A 1.55ns 0.015 mm² 64-bit Quad Number Comparator

Minsu Kim, Joo-Young Kim and Hoi-Jun Yoo

Department of EECS
KAIST
Daejeon, Republic of Korea
beatin@eeinfo.kaist.ac.kr

Abstract—This paper proposes a fast and small area 64-bit quad binary number comparator. Proposed bit-wise comparing logic chain (BCLC) and sequential strobes (SS) scheme enables 1.55ns 64-bit quad binary number comparison, which is 16% improvement compared to conventional comparator. With the help of BCLC and SS scheme, the proposed quad binary number comparator consumes 0.015 mm² in 0.18 um CMOS technology. Compared to previous works, the proposed comparator shows 9% reduction of transistor count and 13% area reduction.

I. INTRODUCTION

The digital comparator is one of the most common arithmetic components in digital systems such as DSPs and application-specific processors. Not only general purpose processors but also application-specific processors such as vision processors [2] and media processors [3] heavily utilize compare instructions in their computation. Sorting operation intrinsically involves numerous comparisons to organize large amounts of data in descending/ascending order and object recognition algorithm exhaustively use comparisons at its key-point localization stage [1-2]. Therefore, designing fast and area efficient binary number comparator is significant to overall system performance. Particularly, for image processing applications, simultaneous comparison of quad binary numbers is frequently required [2-3]. In addition, implementing comparators for more than 2 operands can be advantageous for future wide issue processors and SIMD DSPs.

Previously, one of the straightforward and easy approaches to implement a comparator is to exploit a parallel adder. However, this approach suffers from large transistor count and results in large silicon area. Therefore, to achieve fast operation with small area in binary number comparator, several researches [4]-[6] have been published. In previous work [6], Bitwise Competition Logic (BCL) is proposed to find the larger one between two binary numbers by recognizing the locations of the first 1 form the MSB. Even though the previous works achieve impressive improvement in transistor count and comparison speed, its architecture is limited to comparing 2 binary numbers.

In this paper, we propose an efficient quad binary number comparator that can compare 4 numbers simultaneously to extend internal computation capacity for data intensive applications. Proposed bit-wise comparing logic chain, sequential strobes scheme enables a fast quad binary comparison with small area. And new time difference sensing circuit is also proposed. As a result, the proposed comparator produces high performance with small physical area. The comparator is implemented in 0.18um CMOS technology.

The rest of this paper is organized as follows. Section II describes how to implement a two binary number comparator. In section III, the method for extending the comparator from two binary numbers to four binary numbers and the operation of time difference sensing circuit are explained. Performance evaluation and comparison with conventionally implemented comparators are shown in section IV and V. Finally, conclusion of this paper is given in section VI.

II. FAST AND SMALL AREA TWO BINARY NUMBER COMPARATOR

In the work [6], two inputs, A[i] and B[i] are pre-encoded into (A[i] · ~B[i]) and (~A[i] · B[i]) respectively to detect bit-wise difference. And after pre-encoding inputs, comparator decides the larger one by detecting which has the earlier first 1 from the MSB. But this approach is limited to comparing 2 binary numbers. Therefore we propose the

![Figure 1. The comparison of two binary numbers](image)

978-1-4244-2782-6/09/$25.00 ©2009 IEEE
comparison method without pre-encoding which limits the input numbers to 2.

Fig. 1 shows how to compare two binary numbers conceptually. Each bit of the two numbers is compared from the MSB to the LSB sequentially. Until the first detection of the bit-wise difference, the decision is postponed. And then, at the digit of first bit-wise difference, larger value is decided by the choosing the number which holds ‘1’.

To implement this comparison, the bit-wise comparing logic which is implemented using dynamic circuits shown in Fig. 2 can be exploited. Aline is pulled down to GND, when A is smaller than B and vice versa for Bline. And when A and B have the same value, two lines stay both ‘1’ and the decision is postponed to the next bit.

To implement multi-bit comparator, the bit-wise comparing logic have to be combined into bit-wise comparing logic chain (BCLC) as shown in Fig. 3. Before start of comparison, both lines and outputs are pre-charged to VDD. And sequential strobes (SS) which have constant time difference are exerted from the MSB to the LSB. And sensing circuit can detect which line falls to GND first. If one line is connected to GND first, the corresponding output also becomes ‘0’ and the other output is isolated from the line and kept at VDD by the sensing circuit. When two numbers are equal, both outputs are kept at VDD. This scheme is motivated from [7], which uses sequentially triggering signals. Since the operation speed depends on not logic’s speed but strobe signals’ time difference, the fast operation is possible.

III. PROPOSED QUAD BINARY NUMBER COMPARATOR

Based on above comparator circuit, it is possible to extend the number of compared numbers to the four beyond two.

The flow for comparing the magnitude of four binary numbers is described in Fig. 4. Each bit of four numbers is compared from the MSB to the LSB sequentially. If the case that a certain number’s bit is ‘0’ and one of others’ bit is ‘1’ at same bit position is checked, that number is not the largest among four binary numbers. The numbers which are proved not to the largest are eliminated step by step. Finally, the remaining number at the last is the largest one.

As shown in Fig. 5, a quad binary number comparator can be implemented using BCLC and SS. There are four parallel lines named Aline, Bline, Cline and Dline. If A[i] is ‘0’ and at least one of the bit of other inputs is ‘1’, Aline is pulled down. Therefore, bit-wise comparing logic is modified into \( (A[i] + B[i] + C[i] + D[i]) \) in Aline. The logics for other lines are implemented in the same manner.

