EXPLORING ALGEBRAIC METHODS FOR CIRCUIT VERIFICATION

Daniela Kaufmann

TU Wien

Vienna, Austria

Dagstuhl Seminar

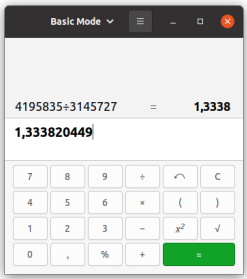
Theory and Practice of SAT and Combinatorial Solving

Dagstuhl, Germany

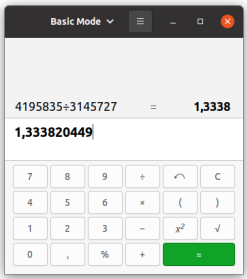
October 9-14, 2022

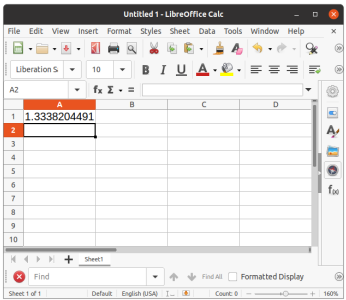
$$4\,195\,835 \div 3\,145\,727 = ?$$

$$4\,195\,835 \div 3\,145\,727 = ?$$

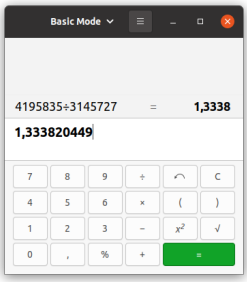


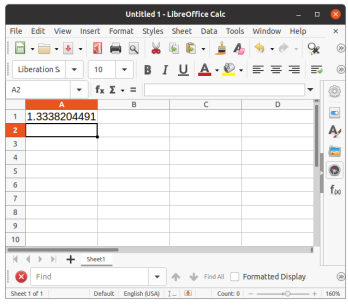
$$4\,195\,835 \div 3\,145\,727 = ?$$

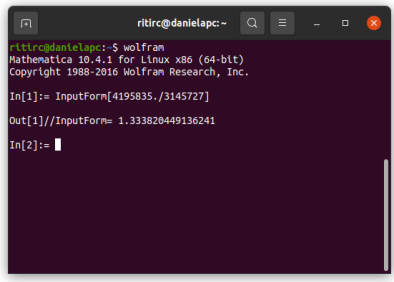


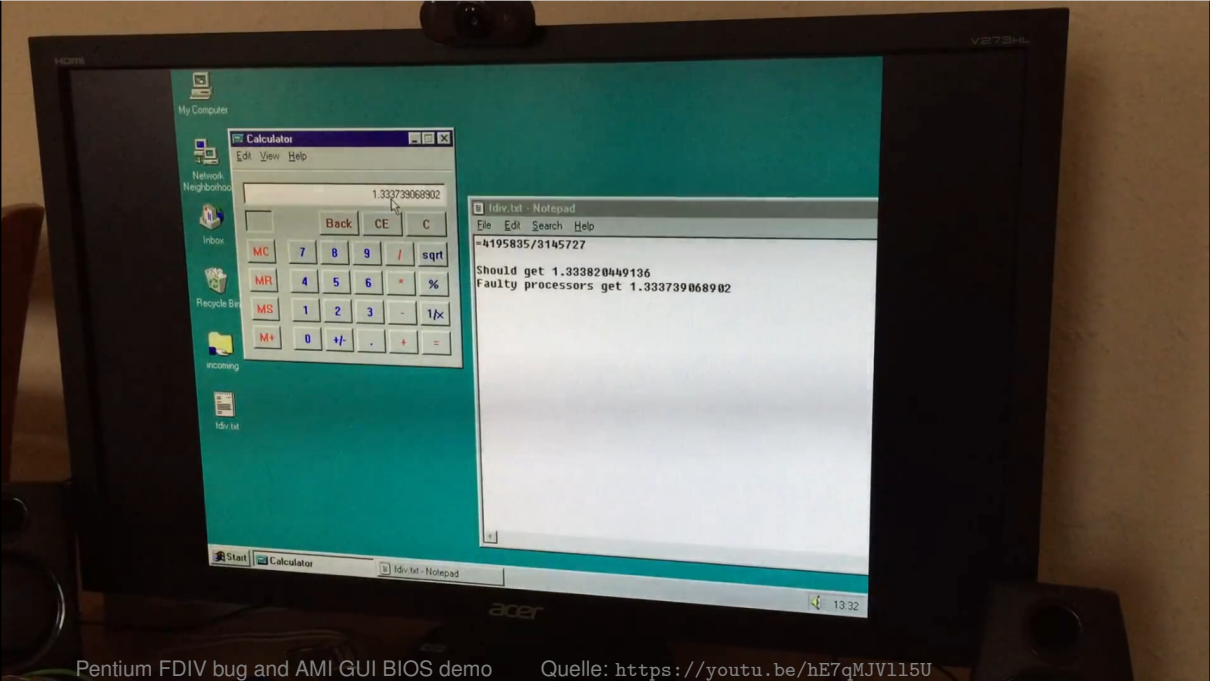


$$4\,195\,835 \div 3\,145\,727 = ?$$









Pentium FDIV bug and AMI GUI BIOS demo Quelle: https://youtu.be/hE7qMJV115U

Intel Pentium FDIV bug 1994

http://neology.com.au/portfolios/a80502-90-sx923/

- Affected floating point unit (FPU) in early Intel processors.
- Processor might return incorrect result for division.
- Cost in 1994: 500 million dollars.

Even more than 25 years later, verification of arithmetic circuits is considered to be hard. Especially proving the correctness of gate-level integer multiplier circuits is a challenge.

Multiplication

$$\begin{array}{r} 1\ 1 \cdot 1\ 1 \\ \hline \end{array}$$

$3 \cdot 3 = 9$

Multiplication

$$\begin{array}{rrcrr} 1 & 1 & \cdot & 1 & 1 \\ \hline & & & 1 & 1 \end{array}$$

$3 \cdot 3 = 9$

Multiplication

$$
\begin{array}{cccc}
1 & 1 & \cdot & 1 & 1 \\
\hline
 & & & 1 & 1 \\
 & & 1 & 1 & \\
\hline
\end{array}
$$

$3 \cdot 3 = 9$

Multiplication

$$
\begin{array}{rrcrr}
1 & 1 & \cdot & 1 & 1 \\
\hline
 & & & 1 & 1 \\
 & & 1 & 1 & \\
 & & & 0 & \\
\hline
 & & & & 1
\end{array}
$$

$3 \cdot 3 = 9$

Multiplication

$$
\begin{array}{rrcrr}
1 & 1 & \cdot & 1 & 1 \\
\hline
 & & & 1 & 1 \\
 & & 1_{1} & 1_{0} & \\
\hline
 & & & 0 & 1
\end{array}
$$

$3 \cdot 3 = 9$

Multiplication

$$
\begin{array}{rrcrr}
1 & 1 & \cdot & 1 & 1 \\
\hline
 & & & 1 & 1 \\
 & & 1 & 1 & \\
 & 1 & 1 & 0 & \\
\hline
 & & 0 & 0 & 1
\end{array}
$$

$3 \cdot 3 = 9$

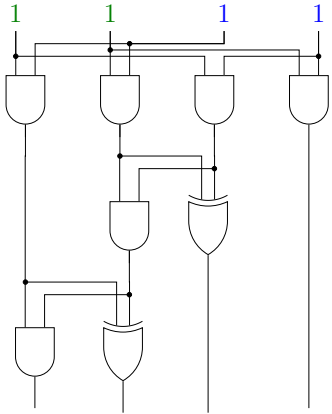
Multiplication

$$
\begin{array}{cccc}
1 & 1 & \cdot & 1 & 1 \\
\hline
 & & & 1 & 1 \\
 & & 1 & 1 & \\
 & 1 & 1 & 0 & \\
\hline
 & 1 & 0 & 0 & 1
\end{array}
$$

$3 \cdot 3 = 9$

Multiplication

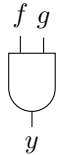
$$
\begin{array}{cccc}
1 & 1 & \cdot & 1 & 1 \\
\hline
 & & & 1 & 1 \\
 & & 1 & 1 & \\
 & 1 & 1 & 0 & \\
\hline
 & 1 & 0 & 0 & 1
\end{array}
$$



$3 \cdot 3 = 9$

Multiplication

AND-Gate



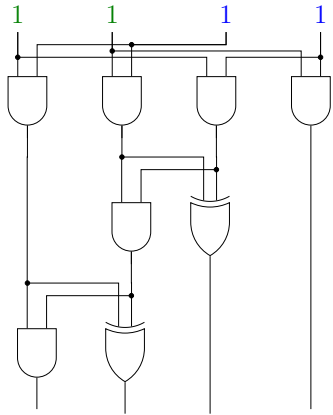
$f \wedge g = y$

f	g	y
0	0	0
0	1	0
1	0	0
1	1	1

XOR-Gate

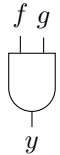
$f \oplus g = y$

f	g	y
0	0	0
0	1	1
1	0	1
1	1	0



Multiplication

AND-Gate



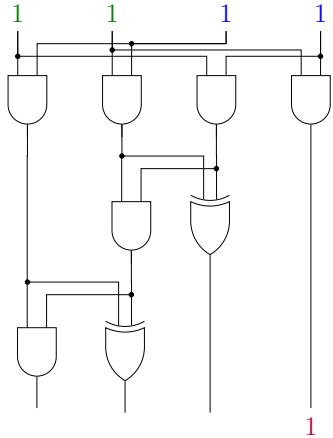
$f \wedge g = y$

f	g	y
0	0	0
0	1	0
1	0	0
1	1	1

XOR-Gate

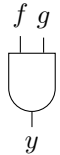
$f \oplus g = y$

f	g	y
0	0	0
0	1	1
1	0	1
1	1	0



Multiplication

AND-Gate



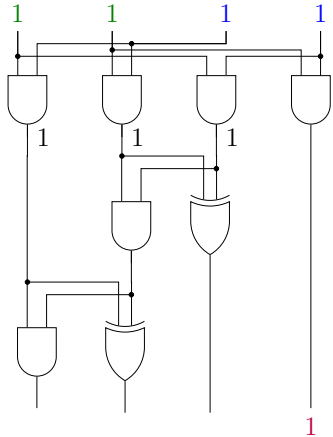
$f \wedge g = y$

f	g	y
0	0	0
0	1	0
1	0	0
1	1	1

XOR-Gate

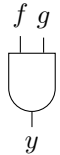
$f \oplus g = y$

f	g	y
0	0	0
0	1	1
1	0	1
1	1	0



Multiplication

AND-Gate



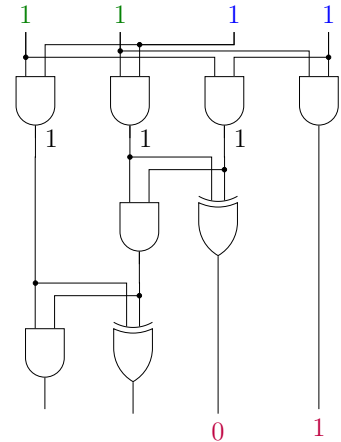
$f \wedge g = y$

f	g	y
0	0	0
0	1	0
1	0	0
1	1	1

XOR-Gate

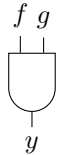
$f \oplus g = y$

f	g	y
0	0	0
0	1	1
1	0	1
1	1	0



Multiplication

AND-Gate



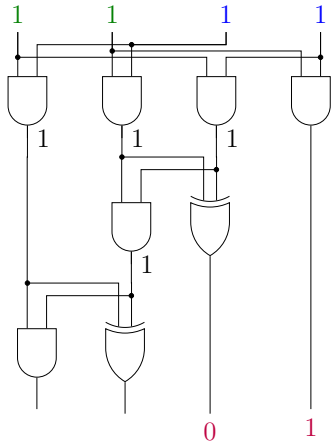
$f \wedge g = y$

f	g	y
0	0	0
0	1	0
1	0	0
1	1	1

XOR-Gate

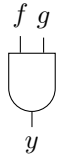
$f \oplus g = y$

f	g	y
0	0	0
0	1	1
1	0	1
1	1	0



Multiplication

AND-Gate



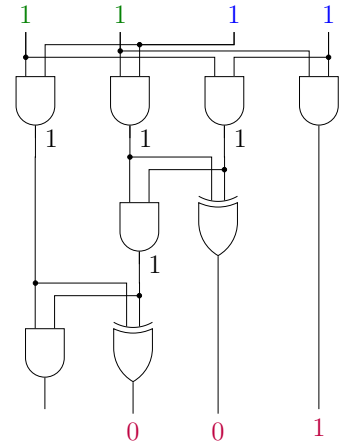
$f \wedge g = y$

f	g	y
0	0	0
0	1	0
1	0	0
1	1	1

XOR-Gate

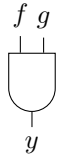
$f \oplus g = y$

f	g	y
0	0	0
0	1	1
1	0	1
1	1	0



Multiplication

AND-Gate



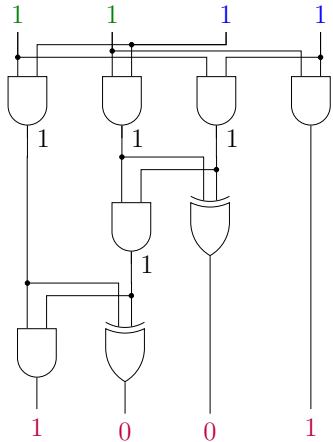
$f \wedge g = y$

f	g	y
0	0	0
0	1	0
1	0	0
1	1	1

XOR-Gate

$f \oplus g = y$

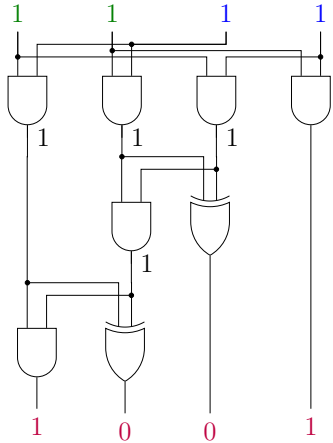
f	g	y
0	0	0
0	1	1
1	0	1
1	1	0



Is the circuit correct?

