USE OF THE CHINESE ABACUS METHOD FOR DIGITAL ARITHMETIC FUNCTIONS

Franco Maloberti 1, Chen Gang 1, 2

(1) Integrated Microsystems Laboratory, University of Pavia, Via Ferrata 1, 27100, Pavia, Italy
Tel. +39 382 505 205, Fax. +39 382 505 677
(2) Department of Electronics, Vocational and Technical College, Hunan Normal University,
Yuelushan, 410081 Changsha, P.R. China

ABSTRACT
This paper analyses the basic reasons that the Chinese abacus is used to perform arithmetic function as a popular and efficient technique. Proper electronic circuits, based on pass transistor and domino logic, are achieved to realize the same functions as the Chinese Abacus. Simulations with 0.5 μm CMOS technology show that a parallel 8 bit adder can run at 500 MHz. Moreover, a pipeline 8 bit adder and 8 x 8 bit multiplier can run at 1 GHz and 800 MHz respectively. Also, the compactness of the physical layout leads to a pretty small area for the circuits.

1. INTRODUCTION
The Chinese Abacus is a very popular and efficient technique used to perform arithmetic functions. It was used for centuries in many parts of the world (mainly in China) and it is still used in shops and small commercial enterprises. The main feature of the Chinese Abacus is speed of use: a well trained operator is often able to compete with electronic pocket calculators.

This observation leads us to analyse the basic reasons behind the speed displayed and, possibly, to transfer the same features to an electronic circuit. This paper shows that the use of the Chinese Abacus approach leads to promising results comparing them to similar published implementation [1] when, for example, a 0.5 μm CMOS process is adopted. The speed for an 8 bit pipeline full adder is as high as 1 GHz and a parallel 8 x 8 bit multiplier can run at 800 MHz. Moreover, the compactness of the physical layout leads to a relatively small area for the circuits.

The Chinese Abacus is made up of a set of unity elements representing the various decades of decimal number. Each element is made up of five beads each having a unity weight and two beads having a weight of five. The coding rule is thermometric, thus to represent a number lower than 5 the same number of beads will be raised in the main part of the unit. For numbers higher than 5 one bead with weight 5 will be lowered. The key feature of the Chinese Abacus is the use of upper beads: it allows the operator to minimise the transmission of rests, that, even in electronic circuits, limits speed. The number representation in the Chinese Abacus refers to the digital numeric system. As we are mostly interested in binary coding, we will use 4 lower beads instead of 5 to represent an octal numbering system.

2. CHINESE-BEAD BASIC BLOCKS
Basic functions of the Chinese Abacus approach are:
• Binary-to Thermometric (B/T) (or Digital to Thermometric) transformation
• Shift-up (SU) operation
• Thermometric-to Abacus (T/A) transformation
and to achieve the final result with a binary representation
• Abacus-to Binary (A/B) transformation

The above function sis carried out using pass transistors [2]. Fig. 1 shows a converter from 3 binary-encoded to thermometric representation (B/T). The control is given by the inputs b0, b1 and their complements b0, b1, b2. The output c0, c1, ..., c6 are made by a thermometric 0 representation or high impedance [3]. When the outputs are in the high impedance they are set to 1 during Cl=0 by the precharge to a
logic 1.

The SU function is achieved by the circuit shown in Fig. 2. It shifts up (SU) the input by one position and provides an extra one on the output, d6. Fig. 3 shows the thermometric-to-abacus (T/A) transformation. The logic input d3 is used to switch the inputs d6, d1, and d2 or the inputs d4, d3, and d5 towards the lower beads e0, e1, and e2. The input d3 itself constitutes the value of the upper bead, f0. The A/B function is performed by the circuit in Fig. 4.

The speed of the basic blocks described above was simulated using a digital CMOS 0.5 µm technology. Fig. 5 shows the transient response of the circuit in Fig. 1 cascaded with T/A converter in Fig. 3 in the worst-case situation. We have two plots. One is the direct output of the block and the other is the response passed through a CMOS inverter. It can be observed that the delay between the clock edge and the data at the output is less than 0.5 nsec. An excellent operation speed, compared with recent works [4], [5] can be expected.

3. THE CIRCUIT OF SUM OPERATION

The basic blocks discussed in the previous section are used here to achieve a N bits full adder (We will assume N=8 but the method can be extended to any N value). The operation required is:

\[ G = A + B \]  
\[ A = a_N 2^N + a_{N-1} 2^{N-1} + \ldots + a_1 2^1 + a_0 2^0 \]  
\[ B = b_N 2^N + b_{N-1} 2^{N-1} + \ldots + b_1 2^1 + b_0 2^0 \]  
\[ G = g_N 2^N + g_{N-1} 2^{N-1} + \ldots + g_1 2^1 + g_0 2^0 \]

