VLSI Signal Processing

Programmable DSP Architectures

Chih-Wei Liu
VLSI Signal Processing Lab
Department of Electronics Engineering
National Chiao Tung University
Outline

- **DSP Arithmetic**
- Stream Interface Unit & A Simple DSP Core
- Register Organization for DSP
DSP Arithmetic

- Dynamic range
- Precision

Floating-point (FP) arithmetic
- Fractional (1.x) operations with automatic rounding
- Automatic radix-point tracking in “exponent” with hardware to use the full precision of the “mantissa”
- Unnecessary dynamic range for most DSP applications with poor speed, silicon area, & power consumption

Integer arithmetic
- Programmer must estimate the dynamic range to prevent overflow
- Tradeoffs between quality (precision) & speed (for explicit exponent tracking & data rounding)
Integer Arithmetic

- Assume a 256-level grayscale image is going to be filtered with a 7-tap FIR with symmetric coefficients 
  \[-0.0645, -0.0407, 0.4181, 0.7885\]
- For 24-bit integer units,
  - 8 bits reserved for inputs & 3 bits for 7-element accumulations
  Thus, the coefficients has at most 13-bit precision
- The fractional coefficients are multiplied by \(2^{13} (8192)\) to integer numbers \([-528, -333, 3425, 6459]\)
- After the inner product (i.e. multiply-accumulate), the result is divided by 8192 (arithmetically right shifted by 13 bits) to normalize the results (with radix point identical to inputs)
- If overflow occurs, the result is saturated (optional)

Note: It is possible to use the 24 bits more efficiently for higher precision with more complicated data manipulations
Static FP Arithmetic

- Effective compromise between FP and integer units
  - Fractional operations with automatic rounding as FPU
  - Shrunken (1-bit) aligners & normalizers from FPU
  - Static exponent tracking (radix point) as integer units
  - Almost FP quality (precision) with much simpler hardware (thus speed, area, & power) as the integer units

- SFP arithmetic utilizes the bits more efficiently for precision
  - No exponent (more free bits for precision)
  - Automatic rounding with fractional multiplications & more frequent normalization than integer units to reduce leading sign-bits
SFPU for Linear Transforms

- For $N$-bit data samples
  - (N+1)-bit adder/subtractor with input aligners & output normalizer (in our embodiment, all are 1-bit right shifters)
  - $N$-bit fractional multiplier with 1-bit output normalizer (left shifter)
  - $N$-bit barrel shifter with arithmetic right shifts
- FP arithmetic
  - Floating-point adder & multiplier only
- Integer unit
  - Integer adder, multiplier & barrel shifter
PEV Analysis

- Static simulation of FP arithmetic
- Each edge in the FP SDFG is associated with a peak estimation vector (PEV) \([M \ r]\)
  - \(M\) denotes the (worst-case) maximum magnitude
  - \(r\) denotes the radix point (similar to the “exponent” in FP)

PEV calculation rules
- Keep \(M\) should be in the range between 1 and 0.5 by carrying out “\(M\) is divided (or multiplied) by 2” and “\(r\) minus (or plus) 1” simultaneously
- \(r\) should be identical before addition or subtraction
- \([M_1 \ r_1] \times [M_2 \ r_2] = [M_1 \times M_2 \ r_1 + r_2]\)
Example

- Align the radix point \((r)\) before summation
- Keep the maximum magnitude \((M)\) in the range between 0.5 & 1 to prevent overflow & maximize precision
Over Estimation & Affine Arithmetic

Represent the intermediates in the affine form to record the respective contribution of each independent variable.

Ignoring the signal correlation may over-estimate the worst-case magnitude.

VLSI Signal Processing 2004, Prof. Chih-Wei Liu
Results

<table>
<thead>
<tr>
<th>2-D DCT Kernel</th>
<th>Cycle Count</th>
<th>PSNR (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single-precision FP</td>
<td>672</td>
<td>-</td>
</tr>
<tr>
<td>16-bit fixed-point (integer)</td>
<td>848</td>
<td>33.22</td>
</tr>
<tr>
<td>32-bit fixed-point (integer)</td>
<td>672</td>
<td>36.10</td>
</tr>
<tr>
<td>16-bit SFP</td>
<td>720*</td>
<td>38.12**</td>
</tr>
<tr>
<td>24-bit SFP</td>
<td>720*</td>
<td>62.14</td>
</tr>
</tbody>
</table>

* 1,120 cycles without embedded 1-bit aligners & normalizers
** 36.58dB without affine arithmetic

- The first three SDFG are derived from the C source codes from IJG
- The two SFP SDFG are derived from the single-precision FP SDFG (through PEV analysis)
Outline