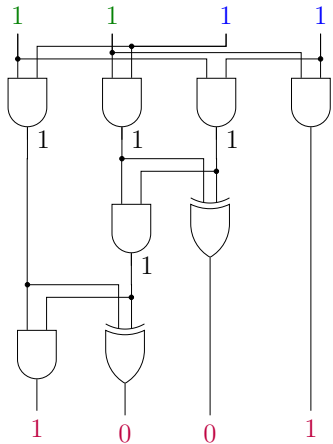
Multiplier circuit

$$
\begin{array}{cccc}
1 & 1 & \cdot & 1 & 1 \\
\hline
 & & & 1 & 1 \\
 & & 1 & 1 & \\
 & 1 & 1 & 0 & \\
\hline
1 & 0 & 0 & 1 &
\end{array}
$$



Multiplier circuit

Circuit seems correct!



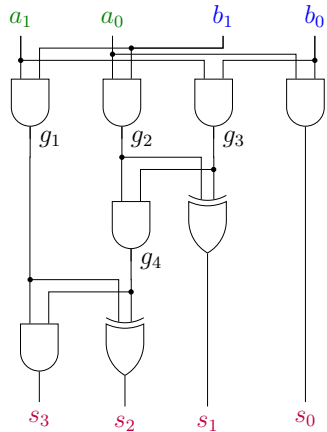
Is the circuit really correct?

Multiplier Circuits

Given: Gate-level multiplier for fixed bit-width.

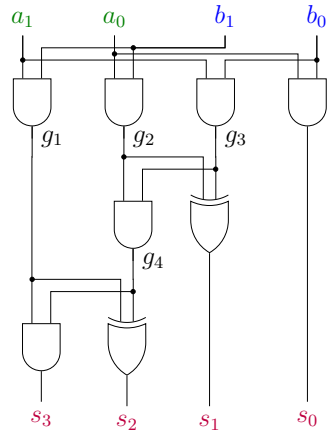
Question: For all possible $a_i, b_i \in \mathbb{B}$:

$$(2a_1 + a_0) * (2b_1 + b_0) = 8s_3 + 4s_2 + 2s_1 + s_0?$$



Simulation

$a_1 a_0$	·	$b_1 b_0$
00	·	00
00	·	01
00	·	10
00	·	11
01	·	00
01	·	01
01	·	10
01	·	11
10	·	00
10	·	01
10	·	10
10	·	11
11	·	00
11	·	01
11	·	10
11	·	11

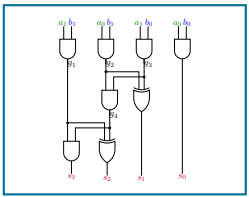


Simulation

Bit-width	Cases	
2	16	
3	64	
4	256	
5	1 024	
6	4 096	
7	16 384	
8	65 536	
⋮	⋮	
32	18 446 744 073 709 551 616	quintillion
⋮	⋮	
64	340 282 366 920 938 463 463 374 607 431 768 211 456	undecillion

Formal Verification

System



Specification

For all $a_i, b_i \in \mathbb{B}$:

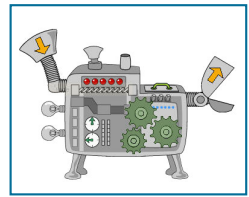
$(2a_1 + a_0) * (2b_1 + b_0) =$

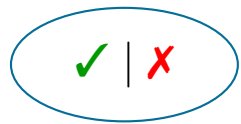
$8s_3 + 4s_2 + 2s_1 + s_0$?

Mathematical Model

$$
\begin{aligned}
B = \{ \\
& x - a_0 * b_0, \\
& y - a_1 * b_1, \\
& s_0 - x * y, \\
& \dots \\
\}
\end{aligned}
$$

Automated Decision Process





Formal Verification Techniques

Satisfiability Checking (SAT)

- SAT 2016 Competition

  [Biere SATComp'16]

- Exponential run-time of solvers

Formal Verification Techniques

Satisfiability Checking (SAT)

- SAT 2016 Competition
  [Biere SATComp'16]
- Exponential run-time of solvers

Decision Diagrams

- First technique to detect Pentium bug
  [ChenBryant DAC'95]
- Requires knowledge of the layout

Formal Verification Techniques

Satisfiability Checking (SAT)

- SAT 2016 Competition
  [Biere SATComp'16]
- Exponential run-time of solvers

Decision Diagrams

- First technique to detect Pentium bug
  [ChenBryant DAC'95]
- Requires knowledge of the layout

Theorem Proving

- Used in industry, e.g., ACL2
  [TemelSlobodovaHunt CAV'20]
- Requires manual effort

Formal Verification Techniques

Satisfiability Checking (SAT)

- SAT 2016 Competition
  [Biere SATComp'16]
- Exponential run-time of solvers

Decision Diagrams

- First technique to detect Pentium bug
  [ChenBryant DAC'95]
- Requires knowledge of the layout

Theorem Proving

- Used in industry, e.g., ACL2
  [TemelSlobodovaHunt CAV'20]
- Requires manual effort

Algebraic Approach

- Seminal work: [LvKallaEnescu TCAD'13, CiesielskiYuBrownLiuRossi DAC'15]
  [SayedGroßeKühneSoekenDrechsler DATE'16]
- Polynomial encoding
- Works for non-trivial multiplier designs

Algebraic Approach

Multipliers

- Finite Field Arithmetic

  [LvKallaEnescu TCAD'13, SuYasinYuCiesielski ISCAS'18]

- Integer Arithmetic

  [CiesielskiYuBrownLiuRossi DAC'15, SayedGroßeKühneSoekenDrechsler DATE'16, RitircBiereKauers FMCAD'17, MahzoonGroßeDrechsler ICCAD'18]

- Floating Point Arithmetic

  [SayedGroßeSoekenDrechsler FMCAD'16]

Dividers

- Division by a fixed constant

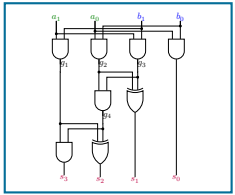
  [YasinSuPillementCiesielski ISVLSI'19]

- General Divider Circuits

  [SchollKonrad DAC'20, SchollKonradMahzoonGroßeDrechsler DATE'21]

Basic Idea of Algebraic Approach

Multiplier



Polynomials

$$
\begin{aligned}
B = \{ & \\
& x - a_0 * b_0, \\
& y - a_1 * b_1, \\
& s_0 - x * y, \\
& \dots \\
\} &
\end{aligned}
$$

Specification

$$\sum_{i=0}^{2n-1} 2^i s_i - (\sum_{i=0}^{n-1} 2^i a_i)(\sum_{i=0}^{n-1} 2^i b_i)$$

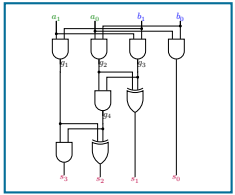
Implication

$= 0$ ✓

$\neq 0$ ✗

Basic Idea of Algebraic Approach

Multiplier



Polynomials

$$
\begin{aligned}
B = \{ & \\
& x - a_0 * b_0, \\
& y - a_1 * b_1, \\
& s_0 - x * y, \\
& \dots \\
\} &
\end{aligned}
$$

Specification

$$\sum_{i=0}^{2n-1} 2^i s_i - (\sum_{i=0}^{n-1} 2^i a_i)(\sum_{i=0}^{n-1} 2^i b_i)$$

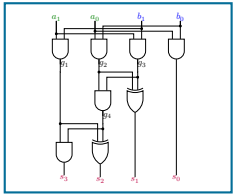
Roots

$= 0$ ✓

$\neq 0$ ✗

Basic Idea of Algebraic Approach

Multiplier



Polynomials

$$
\begin{aligned}
B = \{ & \\
& x - a_0 * b_0, \\
& y - a_1 * b_1, \\
& s_0 - x * y, \\
& \ldots \\
\} &
\end{aligned}
$$

Specification

$$\sum_{i=0}^{2n-1} 2^i s_i - (\sum_{i=0}^{n-1} 2^i a_i)(\sum_{i=0}^{n-1} 2^i b_i)$$

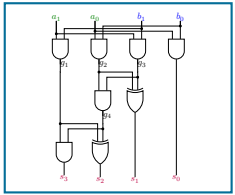
Ideal Membership

$= 0$ ✓

$\neq 0$ ✗

Basic Idea of Algebraic Approach

Multiplier



Polynomials

$$
\begin{aligned}
B = \{ & \\
& x - a_0 * b_0, \\
& y - a_1 * b_1, \\
& s_0 - x * y, \\
& \ldots \\
\} &
\end{aligned}
$$

Specification

$$\sum_{i=0}^{2n-1} 2^i s_i - (\sum_{i=0}^{n-1} 2^i a_i)(\sum_{i=0}^{n-1} 2^i b_i)$$

Ideal Membership

$= 0$ ✓

$\neq 0$ ✗

Multiplier Specification

Unsigned Integers:

$$0 = \sum_{i=0}^{2n-1} 2^i s_i - \left(\sum_{i=0}^{n-1} 2^i a_i\right)\left(\sum_{i=0}^{n-1} 2^i b_i\right) \quad \in \mathbb{Z}[X]$$

Multiplier Specification

Unsigned Integers:

$$\sum_{i=0}^{2n-1} 2^i s_i - \left(\sum_{i=0}^{n-1} 2^i a_i\right)\left(\sum_{i=0}^{n-1} 2^i b_i\right) \quad \in \mathbb{Z}[X]$$

Multiplier Specification

Unsigned Integers:

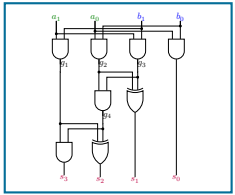
$$\sum_{i=0}^{2n-1} 2^i s_i - \left(\sum_{i=0}^{n-1} 2^i a_i\right)\left(\sum_{i=0}^{n-1} 2^i b_i\right) \quad \in \mathbb{Z}[X]$$

Signed Integers:

$$-2^{2n-1}s_{2n-1} + \sum_{i=0}^{2n-2} 2^i s_i - \left(-2^{n-1}a_{n-1} + \sum_{i=0}^{n-2} 2^i a_i\right)\left(-2^{n-1}b_{n-1} + \sum_{i=0}^{n-2} 2^i b_i\right) \quad \in \mathbb{Z}[X]$$

Basic Idea of Algebraic Approach

Multiplier



Polynomials

$$
\begin{aligned}
B = \{ & \\
& x - a_0 * b_0, \\
& y - a_1 * b_1, \\
& s_0 - x * y, \\
& \dots \\
\} &
\end{aligned}
$$

Specification

$$\sum_{i=0}^{2n-1} 2^i s_i - (\sum_{i=0}^{n-1} 2^i a_i)(\sum_{i=0}^{n-1} 2^i b_i)$$

