An improved high-speed residue-to-binary converter based on the Chinese Remainder Theorem

Ph.D. Maciej CZYŻAK

He received his Ph.D. in 1985 from the Gdańsk University of Technology (GUT). From 1984 to 1990 he was a researcher and lecturer at the Institute of Informatics, Faculty of Electronics, GUT. Since 1994 he has been a lecturer of informatics at the Faculty of Electrical and Control Engineering, GUT. His research interests are in fast digital signal processing, computer arithmetic and VLSI design.

e-mail: m.czyzak@el.ly.ug.edu.pl

Abstract

A new high-speed residue-to-binary converter for five bit moduli based on the Chinese Remainder Theorem is presented. The orthogonal projections are computed by mapping using five-variable logic functions. The sum of projections is calculated using the Wallace tree. The output carry-save representation is partitioned into four segments in such a way that the sum of the numbers represented by the low-order segments does not exceed the Residue Number System (RNS) range $M$. The bits of the high-order segments are compressed by the small carry-propagate adder that in effect diminishes the size of the modulo $M$ generator used to reduce the number represented by the high-order segments. The obtained sum is smaller than $2M$, thus the effective two-operand final modulo $M$ adder can be used. The proposed converter can be pipelined on the full-adder level.

Keywords: residue number system, residue-to-binary conversion.

Ulepszony szybki konwerter z systemu resztoowego do systemu binarnego oparty na chińskim twierdzeniu o resztaх

Streszczenie

Zaprezentowano nową architekturę konwertera z systemu resztoowego do systemu binarnego dla modułów 5-bitowych opartą na chińskim twierdzeniu o resztaх. Projekcje ortogonalne określane są poprzez odczyty z pamięci ich obliczonych wartości. Pamięć symulowana jest poprzez użycie funkcji logicznych o liczbie zmiennych równej bitowej długości modułu. Suma projekcji obliczana jest przy użyciu sumatora wielooperandowego opartego na drzewie Wallace’a. Wszechstronne wektory sumy i przeniesienia są dzielone na cztery segmenty w taki sposób, że suma liczb reprezentowanych przez bity o małych wagach nie przekracza zakresu liczbowego systemu resztoowego, $M$. Bity należące do segmentów o starszych wagach są dodawane w niewielkim sumatorze, co w efekcie umożliwia znaczne zmniejszenie rozmiaru generatora stosowanego do redukcji modulo $M$ liczy reprezentowanej przez te bity. Suma otrzymana po zsumowaniu liczby reprezentowanej przez segmenty o małych wagach i zredukowanej liczby reprezentowanej przez segmenty starszych wagach nie przekracza $2M$, co umożliwia zastosowanie efektywnego sumatora końcowego modulo $M$. Konwerter w proponowanej konfiguracji może pracować potokowo na poziomie pełnego sumatora.

Słowa kluczowe: system resztoowy, konwersja z systemu resztoowego do systemu binarnego.

1. Introduction

The digital processing of signals in real time requires high-speed architectures. The Residue Number System (RNS) [1] is an effective tool for these applications where add-multiply operations dominate since it permits high-speed, low-level pipelined realization of addition, subtraction and multiplication. The RNS replaces operations in a large integer ring by the set of operations in smaller integer rings. Addition, subtraction and multiplication are performed independently on the corresponding digits of the representations of processed numbers without carries between the positions. In the RNS, the nonnegative integer $X$ from the number range $[0,M-1]$ is represented by $n$-tuple $\left[ X_{m_1}, X_{m_2}, \ldots, X_{m_n} \right]$, where the digit $\lfloor X_{m_j} \rfloor$ is the smallest nonnegative residue from the division of $X$ by $m_j$, $j=1,2,3,\ldots,n$, where $m_j$ are the elements of the system base, $B = \{m_1, m_2, \ldots, m_n\}$ and $M = \prod_{j=1}^{n} m_j$. The moduli $m_j$ should be pair-wise co-prime in order to preserve the division of $\lfloor X \rfloor$ by the formula $\lfloor X \rfloor = \sum_{j=1}^{n} \lfloor X \rfloor_{m_j}$, with the use of the CRT [1], is given by the formula

$$X = \sum_{j=1}^{n} X_{m_j}$$

(1)
3. The conversion algorithm

Given below is a step-by-step specification of the residue-to-binary conversion algorithm.