<table>
<thead>
<tr>
<th>Table 1. Bit-wise comparing logics</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Maximum search</strong></td>
</tr>
<tr>
<td>Aline ( (A \cdot (B+C+D)) )</td>
</tr>
<tr>
<td>Bline ( (B \cdot (A+C+D)) )</td>
</tr>
<tr>
<td>Cline ( (C \cdot (A+B+D)) )</td>
</tr>
<tr>
<td>Dline ( (D \cdot (A+B+C)) )</td>
</tr>
<tr>
<td><strong>Minimum search</strong></td>
</tr>
<tr>
<td>Aline ( (A \cdot (B+C+D)) )</td>
</tr>
<tr>
<td>Bline ( (B \cdot (A+C+D)) )</td>
</tr>
<tr>
<td>Cline ( (C \cdot (A+B+D)) )</td>
</tr>
<tr>
<td>Dline ( (D \cdot (A+B+C)) )</td>
</tr>
</tbody>
</table>
case of performing comparison for minimum number
decision, bit-wise comparing logic is modified from \(\overline{A[i]} \cdot (B[i] + C[i] + D[i])\) to \(\overline{A[i]} \cdot (\overline{B[i]} + \overline{C[i]} + \overline{D[i]})\) in
Aline, and other lines are applied as the same way. The bit-
wise comparing logics are summarized in Table 1.

The operation of sensing circuit is as follows. As shown
in Fig. 6, after three lines fall to GND and corresponding
outputs also fall to GND, the last remaining output is
separated from line and pulled up to VDD by 3 serial PMOS.
Only maximum binary number’s output is left keeping VDD,
lastly. Therefore, the sensing circuit can detect timing
difference of signals among Aline, Bline, Cline and Dline.

Using the inverter chain, SS signals with constant time
interval are generated. The time interval is about 60ps which
is long enough for sensing circuit to detect time deference in
0.18\(\mu\)m process. Process mismatch simulations are carried
out to verify the reliability.

Because operation speed depends on strobe signal’s time
deference, the operation speed of quad binary number
comparator is similar with that of two binary numbers
comparator, which means that speed penalty is not so much.
Since increasing of transistor count is not quite large,
overhead area is very small. And as the same way as applied
above, a comparator that has a greater number of inputs can
be also implemented.

IV. IMPLEMENTATION OF 64-BIT QUAD BINARY NUMBER
COMPARATOR

Over 8-bit comparator, it is possible to implement
comparator for long binary number. If the length of input is
longer, operation time also becomes longer proportional to it.
Therefore, having long bits in one line is not desirable. We
implement 64-bit comparator exploiting hierarchical
approach as shown in Fig. 7. In the first stage, 64-bits are
divided into eight segments, and then they are compared in parallel. For the equality cases, this first stage is implemented by using 2 number comparators, muxes and additional logics. In the second stage, 8bit 4 number comparator performs the rest operation with the results of first stage.

The simulation result for the worst input case is shown in Fig. 8. And the layout of the comparator is shown in Fig. 9. As a result, proposed comparator shows 1.55 ns delay, 3376 transistor count, and 14646μm² area. It consumes 7.86mW in 1.8V when it operates at 200MHz frequency.

V. PERFORMANCE COMPARISON

Table 2. shows the performance comparison with conventional comparators. Elaborate scaling down is done on the some conventional results to compensate the processing gap. Since there is no 4 number binary comparator proposed before, conventional 4 number binary comparator’s performance is evaluated. For implementing conventional 4 number binary comparator, two stages of 64-bit comparator are required and three 64-bit comparators are used and there are some additional logics and muxes between stages. In evaluation of delay, comparator in work [5] is used, and in evaluation of transistor counter and area, comparator in work [6] is used. Since the best results among previous works are used for the evaluation, this conventional comparator’s performance is the best case in terms of both delay and area. As a result, compared to conventional comparator, proposed quad binary number comparator shows 16%, 9%, 13% improved result in operation speed, transistor count, and physical area, respectively.

VI. CONCLUSION

We propose a quad binary number comparator with high speed and small area. Using bit-wise comparing logic chain (BCLC) and sequential strobes (SS) scheme, quad binary number comparator is implemented efficiently. Compared to conventionally implemented comparator, it shows 16%, 9%, 13% improved result in operation speed, transistor count, and physical area, respectively.

REFERENCES


<table>
<thead>
<tr>
<th>Comparator Type</th>
<th>Delay</th>
<th>Transistor Count</th>
<th>Area</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 number binary</td>
<td>1.70 ns (scaled)</td>
<td>1640</td>
<td>6676 μm² (scaled)</td>
</tr>
<tr>
<td>Conventional comparator</td>
<td>0.8 ns (scaled)</td>
<td>3386</td>
<td>52785 μm² (scaled)</td>
</tr>
<tr>
<td>Proposed</td>
<td>1.55 ns</td>
<td>3376</td>
<td>14635 μm²</td>
</tr>
<tr>
<td>Improvement</td>
<td>16 %</td>
<td>9 %</td>
<td>13 %</td>
</tr>
</tbody>
</table>

* scale factor (0.6μm → 0.18μm) delay : 3.15, area : 10.9 [6]