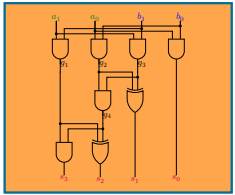
Ideal Membership

$= 0$ ✓

$\neq 0$ ✗

Basic Idea of Algebraic Approach

Multiplier



Polynomials

$$
\begin{aligned}
B = \{ & \\
& x - a_0 * b_0, \\
& y - a_1 * b_1, \\
& s_0 - x * y, \\
& \dots \\
\}
\end{aligned}
$$

Specification

$$\sum_{i=0}^{2n-1} 2^i s_i - (\sum_{i=0}^{n-1} 2^i a_i)(\sum_{i=0}^{n-1} 2^i b_i)$$

Ideal Membership

$= 0$ ✓

$\neq 0$ ✗

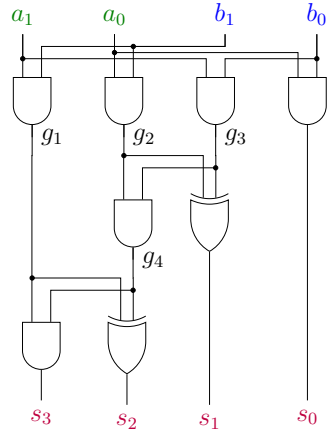
From Circuits to Polynomials

AND-Gate $f \wedge g = y$

f	g	y
0	0	0
0	1	0
1	0	0
1	1	1

XOR-Gate $f \oplus g = y$

f	g	y
0	0	0
0	1	1
1	0	1
1	1	0



From Circuits to Polynomials

AND-Gate $f \wedge g = y$

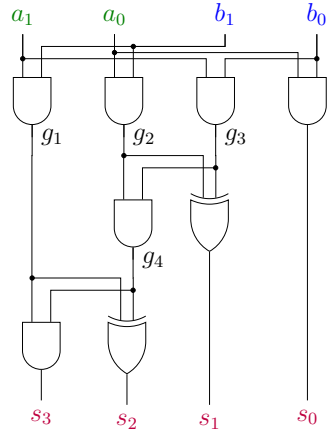
f	g	y
0	0	0
0	1	0
1	0	0
1	1	1

$$-y + fg$$

XOR-Gate $f \oplus g = y$

f	g	y
0	0	0
0	1	1
1	0	1
1	1	0

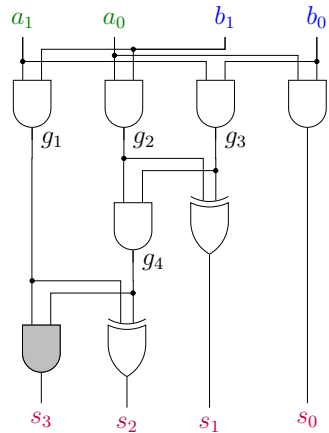
$$-y + f + g - 2fg$$



From Circuits to Polynomials

Gate polynomials $G(C)$.

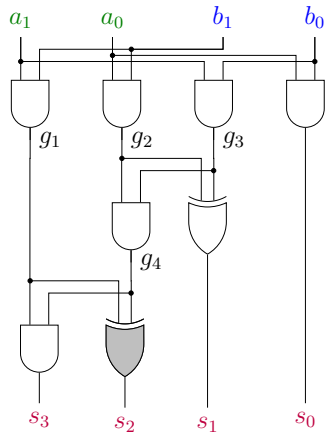
$$s_3 = g_1 \wedge g_4 \qquad -s_3 + g_4 g_1,$$



From Circuits to Polynomials

Gate polynomials $G(C)$.

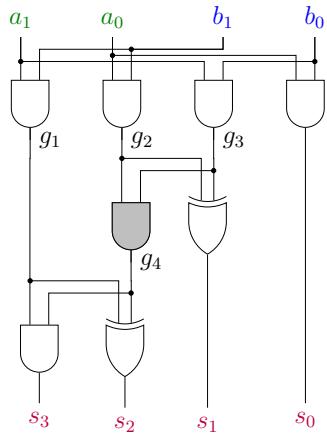
$$
\begin{aligned}
s_3 = g_1 \wedge g_4 \qquad & -s_3 + g_4 g_1, \\
s_2 = g_1 \oplus g_4 \qquad & -s_2 - 2g_4 g_1 + g_4 + g_1,
\end{aligned}
$$



From Circuits to Polynomials

Gate polynomials $G(C)$.

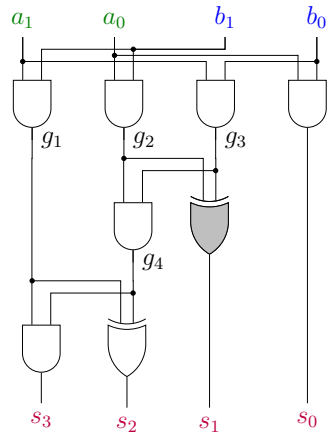
$$
\begin{aligned}
s_3 = g_1 \wedge g_4 \qquad & -s_3 + g_4 g_1, \\
s_2 = g_1 \oplus g_4 \qquad & -s_2 - 2g_4 g_1 + g_4 + g_1, \\
g_4 = g_2 \wedge g_3 \qquad & -g_4 + g_2 g_3,
\end{aligned}
$$



From Circuits to Polynomials

Gate polynomials $G(C)$.

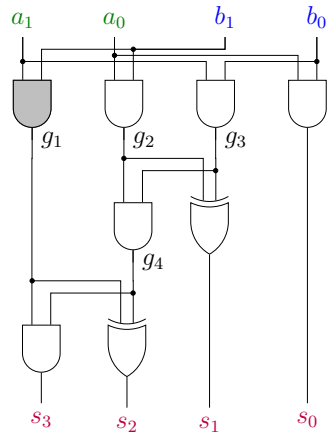
$$
\begin{aligned}
s_3 = g_1 \wedge g_4 \quad & -s_3 + g_4 g_1, \\
s_2 = g_1 \oplus g_4 \quad & -s_2 - 2g_4 g_1 + g_4 + g_1, \\
g_4 = g_2 \wedge g_3 \quad & -g_4 + g_2 g_3, \\
s_1 = g_2 \oplus g_3 \quad & -s_1 - 2g_2 g_3 + g_2 + g_3,
\end{aligned}
$$



From Circuits to Polynomials

Gate polynomials $G(C)$.

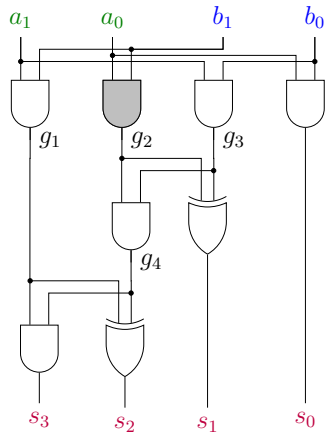
$$
\begin{aligned}
s_3 = g_1 \wedge g_4 \quad & -s_3 + g_4 g_1, \\
s_2 = g_1 \oplus g_4 \quad & -s_2 - 2g_4 g_1 + g_4 + g_1, \\
g_4 = g_2 \wedge g_3 \quad & -g_4 + g_2 g_3, \\
s_1 = g_2 \oplus g_3 \quad & -s_1 - 2g_2 g_3 + g_2 + g_3, \\
g_1 = a_1 \wedge b_1 \quad & -g_1 + a_1 b_1,
\end{aligned}
$$



From Circuits to Polynomials

Gate polynomials $G(C)$.

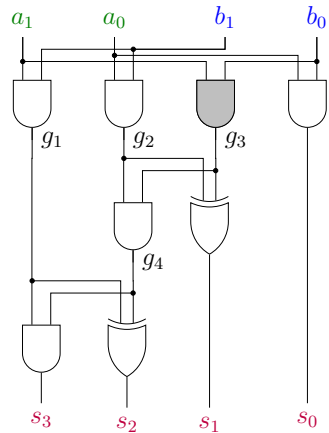
$$
\begin{aligned}
s_3 = g_1 \wedge g_4 \quad & -s_3 + g_4 g_1, \\
s_2 = g_1 \oplus g_4 \quad & -s_2 - 2g_4 g_1 + g_4 + g_1, \\
g_4 = g_2 \wedge g_3 \quad & -g_4 + g_2 g_3, \\
s_1 = g_2 \oplus g_3 \quad & -s_1 - 2g_2 g_3 + g_2 + g_3, \\
g_1 = a_1 \wedge b_1 \quad & -g_1 + a_1 b_1, \\
g_2 = a_0 \wedge b_1 \quad & -g_2 + a_0 b_1,
\end{aligned}
$$



From Circuits to Polynomials

Gate polynomials $G(C)$.

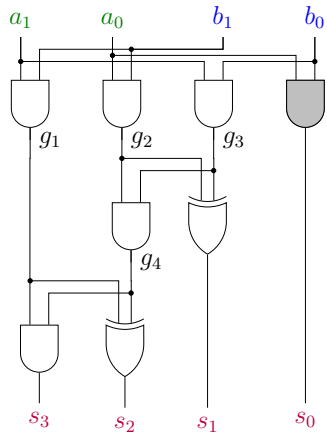
$$
\begin{aligned}
s_3 = g_1 \wedge g_4 \quad & -s_3 + g_4 g_1, \\
s_2 = g_1 \oplus g_4 \quad & -s_2 - 2g_4 g_1 + g_4 + g_1, \\
g_4 = g_2 \wedge g_3 \quad & -g_4 + g_2 g_3, \\
s_1 = g_2 \oplus g_3 \quad & -s_1 - 2g_2 g_3 + g_2 + g_3, \\
g_1 = a_1 \wedge b_1 \quad & -g_1 + a_1 b_1, \\
g_2 = a_0 \wedge b_1 \quad & -g_2 + a_0 b_1, \\
g_3 = a_1 \wedge b_0 \quad & -g_3 + a_1 b_0,
\end{aligned}
$$



From Circuits to Polynomials

Gate polynomials $G(C)$.

$$
\begin{aligned}
s_3 = g_1 \wedge g_4 \quad & -s_3 + g_4 g_1, \\
s_2 = g_1 \oplus g_4 \quad & -s_2 - 2g_4 g_1 + g_4 + g_1, \\
g_4 = g_2 \wedge g_3 \quad & -g_4 + g_2 g_3, \\
s_1 = g_2 \oplus g_3 \quad & -s_1 - 2g_2 g_3 + g_2 + g_3, \\
g_1 = a_1 \wedge b_1 \quad & -g_1 + a_1 b_1, \\
g_2 = a_0 \wedge b_1 \quad & -g_2 + a_0 b_1, \\
g_3 = a_1 \wedge b_0 \quad & -g_3 + a_1 b_0, \\
s_0 = a_0 \wedge b_0 \quad & -s_0 + a_0 b_0
\end{aligned}
$$



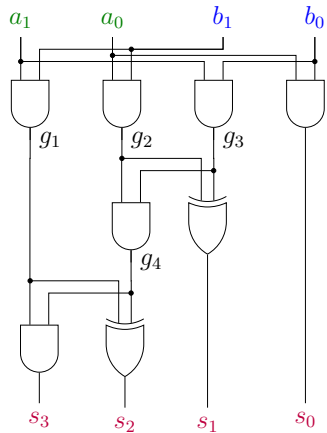
From Circuits to Polynomials

Gate polynomials $G(C)$.

$$
\begin{aligned}
s_3 = g_1 \wedge g_4 \quad & -s_3 + g_4 g_1, \\
s_2 = g_1 \oplus g_4 \quad & -s_2 - 2g_4 g_1 + g_4 + g_1, \\
g_4 = g_2 \wedge g_3 \quad & -g_4 + g_2 g_3, \\
s_1 = g_2 \oplus g_3 \quad & -s_1 - 2g_2 g_3 + g_2 + g_3, \\
g_1 = a_1 \wedge b_1 \quad & -g_1 + a_1 b_1, \\
g_2 = a_0 \wedge b_1 \quad & -g_2 + a_0 b_1, \\
g_3 = a_1 \wedge b_0 \quad & -g_3 + a_1 b_0, \\
s_0 = a_0 \wedge b_0 \quad & -s_0 + a_0 b_0
\end{aligned}
$$

Boolean value constraints $B(C)$.

$$
\begin{aligned}
a_1, a_0 \in \mathbb{B} \quad & a_1(1-a_1),\ a_0(1-a_0), \\
b_1, b_0 \in \mathbb{B} \quad & b_1(1-b_1),\ b_0(1-b_0)
\end{aligned}
$$



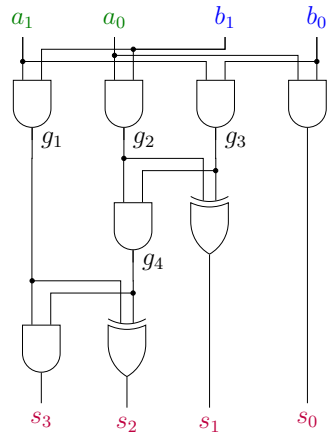
From Circuits to Polynomials

Gate polynomials $G(C)$.

