Optimization of Arithmetic Datapaths with Finite Word-Length Operands

Sivaram Gopalakrishnan\textsuperscript{1}, Priyank Kalla\textsuperscript{1} and Florian Enescu\textsuperscript{2}

\textsuperscript{1}Department of Electrical and Computer Engineering, University of Utah, Salt Lake City, UT-84112

\textsuperscript{2}Department of Mathematics and Statistics, Georgia State University, Atlanta, GA-30303
Outline

- Introduction to the datapath optimization problem
  - Our Focus: Arithmetic with Finite Word-Length Operands
- Problem Modeling
  - Polynomial Functions over Finite Integer Rings
- Previous Work and Limitations
- Approach and Contributions
  - Reducibility of Polynomials over Finite Rings
  - Cost Model
  - Integrated CAD Approach
- Results: Area optimization
- Conclusions & Future Work
The Optimization Problem
Polynomials over Bit-Vectors?

- Quadratic filter design for polynomial signal processing
- \[ y = a_0 \cdot x_1^2 + a_1 \cdot x_1 + b_0 \cdot x_0^2 + b_1 \cdot x_0 + c \cdot x_0 \cdot x_1 \]
- Coefficients/variables implemented with specific bit-vector sizes
Fixed-Size (m) Data-path: Modeling

- Control the datapath size: Fixed size bit-vectors (m)

- Bit-vector of size m: integer values in 0, ..., 2^m - 1

```
Fixed-size (m) bit-vector arithmetic
-> Polynomials reduced \( \% 2^m \)
-> Algebra over the ring \( \mathbb{Z}_{2^m} \)
```
Multiple Bit-Width Operands

- Bit-vector operands with different word-lengths

- Input variables: \( \{x_1, \ldots, x_d\} \)  
  Output variables: \( f, g \)

- Input bit-widths: \( \{n_1, \ldots, n_d\} \)  
  Output width: \( m \)

\[
x_1 \in \mathbb{Z}_{2^{n_1}}, x_2 \in \mathbb{Z}_{2^{n_2}} \ldots f, g \in \mathbb{Z}_{2^{m}}
\]

- Model as polynomial function

\[
\mathbb{Z}_{2^{n_1}} \times \mathbb{Z}_{2^{n_2}} \times \ldots \times \mathbb{Z}_{2^{n_d}} \rightarrow \mathbb{Z}_{2^{m}}
\]
**Arithmetic Datapath: Implementation**

- **Signal Truncation: Unsigned/Overflow Arithmetic**
  - Keep lower order m-bits, ignore higher bits
  - \( f \% 2^m \)

- **Fractional Arithmetic with rounding**
  - Keep higher order m-bits, round lower order bits

- **Saturation Arithmetic**
  - Saturate at overflow
  - If( \( x[7:0] > 255 \) ) then \( x[7:0] = 255; \)
  - Used in image-processing applications
Conventional Methods

- Extracting control-dataflow graphs (CDFGs) from RTL
  - Scheduling
  - Resource sharing
  - Retiming
  - Control synthesis

- Algebraic Transforms
  - Factorization
  - Common Sub-expression Elimination
  - Term-rewriting
  - Tree-Height Reduction

- Overlook the effect of bit-vector size (m)
Previous Work

- Polynomial models for complex computational blocks
- Guiding Synthesis engines using Groebner’s basis
  [Peymandoust and De Micheli, TCAD 02]
  - Given polynomial $F$ and Library elements $<I_1, \ldots, I_n>$
  - $F = h_1 I_1 + \ldots + h_n I_n$
- Computations over $\mathbb{R}, \mathbb{Q}, \mathbb{Z}, \mathbb{Z}_p$ (Galois Fields)
  - Unique Factorization Domains (UFDs): *Uniquely* factorize into irreducibles
  - Polynomial approximation (do not account the effect of bit-vector size)
- Datapath allocation for multiple-wordlength operands
  [Constantinides et al, TVLSI 05]
  - Operates on the given expression
Why is the Problem Difficult?