- DSP Arithmetic
- *Stream Interface Unit & A Simple DSP Core*
- Register Organization for DSP
Review – Folding Transform

- Procedure
  - Operation scheduling
  - Delay calculation
    \[ D_F(U \rightarrow V) = N \cdot w(e) - P_U + v - u \]
  - Dataflow optimization

\[
\begin{array}{c}
\text{iteration 0} \\
W, w = 0 \\
U, u = 1 \\
w(e) = 1D
\end{array}
\]

\[
\begin{array}{c}
\text{iteration 1} \\
W, w = 0 \\
U, u = 1 \\
D_F = 3
\end{array}
\]

\[
\begin{array}{c}
\text{iteration 2} \\
W, w = 0 \\
U, u = 1 \\
D_F = 3
\end{array}
\]

\[
\begin{array}{c}
\text{time} \\
0, 1, 2, 3, 4, 5, 6, 7, 8, 9
\end{array}
\]

VLSI Signal Processing 2004, Prof. Chih-Wei Liu
Stream Interface Unit (SIU)

- Data generation by stream interface unit (SIU)
  - Routing (interconnection)
  - Buffering (storage)

  *i.e. functional units perform data manipulations only*

- Basic SIU architectures
Memory-based SIU

- Static queue
  - Using pointer instead of data movement

- Multi-port memory
  - Output-queue: \( N \) read; 1 write
  - Input-queue: 1 read; \( N \) write
DSP-lite Core

I/O Buffer

DCT coef. block # 1

DCT coef. block # 2

source image

Main Memory

DCT coef.

ping

pong

image block # 3

image block # 2

Embedded DMA Controller

Microinstruction Memory

AMBA AHB

2R/1W memory

1R/1W memory

1R/1W memory

SIU-based DSP Datapath

VLSI Signal Processing 2004, Prof. Chih-Wei Liu
Software Generation

- Floating-Point (FP) SDFG
- FP to SFP Converter
- Static FP (SFP) SDFG
- ILP-based Scheduler
- Scheduled SFP SDFG
- Micro-code Generator
- Micro-code

- SFPU Configuration
- Datapath Configuration
- Memory Configuration

VLSI Signal Processing 2004, Prof. Chih-Wei Liu
Scheduling

- ASAP, ALAP, and scheduling ranges

(a) (b) (c)
The Boolean variable $x_{ij}$ indicates if the vertex $i$ is scheduled at time $j$.

- **Resource constraints** (operations cannot exceed the resources)
  - $x_{0.0} + x_{1.0} \leq 1$; $x_{0.1} + x_{1.1} \leq 1$; $x_{0.2} + x_{1.2} \leq 1$ (for input)
  - $x_{2.1} + x_{3.1} \leq 1$; $x_{2.2} + x_{3.2} \leq 1$; $x_{2.3} + x_{3.3} \leq 1$ (for adder)
  - $x_{5.3} + x_{6.3} \leq 1$; $x_{5.4} + x_{6.4} \leq 1$; $x_{5.5} + x_{6.5} \leq 1$ (for output)

- **Allocation constraints** (each node executes only once)
  - $x_{0.0} + x_{0.1} + x_{0.2} = 1$
  - $x_{1.0} + x_{1.1} + x_{1.2} = 1$
  - $x_{2.1} + x_{2.2} + x_{2.3} + x_{2.4} = 1$
  - $x_{3.1} + x_{3.2} + x_{3.3} = 1$
  - $x_{4.2} + x_{4.3} + x_{4.4} = 1$
  - $x_{5.2} + x_{5.3} + x_{5.4} + x_{5.5} = 1$
  - $x_{6.3} + x_{6.4} + x_{6.5} = 1$

- **Dependency constraints** (for each edge)
  - $2x_{0.0} + 2x_{0.1} + 3x_{0.2} - 2x_{2.1} - 3x_{2.2} - 4x_{2.3} - 5x_{2.4} \leq -1$
  - $2x_{1.0} + 2x_{1.1} + 3x_{1.2} - 2x_{2.1} - 3x_{2.2} - 4x_{2.3} - 5x_{2.4} \leq -1$
  - $3x_{0.0} + 2x_{0.1} + 3x_{0.2} - 2x_{3.1} - 3x_{3.2} - 4x_{3.3} \leq -1$
  - $3x_{1.0} + 2x_{1.1} + 3x_{1.2} - 2x_{3.1} - 3x_{3.2} - 4x_{3.3} \leq -1$
  - $2x_{2.1} + 3x_{2.2} + 4x_{2.3} + 5x_{2.4} - 3x_{5.2} - 4x_{5.3} - 5x_{5.4} - 6x_{5.5} \leq -1$
  - $2x_{3.1} + 3x_{3.2} + 4x_{3.3} - 3x_{4.2} - 4x_{4.3} - 5x_{4.4} \leq -1$
  - $3x_{4.2} + 4x_{4.3} + 5x_{4.4} - 4x_{6.3} - 5x_{6.4} - 6x_{6.5} \leq -1$
Port Constraints