$$
\begin{aligned}
s_3 = g_1 \wedge g_4 \quad & -s_3 + g_4 g_1, \\
s_2 = g_1 \oplus g_4 \quad & -s_2 - 2g_4 g_1 + g_4 + g_1, \\
g_4 = g_2 \wedge g_3 \quad & -g_4 + g_2 g_3, \\
s_1 = g_2 \oplus g_3 \quad & -s_1 - 2g_2 g_3 + g_2 + g_3, \\
g_1 = a_1 \wedge b_1 \quad & -g_1 + a_1 b_1, \\
g_2 = a_0 \wedge b_1 \quad & -g_2 + a_0 b_1, \\
g_3 = a_1 \wedge b_0 \quad & -g_3 + a_1 b_0, \\
s_0 = a_0 \wedge b_0 \quad & -s_0 + a_0 b_0
\end{aligned}
$$

Boolean value constraints $B(C)$.

$$
\begin{aligned}
a_1, a_0 \in \mathbb{B} \quad & -a_1^2 + a_1,\ -a_0^2 + a_0, \\
b_1, b_0 \in \mathbb{B} \quad & -b_1^2 + b_1,\ -b_0^2 + b_0
\end{aligned}
$$

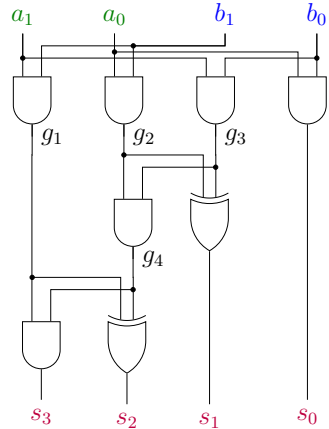


Ideals Associated to Circuits

[RitircBiereKauers FMCAD'17, KaufmannBiereKauers FMSD'20]

More circuit relations:

- $-s_0 + a_0 b_0$ AND-gate
- $-a_1^2 + a_1$ a_1 Boolean
- $-g_2^2 + g_2$ g_2 Boolean
- $s_1 g_4$ XOR-AND constraint
- ...



Polynomial Circuit Constraints.

A polynomial $p \in \mathbb{Z}[X]$ is a polynomial circuit constraint (PCC) if for all

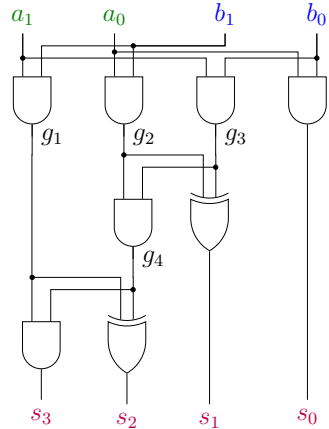
$$(a_0, \ldots, a_{n-1}, b_0, \ldots, b_{n-1}) \in \{0,1\}^{2n}$$

and resulting values $g_1, \ldots, g_k, s_0, \ldots, s_{2n-1}$ implied by the gates of the circuit C the substitution of these values into p gives zero.

Ideals Associated to Circuits

[RitircBiereKauers FMCAD'17, KaufmannBiereKauers FMSD'20]

- The set of all PCCs for C is denoted by $I(C)$.
- $I(C)$ contains all relations of the circuit C.
- $I(C)$ is an ideal.



Ideal. A subset $I \subseteq R[X]$ is called an ideal if

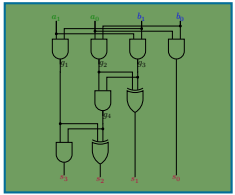
$$\forall\, p, q \in I : p+q \in I \quad \text{and} \quad \forall\, p \in R[X]\, \forall\, q \in I : pq \in I.$$

Multiplier. A circuit C is called a multiplier if

$$\sum_{i=0}^{2n-1} 2^i s_i - \left(\sum_{i=0}^{n-1} 2^i a_i\right)\left(\sum_{i=0}^{n-1} 2^i b_i\right) \quad \in \quad I(C).$$

Basic Idea of Algebraic Approach

Multiplier



Polynomials

$$B = \{$$
$$x - a_0 * b_0,$$
$$y - a_1 * b_1,$$
$$s_0 - x * y,$$
$$\ldots$$
$$\}$$

Specification

$$\sum_{i=0}^{2n-1} 2^i s_i - (\sum_{i=0}^{n-1} 2^i a_i)(\sum_{i=0}^{n-1} 2^i b_i)$$

Ideal Membership

$= 0$ ✓

$\neq 0$ ✗

Ideal Membership Problem

Basis. A set $G = \{g_1, \ldots, g_m\} \subseteq R[X]$ is called a **basis** of an ideal I if

$$I = \{h_1 g_1 + \cdots + h_m g_m \mid h_1, \ldots, h_m \in R[X]\} = \langle G \rangle$$

Given an arbitrary basis G of I it is not obvious how to decide ideal membership.

Ideal Membership Problem

Basis. A set $G = \{g_1, \ldots, g_m\} \subseteq R[X]$ is called a **basis** of an ideal I if

$$I = \{h_1 g_1 + \cdots + h_m g_m \mid h_1, \ldots, h_m \in R[X]\} = \langle G \rangle$$

Given an arbitrary basis G of I it is not obvious how to decide ideal membership.

Solution: We need a basis with certain structural properties, called Gröbner basis.

Ideal Membership Problem

Basis. A set $G = \{g_1, \ldots, g_m\} \subseteq R[X]$ is called a **basis** of an ideal I if

$$I = \{h_1 g_1 + \cdots + h_m g_m \mid h_1, \ldots, h_m \in R[X]\} = \langle G \rangle$$

Given an arbitrary basis G of I it is not obvious how to decide ideal membership.

Solution: We need a basis with certain structural properties, called Gröbner basis.

Gröbner basis. [Buchberger'65]

- Offers unique decision procedure for ideal membership problem.
- Every ideal has a finite Gröbner basis.
- Given an arbitrary basis of an ideal, we are able to compute a Gröbner basis.

Ideal Membership Problem

Basis. A set $G = \{g_1, \ldots, g_m\} \subseteq R[X]$ is called a **basis** of an ideal I if

$$I = \{h_1 g_1 + \cdots + h_m g_m \mid h_1, \ldots, h_m \in R[X]\} = \langle G \rangle$$

Given an arbitrary basis G of I it is not obvious how to decide ideal membership.

Solution: We need a basis with certain structural properties, called Gröbner basis.

Gröbner basis. [Buchberger'65]

- Offers unique decision procedure for ideal membership problem.
- Every ideal has a finite Gröbner basis.
- Given an arbitrary basis of an ideal, we are able to compute a Gröbner basis.
- Computing a Gröbner basis and deciding ideal membership is EXPSPACE-complete.[Mayr STACS'89]

Ideal Membership Problem

[RitircBiereKauers FMCAD'17]

- We can deduce some elements of $I(C)$:
  - Gate polynomials $G(C)$
  - Boolean value constraints $B(C)$
- Let $J(C) = \langle G(C) \cup B(C) \rangle$.
- Lexicographic term order: Reverse topological
  Output variable of a gate is greater than input variables [LvKallaEnescu TCAD'13].

Theorem

$G(C) \cup B(C)$ is a Gröbner basis for $J(C)$.

Proof idea: Application of Buchberger's Product criterion.

Soundness and Completeness

[RitircBiereKauers FMCAD'17]

Theorem

For all acyclic circuits C, we have $J(C) = I(C)$.

- $J(C) \subseteq I(C)$: soundness
- $I(C) \subseteq J(C)$: completeness

Non-Incremental Checking Algorithm

Verification Algorithm

Reduce specification $\sum_{i=0}^{2n-1} 2^i s_i - \left(\sum_{i=0}^{n-1} 2^i a_i\right)\left(\sum_{i=0}^{n-1} 2^i b_i\right)$ by elements of $G(C) \cup B(C)$

until no further reduction is possible, then C is a multiplier iff remainder is zero.

Verification

$$
\begin{aligned}
&G(C) \cup B(C) = \{ \\
&\quad -s_3 + g_1 g_4, \\
&\quad -s_2 - 2g_1 g_4 + g_4 + g_1, \\
&\quad -g_4 + g_2 g_3, \\
&\quad -s_1 - 2g_2 g_3 + g_3 + g_2, \\
&\quad -g_1 + a_1 b_1, \\
&\quad -g_2 + a_0 b_1, \\
&\quad -g_3 + a_1 b_0, \\
&\quad -s_0 + a_0 b_0, \\
&\quad -a_1^2 + a_1, \\
&\quad -a_0^2 + a_0, \\
&\quad -b_1^2 + b_1, \\
&\quad -b_0^2 + b_0\}
\end{aligned}
$$

$$8s_3 + 4s_2 + 2s_1 + s_0 - 4a_1 b_1 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0$$

Verification

$$
\begin{aligned}
&G(C) \cup B(C) = \{ \\
&-s_3 + g_1 g_4, \\
&-s_2 - 2g_1 g_4 + g_4 + g_1, \\
&-g_4 + g_2 g_3, \\
&-s_1 - 2g_2 g_3 + g_3 + g_2, \\
&-g_1 + a_1 b_1, \\
&-g_2 + a_0 b_1, \\
&-g_3 + a_1 b_0, \\
&-s_0 + a_0 b_0, \\
&-a_1^2 + a_1, \\
&-a_0^2 + a_0, \\
&-b_1^2 + b_1, \\
&-b_0^2 + b_0\}
\end{aligned}
$$

$$8s_3 + 4s_2 + 2s_1 + s_0 - 4a_1 b_1 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0$$

$$8g_1 g_4 + 4s_2 + 2s_1 + s_0 - 4a_1 b_1 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0$$

Verification

$$
\begin{aligned}
&G(C) \cup B(C) = \{ \\
&\quad -s_3 + g_1 g_4, \\
&\quad -s_2 - 2g_1 g_4 + g_4 + g_1, \\
&\quad -g_4 + g_2 g_3, \\
&\quad -s_1 - 2g_2 g_3 + g_3 + g_2, \\
&\quad -g_1 + a_1 b_1, \\
&\quad -g_2 + a_0 b_1, \\
&\quad -g_3 + a_1 b_0, \\
&\quad -s_0 + a_0 b_0, \\
&\quad -a_1^2 + a_1, \\
&\quad -a_0^2 + a_0, \\
&\quad -b_1^2 + b_1, \\
&\quad -b_0^2 + b_0\}
\end{aligned}
$$

$$
\begin{aligned}
8s_3 + 4s_2 + 2s_1 + s_0 - 4a_1 b_1 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0 \\
8g_1 g_4 + 4s_2 + 2s_1 + s_0 - 4a_1 b_1 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0 \\
4g_4 + 4g_1 + 2s_1 + s_0 - 4a_1 b_1 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0
\end{aligned}
$$

Verification

$$
\begin{aligned}
&G(C) \cup B(C) = \{ \\
&\quad -s_3 + g_1 g_4, \\
&\quad -s_2 - 2g_1 g_4 + g_4 + g_1, \\
&\quad -g_4 + g_2 g_3, \\
&\quad -s_1 - 2g_2 g_3 + g_3 + g_2, \\
&\quad -g_1 + a_1 b_1, \\
&\quad -g_2 + a_0 b_1, \\
&\quad -g_3 + a_1 b_0, \\
&\quad -s_0 + a_0 b_0, \\
&\quad -a_1^2 + a_1, \\
&\quad -a_0^2 + a_0, \\
&\quad -b_1^2 + b_1, \\
&\quad -b_0^2 + b_0 \}
\end{aligned}
$$

$$
\begin{aligned}
8s_3 + 4s_2 + 2s_1 + s_0 - 4a_1 b_1 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0 \\
8g_1 g_4 + 4s_2 + 2s_1 + s_0 - 4a_1 b_1 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0 \\
4g_4 + 4g_1 + 2s_1 + s_0 - 4a_1 b_1 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0 \\
4g_2 g_3 + 4g_1 + 2s_1 + s_0 - 4a_1 b_1 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0
\end{aligned}
$$

