A Low-Power Array Multiplier Using Separated Multiplication Technique

Chang-Young Han, Hyoung-Joon Park, and Lee-Sup Kim

Abstract—This brief proposes a separated multiplication technique that can be used in digital image signal processing such as finite impulse response (FIR) filters to reduce the power dissipation. Since the 2-D image data have high spatial redundancy, such that the higher bits of input pixels are hardly changed, the redundant multiplication of higher bits is avoided by separating multiplication into higher and lower parts. The calculated values of the higher bits are stored in memory cells, caches, such that they can be reused when a cache hit occurs. Therefore, the dynamic power is reduced by about 14% in multipliers by using the proposed separated multiplication technique (SMT) and in a 1-D 4-tap FIR filter by about 10%.

Index Terms—FIR filter, low power, LRU, multiplier.

I. INTRODUCTION

Digital signal processing (DSP) is the technology at the heart of the next generation of personal mobile communication systems. Most DSP systems incorporate a multiplication unit to implement algorithms such as convolution and filtering. In many DSP algorithms, the multiplier lies in the critical delay path and ultimately determines the performance of the algorithm. Present technologies possess computing capacities that allow the realization of computationally intensive tasks such as speech recognition and real-time digital video. However, the demand for high-performance portable systems incorporating multimedia capabilities has elevated design for low power to the forefront of design requirements in order to maintain reliability and provide longer hours of operation. As a result, a great deal of effort has been invested to reduce the energy dissipation in multipliers in various research fields [1]–[6].

In this brief, we exploit the spatial redundancy of images to reduce the energy dissipation in multipliers and apply this novel multiplier implementation technique to decimated FIR filters by factor 2, which are popularly used in discrete wavelet transform (DWT).

Differential coefficients method (DCM) [7]–[9] is a similar algorithm, which uses highly correlated data. DCM uses differential coefficients to multiply inputs, and compensates for the effect of differential coefficients by adding the previously computed partial product. However, the use of DCM necessitates careful consideration when coefficients are determined in the FIR filter design process. Furthermore, this method is not adequate for a sub-block based 2-D image due to the large memory requirement. Therefore, this brief proposes a new power reduction method for 2-D image processing using the separated multiplication technique.

The remainder of this brief is organized as follows. Section II briefly introduces the motivations for finding a new power reduction in the multiplier. The key idea for reducing power in the multiplication is described in Section III. The proposed idea is simulated and optimal solutions are applied to the decimated FIR filter module in Section IV. The proposed architecture is described in Section V and the results are analyzed in Section VI. Finally, conclusions are presented in Section VII.

II. MOTIVATIONS

A. Review of DCM Algorithm and Its Drawbacks

The direct form of the DCM filter uses coefficients and inputs directly. The DCM computes partial products with difference coefficients first, and then adds the previously computed partial products. We can rewrite the general FIR filter outputs $Y_{j+1}$ with the first-order difference DCM algorithm and obtain (1). The graphical descriptions of direct form and DCM form are shown in Fig. 1.

$$
Y_{j+1} = C_0 X_{j+1} + C_1 X_j + \cdots + C_{N-1} X_{j-N+2} \nonumber
$$

$$
= C_0 X_{j+1} + ((C_1 X_j - C_0 X_j) + C_0 X_j) + \cdots \nonumber
+ (C_{N-1} X_{j-N+2} - C_{N-2} X_{j-N+2} + C_{N-2} X_{j-N+2}) \nonumber
= C_0 X_{j+1} + (dC^1_1 X_j + C_0 X_j) + \cdots \nonumber
+ (dC^1_{N-1} X_{j-N+2} + C_{N-2} X_{j-N+2}) \nonumber
$$

where

$$
dC^k_k = C_k - C_{k-1}, \quad k = 1 to (N-1). \nonumber$$

The DCM needs small bit-width coefficients to reduce the power consumption in computing the partial products. However, power reduction cannot be expected if the bit-width of the coefficient differences is not significantly shorter than the original coefficient bit-width. This scheme cannot be applied to a system if the system uses the existing FIR filter and its coefficient difference is not small. Another drawback of the DCM is that it is not a suitable technique for sub-block based image processing using overlapping memory. After the last data of a certain row is filtered, the data contained in the registers, shown in Fig. 1(b), must be flushed to process the next row. Since the new row processing requires the previous sub-block pixels rather than the current, it requires additional logic to handle the flushing scheme. As a result, the latency increases if DCM is used in sub-block based image processing. Therefore, a new method is needed for efficiently handling sub-block based 2-D image processing.
B. Disadvantages of a Static Array Multiplier With Highly Correlated Inputs