Input: The residue number \( X \), represented as

\[
X = (x_{n-1}, x_{n-2}, \ldots, x_1, x_0)
\]

Output: The binary representation of \( X \).

Step 1: Compute in parallel the orthogonal projections \( X_j \) for the individual \( x_j, j = 1, \ldots, n \).

\[
X_j = \left\lfloor \frac{x_j \cdot N \cdot M}{m_j} \right\rfloor
\]

Step 2: Compute

\[
\sum_{j=1}^{n} X_j = S^{(i)} + C^{(i)} =
\]

using the Wallace tree.

Step 3: Partition the Wallace tree output vectors

\[
S = (s_k, s_{k-1}, \ldots, s_0) \quad \text{and} \quad C = (c_k, c_{k-1}, \ldots, c_0)
\]

into segments

\[
S_H = (s_n^{(i)}, s_{n-1}^{(i)}, \ldots, s_0^{(i)}), \quad S_L = (s_{n-1}^{(i)}, s_{n-2}^{(i)}, \ldots, s_0^{(i)})
\]

with

\[
k = k = \left\lfloor \log_2 M \right\rfloor - 2,
\]

\[
C_{H} = (c_{n}^{(i)}, c_{n-1}^{(i)}, \ldots, c_0^{(i)}), \quad C_{L} = (c_{n-1}^{(i)}, c_{n-2}^{(i)}, \ldots, c_0^{(i)})
\]

so that the inequality is fulfilled

\[
\sum_{j=1}^{k} 2^j + \sum_{j=0}^{k-1} 2^j < M.
\]

Step 4: Compute

\[
S^{(i)} = \sum_{j=k}^{n} c_j^{(i)} 2^j + \sum_{j=k+1}^{n} s_j^{(i)} 2^j
\]

using the carry-propagate adder of the length

\[
l_{CPA} = \left\lfloor \log_2 nM \right\rfloor - \left\lfloor \log_2 M \right\rfloor + 1.
\]

Step 5: Compute by mapping

\[
S^{(i)} \leftrightarrow (s_k, s_{k-1}, \ldots, s_0) \rightarrow \left\lfloor S^{(i)} \right\rfloor = S^{(i)}.
\]

Step 6: Compute

\[
S^{(i)} + C^{(i)} = S^{(i)} + \sum_{j=0}^{k-1} c_j^{(i)} 2^j + \sum_{j=0}^{k-1} s_j^{(i)} 2^j
\]

using the CSA adder.

Step 7: Subtract \( M \) using two's complement

\[
S = S^{(i)} + C^{(i)} = S^{(i)} + 2^{\left\lfloor \log_2 M \right\rfloor} \cdot M
\]

using the CSA adder.
Step 8. Compute in parallel

\[ S^{(b)} = S^{(6)} + C^{(6)} \]  \hspace{1cm} (18a)

and

\[ S^{(7)} = S^{(6)} + C^{(6)}. \]  \hspace{1cm} (18b)

using two CPA adders.

If \( S^{(b)} \geq 2^{\log M} \) set \( \lambda = S^{(6)} \) else set \( \lambda = S^{(7)}. \)

4. Architecture of the new converter

The architecture of the new converter that implements the consecutive steps of the above algorithm is shown in Fig. 1. In the first stage the look-up tables \( LT^{(6)} \), \( LT^{(b)} \) are used, implemented as the \( [\log_2 M]/[\log_2 m] \) logic blocks composed of \( q \)-variable \( [\log_2 m] \) logic functions. (here \( q=5 \)). The residues are added by \( n \)-operand Wallace tree (CSA1). The higher-order segments are added by CPA1 and next the obtained sum is reduced modulo \( M \) by using the look-up table \( LT^{(0)} \). The CSA2 implements carry-save addition of the number represented by low-order segments. The CSA3 makes the subtraction of \( M \) in two’s complement. The CPA2 performs binary addition and the CPA3 two’s complement addition. Subsequently the output carry from the CPA3 is used to select the correct result by using the multiplexer (MUX1). The placement of latches is not shown in Fig. 1. In order to attain pipelining on the FA level, the latches have to be placed inside the logic blocks, since their delay is twice the FA delay.