The sum results from the following partial sums:

\[ G_{10} = A_{10} + B_{10}; \quad A_{10} = a_1 2^1 + a_0 2^0, \quad B_{10} = b_1 2^1 + b_0 2^0 \]  
\[ G_{32} = A_{32} + B_{32}; \quad A_{32} = a_3 2^3 + a_2 2^2, \quad B_{32} = b_3 2^3 + b_2 2^2 \]  
\[ G_{54} = A_{54} + B_{54}; \quad A_{54} = a_5 2^5 + a_4 2^4, \quad B_{54} = b_5 2^5 + b_4 2^4 \]  
\[ G_{76} = A_{76} + B_{76}; \quad A_{76} = a_7 2^7 + a_6 2^6, \quad B_{76} = b_7 2^7 + b_6 2^6 \]

of course the maximum value of the partial sums \( G_{ij} \) is 16. the binary representation requiring 3 bits. The sum \( G \) is then given by:

\[ G = G_{10} + 2^2 G_{32} + 2^4 G_{54} + 2^6 G_{76} \]  

The partial sums \( G_{10} \), \( G_{32} \), \( G_{54} \) and \( G_{76} \) are directly derived with the modified version of the B/T circuit shown in Fig. 6. We have two inputs \((a_i \text{ and } b_i)\) including their complements \( \overline{a_i} \) and \( \overline{b_i} \) with weight 1 and two inputs \((a_i \text{ and } b_i)\) including their complements \( \overline{a_i} \) and \( \overline{b_i} \) with weight 2. The B/T convert also contains a symbol representing the entire block. The six thermometric outputs \( C \) of the B/T block are followed by shift-up by one cell to account for the carry coming from the lower B/T and SU blocks. The maximum thermometric code at the outputs \( D \) corresponds to 7. Therefore, the thermometric to abacus transformation will provide three lower beads and, possibly one upper bead used to activate the shift-up block. The sum \( G \) results from a proper combination of the partial sums, as is stated by equation (9). We can achieve the sum \( G \) either using a parallel or a pipeline solution. Fig. 7 shows a possible parallel implementation: the weights of various \( G_{ij} \) outputs differ by a factor of
4. Thus, when the thermometric code exceeds level 4 the generated upper bead is used as the input of the shift-up by one block of the following block. Since the weight in each block is 4, the length of the critical path (bead #4 of the SU blocks) is halved compared with a conventional solution.

The sum operation can be achieved in a pipeline fashion by a proper use of the control clocks. Just a few modifications are required. Namely, it is necessary to add digital storing elements to preserve logic signals during precharge periods and pass them to the successive pipeline stage. The storing function can be carried out with a simple clocked inverter.

4. THE MULTIPLIER

The bead-code representation of numbers permits the multiplication of two N-bits digital numbers to be effectively implemented. Below we discuss a possible circuit solution for N = 8. The operation required is

\[ P = A \cdot B \]  

where A and B have been defined in (2) and (3). It is well known that the digital representation of P results from the sum of the following binary elements:

\[
\begin{align*}
& a_7 b_1 & P_{7,0} & P_{7,1} & P_{7,2} & P_{7,3} & P_{7,4} & P_{7,5} & P_{7,6} \\
& a_6 b_2 & P_{6,0} & P_{6,1} & P_{6,2} & P_{6,3} & P_{6,4} & P_{6,5} & P_{6,6} \\
& a_5 b_3 & P_{5,0} & P_{5,1} & P_{5,2} & P_{5,3} & P_{5,4} & P_{5,5} & P_{5,6} \\
& a_4 b_4 & P_{4,0} & P_{4,1} & P_{4,2} & P_{4,3} & P_{4,4} & P_{4,5} & P_{4,6} \\
& a_3 b_5 & P_{3,0} & P_{3,1} & P_{3,2} & P_{3,3} & P_{3,4} & P_{3,5} & P_{3,6} \\
& a_2 b_6 & P_{2,0} & P_{2,1} & P_{2,2} & P_{2,3} & P_{2,4} & P_{2,5} & P_{2,6} \\
& a_1 b_7 & P_{1,0} & P_{1,1} & P_{1,2} & P_{1,3} & P_{1,4} & P_{1,5} & P_{1,6} \\
& a_0 b_8 & P_{0,0} & P_{0,1} & P_{0,2} & P_{0,3} & P_{0,4} & P_{0,5} & P_{0,6}
\end{align*}
\]

where the elements of the same column have equal binary weight that increase by a factor of 2 moving from right to left. Of course, the term \( a_0 b_8 \) represents the LSB.