Basic array multipliers, like the Baugh–Wooley scheme, consume low power and exhibit relatively good performance. However, their use can be limited to operands with less than 16-bits (e.g., 8-bits). For operands of 16-bits and over, the modified Booth algorithm reduces the partial product’s numbers by half. Its power dissipation is comparable to the Baugh–Wooley multiplier due to the circuitry overhead in the Booth algorithm. The fastest multipliers adopt the Wallace tree with modified Booth encoding. A Wallace tree would lead, in general, to larger power dissipation and area, due to the interconnect wires. Hence, it is not recommended for low power consumption applications [10], [11]. Therefore, we have chosen an array multiplier for 1-D FIR computations due to its popular usage in image processing.

The array multiplier is composed of rows of adders for recursive shift-addition operations. Sum and carry signals generated in the previous rows are transferred to the next rows as two of three inputs. Hence, the power consumption increases if the transitions of these signals occur frequently. Spatial variance is quite small except for the edges in the 2-D images, i.e., the difference between adjacent pixel values is very slight. This phenomenon is known as spatial redundancy. The digitized pixel value is frequently used as consecutive bit streams in digital image signal processing and the bit transitions in the input stage do not undergo sudden alteration. The higher 4-bits of an 8-bits pixel is highly correlated and its transition ratio is 0.146, the ratio of how many bits are toggled in the input bus with reference 1. The lower significant bits (LSBs) of the pixel data indirectly affect the rows of adders for the calculation of the higher bits of the input. The transitions in the later half rows cause unnecessary power dissipation in the array multiplier because the propagation of sum and carry signals of the previous upper rows give rise to unnecessary transitions in the later half rows. Therefore, a new multiplier architecture that exploits the spatial redundancy of images, consumes less power, and is adequate for sub-block based image processing is needed. Consequently, this brief proposes a new power reduction method, a separated multiplication technique, to overcome these problems.

III. SEPARATED MULTIPLICATION TECHNIQUE (SMT)

The SMT algorithm is summarized in (2) and shown in Fig. 2. The key idea is as follows: The multiplication is separated to the highly correlated part and the random part. The former results are stored in a buffer and reused when its results are valid. This scheme can reduce power dissipation because the multiplier handling the higher significant bits of a pixel is not required to activate.

\[
\text{If } \{\text{unsigned multiplication}\}\}
\]

\[
A(N - \text{bits}) \times B(M - \text{bits})
\]

\[
= \left( \sum_{k=0}^{N-1} a_k 2^k \right) \times \left( \sum_{k=0}^{M-1} b_k 2^k \right) \quad (a_k, b_k \in \{0, 1\})
\]

\[
= \left\{ \sum_{k=0}^{N-1} a_k 2^k \times \left( \sum_{k=0}^{M-1} b_k 2^k \right) \right\} + \left\{ \sum_{k=0}^{N-1} a_k 2^k \times \left( \sum_{k=0}^{M-1} b_k 2^{k+1} \right) \right\} \quad (l \equiv l - \text{bits from MSB})
\]

\[
= \text{higher} - \text{bits computation} + \text{lower} - \text{bits computation}
\]

(2)
For example, an $8 \times 16$ array multiplier is divided into two small $4 \times 16$ array multipliers and an adder for a complete multiplication. One multiplier is for the lower 4 bits computation and the other is for the higher 4 bits. The buffer, a simplified small cache, is used to store the multiplication results of the higher bits of the inputs and the filter coefficients. The multiplication results can be reused if the same higher bits coincide with tags in a cache. The tags are not addresses as in a general cache, but rather are the partial bits from the most significant bit (MSB) of the pixel. The multiplier for higher bits is activated occasionally when cache misses occur and does not activate when the hit signal is high.

We can obtain an exact multiplication result by adding two multiplier results or by summing the lower-bits multiplier result and cache data in the case of a hit. The unnecessary transitions can be removed in the bottom four rows as long as the higher 4 bits of the current pixel value is equal to one of the indexes of cache. Since a SMT can prevent redundant transitions in the multiplier, the power consumption can be dramatically reduced.

