Mapping DSP Algorithms Into FPGAs

Sean Gallagher, Senior DSP Specialist, Xilinx Inc.
seang@xilinx.com
215-990-4616
Agenda

- History of Algorithm implementations in FPGAs
- Why FPGAs for Signal Processing
- Overview of Xilinx FPGA
- Interesting Algorithms for FPGA implementation
  - Critically sampled channelizer
  - Divide and Conquer DFT
  - Winograd FFT
- The Xilinx DSP tool flow
Systolic Array processing techniques were established in the ‘70s
- S.Y. Kung, others

FPGA technology invented by Xilinx in 1984
- Glue logic integration
- Super Computing Research Center (SRC) built Splash I and II coprocessing boards in early ’90s
- Board of 32 Xilinx FPGAs slaved to a Sun workstation
- Computation speeds of 6-7 times greater than a Cray II computer
My History With FPGAs

_visited SRC in early ‘90s to sell synthesis tools_
  – Had no clue what they were talking about

Pursued MSCE at Villanova focused on algorithms in FPGAs
  – They had no idea what I was talking about
  – Master’s thesis in ‘95, Implementing Algorithms in FPGAs

Came to Xilinx in 2001 as DSP Specialist
  – Still learning
Emerging Applications Drive Demand for Next Generation FPGAs

- **Lowest Power and Cost**
  - Handheld portable ultrasound
  - Digital SLR lens control module
  - Software defined radio

- **Industry’s Best Price-Performance**
  - Wireless LTE infrastructure
  - 10G PON OLT line card
  - LED backlit and 3D video displays
  - Medical imaging
  - Avionics imaging

- **Industry’s Highest System Performance and Capacity**
  - 100GE line card
  - 300G bridge
  - Terabit switch fabric
  - 100G OTN
  - MUXPONDER
  - RADAR
  - ASIC emulation
  - Test & Measurement

- **Next Gen Wireless Communications**
- **Automotive Infotainment**
- **Next Gen Wired Communications**
- **Consumer**
- **Aerospace & Defense**
- **Test & Measurement**
- **Medical Imaging**
- **Audio Video Broadcast**

Emerging Applications Drive Demand for Next Generation FPGAs
Why FPGA for Signal Processing?

- How much computational power do you need?

256-tap Filter Example

Conventional DSP Processor – Serial implementation

Data In

Coefficients

MAC Unit

256 loops needed to process samples

Reg

Data Out

\[
\frac{1 \text{ GHz}}{256 \text{ clock cycles}} = 4 \text{ MSPS}
\]

Virtex-4 Parallel Implementation Consumes Zero Logic Resources

Data in

\[ \begin{align*}
C_0 & \times \times \times \\
C_1 & \times \times \times \\
C_2 & \times \times \times \\
C_3 & \times \times \times \\
C_4 & \times \times \times \\
C_5 & \times \times \times \\
C_6 & \times \times \times \\
C_7 & \times \times \times \\
C_{254} & \times \times \times \\
C_{255} & \times \times \times \\
\end{align*}\]

Data out

\[ \begin{align*}
\text{Filters Implemented Entirely Within the DSP48 Slice}
\end{align*}\]

\[ \text{500 MHz Performance Regardless of Filter Size} \]

\[
\frac{500 \text{ MHz}}{1 \text{ clock cycle}} = 500 \text{ MSPS}
\]
- How many MACs (multiply accumulator) do you need?
- For Example, in FIR Filter,

\[
\text{Number of MACs required} = \frac{\text{OutputDataRate} \times \text{NumberOfTaps} \times \text{NumberOfChannels}}{\text{InputDataRate} \times \text{ClockRate}}
\]

FPGAs can meet various throughput requirement
Parallelism enables efficient implementation of multi-channel into a single FPGA

Many low sample rate channels can be multiplexed (e.g. TDM) and processed in the FPGA, at a higher rate

Many of Xilinx IPs takes advantage of multi-channel implementation - FIRCompiler, FFT
FPGA + DSP Processor