Verification

$$
\begin{aligned}
&G(C) \cup B(C) = \{ \\
&\quad -s_3 + g_1 g_4, \\
&\quad -s_2 - 2g_1 g_4 + g_4 + g_1, \\
&\quad -g_4 + g_2 g_3, \\
&\quad -s_1 - 2g_2 g_3 + g_3 + g_2, \\
&\quad -g_1 + a_1 b_1, \\
&\quad -g_2 + a_0 b_1, \\
&\quad -g_3 + a_1 b_0, \\
&\quad -s_0 + a_0 b_0, \\
&\quad -a_1^2 + a_1, \\
&\quad -a_0^2 + a_0, \\
&\quad -b_1^2 + b_1, \\
&\quad -b_0^2 + b_0\}
\end{aligned}
$$

$$
\begin{aligned}
8s_3 + 4s_2 + 2s_1 + s_0 - 4a_1 b_1 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0 \\
8g_1 g_4 + 4s_2 + 2s_1 + s_0 - 4a_1 b_1 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0 \\
4g_4 + 4g_1 + 2s_1 + s_0 - 4a_1 b_1 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0 \\
4g_2 g_3 + 4g_1 + 2s_1 + s_0 - 4a_1 b_1 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0 \\
4g_1 + 2g_3 + 2g_2 + s_0 - 4a_1 b_1 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0
\end{aligned}
$$

Verification

$$
\begin{aligned}
&G(C) \cup B(C) = \{ \\
&\quad -s_3 + g_1 g_4, \\
&\quad -s_2 - 2g_1 g_4 + g_4 + g_1, \\
&\quad -g_4 + g_2 g_3, \\
&\quad -s_1 - 2g_2 g_3 + g_3 + g_2, \\
&\quad -g_1 + a_1 b_1, \\
&\quad -g_2 + a_0 b_1, \\
&\quad -g_3 + a_1 b_0, \\
&\quad -s_0 + a_0 b_0, \\
&\quad -a_1^2 + a_1, \\
&\quad -a_0^2 + a_0, \\
&\quad -b_1^2 + b_1, \\
&\quad -b_0^2 + b_0\}
\end{aligned}
$$

$$
\begin{aligned}
8s_3 + 4s_2 + 2s_1 + s_0 - 4a_1 b_1 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0 \\
8g_1 g_4 + 4s_2 + 2s_1 + s_0 - 4a_1 b_1 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0 \\
4g_4 + 4g_1 + 2s_1 + s_0 - 4a_1 b_1 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0 \\
4g_2 g_3 + 4g_1 + 2s_1 + s_0 - 4a_1 b_1 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0 \\
4g_1 + 2g_3 + 2g_2 + s_0 - 4a_1 b_1 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0 \\
2g_3 + 2g_2 + s_0 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0
\end{aligned}
$$

Verification

$$
\begin{aligned}
&G(C) \cup B(C) = \{ \\
&-s_3 + g_1 g_4, \\
&-s_2 - 2g_1 g_4 + g_4 + g_1, \\
&-g_4 + g_2 g_3, \\
&-s_1 - 2g_2 g_3 + g_3 + g_2, \\
&-g_1 + a_1 b_1, \\
&-g_2 + a_0 b_1, \\
&-g_3 + a_1 b_0, \\
&-s_0 + a_0 b_0, \\
&-a_1^2 + a_1, \\
&-a_0^2 + a_0, \\
&-b_1^2 + b_1, \\
&-b_0^2 + b_0\}
\end{aligned}
$$

$$
\begin{aligned}
8s_3 + 4s_2 + 2s_1 + s_0 - 4a_1 b_1 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0 \\
8g_1 g_4 + 4s_2 + 2s_1 + s_0 - 4a_1 b_1 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0 \\
4g_4 + 4g_1 + 2s_1 + s_0 - 4a_1 b_1 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0 \\
4g_2 g_3 + 4g_1 + 2s_1 + s_0 - 4a_1 b_1 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0 \\
4g_1 + 2g_3 + 2g_2 + s_0 - 4a_1 b_1 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0 \\
2g_3 + 2g_2 + s_0 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0 \\
2g_3 + s_0 - 2a_1 b_0 - a_0 b_0
\end{aligned}
$$

Verification

$$
\begin{aligned}
G(C) \cup B(C) = \{ & \\
& -s_3 + g_1 g_4, \\
& -s_2 - 2g_1 g_4 + g_4 + g_1, \\
& -g_4 + g_2 g_3, \\
& -s_1 - 2g_2 g_3 + g_3 + g_2, \\
& -g_1 + a_1 b_1, \\
& -g_2 + a_0 b_1, \\
& -g_3 + a_1 b_0, \\
& -s_0 + a_0 b_0, \\
& -a_1^2 + a_1, \\
& -a_0^2 + a_0, \\
& -b_1^2 + b_1, \\
& -b_0^2 + b_0 \}
\end{aligned}
$$

$$
\begin{aligned}
8s_3 + 4s_2 + 2s_1 + s_0 - 4a_1 b_1 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0 \\
8g_1 g_4 + 4s_2 + 2s_1 + s_0 - 4a_1 b_1 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0 \\
4g_4 + 4g_1 + 2s_1 + s_0 - 4a_1 b_1 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0 \\
4g_2 g_3 + 4g_1 + 2s_1 + s_0 - 4a_1 b_1 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0 \\
4g_1 + 2g_3 + 2g_2 + s_0 - 4a_1 b_1 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0 \\
2g_3 + 2g_2 + s_0 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0 \\
2g_3 + s_0 - 2a_1 b_0 - a_0 b_0 \\
s_0 - a_0 b_0
\end{aligned}
$$

Verification

$$
\begin{aligned}
&G(C) \cup B(C) = \{ \\
&\quad -s_3 + g_1 g_4, \\
&\quad -s_2 - 2g_1 g_4 + g_4 + g_1, \\
&\quad -g_4 + g_2 g_3, \\
&\quad -s_1 - 2g_2 g_3 + g_3 + g_2, \\
&\quad -g_1 + a_1 b_1, \\
&\quad -g_2 + a_0 b_1, \\
&\quad -g_3 + a_1 b_0, \\
&\quad -s_0 + a_0 b_0, \\
&\quad -a_1^2 + a_1, \\
&\quad -a_0^2 + a_0, \\
&\quad -b_1^2 + b_1, \\
&\quad -b_0^2 + b_0\}
\end{aligned}
$$

$$
\begin{aligned}
8s_3 + 4s_2 + 2s_1 + s_0 - 4a_1b_1 - 2a_1b_0 - 2a_0b_1 - a_0b_0 \\
8g_1g_4 + 4s_2 + 2s_1 + s_0 - 4a_1b_1 - 2a_1b_0 - 2a_0b_1 - a_0b_0 \\
4g_4 + 4g_1 + 2s_1 + s_0 - 4a_1b_1 - 2a_1b_0 - 2a_0b_1 - a_0b_0 \\
4g_2g_3 + 4g_1 + 2s_1 + s_0 - 4a_1b_1 - 2a_1b_0 - 2a_0b_1 - a_0b_0 \\
4g_1 + 2g_3 + 2g_2 + s_0 - 4a_1b_1 - 2a_1b_0 - 2a_0b_1 - a_0b_0 \\
2g_3 + 2g_2 + s_0 - 2a_1b_0 - 2a_0b_1 - a_0b_0 \\
2g_3 + s_0 - 2a_1b_0 - a_0b_0 \\
s_0 - a_0b_0 \\
0
\end{aligned}
$$

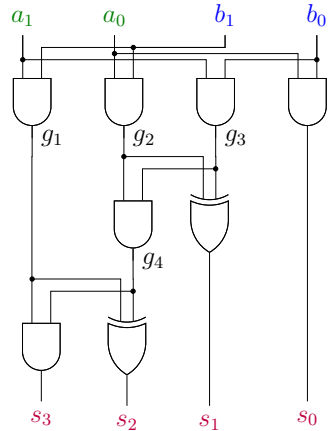
Faulty multiplier

Gate polynomials $G(C)$.

$$
\begin{aligned}
s_3 &= g_1 \wedge g_4 && -s_3 + g_4 g_1, \\
s_2 &= g_1 \oplus g_4 && -s_2 - 2g_4 g_1 + g_4 + g_1, \\
g_4 &= g_2 \wedge g_3 && -g_4 + g_2 g_3, \\
s_1 &= g_2 \oplus g_3 && -s_1 - 2g_2 g_3 + g_2 + g_3, \\
g_1 &= a_1 \wedge b_1 && -g_1 + a_1 b_1, \\
g_2 &= a_0 \wedge b_1 && -g_2 + a_0 b_1, \\
g_3 &= a_1 \wedge b_0 && -g_3 + a_1 b_0, \\
s_0 &= a_0 \wedge b_0 && -s_0 + a_0 b_0
\end{aligned}
$$

Boolean value constraints $B(C)$.

$$
\begin{aligned}
a_1, a_0 &\in \mathbb{B} && a_1(1-a_1),\ a_0(1-a_0), \\
b_1, b_0 &\in \mathbb{B} && b_1(1-b_1),\ b_0(1-b_0)
\end{aligned}
$$



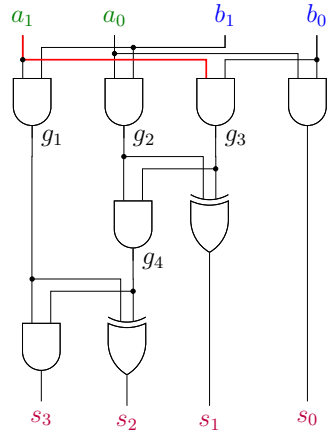
Faulty multiplier

Gate polynomials $G(C)$.

$$
\begin{aligned}
s_3 &= g_1 \wedge g_4 && -s_3 + g_4 g_1, \\
s_2 &= g_1 \oplus g_4 && -s_2 - 2g_4 g_1 + g_4 + g_1, \\
g_4 &= g_2 \wedge g_3 && -g_4 + g_2 g_3, \\
s_1 &= g_2 \oplus g_3 && -s_1 - 2g_2 g_3 + g_2 + g_3, \\
g_1 &= a_1 \wedge b_1 && -g_1 + a_1 b_1, \\
g_2 &= a_0 \wedge b_1 && -g_2 + a_0 b_1, \\
g_3 &= a_1 \wedge b_0 && -g_3 + a_1 b_0, \\
s_0 &= a_0 \wedge b_0 && -s_0 + a_0 b_0
\end{aligned}
$$

Boolean value constraints $B(C)$.

$$
\begin{aligned}
a_1, a_0 &\in \mathbb{B} && a_1(1-a_1),\ a_0(1-a_0), \\
b_1, b_0 &\in \mathbb{B} && b_1(1-b_1),\ b_0(1-b_0)
\end{aligned}
$$



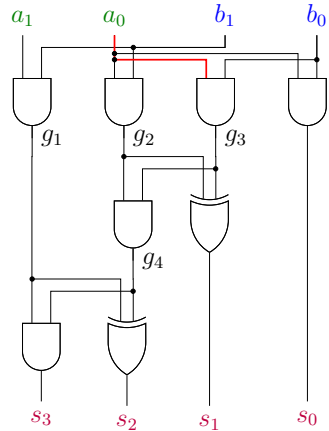
Faulty multiplier

Gate polynomials $G(C)$.

$$
\begin{aligned}
s_3 &= g_1 \wedge g_4 && -s_3 + g_4 g_1, \\
s_2 &= g_1 \oplus g_4 && -s_2 - 2g_4 g_1 + g_4 + g_1, \\
g_4 &= g_2 \wedge g_3 && -g_4 + g_2 g_3, \\
s_1 &= g_2 \oplus g_3 && -s_1 - 2g_2 g_3 + g_2 + g_3, \\
g_1 &= a_1 \wedge b_1 && -g_1 + a_1 b_1, \\
g_2 &= a_0 \wedge b_1 && -g_2 + a_0 b_1, \\
g_3 &= a_1 \wedge b_0 && -g_3 + a_0 b_0, \\
s_0 &= a_0 \wedge b_0 && -s_0 + a_0 b_0
\end{aligned}
$$

Boolean value constraints $B(C)$.

$$
\begin{aligned}
a_1, a_0 &\in \mathbb{B} && a_1(1-a_1),\ a_0(1-a_0), \\
b_1, b_0 &\in \mathbb{B} && b_1(1-b_1),\ b_0(1-b_0)
\end{aligned}
$$



