VLSI Architectures for Low-Power Signal Processing

Indrajit Chakrabarti
Dept. of Electronics & E.C.E.
IIT Kharagpur
Email : indrajit@ece.iitkgp.ernet.in

Reduction of Switching Activity
- CMOS circuits do not dissipate power if they are not switching.
- To reduce switching to the minimal level required to perform computation.
- To power down the complete circuit or its parts.
- To use clock gating.
- To use optimized circuit architectures.

Minimizing the number of Operations
- Vector Quantization to compress a video data stream.
- Basic computation is to compare input image with codebook entries. Mean square error is calculated as:

\[ D_x = \sum_{j=0}^{m} (X_j - C_x)^2 \]

Where \( X_j \) are elements of input vector (each made of 16 pixels) and \( C_x \) are elements of codebook vector. Assume a codebook of size 256.
- Vector Quantization (VQ) encoding algorithms can be either full search or tree search in nature.

Comparing Complexity of VQ Algorithms
- In **full search**, distortion is computed 256 times.
- For each distortion calculation, 16 memory accesses (to fetch codeword entries), 16 subtractions, 16 multiplications and 15 additions.
- Also, 255 comparisons to find minimum of 256 values.
- For **tree-structured VQ**, 2 * \( \log_2 256 \) - 16 distortion calculations; number of comparison is reduced to 8.
- So, computational complexity per input vector (16 pixels) is
  - for full-search technique, (i) no. of memory accesses = 16*256 = 4096, (ii) no. of multiplications = 16*256 = 4096, (iii) no. of subtractions = 16*256 = 4096, and (iv) no. of additions = 15*256 = 3840
  - for tree-search method, these figures are 256, 256, 256, and 240.

Choice of Data Representation
- Introduce redundancy by encoding an \( n \)-bit word into an \( m \)-bit word, where \( m > n \), before transmission. The \( 2^n \) unique data words get mapped to a larger set of \( 2^m \) words.
- Example 1: one-hot coding --- an \( n \)-bit word is encoded for transmission by placing a ‘1’ on \( i \)-th wire where \( 0 \leq i < 2^n - 1 \) is binary value of bit pattern, and ‘0’ on remaining \( m - i \) words.
- One-hot encoding ensures that only one 0 → 1 and 1 → 0 bit transition occurs whenever change of data word is needed.

Choice of Data Representation (Contd.)
- Example 2: Gray coding --- a gray code sequence is a set of numbers in which adjacent numbers have only one bit difference. For example, the eight decimal numbers 1-8, which have 4-bit binary representations of 0001, 0010, 0011, 0100, 0101, 0110, 0111, and 1000, have their corresponding gray code counterparts as 0001, 0011, 0010, 0110, 0111, 0101, 0100, and 1100.
- Gray coding is useful when data sent over a bus are sequential and highly correlated. Gray code always have one transition while number of transitions for binary form approaches 2.
- Gray coding has been applied to code address lines for both instruction access and data access to reduce no. of transitions.
Optimizing Data Form for Arithmetic

Two’s complement implementation of an Accumulator

Two’s Complement Implementation of Accumulator

- Accumulated result (adding 1024, 4-bit numbers) is transferred to accumulator register ACC at 64 KHz.
- MSB of input, bit 3 is tied to bits 4-13 of adder input for sign extension.
- So, whenever input switches sign, MSB changes, which makes all the ten higher order adder input bits switch.
- For uniformly distributed inputs, sign extension bits 3-13 have transition probability close to 0.5
- As accumulator is a low-pass filter, the bits of ACC_OUT have low transition probability.
- Adder output (before ACC latch) has high switching activity due to glitching whenever input changes sign.

Sign Magnitude Implementation of Accumulator

Two Accumulator datapaths used; one sums all positive numbers, the other sums all negative numbers

Trading Area for Switching Activity

- Latched sign-bit of input register generates gated clocks that enables positive datapath latch or negative datapath latch.
- After 1024 cycles, positive and negative accumulated values are transferred to separate registers.
- A subtract operation is performed at a lower frequency.
- Low power is ensured as there is no sign-extension and so adder has a low switching activity in higher-order bits. S-M performs better than 2’s-C for rapidly changing input.
- This implementation needs control circuitry (overhead capacitance) to generate timing signals for latches.
- When input changes very slowly, S-M performs worse.