- FPGA enables DSP processor acceleration – mapping speed critical loop of DSP code to FPGA
- FPGAs enables consolidation of glue logic, memory, interfaces, ASSP
- For detail on interface (EMIF, VLYNQ, LinkPort), see http://www.xilinx.com/esp/wireless.htm
6 Series Xilinx FPGAs

*Now Shipping*

- **Virtex-6** - Industry leading DSP performance
- **Spartan-6** Industry leading DSP cost / performance

<table>
<thead>
<tr>
<th></th>
<th>Spartan-6</th>
<th>Virtex-6</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Industries Best Price/Performance</strong></td>
<td>3.8K – 147K</td>
<td>74K – 567K</td>
</tr>
<tr>
<td><strong>Industries Highest System Performance</strong></td>
<td>8-180</td>
<td>288-2016</td>
</tr>
<tr>
<td><strong>Max Transceivers</strong></td>
<td>8</td>
<td>72</td>
</tr>
<tr>
<td><strong>Transceiver Performance</strong></td>
<td>3.125 Gbps</td>
<td>6.6 Gbps 11.18 Gbps</td>
</tr>
<tr>
<td><strong>Memory</strong></td>
<td>4,824 Kbits</td>
<td>38,309 Kbits</td>
</tr>
<tr>
<td><strong>Max. SelectIO</strong></td>
<td>576</td>
<td>1200</td>
</tr>
<tr>
<td><strong>SelectIO Voltages</strong></td>
<td>1.2v to 3.3v</td>
<td>1v to 2.5v</td>
</tr>
</tbody>
</table>

- Now Shipping
Introducing the 7 Series FPGAs

- **Industry’s Lowest Power and First Unified Architecture**
  - Spanning Low-Cost to Ultra High-End applications
- **Three new device families with breakthrough innovations in power efficiency, performance-capacity and price-performance**

<table>
<thead>
<tr>
<th></th>
<th>ARTIX.7</th>
<th>KINTEX.7</th>
<th>VIRTEX.7</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Lowest Power &amp; Cost</strong></td>
<td>20K – 355K</td>
<td>30K – 410K</td>
<td>285K – 2,000K</td>
</tr>
<tr>
<td><strong>DSP Slices</strong></td>
<td>40 – 700</td>
<td>120 – 1540</td>
<td>700 – 3,960</td>
</tr>
<tr>
<td><strong>Max. Transceivers</strong></td>
<td>4</td>
<td>16</td>
<td>80</td>
</tr>
<tr>
<td><strong>Transceiver Performance</strong></td>
<td>3.75Gbps</td>
<td>6.6Gbps</td>
<td>10.3Gbps</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10.3Gbps</td>
<td>13.1Gbps</td>
</tr>
<tr>
<td><strong>Memory Performance</strong></td>
<td>800Mbps</td>
<td>2133Mbps</td>
<td>2133Mbps</td>
</tr>
<tr>
<td><strong>Max. SelectIO™</strong></td>
<td>450</td>
<td>500</td>
<td>1200</td>
</tr>
<tr>
<td><strong>SelectIO™ Voltages</strong></td>
<td>3.3V and below</td>
<td>3.3V and below</td>
<td>3.3V and below</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.8V and below</td>
<td>1.8V and below</td>
</tr>
</tbody>
</table>
Bridging the DSP Performance Gap with 7-Series

- **DSP Performance**
  - 4752 GMAC
  - 770 GMAC
  - 2000 GMAC
  - 90 GMAC
  - 33 GMAC

- **6-Series**
  - Virtex-6
  - Spartan-6

- **7-Series**
  - Virtex-7
  - Kintex-7

- *Peak performance for symmetric filters*

- Time

- Algorithm Complexity

- Multi-core DSP Architectures

* XILINX®
FPGA Resource

Challenge: How do we make the best use of these resources in most efficient manner?
DSP Performance through the DSP48E1 Slice
Virtex-6, Artex-7, Kintex-7, Virtex-7

- 2 DSP48E1 Slices / Tile
- Column Structure to avoid routing delay
- Pre-adder, 25x18 bit multiplier, accumulator
- Pattern detect, logic operation, convergent/symmetric rounding
- 638 MHz Fmax
Hardened Pre-Adder leverages filter symmetry to reduce Logic, Power and Routing

