Formal Verification of Galois Field Arithmetic Circuits using Computer Algebra Techniques

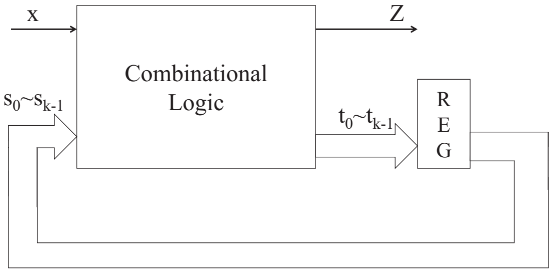
Priyank Kalla

Associate Professor
Electrical and Computer Engineering, University of Utah
kalla@ece.utah.edu
`http://www.ece.utah.edu/~kalla`

Some Background...

- My interests – Design Automation and Verification
  - Formal Verification of RTL-descriptions
  - Word-level abstractions from designs, symbolic techniques
  - Verification of finite-precision arithmetic
- Equivalence check: specification (*Spec*) vs implementation (*Impl*)
  - RTL-1 & RTL-2: same function?
  - Word-level spec (polynomial, RTL) vs gate-level circuit: same function?
  - RTL: functions over *k*-bit-vectors
    - k-bit-vector $\mapsto$ integers $\pmod{2^k} = \mathbb{Z}_{2^k}$
    - k-bit-vector $\mapsto$ Galois (Finite) field $\mathbb{F}_{2^k}$
- Approach: **Computer Algebra Techniques**
  - Model: Polynomial functions over $\mathbb{Z}_{2^k}$ or $\mathbb{F}_{2^k}$
  - Devise decision procedures for polynomial function equivalence
  - Commutative algebra, algebraic geometry + contemporary verification
- This talk, mostly about verification over Galois fields

Typical Sequential Circuit



- Primary input(s): x, primary output(s): Z
- Pseudo inputs: $\{s_0, s_1, \dots, s_{k-1}\}$
- Pseudo outputs: $\{t_0, t_1, \dots, t_{k-1}\}$

A FSM Example

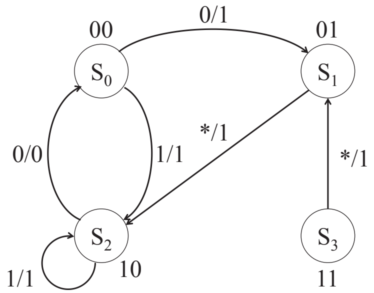


Figure: State Transition Graph

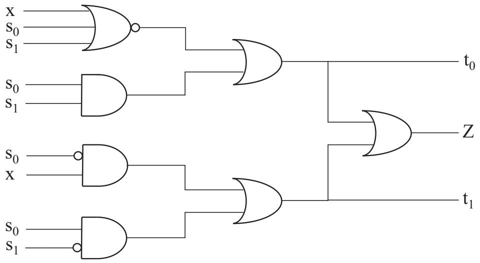


Figure: Corresponding Gate-level Circuit

Breadth-First Traversal Algorithm

ALGORITHM 1: Breadth-first Traversal Algorithm

```
**Input**: Transition functions $\Delta$, initial state $S^0$
$from^0 = reached = S^0$;
repeat
    $i \leftarrow i + 1$;
    $to^i \leftarrow \mathrm{Img}(\Delta, from^{i-1})$;
    $new^i \leftarrow to^i \cap \overline{reached}$;
    $reached \leftarrow reached \cup new^i$;
    $from^i \leftarrow new^i$;
until $new^i == 0$;
return $reached$
```

Image function, states intersection, union and complement in this algorithm will be implemented in computer algebra and algebraic geometry.