Ordering of Input Signals

- Consider the problem of multiplying a signal with a constant coefficient, a common operation in signal processing.
- Multiplication is decomposed into shift-add operations and using canonical signed-digit representation.
  
  \[ \text{IN} \times (1 + 2^{-7} + 2^{-8}) = \text{IN} + \text{IN} \times 2^{-7} + \text{IN} \times 2^{-8} \]

- Lower power in IMP_2 compared to IMP_1: first adder in IMP_2 has small output amplitude and so lower switching act.

Glitching Reduction: Chain vs. Tree

- Due to finite delay from one block to next, a node can have multiple transitions in a single clock cycle
- In the chain, bottommost adder computes thrice per cycle
- In tree, signal paths are balanced and the “extra” transitions are reduced

[Images and diagrams are not transcribed but are integral to the explanations provided.]
**Resource Sharing**

- In direct mapping approach, there is a one-to-one correspondence between operations on signal flow graph and operators in final implementation.
- If there is area constraint or no need of high throughput, then time-multiplexed architectures are adopted.
- In time-multiplexed architecture, multiple operations on a signal flow graph are mapped onto the same functional unit.
- Which architecture leads to lower switching activity?
- Resource sharing (in time-multiplexed implementation) can destroy signal correlations and increase switching activity.

**Architectures Used in Practice**

- General purpose processors have low area and energy efficiency.
- Programmable digital signal processors involve much less energy.
- Direct mapping of algorithms into hardware gives efficient architectures.
  - Energy efficiency: 100-1000 MOPS/mW
  - Area efficiency: 100-1000 MOPS/mm²

---

**A simple signal flow graph**

A three-tap FIR filter

**Direct Mapped Architecture**

- Map operations of a signal flow graph into functional units.
- Obtain maximum parallelism.
- Use minimum clock rate and supply voltage to ensure reduced energy per operation.
- A high level of parallelism allows larger computational rates than uniprocessors without needing high clock rates.

---

**Direct-mapped Implementation of 3-tap FIR filter**

Two registers, three multipliers and two adders.

**Resource-shared Implementation of 3-tap FIR filter**

A single multiplier and a single adder.
Transformations to Minimize Power

- Transformations are changes done in computational structure while preserving the input-output behaviour.
- To optimize the number and type of computational modules, their interconnection and sequencing of operations.
- To optimize power dissipation while meeting throughput.
- Two key approaches to reduce power for a fixed throughput:
  - reduce supply voltage by utilizing speed-up transformation.
  - reduce effective capacitance switched.

Speed-up Transformations

- To reduce the number of control steps, so that slower control clock cycles can be used for a fixed throughput.
- This leads to a reduction in supply voltage.
- Consider a first order IIR filter shown below:

$$ x_n \rightarrow + \rightarrow A \rightarrow + \rightarrow D \rightarrow y_n $$

Assuming each operation takes one control cycle, critical path of this structure: 2

$$ \frac{T_{sample}}{T_{clock}} = 2 \text{ in this example} $$

Loop Unrolling (for IIR filter example)

- Two output samples are computed in parallel based on two input samples:
  $$ Y_{n+1} = X_n + A \cdot Y_n + A \cdot X_{n+1} + A^2 \cdot Y_{n+2} $$
  $$ Y_n = X_n + A \cdot Y_{n-1} + A \cdot X_{n-1} + A^2 \cdot Y_{n+1} $$

Critical path is 4 here, but 2 samples are processed in parallel; so, effective C.P. is unchanged. Thus, supply voltage cannot be altered.

Transformations Enabled by Loop Unrolling

- After loop unrolling, distributivity and constant propagation, the two output samples are represented as:
  $$ Y_{n+1} = X_n + A \cdot Y_n + A \cdot X_{n+1} + A^2 \cdot Y_{n+2} $$
  $$ Y_n = X_n + A \cdot Y_{n-1} + A \cdot X_{n-1} + A^2 \cdot Y_{n+1} $$

Implementation of the transformed solution has a critical path of 3 for processing two samples. Thus, effective C.P. is reduced, and supply voltage and so, power consumption can be decreased.