- To prevent memory conflict in the 1R/1W SIU memory in the DSP-lite core
- For adder memory, $I_0$ & $I_1$ can not be scheduled at the same time
  \[ x_{0.0} + x_{1.0} \leq 1, \quad x_{0.1} + x_{1.1} \leq 1 \]
  \[ x_{0.2} + x_{1.2} \leq 1 \]
- For output memory, $A_0$ & $M_0$ cannot be schedule at the same time
  \[ x_{2.2} + x_{4.2} \leq 1, \quad x_{2.3} + x_{4.3} \leq 1 \]
  \[ x_{2.4} + x_{4.4} \leq 1 \]
Performance Evaluation

- **DSP-lite**
  - I/O registered FU to allow full-cycle routing in SIU
  - Latency: 2 cycles for registered I/O

- **ADSP-218x**
  - Single-cycle multiplier, adder, and barrel shifter with distributed register files

<table>
<thead>
<tr>
<th></th>
<th>ADSP-218x (80MHz)</th>
<th>DSP-lite (100MHz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>3rd-order Lattice Filter</td>
<td>32</td>
<td>13</td>
</tr>
<tr>
<td>2nd-order Biquad Filter</td>
<td>13</td>
<td>14</td>
</tr>
<tr>
<td>16-point complex FFT</td>
<td>874</td>
<td>268</td>
</tr>
<tr>
<td>8-point 1-D DCT</td>
<td>154</td>
<td>47</td>
</tr>
<tr>
<td>8x8 2-D DCT</td>
<td>2,452</td>
<td>720</td>
</tr>
</tbody>
</table>
Silicon Implementation

- TSMC 0.35um 1P4M CMOS Technology
- Area: 2.8mm$^2$
- Freq: 100MHz
- Power: 122mW
Outline

- DSP Arithmetic
- Stream Interface Unit & A Simple DSP Core
- *Register Organization for DSP*
Centralized Register File (RF)

- \( N \) functional units (FU)
  - including load/store units (L/S) & arithmetic units (AU)
- \( P \) access ports (~3\( N \))
- \( n \) registers (\( \alpha N \))

Hardware complexity
- Area: \( n \cdot P^2 \)
- Delay: \( n^{1/2} \cdot P \)
- Power: \( n \cdot P^2 \)
Banking

- Mapping of logical to physical ports
  - $P$-port register file can be constructed with $p$-port banks (smaller SRAM blocks)
  - $p = P$ centralized RF
  - $p < P$ and $B$ banks; arbitrary port mapping requires a $P$-to-$Bp$ crossbar

- Examples
  - RF organized as even/odd banks to support multi-word data operands
  - $p = P$ and non-accessed banks are powered off to save energy
  - $p < P$ and port conflicts are resolved either in HW (stall) or SW (carefully operation scheduling & register allocation)
FU Clustering

- Explore the spatial locality of computations
- Each cluster has its RF with minimal inter-cluster communication

- Hierarchical RF
  - L/S units and AU have distinct RFs
  - The RF for L/S can be regarded as an additional memory hierarchy
- Distributed RF (DSP-lite)
Inter-Cluster Communication (1/2)

- Copy operations
  - STM & HP: Lx
  - BOPS: ManArray (dedicated instruction slot)
  - For hierarchical RF, copy operations can be viewed as special load/store instructions

- Extended accesses
  - TI C’6x (extended reads)
  - For hierarchical RF, extended accesses can be viewed as FU has limited operands from memory (similar to CISC)
Shared storage
- Sun MAJC (with replicates for separate reads)
- For hierarchical & clustered RF, the cache RF may work as the shared storage

Register permutation
- Special case of shared storage
- Each cluster access an exclusive bank
  - For hierarchical RF with permutation functions as a ping-pong buffer
Ring-structure RF (1/2)

- 8 RF (each with 8 elements) are implemented,
  - 4 local (private) for each functional unit
    - 32-bit address registers & GPR in control & LS units
    - 40-bit accumulators in ALU/MAC units
  - 4 shared are concatenated as a ring with *dynamic port mapping*
- Otherwise, a 16-port (8R/8W) centralized register file is required

