# POWER-DELAY TRADEOFFS IN RESIDUE NUMBER SYSTEM

*Alberto Nannarelli, Gian Carlo Cardarilli and Marco Re*

Department of Electrical Engineering
University of Rome îTor Vergataî
Rome, 00133 ITALY

## ABSTRACT

In this paper we present some tradeoffs between delay and power consumption in the design of digital processors based on the Residue Number System (RNS). We focus on reducing the switching capacitance, and therefore the power, in modular adders and isomorph multipliers. Results on architectures such as FIR Ýlters, show that the techniques used to reduce the switching capacitance not only lead to more power efÝcient circuits, but also to a better performance.

## 1. INTRODUCTION

In CMOS circuits the main contribution to power dissipation is the dynamic dissipation due to the charging and discharging of load capacitances. The well known expression $P \propto V_{DD}^2 C_L a f$ shows the design parameters to change in order to modify the power Ýgures of a system. We indicate with $a$ the average activity of a circuit related to the clock frequency $f$ ($0 \leq a \leq 1$). Usually the main constraint to be met in system design is the timing constraint (e.g. clock cycle). Therefore, reduction of power dissipation by lowering $f$ is not implemented for these systems. On the other hand, lowering the supply voltage $V_{DD}$, although it leads to a quadratic reduction of the power consumed, is also not implemented because the propagation delay of a circuit increase at a rate proportional to $V_{DD}/(V_{DD} - V_T)^2$ [1]. As a consequence, in most cases power optimization is carried out by minimizing the so called switching capacitance $aC_L$: by reducing the switching activity in some parts of the circuit.

In a Residue Number System, deÝned by a set of relatively prime integers $\{m_1, m_2, \ldots, m_P\}$ with dynamic range $M = \prod m_i$, any integer $X \in \{0, 1, \ldots M-1\}$ has a unique representation given by:

$$X \overset{RNS}{\rightarrow} (\langle X \rangle_{m_1}, \langle X \rangle_{m_2}, \ldots, \langle X \rangle_{m_P})$$

This work was partially supported by MIUR National Project: Optimization of Digital Signal Processing Structures.

where $\langle X \rangle_{m_i}$ denotes the operation $X \ mod \ m_i$. Operations on single moduli are done in parallel

$$Z = X \ op \ Y \quad \overset{RNS}{\rightarrow} \quad \begin{cases} Z_{m_1} = \langle X_{m_1} \ op \ Y_{m_1} \rangle_{m_1} \\ Z_{m_2} = \langle X_{m_2} \ op \ Y_{m_2} \rangle_{m_2} \\ \ldots \quad \ldots \quad \ldots \\ Z_{m_P} = \langle X_{m_P} \ op \ Y_{m_P} \rangle_{m_P} \end{cases}$$

Therefore, the RNS allows the decomposition of a given dynamic range in slices of narrower bit-width on which the computation can be implemented in parallel [2].

This parallelization usually implies a shortening of the critical path of a system, because narrower datapath have shorter carry chains and consequently smaller delays. Taking advantage of this parallelism, in [3] the power dissipation in RNS structures is reduced by lowering the supply voltage until the desired value of delay is obtained. In [4], we presented an approach to reduce power dissipation in a RNS Ýlter without penalizing its speed. We used the time slack available in the non critical parallel paths and reduced $V_{DD}$ for these moduli until delays in all paths are equalized to the critical one. In this way, we reduce the power dissipation without affecting the overall performance. However, this multiple-$V_{DD}$ approach is very technology dependent and cannot be used if only one supply voltage is available for the whole chip, or it is not very effective if $V_{DD}$ cannot be easily scaled to the desired value.

In this work, we focus on power reduction techniques applied to the switching capacitance of RNS paths which are not the critical path. Our objective is to reduce the power dissipation without penalizing the overall performance.

Results show that reducing the switching capacitance, in some applications such as FIR Ýlters, not only leads to more power efÝcient circuits, but also to better performance.

## 2. MODULAR ADDITION

The modular addition $\langle a + b \rangle_m$, consists of two binary additions. If the result of $a + b$ exceeds the modulus (it is larger than $m - 1$), we have to subtract the modulus $m$. In

0-7803-7761-3/03/$17.00 ©2003 IEEE

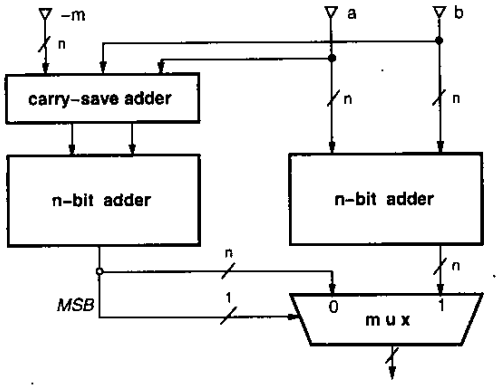


Figure 1: Structure of modular adder.