5. Time-hardware complexity of the new converter

The method of evaluation of time-hardware complexity depends on purpose of evaluation and the type of a VLSI circuit. For evaluation of ASIC architectures based on standard cell library(STCL), a rough estimate of the circuit area can be obtained by summing the area of the required logic elements, expressed in \( \mu \text{cm}^2 \) or GE (Gate Equivalents), where the GE is the area of a two-input NAND with fan-out=1. However, the use of the FA area is more common. Also the FA delay \( t_{\text{FA}} \) is used as the unit. The STCL also allows a rough estimate of power consumption for the required frequency of operation. The models of logic components used here are from Samsung 0.18\( \mu \text{m} \) STDL 130 [17].

In the next subsections we shall analyze the hardware amount (HA) and delay of the new converter.

5.1. Hardware amount

The hardware amount (HA) of the converter architecture shown in Fig. 1 can be estimated in the following manner

\[ A_{\text{CONV}} = n \cdot \left[ [\log_2 M]/[\log_2 m] \right] A_{\text{CSA0}} + \left[ [\log_2 M]/[\log_2 m] \right] A_{\text{CSA1}} + \left[ [\log_2 M]/[\log_2 m] \right] A_{\text{CSA2}} \]

\[ + \left[ [\log_2 M]/[\log_2 m] \right] A_{\text{CSA3}} + \left[ [\log_2 M]/[\log_2 m] \right] A_{\text{FA}} + \left[ [\log_2 M]/[\log_2 m] \right] A_{\text{MUX}} \]  \hspace{1cm} (19)

where \( A_{\text{CSA0}} \) is the area of the block of \( q \)-variable \( i \) logic functions, \( A_{\text{CSA1}} = r \cdot A_{\text{FA}} \), is the area of the Ripple Carry Adder(RCA), where \( A_{\text{FA}} \) denotes the area of the FA, and

\[ A_{\text{CSA2}} = (p-2) \cdot [\log_2 M] \cdot A_{\text{FA}} + [\log_2 p] \cdot A_{\text{FA}}. \]  \hspace{1cm} (20)

The area of the Wallace tree with \( p \) operands of \( [\log_2 M] \)-bits. The CPAs of \( [\log_2 M] \)-bits are assumed in the Column Compression (CC) form as in [10] with the area of \( k \cdot [\log_2 k] \cdot A_{\text{FA}} \) for \( k \)-bit length. \( A_{\text{MUX}} \) is the area of a \( 2 \)-input multiplexer, and this area is here given by

\[ A_{\text{MUX}} = \Theta(n/2) \cdot A_{\text{MUX}}. \]  \hspace{1cm} (21)

The details of this design are given in [15]. It is assumed that the block of five \( 4 \)-variable logic functions is realized jointly as a block with a regular structure denoted as \( LF^{(4)} \). In the considered realization of the converter \( q \)-variable logic functions with \( q=5 \) are needed. Such a function is implemented using two multiplexed functions of \( 4 \)-variables realized jointly as a block with the suitable buffering. Using the data from [17] we receive \( A_{\text{FA}} = 87.7 \text{ GE} \) and \( A_{\text{MUX}} = 178 \text{ GE} \).

5.2. Delay evaluation

The converter delay can be evaluated as follows

\[ t_{\text{con}} = t_{\text{FA}} + \Theta(l/q) \cdot t_{\text{FA}} + t_{\text{FA}} + t_{\text{FA}} + 2t_{\text{FA}} + t_{\text{FA}} + t_{\text{FA}}. \]  \hspace{1cm} (21)

The delays of logic blocks are \( t_{\text{FA}} = 0.90 \text{ ns} \) and \( t_{\text{FA}} = 1.25 \text{ ns} \). \( \Theta() \) denotes the number of levels in the \( n \)-operand CSA tree[16], p.102). For the first CPA in (21) in the RCA form, due to the fast carry propagation in the FA from STDL 130 \( t_{\text{FA}} = 0.54 \text{ ns} \), \( t_{\text{CC0}} = 0.28 \text{ ns} \), we have

\[ t_{\text{CPA1}} = \Theta(k/2) \cdot t_{\text{FA}}. \]  \hspace{1cm} (22)

The second adder in the CC form [10] has the delay

\[ t_{\text{CPA2}} = \Theta([\log_2 k] + 1) \cdot t_{\text{FA}}. \]  \hspace{1cm} (23)
TheMUX(2-1) delay is equal to $t_{MUX(2-1)}=0.7\cdot t_{FA}$, for the data from [17].