There is another array type multiplier using the Booth algorithm. The modified radix-4 Booth encoding array multiplier can reduce the rows of adders by half, but redundant transitions that occur by the lower bits of input are still transferred to the rows of adders for the higher bits computation. Hence, the power dissipation does not reduce by half even if its half rows of adders are eliminated. Due to the additional control logics, its power dissipation is comparable to the basic array multiplier. The power efficiency of SMT, also, is not doubled due to the additional cache, although it prevents redundant transitions by separating multiplication units. However, the power efficiency with SMT does rapidly increase as the bit-width of the highly correlated input widens because the power consumption in the cache is almost unaltered and SMT saves more rows of adders to one summed result. In contrast, the power efficiency with the Booth algorithm barely changes. In addition, the SMT algorithm can be applied to the modified Booth algorithm array multipliers due to its regular layout structure and local interconnect.

IV. OPTIMIZATION THROUGH SIMULATIONS

A. Optimal Cache Size

Fig. 3 shows the simulation results of real images with high-level C-language. The $x$ axis represents the number of cache entries and the $y$ axis is the hit ratio. The pixel is an 8-bits grayscale data, and hence eight curves are displayed. The highest curve (MSB1) implies that the cache-hit ratio is 93.77% when only the highest bit of the input pixels is compared with the tags and a cache stores just one entry. The second upper curve also shows that the hit ratio is 99.46% with three cache entries when two bits from the MSB are compared. The remaining curves are simulated in the same manner.

Now, let us determine how many bits must be compared from the MSB, and how many cache entries are needed. The cache-hit ratios are in the range of eighty to one hundred percent when the compared bit-width is less than 5 bits. The area overhead cannot be overcome if eight is selected as the optimal number for the cache entry. Therefore, the optimal solution is when the tag bit-width is four and the cache entry number is also four in the image processing. One cache entry requires 20 bits because of 16-bits filter coefficient and 4-bits data, so that the total size is only 10 Bytes. The cache-hit ratio is 91.60% in optimal conditions with the ideal least recently used (LRU) replacement strategy.

B. Replacement Strategy

When a miss occurs in a cache, the least recently used data must be discarded. Many hardware components are required to check and gather information when certain data are rejected in the next miss.

Hence, pseudo LRU is preferred in many caches in order to reduce hardware overhead. Many pseudo LRU solutions have been proposed in the literature and most of them considered random data. It is necessary to re-design an efficient replacement strategy for image processing since image data have high correlation.

The SMT exploits the spatial redundancy once more for a suitable cache replacement strategy for image processing. There are two curves in Fig. 4 and the decimated FIR filter module is experimented. This
interface-multipliers. Since the cache is unidirectional, its design is
However, in our architecture, the only link of a cache is to the internal
by Verilog–HDL simulation.
The nonseparated multiplication processors. Its functionality is verified
the lower bits multiplication. The final result is the same as the output of
the adder is necessary in order to add the results of the higher and the
are prepared in order to prevent transferring the multiplication result
the entries of a cache. The registers located in back of the multipliers
the new result of the multiplier instead and it is stored back to one of
in a cache. If the higher bits of input coincide with the tags in a cache,
rent multiplication results and previously calculated and stored results
The architecture contains multiplexers for choosing between the cur-
fig. shows the optimal cache size, 4-bits tag and 4-entries in a cache.
One is for ideal LRU strategy and the other is for the proposed scheme.
The difference in hit ratio of these two types is so slight that it is un-
necessary to select the ideal LRU with hardware overhead. However,
the proposed LRU logic is simplified to \( \log_2(N) \)-bits incrementer for
N-cache entries because the cache-hit probability increases as the ham-
mimg distance from the current pixel decreases. It is proved that the ef-
efect is the same as an ideal LRU through simulations when a cache has
4 or more entries. Therefore, the proposed scheme is more efficient for
image data processing than the ideal LRU due to low cost hardware and
better hit ratio when the cache entry number is larger than 4.

For example, consider a cache with four entries, index #0 to #3. Ini-
tially, each cache entry will be stored from #0 to #3 in turn. Then, when
a miss occurs, unconditionally entry #0 is updated like first-in–first-out
in our scheme. This is the entry farthest from the last location, entry #3.
This is again to exploit the spatial redundancy of an image, that is, the
adjacent pixel value has a high probability of being the same. It is pos-
sible to implement a fully associative cache with very small LRU logic.