Pipelining to Reduce Power Consumption

- Pipelining can now be applied to the structure.
- Critical path is now reduced to 2 cycles.

Critical Path is same as original datapath (2 cycles). Sampling rate is half the original rate. This allows reduction in supply voltage, and consequent reduction of power, in spite of increase in effective capacitance.

Operation Reduction

- Reduction in number of operations reduces effective capacitance.
- Computation of polynomials by Horner’s scheme is often used in filter design and FFT calculations:
  $$ X^2 + AX + B = X \cdot (X+A) + B $$

Transformed flow graph has one less multiplication, and so has lower capacitance and power.
Operation Substitution

- Multiplications are substituted with additions

\[(A_r j A_i)(X_r j X_i) = (A_r X_r - A_i X_i) + j(A_i X_r + A_r X_i) = Y_r + jY_i\]

IMP_2 has lower effective capacitance, but longer critical path than IMP_1. So, voltage in IMP_1 can be lower than that in IMP_2.

Common Sub-Expression Elimination

- To scale the coefficients to minimize number of add-shift operations

\[A = IN * 0011\]
\[B = IN * 0011\]
\[A = (IN >> 4 + IN >> 3)\]
\[B = (IN >> 4 + IN >> 3 + IN >> 2)\]

- Multiplying an input with two coefficients
  - brute-force implementation: two outputs computed in parallel, requiring 5 shifts and 3 additions
  - uses common terms in constants to share some of the shift-add operations. Total shift-add operations reduced in number

Architecture for Motion Estimation

- To eliminate unnecessary computation using conservative approximation
- To compute a conservative estimate of the exact distortion value for each candidate macro-block before computing the exact distortion
- If the conservative estimate of distortion is larger than the minimum distortion found so far, then the exact distortion need not be computed.
- If power consumed in computing the estimate is very small, power is saved owing to the skipped distortion computation

Motion Estimation Architecture (contd.)

- A block-matching process with search range \(w\) has a area of \((2w+N)^2\) pels and

\[D(u, v) = \sum_{i=1}^{N} \sum_{j=1}^{N} |r(i, j) - s(i+u, j+v)|\]

where \(r(i, j)\) and \(s(i+u, j+v)\) are intensity values at position \((i, j)\) of the reference block and at position \((i+u, j+v)\) in the candidate macroblock \((u, v)\) in search area \(F\) respectively.

- Motion vector \((u_{opt}, v_{opt})\) is the displacement of candidate macroblock with minimum distortion \(D_{opt}(F)\) relative to the reference macroblock

Full-Search Block-Matching Algorithm

- In full-search block-matching motion estimation, each reference macroblock of size \(N\times N\) pels is compared to all candidate macroblocks in search area \(F\) to find best match.

- Candidate block with the minimum distortion (sum of absolute differences in intensity values) is considered the best match.

Motion Vector Computation

- Distortion for candidate block at position \((u, v)\) assuming the size of a macroblock is \(N\times N\), is defined as:

\[D(u, v) = \sum_{i=1}^{N} \sum_{j=1}^{N} |r(i, j) - s(i+u, j+v)|\]

where \(r(i, j)\) and \(s(i+u, j+v)\) are intensity values at position \((i, j)\) of the reference block and at position \((i+u, j+v)\) in the candidate macroblock \((u, v)\) in search area \(F\) respectively.

- Motion vector \((u_{opt}, v_{opt})\) is the displacement of candidate macroblock with minimum distortion \(D_{opt}(F)\) relative to the reference macroblock

\[D(u_{opt}, v_{opt}) = D_{min}(F) \leq D(u, v) \quad \forall (u, v) \in F\]
Complexity of Block-Matching

- A block-matching process with a search range $w$ has an area of $(2w+N)^2$ pels and $(2w+1)$ candidate macroblocks in each horizontal and vertical directions, or a total of $(2w+1)^2$ candidate macroblocks for each reference macroblock.
- The distortion value is computed for each candidate macroblock.
- The minimum value $D_{\text{min}}(V)$ is found from the pool of $(2w+1)^2$ candidates.
- The block-matching process generates a motion vector $(u_{opt}, v_{opt})$ and the corresponding distortion value, $D_{\text{min}}(V)$.

Conservative Approximation

- To reduce power consumption, replace $D(u, v)$ with a simpler function (conservative estimate).
  