Verification of a faulty multiplier

$$
\begin{aligned}
G(C) \cup B(C) = \{ & \\
& -s_3 + g_1 g_4, \\
& -s_2 - 2g_1 g_4 + g_4 + g_1, \\
& -g_4 + g_2 g_3, \\
& -s_1 - 2g_2 g_3 + g_3 + g_2, \\
& -g_1 + a_1 b_1, \\
& -g_2 + a_0 b_1, \\
& -g_3 + a_0 b_0, \\
& -s_0 + a_0 b_0, \\
& -a_1^2 + a_1, \\
& -a_0^2 + a_0, \\
& -b_1^2 + b_1, \\
& -b_0^2 + b_0 \}
\end{aligned}
$$

$$
\begin{aligned}
8s_3 + 4s_2 + 2s_1 + s_0 - 4a_1 b_1 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0 \\
8g_1 g_4 + 4s_2 + 2s_1 + s_0 - 4a_1 b_1 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0 \\
4g_4 + 4g_1 + 2s_1 + s_0 - 4a_1 b_1 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0 \\
4g_2 g_3 + 4g_1 + 2s_1 + s_0 - 4a_1 b_1 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0 \\
4g_1 + 2g_3 + 2g_2 + s_0 - 4a_1 b_1 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0 \\
2g_3 + 2g_2 + s_0 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0 \\
2g_3 + s_0 - 2a_1 b_0 - a_0 b_0
\end{aligned}
$$

Verification of a faulty multiplier

$$
\begin{aligned}
G(C) \cup B(C) = \{ & \\
& -s_3 + g_1 g_4, \\
& -s_2 - 2g_1 g_4 + g_4 + g_1, \\
& -g_4 + g_2 g_3, \\
& -s_1 - 2g_2 g_3 + g_3 + g_2, \\
& -g_1 + a_1 b_1, \\
& -g_2 + a_0 b_1, \\
& -g_3 + a_0 b_0, \\
& -s_0 + a_0 b_0, \\
& -a_1^2 + a_1, \\
& -a_0^2 + a_0, \\
& -b_1^2 + b_1, \\
& -b_0^2 + b_0 \}
\end{aligned}
$$

$$
\begin{aligned}
8s_3 + 4s_2 + 2s_1 + s_0 - 4a_1 b_1 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0 \\
8g_1 g_4 + 4s_2 + 2s_1 + s_0 - 4a_1 b_1 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0 \\
4g_4 + 4g_1 + 2s_1 + s_0 - 4a_1 b_1 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0 \\
4g_2 g_3 + 4g_1 + 2s_1 + s_0 - 4a_1 b_1 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0 \\
4g_1 + 2g_3 + 2g_2 + s_0 - 4a_1 b_1 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0 \\
2g_3 + 2g_2 + s_0 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0 \\
2g_3 + s_0 - 2a_1 b_0 - a_0 b_0 \\
s_0 - 2a_1 b_0 + a_0 b_0
\end{aligned}
$$

Verification of a faulty multiplier

$$
\begin{aligned}
G(C) \cup B(C) = \{ & \\
& -s_3 + g_1 g_4, \\
& -s_2 - 2g_1 g_4 + g_4 + g_1, \\
& -g_4 + g_2 g_3, \\
& -s_1 - 2g_2 g_3 + g_3 + g_2, \\
& -g_1 + a_1 b_1, \\
& -g_2 + a_0 b_1, \\
& -g_3 + a_0 b_0, \\
& -s_0 + a_0 b_0, \\
& -a_1^2 + a_1, \\
& -a_0^2 + a_0, \\
& -b_1^2 + b_1, \\
& -b_0^2 + b_0\}
\end{aligned}
$$

$$
\begin{aligned}
8s_3 + 4s_2 + 2s_1 + s_0 - 4a_1 b_1 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0 \\
8g_1 g_4 + 4s_2 + 2s_1 + s_0 - 4a_1 b_1 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0 \\
4g_4 + 4g_1 + 2s_1 + s_0 - 4a_1 b_1 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0 \\
4g_2 g_3 + 4g_1 + 2s_1 + s_0 - 4a_1 b_1 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0 \\
4g_1 + 2g_3 + 2g_2 + s_0 - 4a_1 b_1 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0 \\
2g_3 + 2g_2 + s_0 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0 \\
2g_3 + s_0 - 2a_1 b_0 - a_0 b_0 \\
s_0 - 2a_1 b_0 + a_0 b_0
\end{aligned}
$$

Verification of a faulty multiplier

$$
\begin{aligned}
G(C) \cup B(C) = \{ & \\
& -s_3 + g_1 g_4, \\
& -s_2 - 2g_1 g_4 + g_4 + g_1, \\
& -g_4 + g_2 g_3, \\
& -s_1 - 2g_2 g_3 + g_3 + g_2, \\
& -g_1 + a_1 b_1, \\
& -g_2 + a_0 b_1, \\
& -g_3 + a_0 b_0, \\
& -s_0 + a_0 b_0, \\
& -a_1^2 + a_1, \\
& -a_0^2 + a_0, \\
& -b_1^2 + b_1, \\
& -b_0^2 + b_0 \}
\end{aligned}
$$

$$
\begin{aligned}
& 8s_3 + 4s_2 + 2s_1 + s_0 - 4a_1 b_1 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0 \\
& 8g_1 g_4 + 4s_2 + 2s_1 + s_0 - 4a_1 b_1 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0 \\
& 4g_4 + 4g_1 + 2s_1 + s_0 - 4a_1 b_1 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0 \\
& 4g_2 g_3 + 4g_1 + 2s_1 + s_0 - 4a_1 b_1 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0 \\
& 4g_1 + 2g_3 + 2g_2 + s_0 - 4a_1 b_1 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0 \\
& 2g_3 + 2g_2 + s_0 - 2a_1 b_0 - 2a_0 b_1 - a_0 b_0 \\
& 2g_3 + s_0 - 2a_1 b_0 - a_0 b_0 \\
& s_0 - 2a_1 b_0 + a_0 b_0 \\
& -2a_1 b_0 + 2a_0 b_0
\end{aligned}
$$

Counter Example

Remainder: $-2a_1b_0 + 2a_0b_0 \neq 0$

Counter Example

Remainder: $-2a_1b_0 + 2a_0b_0 \neq 0$

$a_1 = 0, \quad a_0 = 1$
$b_1 = 0, \quad b_0 = 1$

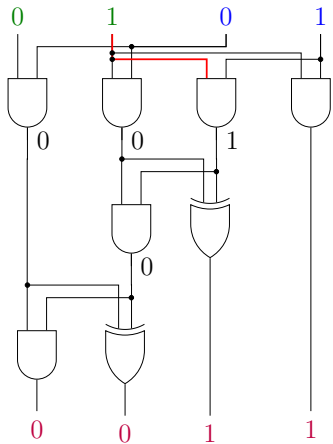
$01 \cdot 01 = 0001$

Counter Example

Remainder: $-2a_1b_0 + 2a_0b_0 \neq 0$

$a_1 = 0, \quad a_0 = 1$
$b_1 = 0, \quad b_0 = 1$

$01 \cdot 01 = 0001$



Error Correction using Algebra

Uses the remainder polynomial to compute rectification functions that nullify errors.

- Single-fix [FarahmandiMishra ICCD'17, MahzoonGroßeDrechsler ISVLSI'18]
- Multi-fix [RaoOndricekKallaEnescu VLSI-SoC'21, SabbaghAlizadeh ETS'21]

Limited application, as they rely on existence of remainder polynomial.

Non-Incremental Checking Algorithm

Verification Algorithm

Reduce specification $\sum_{i=0}^{2n-1} 2^i s_i - \left(\sum_{i=0}^{n-1} 2^i a_i\right)\left(\sum_{i=0}^{n-1} 2^i b_i\right)$ by elements of $G(C) \cup B(C)$

until no further reduction is possible, then C is a multiplier iff remainder is zero.

Non-Incremental Checking Algorithm

Verification Algorithm

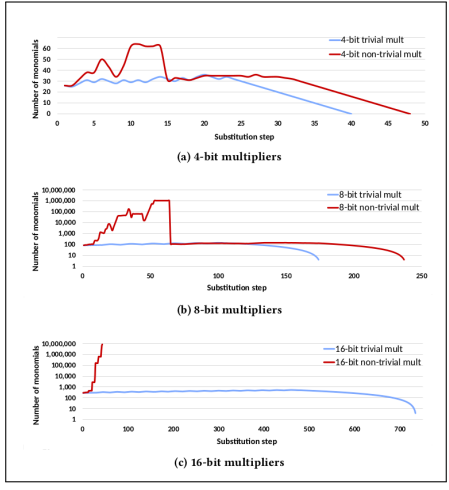
Reduce specification $\sum_{i=0}^{2n-1} 2^i s_i - (\sum_{i=0}^{n-1} 2^i a_i)(\sum_{i=0}^{n-1} 2^i b_i)$ by elements of $G(C) \cup B(C)$ until no further reduction is possible, then C is a multiplier iff remainder is zero.

Computational Problems

- The number of monomials in the intermediate results increases drastically.
- 8-bit multiplier cannot be verified within 20 minutes.

Non-Incremental Checking Algorithm

[MahzoonGroßeDrechsler ICCAD'18]

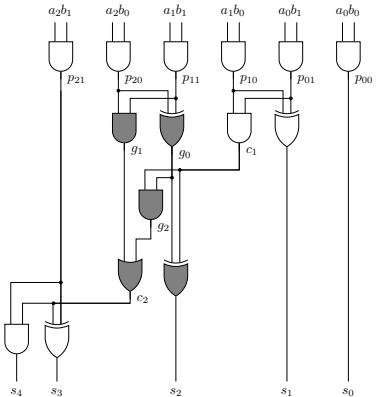


Strategies

1. Preprocessing
   - Variable Elimination [MahzoonGroßeDrechsler DAC'19, RitircBiereKauers DATE'18]
2. Reduction
   - Incremental Algorithm [RitircBiereKauers FMCAD'17]
   - Dynamic Reduction Order [MahzoonGroßeSchollDrechsler DATE'20]
3. Tricky: OR Gates in final stage adder
   - Include SAT or BDDs [KaufmannBiereKauers FMCAD'19, DrechslerMahzoon ISEEIE'22]
   - Dual variables [KaufmannBeameBiereNordström DATE'22]

Preprocessing - Variable Elimination

[RitircBiereKauers DATE'18, MahzoonGroßeDrechsler DAC'19]



Full adder

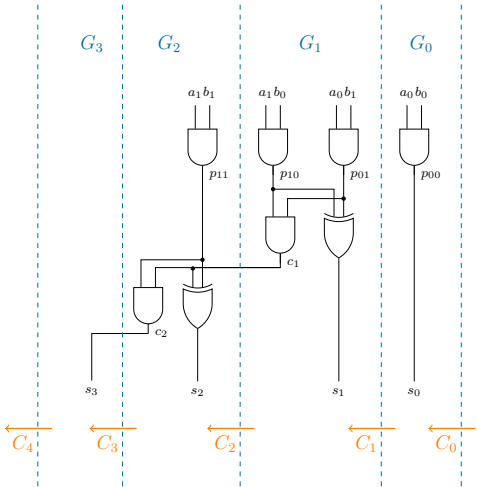
$$\left.\begin{aligned} &-c_2 + g_2 + g_1 - g_2 g_1, \\ &-s_2 + g_0 + c_1 - 2g_0 c_1, \\ &-g_2 + g_0 c_1, \\ &-g_1 + p_{20} p_{11}, \\ &-g_0 + p_{20} + p_{11} - 2p_{20} p_{11} \end{aligned}\right\} -2c_2 - s_2 + p_{20} + p_{11} + c_1$$

Theorem ([RitircBiereKauers DATE'18])

Local elimination of variables preserves Gröbner basis.

Incremental Verification

[RitircBiereKauers FMCAD'17]



Let $P_k = \sum_{k=i+j} a_i b_j$.