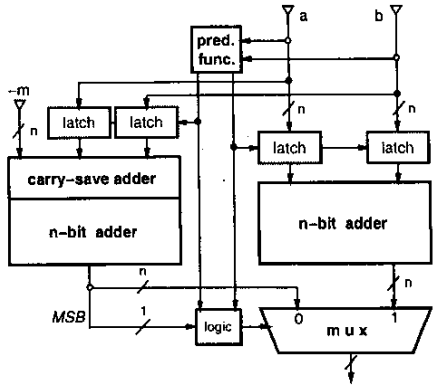


Figure 2: modular adder with latches and $F$.

order to speed-up the operation two additions are executed in parallel: $(a+b)$ and $(a+b-m)$.
If the sign of the three-term addition is negative it means that $(a+b) < m$ and the modular sum is $(a+b)$, otherwise the modular addition is the result of $(a+b-m)$. The above algorithm can be implemented with a carry-save adder (CSA), two n-bit ($n = \lceil \log_2 m \rceil$) adders[1] (nCPA), and a multiplexer as shown in Figure 1.

The modular adder can be simpliÝed when $m = 2^n$ (only a n-bit adder is required), and when $m = 2^n - 1$ (implemented with two n-bit adders in parallel, one with carry-in $c_{in} = 0$ the other with $c_{in} = 1$ and then selecting the correct result according to the $c_{out}$ computed in the adder with $c_{in} = 0$).

In the general case of Figure 1, the delay of the modular adder is

$$t_{modadd} = t_{CSA} + t_{nCPA} + t_{MUX} \ .$$

The power dissipation is that of two additions executed in parallel. We can reduce the power by switching off the path producing the unused result in the modular adder of Figure 1 by predicting if the binary sum of the two operands will exceed the modulus. In this way, according to the prediction, we select the path (left of right adder) in the modular adder to be active and switch off the other. This activation/deactivation can be done, for combinational logic, by using transparent latches which block the propagation of the new values in the adder to be turned off (Figure 2). With this new scheme, the prediction function must be well chosen because the delay of the resulting circuit is in the critical path as it is derived from Figure 2:

$$t'_{modadd} = t_{prediction} + t_{latch} + t_{modadd}$$

[1] Adders are implemented in a carry-look-ahead scheme. An extra inverter is required in the $(a+b-m)$ adder to obtain the sign.

The optimal prediction would be to implement the function:

$$F(a,b,m) = \text{sign of } (a+b-m)$$

because with this function we could always disable one of the two paths. However, the required sign-detector has roughly the same complexity of a n-bit adder and, therefore, the scheme will not be convenient in terms of both critical path (almost doubled) and power dissipation. For this reason, we have to design a simpler, but less precise, prediction function which will determine three cases (refer to Figure 2):

1. ENABLE RIGHT: correct result produced by adder at right (e.g. $a+b < m$);
2. ENABLE LEFT: correct result produced by adder at left (e.g. $a+b \geq m$);
3. ENABLE BOTH: the prediction function is unable to predict and both adders are enabled to produce the correct result.

It is clear, that the better the prediction function is, the smaller number of input combinations fall in case ENABLE BOTH.

As an example, for $m = 11$ we can choose the following prediction function $F$ (indicating with $a_3$ and $a_0$ the MSB and the LSB of $a$, respectively):

$$\begin{array}{ll} F_R = \overline{a_3}\,\overline{b_3}(a_2+b_2) & a < 7 \text{ AND } b < 3, \text{ or v.v.} \\ F_L = a_3 b_2 + a_2 b_3 + a_3 b_3 & a \geq 8 \text{ AND } b \geq 4, \text{ or v.v.} \end{array}$$

$$F(a,b,11) = \begin{cases} \text{ENABLE RIGHT} & \text{if } F_R = 1 \\ \text{ENABLE LEFT} & \text{if } F_L = 1 \\ \text{ENABLE BOTH} & \text{otherwise} \end{cases}$$

For $m = 11$ and $F$ deÝned above, in 40% of all possible input combinations the adder at right is enabled, in 27% the adder at left, and in 33% both adders are enabled. We implemented the original mod 11 adder and the new scheme,

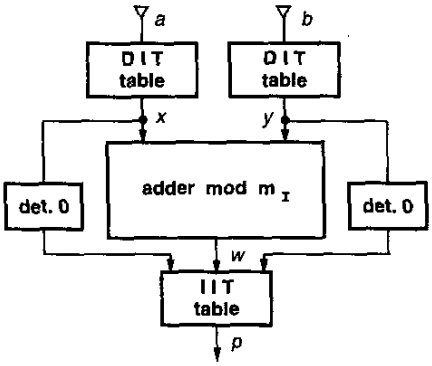


Figure 3: Structure of isomorphic multiplier.