V. PROPOSED LOW POWER ARCHITECTURE

A. Proposed Array Multiplier Architecture

Fig. 5 shows the proposed low power multiplier architecture using
SMT. This algorithm is applied to a 1-D FIR filter module in Fig. 6. The
architecture contains multiplexers for choosing between the cur-
rent multiplication results and previously calculated and stored results
in a cache. If the higher bits of input coincide with the tags in a cache,
the multiplexers select the stored value. If not, the multiplexers choose
the new result of the multiplier instead and it is stored back to one of
the entries of a cache. The registers located in back of the multipliers
are prepared in order to prevent transferring the multiplication result
directly to the next adder and to give time to load a cache data on bus.
The adder is necessary in order to add the results of the higher and the
lower bits multiplication. The final result is the same as the output of
the nonseparated multiplication processors. Its functionality is verified
by Verilog–HDL simulation.

A cache typically lies between the processor and the main memory.
However, in our architecture, the only link of a cache is to the internal
interface-multipliers. Since the cache is unidirectional, its design is
simplified. The modified cache architecture is based on the conven-
tional model and consists of TagRAM, DataRAM, valid-bit, and, a con-
troller. The tags are stored in TagRAM and the multiplication results in
DataRAM.

The conventional cache uses an address to reference valid data. How-
ever, a tag is not an address but the higher bits of pixel in our architec-
ture. In addition, each cache entry has a single bit that indicates whether
the entry is valid. All entries are initially set as invalid and the reset
signal is also on. The simplified cache architecture is outlined in Fig. 5.
If the cache-hit signal is high and the valid bit is on, the stored multi-
plication result in DataRAM is loaded on a data bus. When a miss oc-
curs, the new multiplication result is stored to DataRAM through the
data bus. The discarded data is determined by the proposed replace-
ment strategy. The cache is designed with full-associativity because
of its small size. Full-associativity is quite effective for this algorithm.
D-Latches are used as cache memory cells because the datapath library
does not contain a SRAM cell, the popular basic cache cell type.

B. SMTs Application to 1-D FIR Filter

The 1-D 4-tap FIR filter structure in Fig. 6 is based on the semi-re-
cursive pyramid algorithm [12] and its filter coefficients are taken from
Daubeches’s work [13]. This filter structure is more complex than a
general FIR filter architecture, direct form, in Fig. 1(a). The reasons are
as follows: First, its operation is based on \( 8 \times 8 \) sub-block based over-
lapping and decimation filtering by factor 2. The registers, D0–D5,
are prepared for sub-block processing and decimation. Second, high-
and low-pass filtering is achieved at every other clock in the same filter
module. Besides, the FIR filter architecture of [12] is slightly modified
for effective cache usage. We only use low-pass filter coefficients of
Daubeches 4-Tap coefficients due to complementariness of their high-
and low-pass filter coefficients. The high pass filter coefficients are
only negation and reverse order combinations of low-pass filter coeffi-
cients. One clock is for low-pass filtering with original low-pass filter
coefficients and the other is for high pass filtering with combinations
of the negation and reverse order low-pass filter coefficients. Hence, it
is necessary to redesign the final adder stages such that they share the
subtract operation for high pass filtering. This induces a slight increase
in the transitions in the output port of the multiplier for lower-bits com-
putation. Control signals are needed to externally manage the operation
of elements according to cache hits/misses. These signals are generated
together with cache control signals in the cache control unit.
TABLE I
EXAMPLE FOR POWER ESTIMATION PER CELLS

\[ VDD = 5.5 \text{V} \quad F = 20 \text{MHz} \]