No restriction to coefficient table size

Filter symmetry exploited to pre-add tap delay values and reduce multiplies by 50%
Greater Flexibility with Fully Independent Multipliers

- Full, independent access to every multiplier
- One accumulator for each multiplier
- 5 Interconnects support up to 50 bit multiplies per tile
25x18 Multiplier

- Single DSP slice supports up to 25x18 multiplies
  - 50% fewer DSP resources required for high-precision multiplies
  - Efficient FFT Implementations
  - Efficient single-precision floating-point implementations
- Single DSP Tile supports up to 50x36 multiplies
- Delivers higher performance and lower power
Only FPGA architecture that supports pattern detection
  - Pattern can be constant (set by attribute) or C input

Efficient implementation of rounding modes
  - Symmetric
  - Convergent
  - Saturation
DSP48E1 slice provides an accumulator for each multiplier
- 2X more than competitive architectures

Up to 48-bits accumulation per DSP slice
- 25x18 multiply

Up to 96-bits accumulation per DSP tile
- 50x36 multiply
DSP IP Portfolio

- **Comprehensive IP portfolio**
- **Constraint Driven**
- **IP can be imported into RTL, System Generator and Platform Studio**

<table>
<thead>
<tr>
<th>Category</th>
<th>IP Blocks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Math</td>
<td>mult, adder, accumulator, divider, trig, CORDIC</td>
</tr>
<tr>
<td>Filters</td>
<td>FIR, CIC</td>
</tr>
<tr>
<td>Memory</td>
<td>RAM, register, FIFO, shift register</td>
</tr>
<tr>
<td>Transforms</td>
<td>FFT, IFFT, LTE FFT</td>
</tr>
<tr>
<td>Processors</td>
<td>MicroBlaze</td>
</tr>
<tr>
<td>Video</td>
<td>Color correction, CFA, pixel correction, image characterization, edge enhancement, noise reduction, statistics, CSC, VFBC, Scaler, timing controller,</td>
</tr>
<tr>
<td>Wireless</td>
<td>DDS, DUC/DDC, MIMO Decoder/encoder, RACH preamble det, DPD, CFR,</td>
</tr>
<tr>
<td>Floating-Point</td>
<td>Add/sub, mult, div, sqrt, compare, convert, FFT</td>
</tr>
</tbody>
</table>
Constraint Driven IP

Interpolate by 2

30.22 MHz  →  61.44 MHz

FIR Compiler 6.0

11 Tap FIR Filter

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Result 1</th>
<th>Result 2</th>
<th>Result 3</th>
<th>Result 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Channels</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>Clock Frequency</td>
<td>122.7</td>
<td>245.4</td>
<td>245.4</td>
<td>368.1</td>
</tr>
<tr>
<td>DSP Slice Count</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>1</td>
</tr>
</tbody>
</table>

- Overclocking automatically used to reduce DSP slice count
- Quick estimates provided by IP compiler GUI
- Insures best results for your design requirements
Interesting Algorithms For FPGA Implementation

- **Critically sampled channelizers**
  - Polyphase with a DFT bank

- **Divide and conquer DFT**
  - Calculating a 1D FFT as a 2DFFT

- **Winograd FFT Transform**
  - Least amount of multiplies
In a FDM digital communication system a common requirement is, for each channel:

- translate the channel to baseband
- shape the channel spectrum
- reduce the sample rate to match the channel bandwidth

This is the function of a channelizer

When the channel spacing’s are equal a computationally efficient structure for performing the above functions is the carrier centered polyphase transform
Baseband Polyphase Filter

\[ h_0(n) = h_0 \quad h_M \quad L \quad h_{N-M} \]

\[ h_1(n) = h_1 \quad h_{M+1} \quad L \quad h_{N-M+1} \]

\[ h_{M-1}(n) = h_{M-1} \quad h_{2M-1} \quad L \quad h_{N-1} \]
Express the filter coefficient set in terms of a course and vernier index $r_1$ and $r_2$ respectively

\[ h(n) = h(r_1 + M r_2) \quad r_1 = 0, K, M - 1, \quad r_2 = 0, K, \frac{N}{M} - 1 \]