Implement Image Function in Computer Algebra

- State variables (word-level) S, T and sets of states such as $from^i$, to^i can always be represented as varieties of ideals.
- Boolean operators can always be converted to operations in $\mathbb{F}_2$

Boolean operator	operation in $\mathbb{F}_2$
$\overline{a}$	$1 + a$
a *and* b	ab
a *or* b	$a + b + ab$
$a \oplus b$	$a + b$

Table: Some Boolean operators and corresponding operations in $\mathbb{F}_2$

- An *elimination ideal* can be built from circuit gates, pseudo input/output word definition and vanishing polynomials

Implement Image Function in Computer Algebra(2)

Elimination ideal to model Image function for example STG:

- Transition functions (bit-level):
$$\begin{aligned} f_1 : &\quad t_0 - (\overline{x} \textit{ and } \overline{s_0} \textit{ and } \overline{s_1}) \textit{ or } (s_0 \textit{ and } s_1) \\ f_2 : &\quad t_1 - (\overline{s_0} \textit{ and } x) \textit{ or } (s_0 \textit{ and } \overline{s_1}) \end{aligned}$$
- Word variable definitions:
$$\begin{aligned} f_3 : &\quad S - s_0 - s_1\alpha \\ f_4 : &\quad T - t_0 - t_1\alpha \end{aligned}$$
- Vanishing polynomials: $f_6 : x^2 - x; f_7 : t_0^2 - t_0; f_8 : t_1^2 - t_1; f_9 : S^4 - S; f_{10} : s_0^2 - s_0; f_{11} : s_1^2 - s_1; f_{12} : T^4 - T$

Add the current state (for example, add initial states in first iteration $f_5 : S$), compute Gröbner basis for ideal $J = \langle f_1, \ldots, f_{12} \rangle$ under elimination term order

intermediate bit-level signals $>$ *bit-level primary inputs/outputs* $> S > T$

result will include a univariate polynomial about *next states* T.

Algebraic Geometry Concepts

Definition

(**Sum of Ideals**) If I and J are ideals in $k[x_1, \ldots, x_n]$, then the **sum** of I and J, denoted by $I + J$, is the set

$$I + J = \{f + g \mid f \in I \text{ and } g \in J\}.$$

Furthermore, if $I = \langle f_1, \ldots, f_r \rangle$ and $J = \langle g_1, \ldots, g_s \rangle$, then $I + J = \langle f_1, \ldots, f_r, g_1, \ldots, g_s \rangle$.

Definition

(**Product of Ideals**) If I and J are ideals in $k[x_1, \ldots, x_n]$, then the **product** of I and J, denoted by $I \cdot J$, is defined to be the ideal generated by all polynomials $f \cdot g$ where $f \in I$ and $g \in J$. Furthermore, let $I = \langle f_1, \ldots, f_r \rangle$ and $J = \langle g_1, \ldots, g_s \rangle$, then

$$I \cdot J = \langle f_i g_j \mid 1 \leq i \leq r, 1 \leq j \leq s \rangle.$$

Algebraic Geometry Concepts(2)

Definition

(**Quotient of Ideals**) If I and J are ideals in $k[x_1, \ldots, x_n]$, then $I : J$ is the set

$$\{f \in k[x_1, \ldots, x_n] \mid f \cdot g \in I, \forall g \in J\}$$

and is called the **ideal quotient** of I by J.

Concepts are adopted by following theorems:

Theorem

If I and J are ideals in $k[x_1, \ldots, x_n]$, then $\mathbf{V}(I + J) = \mathbf{V}(I) \bigcap \mathbf{V}(J)$ and $\mathbf{V}(I \cdot J) = \mathbf{V}(I) \bigcup \mathbf{V}(J)$.

Theorem

If I, J are ideals with only one generator, then $\mathbf{V}(I : J) = \mathbf{V}(I) - \mathbf{V}(J)$.