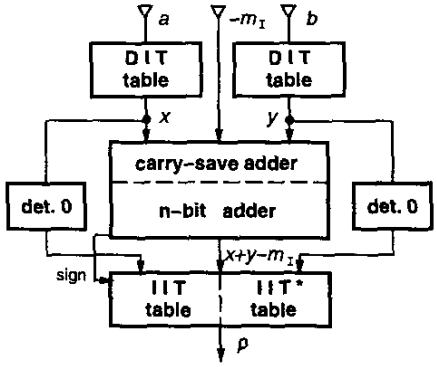


Figure 4: Isomorphic multiplier after Ýrst modiÝcation.

with a library of standard cells (0.35 $\mu m$a,) by using Synopsys Design Analyzer, and obtained the following values for delay and power dissipation:

| | delay $[ns]$ | P $[\mu W]$ @ 100 MHz |
|---|---|---|
| original | 1.45 | 440 |
| new | 1.68 | 420 |
| ratio | 1.15 | 0.95 |

The power reduction obtained (5%) in a favorable case (33% of ENABLE BOTH is quite good for a simple $F$) makes this techniques not very power efÝcient, although the delay overhead is not large and can be accommodated in a small time slack for RNS paths which are not critical. The introduced latches almost offset the reduced switching capacitance in the adders.

## 3. MODULAR MULTIPLICATION

To reduce the complexity of modular multiplication, we use the isomorphism technique [5] to implement the product of residues. This method works quite well for small prime moduli ($m < 64$). By using isomorphisms, the product of the two residues is transformed into the sum of their indices which are obtained by an isomorphic transformation. According to [5], if $m$ is prime there exists a primitive radix $q$ such that its powers modulus $m$ cover the set $[1, m-1]$:

$$p = \langle q^w \rangle_m \quad \text{with } p \in [1, m-1]\,, w \in [0, m-2]\,.$$

Both transformations $p \rightarrow w$ and $w \rightarrow p$ can be implemented with $m$-$1$-entry tables. Therefore, the product of $a$ and $b$ modulus $m$ can be obtained as:

$$\langle a \cdot b \rangle_m = \langle q^w \rangle_m \;, \quad w = \langle x + y \rangle_{m-1} \;, \qquad \begin{matrix} a = \langle q^x \rangle_m \\ b = \langle q^y \rangle_m \end{matrix}$$

In order to implement the modular multiplication the following operations are performed:

1. Two direct isomorphic transformations (DIT) to obtain $x$ and $y$;
2. One modulus $m_I = m - 1$ addition $\langle x + y \rangle_{m_I}$;
3. One inverse isomorphic transformations (IIT) to obtain the product.

The tables are implemented as synthesized multi-level logic and special attention has to be paid when one of the two operands is zero. In this case there exists no isomorphic correspondence and the modular adder has to be bypassed. The delay of the resulting scheme, shown in Figure 3, is

$$t_{isomult} = t_{DIT} + t_{CSA} + t_{nCPA} + t_{MUX} + t_{IIT}\,.$$

Because isomorph multipliers use modular adders in combination with tables, as a Ýrst step, we eliminate one of the adders in parallel and we incorporate the modulus reduction in the IIT table, as follows:

1. we compute: $x + y - m_I$ ($m_I = m - 1$, $x$ and $y$ are the indices of the isomorphism);
2. if the result is positive (e.g. $x + y \geq m_I$) we access the normal IIT table;
   otherwise we access a modiÝed table (IIT*) in which the inverse isomorphism is addressed by $\langle x + y - m_I \rangle_k$, where $k = \lceil \log_2 m_I \rceil$.

In this new scheme, depicted in Figure 4, the IIT tables are more complex (entries are doubled) than in the original scheme, but the delay is not increased because the multiplexer in the last stage of the modular adder is eliminated.