6. Example and comparison

We shall compare the hardware amount and delay of the new converter with that from [8], for the RNS base:

$$B = \{32, 31, 29, 27, 25, 23, 19, 17\}.$$ 

The dynamic range for this base is 37.07 bits.

The hardware amount of the converter from [8] can be expressed as

$$A_{\text{conv},P} = n \cdot A_{\text{ROM}(2^r \log_2 M) + A_{\text{RAM}(\log_2 M) + A_{\text{ROM}(2^r \log_2 M) + A_{\text{ROM}(2^r \log_2 M) + 2 \cdot A_{\text{PROM}(\log_2 M)} + 2 \cdot A_{\text{PROM}(\log_2 M)} + \lceil \log_2 M \rceil A_{\text{MUX}(2^r)}},$$

(24)

where $r$, given in Table 1 for $n \leq 8$, is the maximum number of bits of $S_B$ and $C_L$, which determines the size of the modulo $M$ generator (MSB converter) [8].

<table>
<thead>
<tr>
<th>$n$</th>
<th>$r$</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>5,6</td>
<td>7</td>
</tr>
<tr>
<td>7,8</td>
<td>8</td>
</tr>
</tbody>
</table>

The ROM area is evaluated using the estimates from [8]. However, the maximum number bits per word (bpiw) $P_{\text{MAX}}$, should be relatively small. Here it is assumed that $P_{\text{MAX}} \leq 8$, that gives good estimates. The validity of such estimation can in principle be evaluated by using ROM areas expressed in $\mu m^2$ [17]. These data, however, are only available only for certain ROM sizes.

The delay of the converter from [8] can be written as

$$t_{\text{conv}} = t_{\text{ROM}(2^r \log_2 M) + \Theta(n) \cdot t_{FA} + t_{\text{ROM}(2^r \log_2 M} + 2 \cdot t_{\text{RAM}(\log_2 M)} + t_{\text{MUX}(2^r)}.$$ (25)

The ROM delays expressed in $t_{FA}$ units are $t_{\text{ROM}} \approx 4.2 t_{FA}$, $t_{\text{ROM}} \approx 4.6 t_{FA}$, for 5-bit and 8-bit address, respectively. These delays are quotients of the high-density synchronous diffusion programmable ROM access times and the worst case FA delay from [17].

In Table 2 below the hardware amounts, delays and maximum pipelining rates for three types of converters are given: for the new converter with memoryless (NML) realization and ROM-based (New-ROM) realization and the converter from [8] (Piestrak-ROM). In addition to the above analysis, also an approximate fomulation of the FPGA memoryless implementation of the new converter for the Xilinx Virtex-II Pro [19] has been carried out.

<table>
<thead>
<tr>
<th>Tab. 2. Comparison of converters</th>
</tr>
</thead>
<tbody>
<tr>
<td>NML</td>
</tr>
<tr>
<td>-----</td>
</tr>
<tr>
<td>HA[$A_{in}$]</td>
</tr>
<tr>
<td>Delay $[t_{FA}]$</td>
</tr>
<tr>
<td>Pipelining rate $[r_{FA}]$</td>
</tr>
</tbody>
</table>

It has been observed that about 520 slices are required for the assumed 8-moduli RNS base. Thus the converter requires less than half of the resources of the smallest device of this family (XC2VP2 comprises 1408 slices).

7. Conclusion

An effective CRT-based residue-to-binary converter for five bit moduli has been presented. The proposed technique of reduction of the sum of orthogonal projections to the range $[0,2M]$ allows for memoryless implementation and pipelining on the FA level. A slight increase in the hardware amount (5%) for the 37-bit RNS number range has been observed with a 5% reduction of the delay. For the ROM-based realization for the above number range the use of the new technique allows a hardware reduction by about 20% but at the cost of an increased delay by 10-15%.

8. Literatura

Artykuł recenzowany