<table>
<thead>
<tr>
<th>Unit</th>
<th>% of Active Bits</th>
<th>( \text{intAF} )</th>
<th>Cell Power</th>
<th>Ext Power</th>
<th>Total Power</th>
<th>Area (A)</th>
<th>Area (B)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conventional Architecture</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D/F (pixel)</td>
<td>0.249</td>
<td>0.125</td>
<td>55.720</td>
<td>0.416</td>
<td>114.660</td>
<td>8.392</td>
<td>56</td>
</tr>
<tr>
<td>MUL</td>
<td>0.249</td>
<td>0.037</td>
<td>20.150</td>
<td>0.624</td>
<td>40.030</td>
<td>9.386</td>
<td>976</td>
</tr>
<tr>
<td>D/F (Out)</td>
<td>0.192</td>
<td>0.058</td>
<td>111.520</td>
<td>0.832</td>
<td>225.359</td>
<td>16.780</td>
<td>112</td>
</tr>
<tr>
<td>Sum</td>
<td></td>
<td></td>
<td>437.100</td>
<td>224.180</td>
<td>1144</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Proposed Architecture</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Datapath</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D/F (High)</td>
<td>0.146</td>
<td>0.044</td>
<td>2.342</td>
<td>0.208</td>
<td>52.377</td>
<td>4.196</td>
<td>12</td>
</tr>
<tr>
<td>D/F (Low)</td>
<td>0.454</td>
<td>0.136</td>
<td>27.880</td>
<td>0.208</td>
<td>574.714</td>
<td>4.196</td>
<td>12</td>
</tr>
<tr>
<td>MUL (High)</td>
<td>0.146</td>
<td>0.022</td>
<td>73.520</td>
<td>1.066</td>
<td>148.555</td>
<td>108.580</td>
<td>486</td>
</tr>
<tr>
<td>MUL (Low)</td>
<td>0.454</td>
<td>0.069</td>
<td>875.300</td>
<td>0.884</td>
<td>1754.154</td>
<td>108.580</td>
<td>486</td>
</tr>
<tr>
<td>D/F (High_O)</td>
<td>0.105</td>
<td>0.032</td>
<td>11.710</td>
<td>0.960</td>
<td>252.780</td>
<td>20.990</td>
<td>140</td>
</tr>
<tr>
<td>D/F (Low_O)</td>
<td>0.278</td>
<td>0.083</td>
<td>111.520</td>
<td>1.184</td>
<td>229.855</td>
<td>16.780</td>
<td>112</td>
</tr>
<tr>
<td>MUX</td>
<td>0.374</td>
<td>0.094</td>
<td>88.350</td>
<td>1.340</td>
<td>1843.206</td>
<td>115.800</td>
<td>60</td>
</tr>
<tr>
<td>ADD</td>
<td>0.239</td>
<td>0.060</td>
<td>305.877</td>
<td>0.624</td>
<td>6140.191</td>
<td>29.840</td>
<td>160</td>
</tr>
<tr>
<td>D/F (F_Out)</td>
<td>0.239</td>
<td>0.072</td>
<td>111.520</td>
<td>0.832</td>
<td>2266.642</td>
<td>16.780</td>
<td>112</td>
</tr>
<tr>
<td>3-state Buffer</td>
<td>0.105</td>
<td>0.026</td>
<td>9.240</td>
<td>1.035</td>
<td>201.080</td>
<td>8.540</td>
<td>40</td>
</tr>
<tr>
<td>Cache</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tag_RAM</td>
<td>0.286</td>
<td>0.086</td>
<td>39.032</td>
<td>0.328</td>
<td>797.706</td>
<td>16.780</td>
<td>112</td>
</tr>
<tr>
<td>Data_RAM</td>
<td>0.359</td>
<td>0.108</td>
<td>68.960</td>
<td>1.020</td>
<td>1445.847</td>
<td>48.800</td>
<td>320</td>
</tr>
<tr>
<td>Tri-buffer</td>
<td>0.098</td>
<td>0.025</td>
<td>88.000</td>
<td>0.954</td>
<td>1774.429</td>
<td>34.160</td>
<td>80</td>
</tr>
<tr>
<td>Control &amp; Others</td>
<td>0.324</td>
<td>0.081</td>
<td>54.891</td>
<td>0.364</td>
<td>1116.658</td>
<td>20.064</td>
<td>57</td>
</tr>
<tr>
<td>Cache Sum</td>
<td></td>
<td></td>
<td>5133.640</td>
<td>119.808</td>
<td>569</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total Sum</td>
<td>37780.591</td>
<td>444.828</td>
<td>2209</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cache Ratio</td>
<td>0.136</td>
<td>0.259</td>
<td>0.258</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ratio</td>
<td>0.864</td>
<td>1.840</td>
<td>1.391</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

VI. POWER ANALYSIS OF THE SMT

The power consumption was estimated using a standard library cell in data book—0.6-μm 5-V Datapath Library. The total dynamic power consumption for an element is calculated.

Power Dissipation = \[ (Vdd^2 \times C_{int} \times \text{intAF}) + (Vdd^2 \times C_{ext} \times \text{extAF}) \] \times F. \hspace{1cm} (3)

It is composed of two terms. One is the internal power consumed in each element with consideration of the internal activity factor (intAF), that is, how often the element is activated per clock. The other is the external power consumed by load capacitance with consideration of the external activity factor (extAF), that is, the transition ratio on the output bits of elements, in the worst-case conditions. The numerical value in Figs. 5 and 6 is the percentage of active bits on the bus, that is, how many bits are toggled in the input or the output port of each element. The internally consumed power is taken from the datasheet. The load capacitance, \( C_{ext} \) in pF, is sum of the output capacitance of the previous element and the input capacitance of the following element. The total power consumption includes activity factors in order to take into account that all inputs and outputs are not always active during every clock cycle in each hardware element. Also, \( F \) is the output switching frequency in megahertz. It is assumed to be 20 MHz in this brief, for power estimation. We estimated accurate power distribution and area by the datapath library and verified the functionality by Verilog-XL simulations.