- Invoke the modulation theorem to convert a prototype baseband filter to its equivalent carrier centered, or spectrally shifted version

\[
\begin{align*}
\text{if} & \quad h(n) \Leftrightarrow H(\theta) \\
\text{then} & \quad h(n) e^{j\theta_0 n} \Leftrightarrow H(\theta - \theta_0)
\end{align*}
\]
The coefficients of the carrier centered filter are
\[ g(n) = h(n)e^{j\theta_0 n} \]

Now perform a polyphase partition on the modulated coefficients
\[ g_{r_1}(r_2) = h(r_1 + Mr_2)e^{j\theta_0 (r_1 + Mr_2)} \]
\[ = h(r_1 + Mr_2)e^{j\theta_0 r_1}e^{j\theta_0 Mr_2} \]

Select \( \theta_0 \) so that a single period of the series \( e^{j\theta_0 n} \) is harmonically related to \( M \)
Passband Polyphase Filters

\[ \theta_0 = k \frac{2\pi}{M} \]

\[ g_{r_1}(r_2) = h(r_1 + Mr_2) e^{j\theta_0 r_1} e^{jk \frac{2\pi}{M} r_2} = h(r_1 + Mr_2) e^{jk \frac{2\pi}{M} r_1} \]

- Carrier centered polyphase filter
- the one structure
  - baseband’s the channel
  - shapes the signal
  - reduces the sample rate
• Recovering 2 channels from FDM spectra
• The two sets of filters employ identical coefficients
• Note: the two sets of filters contain the same data
Passband Polyphase Filters

• Only one filter is required because the data is the same in both filters on the previous slide
• Baseband and carrier centered polyphase filter, heterodyne and downsample
Recall that the IDFT of an $M$-point sequence $Y(k)$ is

$$y(n) = \sum_{k=0}^{M-1} Y(k)e^{j2\pi nk/M} \quad n = 0, 1, K, M - 1$$

If the $M$ phase rotators are sequenced over all of the $M$ values of $k$ we recognize that this is the same as computing an IDFT.
• Passband Polyphase Filters

• Carrier centered polyphase filters can also be used for constructing frequency division multiplexed signals
• Baseband and carrier centered polyphase filter, heterodyne and upsample
• Passband Polyphase Filters

Baseband and carrier centered polyphase common filter, heterodyne and upsample
It is possible to compute a one dimensional DFT as a two dimensional DFT

- Ideal for processing high rate data that has been demuxed to multiple paths at a lower rate

Decompose DFT into two dimensions:

\[
X(p, q) = \sum_{m=0}^{M-1} \sum_{l=0}^{L-1} x(l, m)W_N^{(Mp+q)(mL+l)}
\]

But:

\[
W_N^{(Mp+q)(mL+l)} = W_N^{MLmp}W_N^{MLq}W_N^{Mpl}W_N^{lq}
\]

However:

\[
W_N^{Nmp} = 1, W_N^{mqL} = W_N^{mq}/L = W_M^{mq} \quad \text{and} \quad W_N^{mpl} = W_M^{pl}/M = W_L^{pl}
\]

\[
X(p, q) = \sum_{l=0}^{L-1} \left\{ \sum_{m=0}^{M-1} x(l, m)W_M^{mq} \right\}W_L^{lp}
\]
These simplifications lead to:

\[ X(p, q) = \sum_{l=0}^{L-1} \left\{ W_N^{lq} \left[ \sum_{m=0}^{M-1} x(l, m) W_M^{mq} \right] \right\} W_L^{lp} \]

Process Steps:
1. Store signal column-wise
2. Compute the M point DFT for each row
3. Multiply the resulting array by the phase factors \( W_N^{lq} \)
4. Compute the L-point DFT of each column
5. Read the resulting array row wise
Winograd FFT

Developed by mathematician Schmuel Winograd in 1976
• Goal was to reduce the number of multiplies required
• Multiplies minimized but at expense of increased complexity
• Memory mappings became very complex too
• Due to complexity, cost of doing an fft did not significantly go down
• Problem with algorithm is that multiplies and accumulates were separated
  so execution on DSP processor was not efficient