Final Project: Implementation of Energy-Efficient FIR Filters
OUTLINE

- Implementation of FIR filters
- Quartus II & Modelsim
FIR FILTERS

- A general equation of $N$-tap linear phase FIR filters:

$$y[n] = \sum_{k=0}^{N-1} x[n-k] \cdot h[k]$$

$$= \begin{cases} 
\sum_{k=0}^{\frac{N}{2}-1} (x[n-k] + x[n-(N-1-k)]) \cdot h[k], & \text{if } N \text{ is even} \\
\sum_{k=0}^{\frac{N-1}{2}-1} (x[n-k] + x[n-(N-1-k)]) \cdot h[k] + x[n-(\frac{N-1}{2})] \cdot h[\frac{N-1}{2}], & \text{if } N \text{ is odd}
\end{cases}$$
FOLDING OF AN FIR FILTER

- Folding – area reduction / reconfigurable

![Diagram of folding of an FIR filter with symbols and coefficients]

Data Memory

Coefficient Memory
**Pre-added Multiply-Accumulate**

- **Full precision**
  - Accurate but energy-hungry

- **Fixed-width multiplier**
  - Post truncate
  - Direct truncate
  - Low error fixed-width multiplier [2][3]
    - Compromise between accuracy and overhead

- **Static floating point [1]**
  - System level analysis

---


**STATIC FLOATING POINT**

- **Peak estimation vector** \([M \ r]\)
  - \(M\) – maximum magnitude
  - \(r\) – position of radix point

- **Design time analysis**
  - Normalize \(M\) within 0.5 ~ 1
  - Align \(r\) before addition and subtraction
  - \([M_1 \ r_1] \cdot [M_2 \ r_2] = [M_1 \cdot M_2 \ r_1 + r_2]\)

- **Example**
**EXAMPLE: 27-TAP LINEAR PHASE FIR**

- Reduce truncation error
  - Insert shift by SFP analysis
- Power gap between ASIC and processor
  - Register access
  - Instruction access
  - Data load / store

![Example Diagram]
**SFP-based FIR Filter Engine**

- SFP analysis for FIR filter
  - 1-bit shifter after pre-adder
  - 1 shifter to align input of post-adder
  - 1-bit shifter after post-adder

- Cascaded datapath
  - Reduce computing power
  - Reduce register access power
  - Simplify DSP instruction to control signals
  - Decrease execution cycle

---

**Diagram:**
- S1
- S2
- Coef
- Acc
- Direct truncated multiplier
- MUX
- Acc
RESULTS (DESIGN COMPILER)

- **SNR:**
  \[SNR = 10 \cdot [\log(\sum (gold\_out)^2) - \log(\sum (out - gold\_out)^2)]\]

- **Area:** Area @UMC 90nm process
- **Delay:** Worst case delay
- **Power:** Simulation based power analysis

<table>
<thead>
<tr>
<th>Implement</th>
<th>SNR (dB)</th>
<th>Area (μm²)</th>
<th>Delay (ns)</th>
<th>Power (μW)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full Precision</td>
<td>72.59</td>
<td>81,504</td>
<td>20.08</td>
<td>135</td>
</tr>
<tr>
<td>Post Truncate</td>
<td>60.47</td>
<td>76,272</td>
<td>20.46</td>
<td>113</td>
</tr>
<tr>
<td>Direct Truncate</td>
<td>44.54</td>
<td>46,584</td>
<td>18.00</td>
<td>72</td>
</tr>
<tr>
<td>SFP</td>
<td>58.41</td>
<td>60,614</td>
<td>27.25</td>
<td>90</td>
</tr>
</tbody>
</table>
## Results (Quartus II)

- **Area**: number of logic elements (LE)
- **Power**: Vector-free power analysis