Column-Wise Checking Algorithm

Input: Circuit C with sliced Gröbner bases G_i
Output: Determine whether C is a multiplier

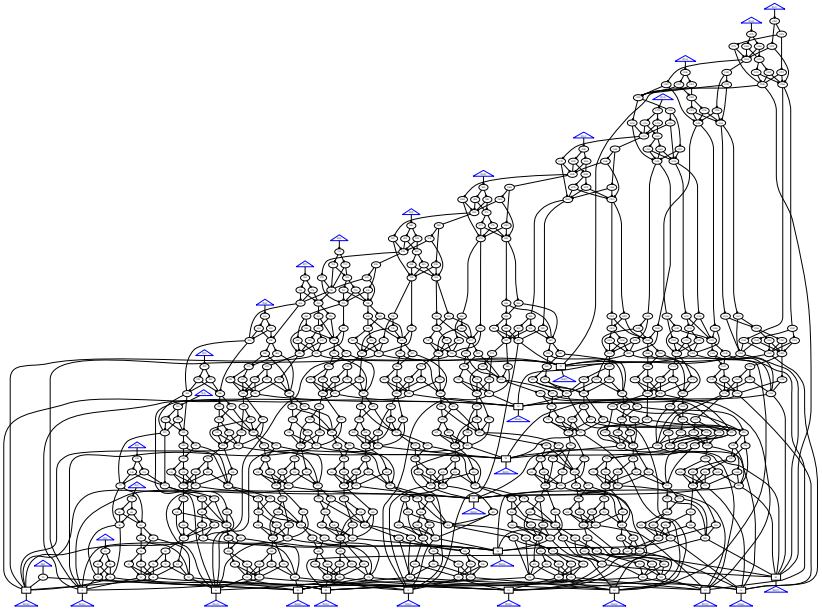
$C_{2n} \leftarrow 0$

for $i \leftarrow 2n-1$ **to** 0

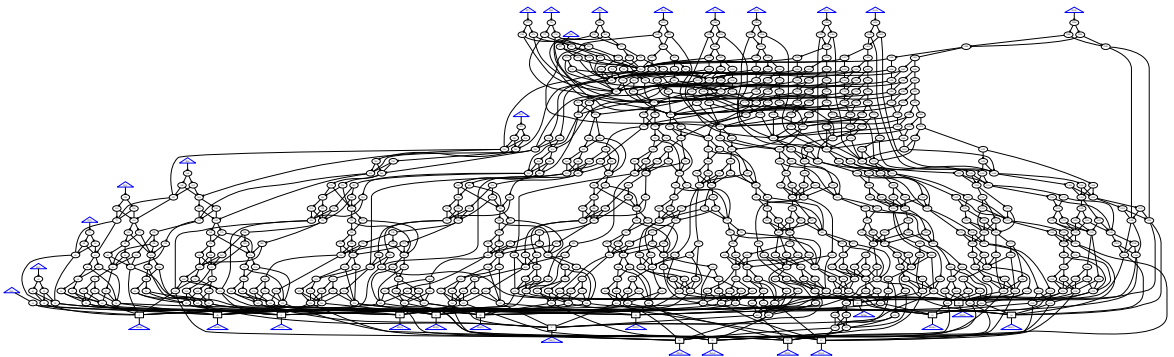
$C_i \leftarrow$ Remainder $(2C_{i+1} + s_i - P_i,\ G_i)$

return $C_0 = 0$

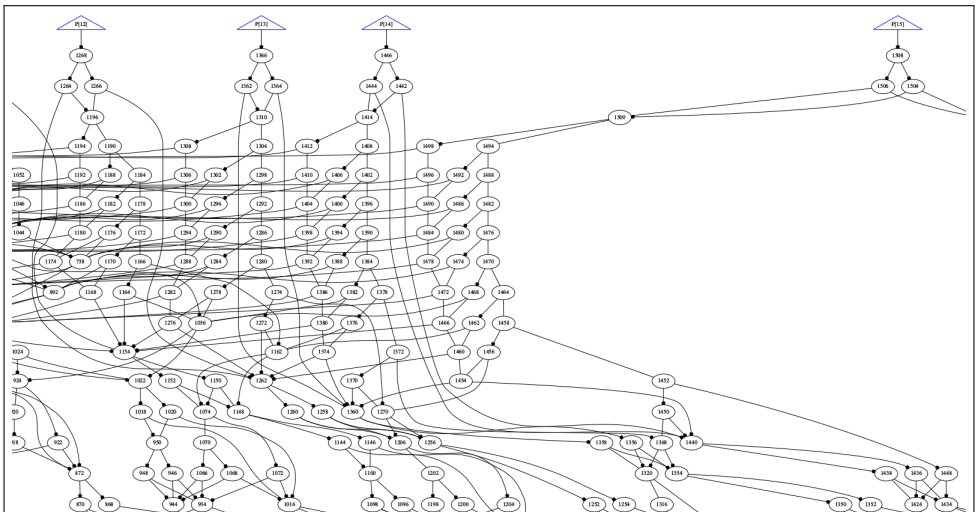
Simple



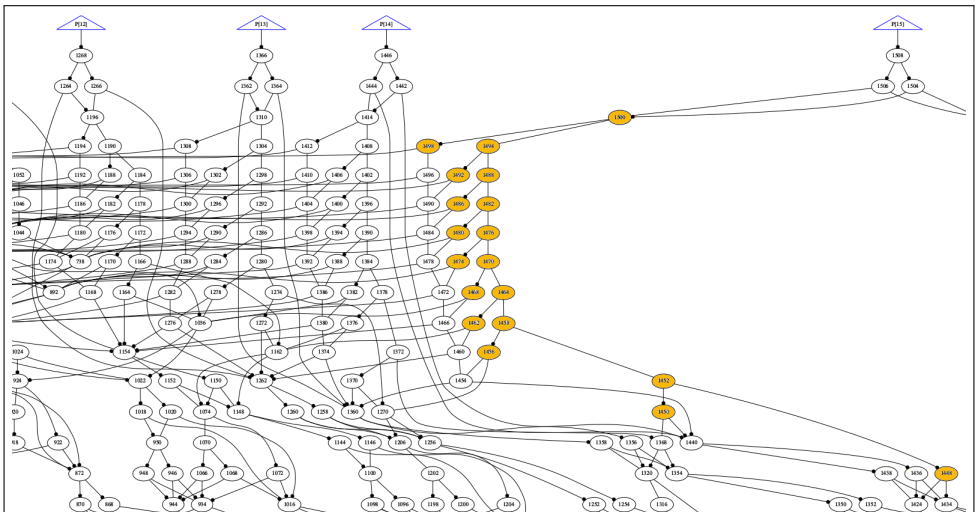
Hard



Hard

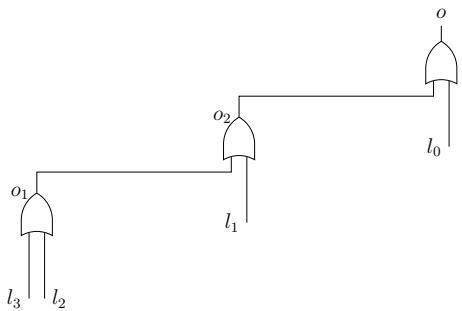


Hard



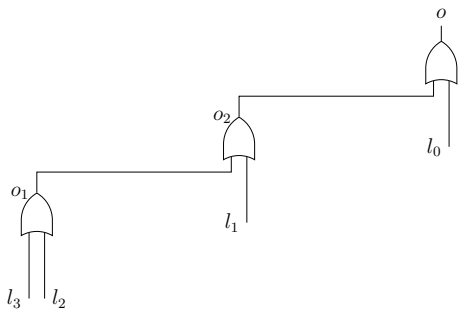
OR Gates

$$
\begin{aligned}
o = o_2 \vee x_0 \quad & -o + o_2 + l_0 - o_2 l_0, \\
o_2 = o_1 \vee l_1 \quad & -o_2 + o_1 + l_1 - o_1 l_1, \\
o_1 = l_3 \vee l_2 \quad & -o_1 + l_3 + l_2 - l_3 l_2
\end{aligned}
$$



OR Gates

$$
\begin{aligned}
o = o_2 \vee x_0 \quad & -o + o_2 + l_0 - o_2 l_0, \\
o_2 = o_1 \vee l_1 \quad & -o_2 + o_1 + l_1 - o_1 l_1, \\
o_1 = l_3 \vee l_2 \quad & -o_1 + l_3 + l_2 - l_3 l_2
\end{aligned}
$$



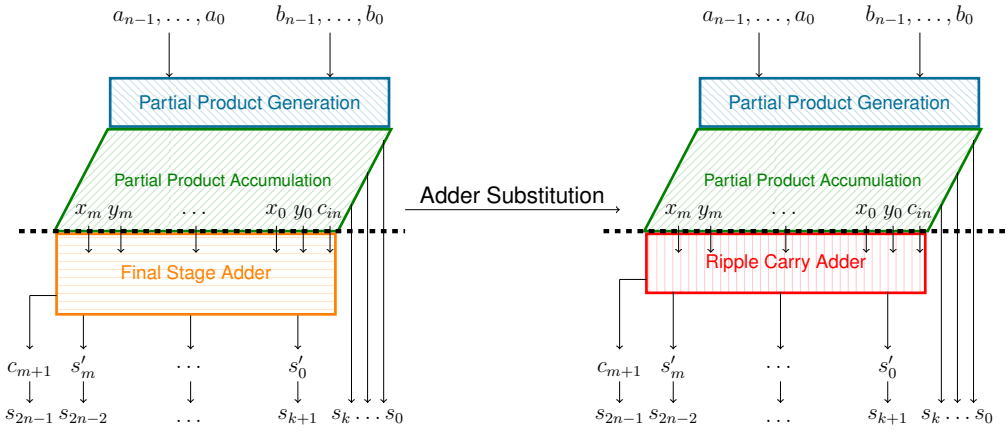
$$o = l_0 + l_1 - l_0 l_1 + l_2 - l_0 l_2 - l_1 l_2 + l_0 l_1 l_2 + l_3 - l_0 l_3 - l_1 l_3 + l_0 l_1 l_3 - l_2 l_3 + l_0 l_2 l_3 + l_1 l_2 l_3 - l_0 l_1 l_2 l_3$$

$15 = 2^4 - 1$ monomials

n OR Gates $\rightarrow 2^{n+1} - 1$ monomials

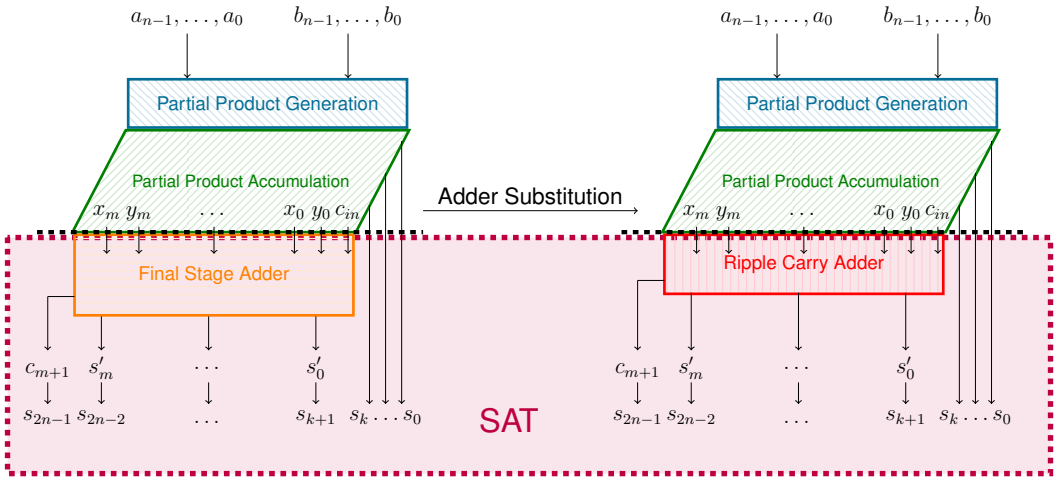
SAT & Computer Algebra

[KaufmannBiereKauers FMCAD'19]



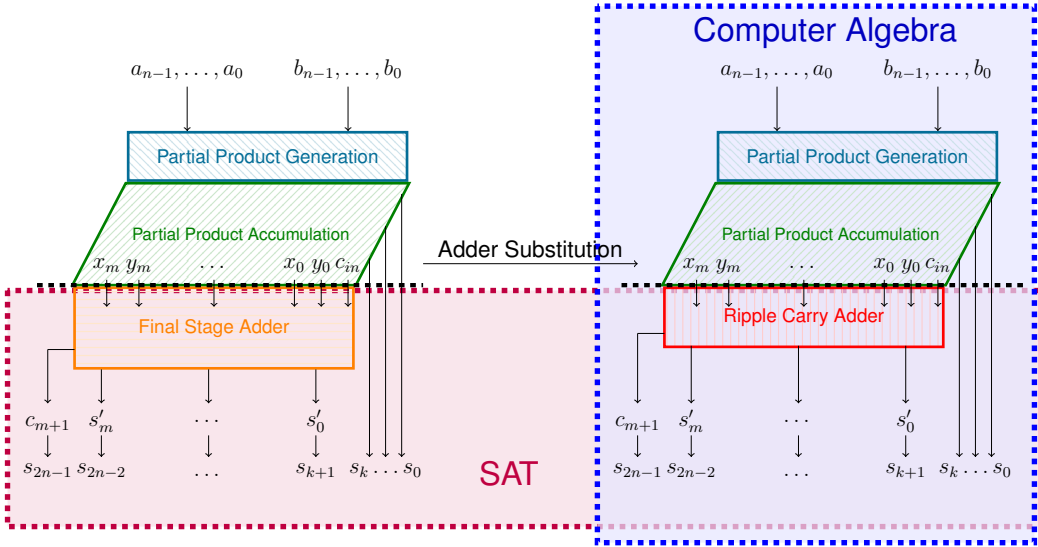
SAT & Computer Algebra

[KaufmannBiereKauers FMCAD'19]



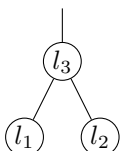
SAT & Computer Algebra

[KaufmannBiereKauers FMCAD'19]



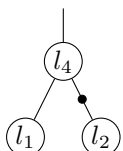
Dual Variables

Provide a shorthand notation for inverters.