- $\mathbb{Z}_2^m$ is a non-UFD
  - $f = x^2 + 6x$ in $\mathbb{Z}_8$ can be factorized as
    \[
    f = (x)(x+6) = (x+2)(x+4)
    \]

- Factorization in non-UFDs is therefore hard !!!
- Scope to explore multiple factorizations
Example: Polynomial Filter

- A Polynomial filter \(f\) over a uniform 16-bit datapath
  \[
  f_1 = 16384x^5 + 19666x^4 + 38886x^3 + 16667x^2 + 52202x + 1
  \]
- Area: 42910 sq. units

- Alternatively, \(f\) can be implemented as
  \[
  f_2 = 3282x^4 + 22502x^3 + 283x^2 + 52202x + 1
  \]
- Area: 28840 sq. units

\[
f_1 \neq f_2, \text{ but } f_1 \% 2^{16} \equiv f_2 \% 2^{16}, f_1[15:0] = f_2[15:0]
\]
Digital Image Rejection Unit

input \( A[11:0], B[7:0] \);
output \( Y_1[15:0], Y_2[15:0] \);

\[
Y_1 = 16384(A^4 + B^4) + 64767(A^2 - B^2) + A - B + 57344AB(A - B)
\]

\[
Y_2 = 24576A^2B + 15615A^2 + 8192AB^2 + 32768AB + A + 17153B^2 + 65535B
\]

- \( Y_1 \neq Y_2 \)
- \( Y_1[15:0] = Y_2[15:0] \)
- \( Y_1 \% 2^{16} \equiv Y_2 \% 2^{16} \)
Problem Modeling

- Polynomial Model:
  - \( Y_1(A_{12}, B_8)^{\%2^{16}} \equiv Y_2(A_{12}, B_8)^{\%2^{16}} \)
  - \( Y_1, Y_2: \mathbb{Z}_{2^{12}} \times \mathbb{Z}_{2^8} \rightarrow \mathbb{Z}_{2^{16}} \) are equal as functions

- Consider \( Y_1 - Y_2 \)
  - \( Y_1 - Y_2 \equiv 0 \% 2^{16} \)
    \[
    Y_1 - Y_2 = 16384(A^4 + B^4) + 32768AB(A + 1) + 49152(A^2 + B^2) \equiv 0 \% 2^{16}
    \]

- \( Y_1 - Y_2 \) vanishes as a function from \( \mathbb{Z}_{2^{12}} \times \mathbb{Z}_{2^8} \rightarrow \mathbb{Z}_{2^{16}} \)

- \( Y_1 - Y_2 \) is known as the *vanishing polynomial*
Vanishing Polynomials for Reducibility

- In \( Z_2^3 \), say \( f(x) = 4x^2 \) and \( V(x) \) is a vanishing polynomial
  - \( f(x) = f(x) - V(x) \)
  - Generate \( V(x) \)
  - \( V(x) = 4x^2 + 4x \equiv 0 \pmod{2^3} \)

- Reduce by subtraction:
  - \[
  \begin{align*}
  &4x^2 \quad f(x) \\
  - &4x^2 + 4x \quad V(x) \\
  = &\underline{4x} - 4x = -4x \pmod{8} = 4x
  \end{align*}
  \]
  - \( 4x^2 \) can be reduced to \( 4x \)
  - \textit{Degree reduction}
Vanishing Polynomials for Reducibility

- Degree is not always reducible

- In $\mathbb{Z}_2^3$, $f(x) = 6x^2$

- Divide and subtract
  - $6x^2 = 2x^2 + 4x^2 \mod 2^3$
  - $4x^2$ can be reduced to $4x$

- $f(x) = 2x^2 + 4x$; *Lower Coefficient*
  - *Coefficient reduction*
Results From Number Theory

- $n!$ divides a product of $n$ consecutive numbers
  - $4!$ divides $99 \times 100 \times 101 \times 102$

- Find least $n$ such that $2^m | n$!
  - *Smarandache Function (SF)*
  - $SF(2^3) = 4$, since $2^3 | 4!$

- $2^m$ divides the product of $n = SF(2^m)$ consecutive numbers
  - $2^3$ divides the product of 4 consecutive numbers
Results From Number Theory

- $F \equiv 0 \mod 2^3$
  - $2^3 \mid F$ in $\mathbb{Z}_2^3$
  - $2^3$ divides the product of 4 consecutive numbers

If $F$ is a product of 4 consecutive numbers then $2^3 \mid F$

- A polynomial as a product of 4 consecutive numbers?