  Removing one summation term from above, one has

  \[
  \sum_{i=1}^{N} \sum_{j=1}^{N} \left| s(i + u, j + v) - r(i, j) \right| 
  \geq \sum_{i=1}^{N} \sum_{j=1}^{N} \left| s(i + u, j + v) - r(i, j) \right| - \sum_{i=1}^{N} \sum_{j=1}^{N} r(i, j) 
  \]

  
  \[
  D(u, v) = \sum_{i=1}^{N} \sum_{j=1}^{N} \left| s(i + u, j + v) - r(i, j) \right| 
  \leq \sum_{i=1}^{N} \sum_{j=1}^{N} \left| s(i + u, j + v) - r(i, j) \right| 
  \]

  
  \[
  (1) 
  \]

Avoid computing $D(u, v)$ if unnecessary

- Define current minimum distortion ($D_{\text{min}}(V_{opt})$) as the minimum distortion value found in the area searched so far ($V_{opt}$).

  \[
  D_{\text{min}}(V_{opt}) = \min_{V} \sum_{i=1}^{N} \sum_{j=1}^{N} \left| s(i + u, j + v) - r(i, j) \right| 
  \]

- Skip computing the exact distortion function $D(-3, 2)$ if $D(-3, 2) \geq D_{\text{min}}(V_{opt})$.

Low-Power Motion Estimator Architecture

- Two main blocks: (a) Motion Estimator (PE array and Block Matching Unit) and (b) Distortion Approximation Unit (DAU).
- Simultaneously, DAU computes $\hat{D}(u+1, v)$ using data available for $D(u, v)$ and ME computes $\hat{D}(u, v)$.
- The disable signal is asserted if $\hat{D}(u+1, v) > D_{\text{min}}(V_{opt})$ which prevents data needed to compute $D(u+1, v)$ to enter PE array. This saves power.
Processing Element

- Each processing element (PE) computes absolute difference
  \( AD_{ij} = |r(i, j) - t(i, j)| \)
- It forwards sum of above AD and partial sum from row below
  \( AD_{ij} + \sum_{k} AD_{jk}(k, j) \)

Processing Element (PE) Array

- Pel data in search area are serially shifted in systolic PE array as follows:
  \( s(1-w,1-w), s(1-w,2-w), \ldots, s(1-w,N+w) \)
  (1st row),
  \( s(1-w,1-w), s(1-w,2-w), \ldots, s(N+w,N+w) \)
  (2nd row),
  \( \ldots \),
  \( s(N+w,1-w), s(N+w,2-w), \ldots, s(N+w,N+w) \)
  (N+2wth row)
- Blocking latch blocks a new candidate macroblock from entering PE circuits on assertion of disable signal from DAU

Block Matching Unit (BMU)

- The BMU generates the distortion value for each candidate macroblock and compares it to the current minimum distortion value.
- \( D_{min} \) and the motion vector are updated according to the result of comparison.

Distortion Approximation Unit (DAU)

\[
|D(u,v) - \sum_i |s(i+u, j+v) - s(i, j)| - SR| = \begin{cases} 
D(u-1,v) - SS(u+1,v-1) - SR, & \text{if } v = w, \ldots, w \text{ (1)} \\
D(u+1,v) - SS(u+1,-v) - SR, & \text{if } v = -w \text{ (2)}
\end{cases}
\]

\( \text{where } SS(u+v) = \sum_{i} |s(i+u, j+v)| \text{ and } SR = \sum_{i} r(i,j) \)

Energy Consumption

- Define: one unit of \( ADE \) (absolute difference equivalent) is the amount of energy consumed in computing one AD
- To calculate \( D(u,v) \) in systolic PE array requires \( N^2 \) ADs and \( N^2 \) additions, assuming no correlation of data
- One AD calculation approximates two additions
- Therefore computing \( D(u,v) \) consumes 1.5 \( N^2 \) ADE
- It may be shown that computation of \( \hat{D}(u+1,v) \) for \( u=N-1 \)
  and \( N \gg 1 \) consumes nearly \( 3N \) ADE
- Thus energy consumption for computing \( \hat{D}(u+1,v) \) is linear in \( N \), while energy consumption for computing \( D(u,v) \) is quadratic in \( N \)

References