As a second step, we can incorporate the addition of the constant $-m_I$ in one of the DIT tables. The scheme of Figure 4 is modiÝed as follows:

1. In one of the DIT table instead of addressing the index of $y$ we address $e = y - m_I$;
2. using a n-bit adder, we compute $w = x + e = x + y - m_I$;

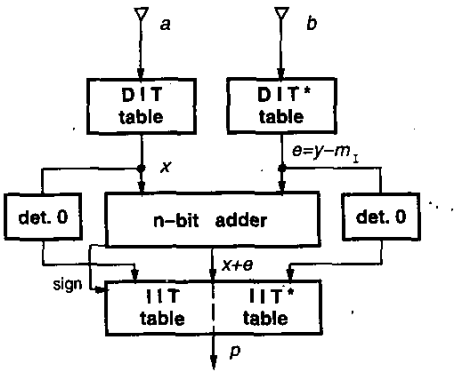


Figure 5: Isomorphic multiplier after second modiÝcation.

3. if $w$ is positive (e.g. $x + y \geq m_I$) we access the normal IIT table;
   otherwise we access the IIT* table, as previously shown.

In this new scheme (Figure 5) we have eliminated the carry-save adder which was in the critical path, while the modiÝcations in DIT* do not affect the table delay.

In conclusion, with this two modiÝcations we reduced the critical path of the isomorph multiplier to:

$$t'_{isomult} = t_{DIT} + t_{nCPA} + t_{IIT^*} .$$

As for the power dissipation, we implemented a $m = 11$ isomorph multiplier[2] and obtained the following results:

| | delay $[ns]$ | P $[\mu W]$ @ 100 MHz |
|---|---|---|
| original | 2.00 | 535 |
| new | 1.60 | 455 |
| ratio | 0.80 | 0.85 |

For this isomorph multiplier the power is reduced by 15% and the delay of the new scheme is 20% shorter.

## 4. APPLICATION TO FIR FILTERS

To evaluate the impact of the techniques presented above we implemented a 16-tap programmable FIR Ýlter using different schemes:

**STD** standard RNS implementation in transposed form (one isomorph multiplier and modular adder per tap);

**CS** carry-save RNS implementation as described in [4] (one isomorph multiplier and carry-save adder per tap);

**NEW** carry-save RNS implementation with new isomorph multipliers (one new isomorph multiplier and carry-save adder per tap).

[2] We implemented a multiplier which is used in FIR Ýlters in which the DIT tables are eliminated as explained in the next section.

| | min. $T_{cycle}$ | area (ND2 equiv.) | power @ 100 MHz |
|---|---|---|---|
| STD | 5.8 ns | 13,500 | 90 mW |
| CS | 4.3 ns | 18,200 | 100 mW |
| NEW | 3.7 ns | 17,100 | 94 mW |

Table 1: Results of FIR Ýlter implementations.

In all schemes DIT tables can be eliminated because the input $x$ is the multiplicand of all the multiplications and the DIT table is incorporated in the binary to RNS conversion, while the coefÝcients of the Ýlter (multiplicators) can be directly loaded at start-up as isomorphism indices. We chose the set of moduli { 11, 13, 29 } which covers a 12-bit dynamic range.

Results in Table 1 show that the CS-Ýlter is about 25% faster than the standard RNS, but consumes about 11% more power due the extra registers introduced to keep the carry-save representation of data. By implementing the new isomorph multiplier with reduced switching capacitance the power overhead is reduced to 5% and the speed-up with respect to the standard is 1.56 (about 37% faster).

## 5. CONCLUSIONS

We applied techniques aiming to reduce the switching capacitance to the design of modular adders and multipliers. For the adders, the power savings are quite small compared to the time overhead introduced. However, the technique of disabling some unused parts of a circuit by using transparent latches can lead to better results if those disabled parts are deeper (in terms of levels of gates) than adders, or if custom low-power latches are used.

The modiÝcations introduced for isomorph multipliers allow a reduction both in delay and power dissipation and make very attractive the use of this modiÝed multipliers in RNS FIR Ýlters.

## 6. REFERENCES

[1] W. Nebel and J. Mermet editors, *Low Power Design in Deep Submicron Electronics*, Kluwer Academic Publishers, 1997.

[2] M.A. Sodestrand, W.K. Jenkins, G. A. Jullien, and F. J. Taylor, *Residue Number System Arithmetic: Modern Applications in Digital Signal Processing*, New York: IEEE Press, 1986.

[3] W.L. Freking and K.K. Parhi, ìLow-power digital Ýlters using residue arithmetic,î *Thirty-First Asilomar Conference on Signals, Systems and Computers*, vol. 1, pp. 739ñ743, 1998.

[4] A. Del Re, A. Nannarelli, and M. Re, ìImplementation of Digital Filters in Carry-Save Residue Number System,î *Proc. of 35th Asilomar Conference on Signals, Systems, and Computers*, pp. 1309ñ1313, Nov. 2001.

[5] I.M. Vinogradov, *An Introduction to the Theory of Numbers*, New York: Pergamon Press, 1955.