<table>
<thead>
<tr>
<th></th>
<th>Implement</th>
<th>SNR (dB)</th>
<th>Area (# LE)</th>
<th>Delay (ns)</th>
<th>Power (mW)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Optimize delay</strong></td>
<td>Full Precision</td>
<td>72.59</td>
<td>684</td>
<td>12</td>
<td>6.29</td>
</tr>
<tr>
<td></td>
<td>Post Truncate</td>
<td>60.47</td>
<td>649</td>
<td>12</td>
<td>5.44</td>
</tr>
<tr>
<td></td>
<td>Direct Truncate</td>
<td>44.54</td>
<td>426</td>
<td>12</td>
<td>4.43</td>
</tr>
<tr>
<td></td>
<td>SFP</td>
<td>58.41</td>
<td>547</td>
<td>16</td>
<td>4.87</td>
</tr>
<tr>
<td><strong>Optimize area</strong></td>
<td>Full Precision</td>
<td>72.59</td>
<td>684</td>
<td>17</td>
<td>0.75</td>
</tr>
<tr>
<td></td>
<td>Post Truncate</td>
<td>60.47</td>
<td>649</td>
<td>16</td>
<td>0.65</td>
</tr>
<tr>
<td></td>
<td>Direct Truncate</td>
<td>44.54</td>
<td>426</td>
<td>16</td>
<td>0.49</td>
</tr>
<tr>
<td></td>
<td>SFP</td>
<td>58.41</td>
<td>520</td>
<td>22</td>
<td>0.59</td>
</tr>
</tbody>
</table>
To Do...

- Realize a linear phase FIR engine with SFP
  - SQNR approximates post-truncated design
  - Reduce critical path delay
  - Power efficient
  - Area efficient

- Grade
  - Architecture & implementation (60%)
  - Report (40%)
OUTLINE

- Implementation of FIR filters
- Quartus II & Modelsim
DOWNLOAD & INSTALLATION

- Download web edition (free) from Altera website
START A NEW PROJECT IN QUARTUS II

- File -> New project wizard
  - Select working directory
  - Define project name
QUARTUS II SETTING

- **Project -> Add/remove files in project**
  - Add design files / test bench / design constraint file to the project

- **Set top module**
- **Set test bench**
  - Properties: type
    - Verilog test bench type
DESIGN CONSTRAINTS

- Set clock name / clock period
- Set input / output delay

```bash
create_clock -name clk -period 16.000 -waveform {0 8} [get_ports clk]
set_output_delay 1 -clock clk [all_outputs]
set_input_delay 1 -clock clk [all_inputs]
```
Optimization Effort

Physical Synthesis Optimizations

Specify options for performing physical synthesis optimizations during synthesis and fitting. These options can affect VQM or EDIF netlists even if WYSIWYG primitive resynthesis is disabled.

Optimize for performance (physical synthesis)
- Perform physical synthesis for combinational logic
- Perform register retiming

Effort level
- Fast (optimize during synthesis; minor compilation time increase)
- Normal (optimize during synthesis and fitting; moderate compilation time increase)
- Extra (optimize during synthesis and fitting; major compilation time increase)

Fitter netlist optimizations
- Perform automatic asynchronous signal pipelining
- Perform register duplication

Optimize for fitting (physical synthesis for density)
- Perform physical synthesis for combinational logic
- Perform logic to memory mapping
OPTIMIZATION TECHNIQUE
Compile Design (1/2)

- Processing -> Start compilation
  - Start synthesis

- Synthesis result
Compile Design (2/2)

- Critical path delay

![Multicorner Timing Analysis Summary](image)
POWER CONSUMPTION ESTIMATION

- Power consumption (vectorless)
  - PowerPlay Power Analyzer

Result

![Image of PowerPlay Power Analyzer interface with results]

- Core Dynamic Thermal Power Dissipation: 3.74 mW
EDA Tool Setting

- Tools->Options->EDA Tool Options
  - Choose the directory of ModelSim-Altera
**MODELSIM-ALTERA**

- Assignments->Settings->EDA Tool Setting
  - Select simulation tool: ModelSim-Altera
**Setup Test Bench (1/2)**

- Assignments -> Settings -> EDA Tool Setting -> Simulation -> Test Benches
Setup Test Bench (2/2)

- Define test bench name
- Select test bench file
Simulation using ModelSim-Altera

- Simulation type
  - RTL simulation
  - Gate level simulation

- Run ModelSim automatically
  - All the files used in test bench should be placed in the directory "./simulation/modelsim/"
  - Check waveform and/or messages in ‘transcript’ window