New Traversal Algorithm using Algebraic Geometry

ALGORITHM 2: Algebraic Geometry based Traversal Algorithm

```
Input: Input-output circuit characteristic polynomial ideal $I_{ckt}$, initial state
       polynomial $\mathcal{F}(S)$
$from^0 = reached = \mathcal{F}(S)$;
repeat
    $i \leftarrow i + 1$;
    $to^i \leftarrow$GB w/ elimination term order$\langle I_{ckt}, from^{i-1} \rangle$;
    $new^i \leftarrow$generator of $\langle to^i \rangle + (\langle T^4 - T \rangle : \langle reached \rangle)$;
    $reached \leftarrow$generator of $\langle reached \rangle \cdot \langle new^i \rangle$;
    $from^i \leftarrow new^i(S \setminus T)$;
until $new^i == 1$;
return $reached$
```

Example Executing New Traversal Algorithm

State encodings are mapped to varieties of ideals, e.g.:

$$\{00, 01\} \to \{0, 1\} = V_{\mathbb{F}_{2^2}}(\langle T^2 + T\rangle)$$

$$\{01, 10, 11\} \to \{1, \alpha, 1+\alpha\} = V_{\mathbb{F}_{2^2}}(\langle T^3 + 1\rangle)$$

- Iteration 0: Assume initial state is $\{00\} \to \{0\}$
- Iteration 1: $reached = from^0 = 0 = V_{\mathbb{F}_{2^2}}(\langle S\rangle)$, $to^1 = \{1, \alpha\} = V_{\mathbb{F}_{2^2}}(\langle T^2 + (1+\alpha)T + \alpha\rangle)$, $new^1 = to^1$, $reached = \{0, 1, \alpha\} = V_{\mathbb{F}_{2^2}}(\langle T^3 + (1+\alpha)T^2 + \alpha T\rangle)$
- Iteration 2:
  $from^1 = new^1(S \setminus T) = \{1, \alpha\} = V_{\mathbb{F}_{2^2}}(\langle S^2 + (1+\alpha)S + \alpha\rangle)$, $to^2 = \{0, \alpha\} = V_{\mathbb{F}_{2^2}}(\langle T^2 + \alpha T\rangle)$, $new^2 = 1$, **Terminate**
- Return $reached = \{0, 1, \alpha\} = V_{\mathbb{F}_{2^2}}(\langle T^3 + (1+\alpha)T^2 + \alpha T\rangle)$

An Application – Sequential Galois Arithmetic Circuits Verification

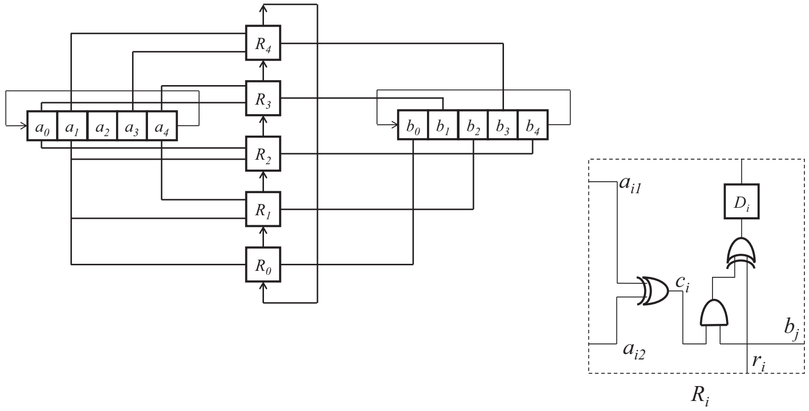


Figure: 5-bit Normal Basis Angew's Sequential Multiplier with Parallel Output (SMPO)

SMPO Protocol

- **Initial** $R_0 = R_1 = R_2 = R_3 = R_4 = 0$
- **Clock 1** $R_0 = a_1 b_0, R_1 = b_2(a_1 + a_4), R_2 = b_4(a_0 + a_1), R_3 = b_1(a_4 + a_0), R_4 = b_3(a_1 + a_3)$
- **Clock 2**
  $R_0 = b_3(a_1 + a_3) + a_0 b_4, R_1 = a_1 b_0 + b_1(a_0 + a_3), R_2 = b_2(a_1 + a_4) + b_3(a_4 + a_0), R_3 = b_4(a_0 + a_1) + b_0(a_3 + a_4), R_4 = b_1(a_4 + a_0) + b_2(a_0 + a_2)$
- ...
- **Clock 5** $R_0 = c_0, R_1 = c_1, R_2 = c_2, R_3 = c_3, R_4 = c_4$, i.e. $R = A \cdot B$.