For our purposes a convenient way to calculate P is to express the sum (11) in the form:

\[
\begin{align*}
& a_7 b_1 & P_{7,0} & P_{7,1} & P_{7,2} & P_{7,3} & P_{7,4} & P_{7,5} & P_{7,6} \\
& a_6 b_2 & P_{6,0} & P_{6,1} & P_{6,2} & P_{6,3} & P_{6,4} & P_{6,5} & P_{6,6} \\
& a_5 b_3 & P_{5,0} & P_{5,1} & P_{5,2} & P_{5,3} & P_{5,4} & P_{5,5} & P_{5,6} \\
& a_4 b_4 & P_{4,0} & P_{4,1} & P_{4,2} & P_{4,3} & P_{4,4} & P_{4,5} & P_{4,6} \\
& a_3 b_5 & P_{3,0} & P_{3,1} & P_{3,2} & P_{3,3} & P_{3,4} & P_{3,5} & P_{3,6} \\
& a_2 b_6 & P_{2,0} & P_{2,1} & P_{2,2} & P_{2,3} & P_{2,4} & P_{2,5} & P_{2,6} \\
& a_1 b_7 & P_{1,0} & P_{1,1} & P_{1,2} & P_{1,3} & P_{1,4} & P_{1,5} & P_{1,6} \\
& a_0 b_8 & P_{0,0} & P_{0,1} & P_{0,2} & P_{0,3} & P_{0,4} & P_{0,5} & P_{0,6}
\end{align*}
\]

again, the elements on the same column have identical weight. Moreover, the weight of columns increases by a factor of 4 when moving from right to left. The generic partial sums \( P_{i,j} \) represents the expression:

\[
P_{i,j} = 2(a_i b_j + a_{i-1} b_{j+1}) + a_{i-1} b_j + a_{i-2} b_{j+1}
\]

where, \( i = 1,3,5,7 \) and \( j = 0,2,4,6; \) moreover, for \( i = 1 \) the last term in (13) must be 0. Using the same principle followed to compute (11) we can group the terms in (12) as follows:

\[
\begin{align*}
& K_{7,3} & H_{7,0} & H_{7,1} \\
& K_{7,7} & H_{7,4} & H_{7,5}
\end{align*}
\]

where the weight of each column increases by a factor of 16 moving from right to left, moreover:

\[
\begin{align*}
& K_{i,m} = 4a_i b_m + a_{i-1} b_{m-2} + P_{i,m-1} \\
& H_{i,j} = 4(P_{i,j} + P_{i-2,j+2} + P_{i-4,j+4})
\end{align*}
\]

and finally, we can represent (14) as:

\[
P = 2^8 Q_7 + Q_0
\]

\[
Q_7 = 16 K_{7,3} + K_{7,7} + H_{7,4} Q_5 + 16(H_{7,0} + H_{7,1}) + H_{7,5}
\]
complexity of the circuits is quite limited: for example, the function \( P_{ij} \) (eq. 13) requires 75 n-channel transistors and 44 p-channel transistors. The higher level functions \( K \) and \( H \) require 173 and 581 total transistors. The count for the 8 x 8 multiplier becomes just 3699 elements.

5. IMPLEMENTATION

The proposed circuits have been simulated with Spice using the transistor models of a 0.5\( \mu \)m CMOS process [6]. The results achieved for the parallel implementation and the pipeline implementation of the adder as well as the 8 x 8 pipeline multiplier are summarized in Table 1. We can observe that for the pipeline implementations the precharge phase and the I/O delay due to the transfer-gate operation are less than 0.5 nsec and 0.6 nsec, for the 8 bit adder and the 8 x 8 multiplier respectively. Therefore the maximum possible clock frequency is, in the nominal case, 1 GHz and 800 MHz respectively.

The total number of transistors required by the circuits is limited, it ranges from 296 to 3699. These figures are quite acceptable for the functions implemented. Moreover a custom layout allows a good compactness to be obtained. A possible layout for the binary-to-thermometric converter shows that we can accommodate 33 transistors within a 21 x 25 \( \mu \)m space, leading to an area per single transistor which is as small as 16 \( \mu \)m\(^2\). Assuming that the overhead for block interconnections is 100% of the basic block area we can estimate that the entire 8 x 8 pipelined multiplier can be accommodated in 0.12 mm\(^2\). The above estimation is rough; nevertheless, the result achieved gives us an idea of the possible chip area of the proposed solution.

REFERENCES


Table 1. Features of Abacus arithmetic circuits

<table>
<thead>
<tr>
<th>PARAMETER</th>
<th>8bit-parall. Adder</th>
<th>8bit-pipel. Adder</th>
<th>8x8 pipel. Multiplier</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max # pass trans.</td>
<td>8</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>Pre-charge delay</td>
<td>0.93 nsec</td>
<td>0.44 nsec</td>
<td>0.51 nsec</td>
</tr>
<tr>
<td>I/O delay</td>
<td>1.01 nsec</td>
<td>0.48 nsec</td>
<td>0.56 nsec</td>
</tr>
<tr>
<td>Max. Clock freq.</td>
<td>500 MHz</td>
<td>1 GHz</td>
<td>800 MHz</td>
</tr>
<tr>
<td># n-ch transistors</td>
<td>208</td>
<td>416</td>
<td>2746</td>
</tr>
<tr>
<td># p-ch transistors</td>
<td>88</td>
<td>220</td>
<td>953</td>
</tr>
<tr>
<td>total # of trans.</td>
<td>296</td>
<td>636</td>
<td>3699</td>
</tr>
</tbody>
</table>