$(x)(x-1)(x-2)(x-3)$
Basis for Factorization: One Variable

- $Y_0(x) = 1$
- $Y_1(x) = (x)$
- $Y_2(x) = (x)(x - 1) = \text{Product of 2 consecutive numbers}$
- $Y_3(x) = (x)(x - 1)(x - 2) = \text{Product of 3 consecutive numbers}$
- $\ldots$
- $\ldots$
- $Y_k(x) = (x - k + 1) Y_{k-1}(x) = \text{Product of } k \text{ consecutive numbers}$

Rule 1: Degree is $k$. If $k \geq n$

where $n = SF(2^m)$, use $Y_k(x)$ (degree reduction)

Straight forward extension to multiple variables with finite word-lengths
Constraints on the Coefficient

- \( F(x) = 4x^2 - 4x = 4(x)(x-1) \mod 2^3 = 0 \mod 2^3 \)
  compensated by constant

- In \( \mathbb{Z}_2^3 \)
  - \( Y_4(x) = (x)(x-1)(x-2)(x-3) \)
    missing factor

Rule 2: if Coefficient \( \geq b_k \) where \( b_k = 2^m/\gcd(k!, 2^m) \), then use
  \[ a_k \cdot b_k \cdot Y_k \] (for coefficient reduction)

- Here, Coefficient of \( F(x) = 4 \), Degree of \( F(x) = 2 \)
- \( b_{<2>} = 2^3/\gcd(2!, 2^3) = 4 \) (coefficient’s value!!!)
Example

- Consider $x^4$ in $\mathbb{Z}_8$

\[ x^4 \]

- $k = 4$, $SF(8) = 4$, So $V(x) = Y_4(x)$ (Rule 1)
  \[ V(x) = x(x-1)(x-2)(x-3) \]

- Degree Reduction

- $6x^3 + 5x^2 + 6x$
  - $k = 3 < SF(8)$, $b_k = 8/(8,6) = 4$, Coefficient = 6
  \[ V(x) = 1.4.Y_3(x) \text{ (Rule 2)} \]
  \[ V(x) = 4.x(x-1)(x-2) \]

- Coefficient Reduction

- $2x^3 + x^2 + 6x$ (Canonical Form)
Our Approach

- Say \( f(x) = a_k x^k + a_{k-1} x^{k-1} + \ldots + a_0 \)
  - In decreasing total degree order

- Given \( f(x) \) and the input/output bit-vector sizes
  - Check if degree can be reduced
  - Check if coefficient can be reduced
  - Perform corresponding reductions to get an intermediate expression
  - Estimate the cost of the intermediate expression
  - Repeat for all monomials …
  - Finally, when \( f(x) \) is in the reduced, minimal, unique form, identify the expression with the least cost
Exploring more solutions

- Consider $f = x^6 + 8x^3 + 8x$ in $\mathbb{Z}_{16}$

- Reduction of $f$ leads to following intermediate forms
  \[
  f = x^6 + 8x^3 + 8x \quad \Rightarrow \quad f_1 = 11x^5 + x^4 + 9x^3 + 8x^2 + 4x
  \]

- Reducing only $8x^3 + 8x$ leads to $0$ (vanishing polynomial!!!)

- $f$ reduces from $x^6 + 8x^3 + 8x$ to $x^6$

- $x^6$ is a better implementation!!!
Cost Model

- Adder(m-bit) = m * Cost (Full Adder)

- MULT(m-bit) = Partial products + Array
Cost Model

- **Constant Multiplier:** Simplification by constant propagation
  - Analyze the bit pattern of the constant
  - Propagate the bits using the array multiplier model
- **Example 1:** 5A, Bit pattern of 5 is \{0101\}
Cost Model

- Constant Multiplier: Simplification by constant propagation
  - Analyze the bit pattern of the constant
  - Propagate the bits using the array multiplier model