VLSI Signal Processing 2004, Prof. Chih-Wei Liu
64-tap FIR Example

Syntax: #, ring offset, instr0, instr1, instr2, instr3 (memory is half-word addressed)

\begin{verbatim}
i0 0; MOV r0,COEF; MOV r0,COEF; MOV r0,0; MOV r0,0;
i1 0; MOV r1,X; MOV r1,X+1; NOP; NOP;
i2 0; MOV r2,Y; MOV r2,Y+2; NOP; NOP;
// assume half-word (16-bit) input & word (32-bit) output
i3 RPT 512,8; // 2 outputs per iteration & total 1024 outputs
i4 0; LW_D r8,r9,(r0)+2; LW_D r8,r9,(r0)+2; MOV r1,0; MOV r1,0;
i5 RPT 15,2; // loop kernel: 60 MAC_V, including 120 multiplication (2 output)
i6 2; LW_D r8,r9,(r0)+2; LW_D r8,r9,(r0)+2; MAC_V r0,r8,r9; MAC_V r0,r8,r9;
i7 0; LW_D r8,r9,(r0)+2; LW_D r8,r9,(r0)+2; MAC_V r0,r8,r9; MAC_V r0,r8,r9;
i8 2; LW_D r8,r9,(r0)+2; LW_D r8,r9,(r0)+2; MAC_V r0,r8,r9; MAC_V r0,r8,r9;
i9 0; MOV r0,COEF; MOV r0,COEF; MAC_V r0,r8,r9; MAC_V r0,r8,r9;
i10 0; ADDI r1,r1,-60; ADDI r1,r1,-60; ADD r8,r0,r1; ADD r8,r0,r1;
i11 2; SW (r2)+4,r8; SW (r2)+4,r8; MOV r0,0; MOV r0,0;
\end{verbatim}

Remarks:
- 35 instruction cycles for 2 output; i.e. 17.5 cycle/output or 3.66 taps/cycle
- SIMD MAC: MAC_V r0,r8,r9; \( r0 = r0 + r8.Hi \times r9.Hi \) & \( r1 = r1 + r8.Lo \times r9.Lo \)

VLSI Signal Processing 2004, Prof. Chih-Wei Liu
Ring-structure RF (2/2)

Each cluster has a ping-pong hierarchical RF
Performance Evaluation

Implementation Results

<table>
<thead>
<tr>
<th></th>
<th>Centralized RF</th>
<th>Ring-Structure RF</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delay</td>
<td>38.46 ns</td>
<td>8.71 ns</td>
</tr>
<tr>
<td>Gate Count</td>
<td>591K</td>
<td>48K</td>
</tr>
<tr>
<td>Area</td>
<td>17.76mm×17.76mm</td>
<td>1.9mm×1.9mm</td>
</tr>
<tr>
<td>(2.34mm×1.53mm)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Power</td>
<td>N.A.</td>
<td>356mW @3.3V 100MHz</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>TI C’55x</th>
<th>TI C’64x</th>
<th>NEC SPXK5</th>
<th>Intel/ADI MSA</th>
<th>Proposed</th>
</tr>
</thead>
<tbody>
<tr>
<td>FIR</td>
<td>NT/2</td>
<td>NT/4</td>
<td>NT/2</td>
<td>NT/2</td>
<td>NT/4</td>
</tr>
<tr>
<td>FFT</td>
<td>4,768</td>
<td>2,403</td>
<td>2,944</td>
<td>3,176</td>
<td>2,340</td>
</tr>
<tr>
<td>Viterbi</td>
<td>1 (0.4)</td>
<td>N.A.</td>
<td>1 (1)</td>
<td>1 (N.A.)</td>
<td>1 (0.84)</td>
</tr>
<tr>
<td>ME</td>
<td>N.A.</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>2</td>
</tr>
</tbody>
</table>
Summary

Static FP arithmetic
- Software (i.e. static) tracks the exponents of the intermediate variables to maximize the precision while preventing overflow
- $62.14 \text{dB}$ & $38.12 \text{dB}$ for 24-bit & 16-bit SFPU respectively

Simple DSP core – DSP-lite
- Software techniques are extensively investigated to reduce the hardware complexity
- SIU-based computing engine has about $3X$ performance over conventional DSP with similar functional units

Ring-structure RF for $N$ functional units
- Partitions the $4N$-port CRF into $2N$ sub-blocks, and each has $2R2W$ ports only
- Has comparable performance with state-of-the-art DSP processors (estimated in cycles) when $N=4$, and saves $91.88\%$ area & $77.35\%$ access time