A. Power Analysis of Multiplier

Power consumption in each hardware component was estimated and Table I lists the estimated power of each cell in the multiplier. Consider the MUL (High) and D/F/F (Low) for the calculation of cell power in Table I. The internal power is calculated as follows: \(-263.90 + (17.80 \times N \times M)\times \text{intAF}, (6.97 \times N) \times \text{intAF} \)-unit is uW per MHz. \( N \) and \( M \) are the input bit-widths of the multiplier or D/F/F. The D/F/F (Low) is activated every clock so that its intAF is set at one. However, the cell power of MUL (High) will be reduced because this element does not activate when a cache hit occurs. Thus, the multiplier for the higher-bit, MUL (High), is activated effectively at 0.084 per clock because cache hit ratio is 90.4% with the optimal cache condition, that is, 4-bits tag and 4-entries and the proposed replacement strategy. This is controlled by the gated clock of cache-hit signal, which disables the unused block. Therefore, energy dissipation in these elements is just 8.4% of typical cell power.

The elements activated when a cache misses are: D/F/F (High), MUL (High), D/F/F (High_O), and the 3-state buffer. The total power is calculated from (3) with VDD 5.5 V and an operating frequency of 20 MHz. The \( \text{MUL (High)} \) item in Table I is a \( 4 \times 16 \) array multiplier and statistically 0.58-bits of the 4-bits input change in every clock cycle because the percentage of active bits in the input is 0.146. The percentage of active bits was measured with a real image data by C language. If random data are used, the percentage of active bits will be 0.5. MUL (High) consumes about 1.48 mW/MHz considering the external capacitance of a linked cell, and its equivalent gates are 496. The same procedure was applied to each cell. The total power consumption has been reduced to 14% in average in the proposed SMT and the ratio of energy consumption in a cache to total power is 13.6%. The power saving effect can be added to total power if the cache is made of SRAM cells instead of D-Latches.

B. Analysis of 1-D FIR Filter

We applied SMT to a 1-D FIR filter whose architecture is described in the previous section. The results for the FIR filter are listed in
Table II. The same type elements are grouped in the table. The effect of SMT in the 1-D FIR filter, regarding the decimated and overlapped operation, is about 10 percentages. Since we selected a somewhat more complex architecture than the general direct FIR form, the result shows relatively higher power dissipation due to additional elements. The portion of the cache to total energy dissipation is also 9%. However, area is increased by about 1.9 times compared to the conventional array multiplier and 1.7 times to the conventional 1-D filter module. This is due to an inserted cache implemented as D-Latches instead of SRAM cells and some additional primitives such as adders and multiplexers. D-Latches occupy much more area compared to SRAM cells—about 2.6 times. The area overhead and power dissipation can be dramatically reduced if the cache is implemented by SRAM cells.

The $4 \times 16$ multiplier is still in the critical delay path but cache data arrive more quickly since the cache size is very small in the proposed architecture. Hence, cache delay is not a problem. The system clock frequency can be increased by about 17% due to the shortened critical path. If we use the same operating frequency, this increase can be used for reducing the power supply voltage, which results in greater power reduction. Additionally, the relative power and area of multipliers occupied in the filter module increase as the filter tap increases. Therefore, the advantageousness of SMT increases as the filter tap increases.

VII. CONCLUSIONS

Many multipliers are needed in convolution or filtering VLSI architecture and their power consumption is about 80–90% of the total consumption in an FIR filter module. Consequently, we focused on power reduction during multiplication and exploited the spatial redundancy of images. Multipliers were separated into two parts, the higher and lower parts of multiplication. We optimized the architecture by separating the grayscale pixel data into the higher 4-bits and the lower 4-bits for multiplication and removed unnecessary multiplications by accessing a cache which stores the higher 4-bits multiplication result. A replacement strategy, employing an incrementor, was proposed for adequate cache architecture for image data processing. The total power was reduced by 14% in the multiplier and 10% in the 1-D 4-tap FIR filter. The speed increase was about 17% due to the use of the smaller multipliers. Therefore, dropping the supply voltage with the same clock frequency and constant throughput can save more power.

REFERENCES


