timing Modeling, Optimization and Simulation (PATMOS 2009)*, pp. 96–105, Sept. 2009.

[4] S.-L. Lu, "Speeding up processing with approximation circuits," *IEEE Computer Magazine*, vol. 37, no. 3, pp. 67–73, Mar. 2004.

[5] D. R. Kelly, B. J. Phillips, and S. Al-Sarawi, "Increasing Throughput of a RISC Architecture Using Arithmetic Data Value Speculation," *Proc. of 43rd Asilomar Conference on Signals, Systems, and Computers*, pp. 915–920, Nov. 2009.