- Example 1: 5A, Bit pattern of 5 is \{0101\}

Cost is $3 \times \text{Cost (HA)}$
## Results

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Poly1</td>
<td>7581</td>
<td>3766</td>
<td>50.3%</td>
<td>37430</td>
<td>20628</td>
<td>44%</td>
<td>Minimal</td>
</tr>
<tr>
<td>Poly2</td>
<td>4820</td>
<td>2393</td>
<td>50.3%</td>
<td>28848</td>
<td>11684</td>
<td>59.49%</td>
<td>Minimal</td>
</tr>
<tr>
<td>Poly3</td>
<td>6227</td>
<td>5465</td>
<td>11.7%</td>
<td>28840</td>
<td>23006</td>
<td>20.2%</td>
<td>Minimal</td>
</tr>
<tr>
<td>Poly_unopt</td>
<td>5196</td>
<td>2994</td>
<td>42.3%</td>
<td>28836</td>
<td>14424</td>
<td>49.9%</td>
<td>Minimal</td>
</tr>
<tr>
<td>Deg4</td>
<td>22731</td>
<td>16361</td>
<td>28%</td>
<td>116684</td>
<td>82718</td>
<td>29.1%</td>
<td>Minimal</td>
</tr>
<tr>
<td>Janez</td>
<td>8907</td>
<td>6163</td>
<td>30.9%</td>
<td>42910</td>
<td>28840</td>
<td>32.7%</td>
<td>Minimal</td>
</tr>
<tr>
<td>Mibench</td>
<td>58510</td>
<td>48226</td>
<td>17.6%</td>
<td>249290</td>
<td>216772</td>
<td>13.04%</td>
<td>Intermed</td>
</tr>
<tr>
<td>IRR</td>
<td>10864</td>
<td>6943</td>
<td>37.3%</td>
<td>54594</td>
<td>37792</td>
<td>30.77%</td>
<td>Minimal</td>
</tr>
<tr>
<td>Antialias</td>
<td>15997</td>
<td>12011</td>
<td>24.9%</td>
<td>79254</td>
<td>59712</td>
<td>24.65%</td>
<td>Intermed</td>
</tr>
<tr>
<td>PSK</td>
<td>18140</td>
<td>18140</td>
<td>&lt;1%</td>
<td>76876</td>
<td>-</td>
<td>-</td>
<td>Orig</td>
</tr>
<tr>
<td>Cubic</td>
<td>47595</td>
<td>47586</td>
<td>&lt;1%</td>
<td>256388</td>
<td>-</td>
<td>-</td>
<td>Orig</td>
</tr>
<tr>
<td>IIR-4</td>
<td>49339</td>
<td>49333</td>
<td>&lt;1%</td>
<td>213408</td>
<td>-</td>
<td>-</td>
<td>Orig</td>
</tr>
</tbody>
</table>

Average area improvement: 23%
## Results

<table>
<thead>
<tr>
<th>Benchmark</th>
<th>Est. Cost</th>
<th>Imp. Cost</th>
<th>Selection</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Orig</td>
<td>Min</td>
<td>Improv</td>
</tr>
<tr>
<td>Poly1</td>
<td>7581</td>
<td>3766</td>
<td>50.3%</td>
</tr>
<tr>
<td>Poly2</td>
<td>4820</td>
<td>2393</td>
<td>50.3%</td>
</tr>
<tr>
<td>Poly3</td>
<td>6227</td>
<td>5465</td>
<td>11.7%</td>
</tr>
<tr>
<td>Poly_unopt</td>
<td>5196</td>
<td>2994</td>
<td>42.3%</td>
</tr>
<tr>
<td>Deg4</td>
<td>22731</td>
<td>16361</td>
<td>28%</td>
</tr>
<tr>
<td>Janez</td>
<td>8907</td>
<td>6163</td>
<td>30.9%</td>
</tr>
<tr>
<td>Mibench</td>
<td>58510</td>
<td>48226</td>
<td>17.6%</td>
</tr>
<tr>
<td>IRR</td>
<td>10864</td>
<td>6943</td>
<td>37.3%</td>
</tr>
<tr>
<td>Antialias</td>
<td>15997</td>
<td>12011</td>
<td>24.9%</td>
</tr>
<tr>
<td>PSK</td>
<td>18140</td>
<td>18140</td>
<td>&lt;1%</td>
</tr>
<tr>
<td>Cubic</td>
<td>47595</td>
<td>47586</td>
<td>&lt;1%</td>
</tr>
<tr>
<td>IIR-4</td>
<td>49339</td>
<td>49333</td>
<td>&lt;1%</td>
</tr>
</tbody>
</table>

Average area improvement: 23%
Conclusions & Future Work

- Area optimization approach for polynomial datapaths implemented with finite word-length operands

- Arithmetic datapaths are modeled as a polynomial function from
  \[ Z_{2^{n_1}} \times Z_{2^{n_2}} \times \cdots \times Z_{2^{n_d}} \rightarrow Z_2^m \]

- \( f(x_1, \ldots, x_d) \mod 2^m \) is reduced to its unique canonical form \( g(x_1, \ldots, x_d) \mod 2^m \)
  - Exploiting the concept of polynomial reducibility over
    \[ Z_{2^{n_1}} \times Z_{2^{n_2}} \times \cdots \times Z_{2^{n_d}} \rightarrow Z_2^m \]

- Cost Model to estimate area at polynomial level

- Reduction procedure + Cost model \( \rightarrow \) Least cost expression for implementation

- Future Work involves extensions for
  - Polynomial Decomposition over such arithmetic
  - Given n-bit ADD/MULTS, synthesize an m-bit datapath
Questions?