Compose Elimination Ideal for 5-bit SMPO

An elimination ideal for the first clock cycle:

- **Gate descriptions:** $a_1 + a_4 + c_1, a_1 + a_0 + c_2, a_0 + a_4 + c_3, a_1 + a_3 + c_4, a_1 b_0 + r_4 + R_0, c_1 b_2 + r_0 + R_1, c_2 b_4 + r_1 + R_2, c_3 b_1 + r_2 + R_3, c_4 b_3 + r_3 + R_4;$
- **Word-level variables:** $A + a_0\alpha^5 + a_1\alpha^{10} + a_2\alpha^{20} + a_3\alpha^9 + a_4\alpha^{18}, B + b_0\alpha^5 + b_1\alpha^{10} + b_2\alpha^{20} + b_3\alpha^9 + b_4\alpha^{18}, r + r_0\alpha^5 + r_1\alpha^{10} + r_2\alpha^{20} + r_3\alpha^9 + r_4\alpha^{18}, R + R_0\alpha^5 + R_1\alpha^{10} + R_2\alpha^{20} + R_3\alpha^9 + R_4\alpha^{18};$
- **Vanishing polynomials:** $a_0^2 + a_0, a_1^2 + a_1, a_2^2 + a_2, a_3^2 + a_3, a_4^2 + a_4, b_0^2 + b_0, b_1^2 + b_1, b_2^2 + b_2, b_3^2 + b_3, b_4^2 + b_4, r_0^2 + r_0, r_1^2 + r_1, r_2^2 + r_2, r_3^2 + r_3, r_4^2 + r_4, R_0^2 + R_0, R_1^2 + R_1, R_2^2 + R_2, R_3^2 + R_3, R_4^2 + R_4, c_1^2 + c_1, c_2^2 + c_2, c_3^2 + c_3, c_4^2 + c_4, A^{32} + A, B^{32} + B, r^{32} + r, R^{32} + R;$
- **Feedback input:** r_{in}.

Fast Abstraction without GB computation

Definition

A lexicographic order constrained by following relation $>_r$: "circuit variables ordered reverse topologically" $>$ "designated word-level output" $>$ "word-level inputs" is called the *Refined Abstraction Term Order (RATO)*.

Example

The elimination ideal for 5-bit SMPO could be rewritten under RATO:

$$\begin{aligned}&(R_0, R_1, R_2, R_3, R_4) > (r_0, r_1, r_2, r_3, r_4)\\&> (c_1, c_2, c_3, c_4, b_0, b_1, b_2, b_3, b_4)\\&> (a_0, a_1, a_2, a_3, a_4) > R > r > (A, B)\end{aligned}$$

Under RATO, most polynomials have relatively prime leading terms/monomials (which means $Spoly \xrightarrow{J+J_0}_{+} 0$) except one pair: word-level polynomial corresponding to outputs and its leading bit-level variable's gate description polynomial.

Example

Candidate pair for 5-bit SMPO is $(f_w, f_g), f_w = R_0 + r_4 + b_0 \cdot a_1, f_g = R_0\alpha^5 + R_1\alpha^{10} + R_2\alpha^{20} + R_3\alpha^9 + R_4\alpha^{18} + R$.
Result after reduction is an abstraction:

$$
\begin{aligned}
& Spoly(f_w, f_g) \xrightarrow{J+J_0}_{+} \\
& r_1 + (\alpha)r_2 + (\alpha^4 + \alpha^2)r_3 + (\alpha^3 + \alpha^2)r_4 + (\alpha^3)b_1a_1 \\
+ & (\alpha^4 + \alpha^2)b_1a_2 + (\alpha^3 + \alpha + 1)b_1a_3 + (\alpha^3 + \alpha)b_1a_4 + (\alpha + 1)b_1A \\
+ & (\alpha^4 + \alpha^2 + \alpha)b_2a_1 + (\alpha^4 + \alpha^3 + \alpha^2 + \alpha)b_2a_4 + (\alpha^3 + \alpha^2 + 1)b_3a_1 \\
+ & (\alpha)b_3a_3 + (\alpha^2 + \alpha + 1)b_4a_1 + (\alpha + 1)b_4a_2 + (\alpha^4 + \alpha^2)b_4a_3 \\
+ & (\alpha^4 + \alpha^3 + \alpha + 1)b_4a_4 + (\alpha^3 + 1)b_4A + (\alpha^4 + \alpha^3 + \alpha^2 + 1)a_1B \\
+ & (\alpha^4 + \alpha^3 + \alpha^2 + 1)R
\end{aligned}
$$

Bit-Level Variable Substitution (BLVS)

Use Gaussian elimination style approach, eliminate other bit-level variables except for one.

Example

Objective: Abstract polynomial $a_i + \mathcal{G}_i(A)$ from $f_0 : a_0\alpha^5 + a_1\alpha^{10} + a_2\alpha^{20} + a_3\alpha^9 + a_4\alpha^{18} + A$. Eliminate variable a_0 by operation

$$\begin{aligned} f_1 =& f_0 \times \alpha^5 + f_0^2 : \\ & a_1 + (\alpha)a_2 + (\alpha^4 + \alpha^2)a_3 + (\alpha^3 + \alpha^2)a_4 \\ & +(\alpha^4 + \alpha^3 + \alpha^2 + 1)A^2 + (\alpha^2 + \alpha)A \end{aligned}$$

Recursively eliminate a_1 from f_1, a_2 from f_2, etc.

Bit-Level Variable Substitution (BLVS) (2)

Example

For 5-bit SMPO example, the result is

$$\begin{cases} a_0 = (\alpha+1)A^{16} + (\alpha^4+\alpha^3+\alpha)A^8 + (\alpha^3+\alpha^2)A^4 \\ \quad +(\alpha^4+1)A^2 + (\alpha^2+1)A \\ a_1 = (\alpha^2+1)A^{16} + (\alpha+1)A^8 + (\alpha^4+\alpha^3+\alpha)A^4 \\ \quad +(\alpha^3+\alpha^2)A^2 + (\alpha^4+1)A \\ a_2 = (\alpha^4+1)A^{16} + (\alpha^2+1)A^8 + (\alpha+1)A^4 \\ \quad +(\alpha^4+\alpha^3+\alpha)A^2 + (\alpha^3+\alpha^2)A \\ a_3 = (\alpha^3+\alpha^2)A^{16} + (\alpha^4+1)A^8 + (\alpha^2+1)A^4 \\ \quad +(\alpha+1)A^2 + (\alpha^4+\alpha^3+\alpha)A \\ a_4 = (\alpha^4+\alpha^3+\alpha)A^{16} + (\alpha^3+\alpha^2)A^8 + (\alpha^4+1)A^4 \\ \quad +(\alpha^2+1)A^2 + (\alpha+1)A \end{cases}$$

By substitution of bit-level variables in remainder from RATO, get next state abstraction $R + \mathcal{F}(A, B)$

Results Comparing to SAT/ABC/BDD

	Word size of the operands k-bits			
Solver	11	18	23	33
Lingeling	593	*TO*	*TO*	*TO*
ABC	6.24	*TO*	*TO*	*TO*
BDD	0.1	11.7	1002.4	*TO*

Table: Runtime for verification of bug-free SMPO circuits over $\mathbb{F}_{2^k}$ for SAT, ABC and BDD based methods. TO = timeout of 14 hrs

Results from our approach

Operand size k	36	66	82	89	100
#variables	183	333	413	448	503
#polynomials	2700	8910	13694	16109	20300
#terms	12996	43626	67322	79299	100100
Runtime(bug-free)	113	3673	15117	28986	50692
Runtime(buggy)	118	4320	15226	31571	58861

Table: Runtime (given in seconds) for verification of bug-free and buggy Angew's SMPO circuits over $\mathbb{F}_{2^k}$ using our approach