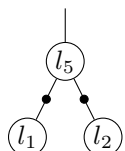
$$l_3 = l_1 \wedge l_2$$

$$-l_3 + l_1 l_2$$



$$l_4 = l_1 \wedge \neg l_2$$

$$-l_4 - l_1 l_2 + l_1$$



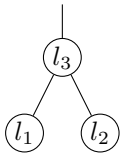
$$l_5 = \neg l_1 \wedge \neg l_2$$

$$-l_5 + l_1 l_2 - l_1 - l_2 + 1$$

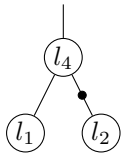
Dual Variables

Provide a shorthand notation for inverters.

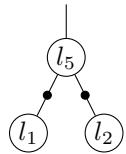
Dual variables.
Whenever two variables $l_i, f_i \in \{0, 1\}$ fulfill the relation $f_i = 1 - l_i$, we have $f_i = \mathrm{dual}(l_i)$.



$$l_3 = l_1 \wedge l_2$$
$$-l_3 + l_1 l_2$$
$$-l_3 + l_1 l_2$$



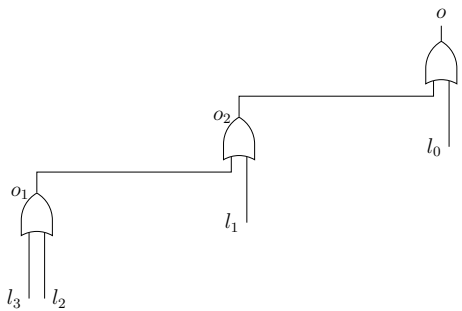
$$l_4 = l_1 \wedge \neg l_2$$
$$-l_4 - l_1 l_2 + l_1$$
$$-l_4 + l_1 f_2$$



$$l_5 = \neg l_1 \wedge \neg l_2$$
$$-l_5 + l_1 l_2 - l_1 - l_2 + 1$$
$$-l_5 + f_1 f_2$$

OR Gates

$$
\begin{aligned}
o = o_2 \vee x_0 \quad & -o + o_2 + l_0 - o_2 l_0, \\
o_2 = o_1 \vee l_1 \quad & -o_2 + o_1 + l_1 - o_1 l_1, \\
o_1 = l_3 \vee l_2 \quad & -o_1 + l_3 + l_2 - l_3 l_2
\end{aligned}
$$

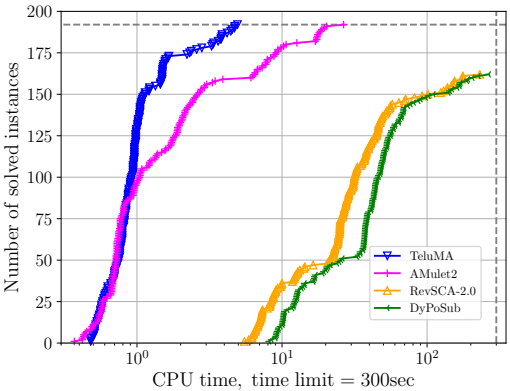


$$o = l_0 + l_1 - l_0 l_1 + l_2 - l_0 l_2 - l_1 l_2 + l_0 l_1 l_2 + l_3 - l_0 l_3 - l_1 l_3 + l_0 l_1 l_3 - l_2 l_3 + l_0 l_2 l_3 + l_1 l_2 l_3 - l_0 l_1 l_2 l_3$$

$$o = 1 - f_0 f_1 f_2 f_3$$

Evaluation - Multiplier Verification

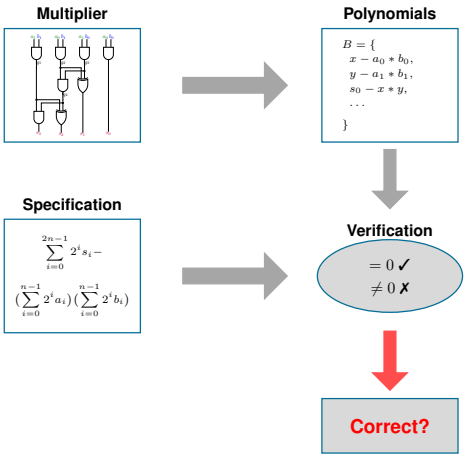
Verification of 192 unsigned 64-bit multipliers



- TeluMA [KaufmannBeameBiereNordström DATE'22]
  variable elimination - incremental reduction - dual var.
- AMulet2 [KaufmannBiere TACAS'21]
  variable elimination - incremental reduction - SAT
- RevSCA-2.0 [MahzoonGroßeDrechsler DAC'19]
  variable elimination
- DyPoSub [MahzoonGroßeSchollDrechsler DATE'20]
  variable elimination - dynamic reduction

Is the circuit really really correct?

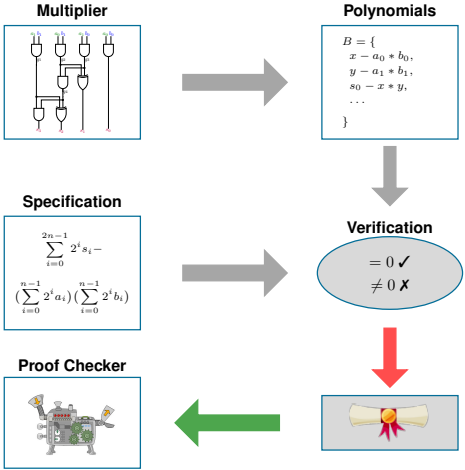
Proofs



Problem:

- Can we trust our own implementation?
- Is the verification process correct?

Proofs



Problem:

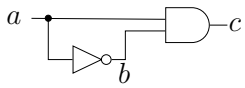
- Can we trust our own implementation?
- Is the verification process correct?

Solution:

Validate result of verification process.

- Generate machine-checkable proofs.
- Check by independent proof checkers.

Practical Algebraic Calculus



$$
\begin{aligned}
G(C) &= \{-b+1-a,\ -c+ab\} \\
B(C) &= \{-a^2+a\} \\
\text{Spec} &= c
\end{aligned}
$$

[RitircBiereKauers SC2'18]

```
* : -b+1-a,        a,         -a*b+a-a^2;
* : -a^2+a,        -1,        a^2-a;
+ : -a*b+a-a^2,    a^2-a,     -a*b;
+ : -a*b,          -c+a*b,    -c;
* : -c,            -1,        c;
```

[KaufmannFleuryBiere FMCAD'20]

```
3   * 1,   a,    -a*b;
1   d;
4   + 3,   2,    -c;
5   * 4,   -1,   c;
```

[KaufmannBiereKauers FMCAD'19]

```
* : -b+1-a,    a,         -a*b;
+ : -a*b,      -c+a*b,    -c;
* : -c,        -1,        c;
```

[KaufmannFleuryBiereKauers FMSD'21]

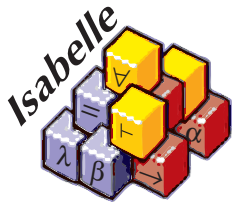
```
3   % 1*(-a) + 2*(-1), c;
```

Is the circuit really really really correct?

PASTÈQUE

[KaufmannFleuryBiere FMCAD'20]

Theorem Prover Isabelle/HOL



Refinement Approach, relying on Isabelle's Refinement Framework

- abstract specification on ideals: specification in ideal
- final step: executable checker

Isabelle's Archive of Formal Proofs 8000 lines of code

Conclusion

- **Is the circuit correct?**
  Yes, because we have tested some cases.
- **Is the circuit really correct?**
  Yes, because we applied algebraic reasoning.
- **Is the circuit really really correct?**
  Yes, because we generated and checked a proof certificate.
- **Is the circuit really really really correct?**
  Yes, because we used a verified proof checker.

References I

[Biere SATComp'16] A. Biere. Collection of Combinational Arithmetic Miters Submitted to the SAT Competition 2016. In SAT Competition 2016, pages 65–66, Dep. of Computer Science Report Series B, University of Helsinki, 2016.

[Buchberger'65] B. Buchberger. Ein Algorithmus zum Auffinden der Basiselemente des Restklassenringes nach einem nulldimensionalen Polynomideal. PhD Thesis, University of Innsbruck, 1965.

[CleggEdmondsImpagliazzo STOC'96] M. Clegg, J. Edmonds, and R. Impagliazzo. Using the Groebner Basis Algorithm to Find Proofs of Unsatisfiability. In Proc. of STOC'96, pages 174–183, ACM, 1996.

[CiesielskiYuBrownLiuRossi DAC'15] M. Ciesielski, C. Yu, W. Brown, D. Liu, and A. Rossi. Verification of Gate-level Arithmetic Circuits by Function Extraction. In Proc. of DAC'15, pages 52:1–52:6, ACM, 2015.

[ChenBryant DAC'95] Y. Chen and R. Bryant. Verification of Arithmetic Circuits with Binary Moment Diagrams. In Proc. of DAC'95, pages 535–541, ACM, 1995.

[DrechslerMahzoon ISEEIE'22] R. Drechsler and A. Mahzoon. Design Modification for Polynomial Formal Verification. In Proc. of ISEEIE'22, pages 187–194, IEEE, 2022.

References II

[FarahmandiMishra ICCD'17] F. Farahmandi and P. Mishra. Automated Debugging of Arithmetic Circuits Using Incremental Gröbner Basis Reduction. In Proc. of ICCD'17, pages 193–200, IEEE, 2017.

[KaufmannBeameBiereNordström DATE'22] D. Kaufmann, P.Beame, A. Biere and J. Nordström. Adding Dual Variables to Algebraic Reasoning for Gate-Level Multiplier Verification. In Proc. of DATE'22, pages 1431–1436, IEEE, 2022.

[KaufmannBiere TACAS'21] D. Kaufmann and A. Biere. AMulet 2.0 for Verifying Multiplier Circuits. Aceppted at TACAS'21, 2021.

[KaufmannBiereKauers FMCAD'19] D. Kaufmann, A. Biere and M. Kauers. Verifying Large Multipliers by Combining SAT and Computer Algebra. In Proc. of FMCAD'19, pages 28–36, IEEE, 2019.

[KaufmannBiereKauers FMSD'20] D. Kaufmann, A. Biere and M. Kauers. Incremental Column-Wise Verification of Arithmetic Circuits Using Computer Algebra. In FMSD, vol 56, pages 22–54, 2020.

[KaufmannFleuryBiere FMCAD'20] D. Kaufmann, M. Fleury and A. Biere. Pacheck and Pastèque Checking Practical Algebraic Calculus Proofs. In Proc. of FMCAD'20, pages 264–269, TU Vienna Academic Press, 2020.

References III

[KaufmannFleuryBiereKauers FMSD'21] D. Kaufmann, M. Fleury, A. Biere and M. Kauers. Practical Algebraic Calculus and Nullstellensatz with the Checkers Pacheck and Pastèque and Nuss-Checker. In FMSD, online first, 2021.

[LvKallaEnescu TCAD'13] J. Lv, P. Kalla, F. Enescu. Efficient Gröbner Basis Reductions for Formal Verification of Galois Field Arithmetic Circuits. In IEEE TCAD, vol. 32, pages 1409–1420, 2013.

[Mayr STACS'89] E. Mayr. Membership in polynomial ideals over Q is exponential space complete. In Proc. of STACS'89, pages 400–406, Springer, 1989

[MahzoonGroßeDrechsler DAC'19] A. Mahzoon, D. Große and R. Drechsler. RevSCA: Using Reverse Engineering to Bring Light into Backwards Rewriting for Big and Dirty Multipliers. In Proc. of DAC'19, pages 185:1–185:6, ACM, 2019.

[MahzoonGroßeDrechsler ICCAD'18] A. Mahzoon, D. Große and R. Drechsler. PolyCleaner: Clean your Polynomials before Backward Rewriting to verify Million-gate Multipliers. In Proc. of ICCAD'18, pages 129:1–129:8, ACM, 2018.

References IV

[MahzoonGroßeDrechsler ISVLSI'18] A. Mahzoon, D. Große and R. Drechsler. Combining Symbolic Computer Algebra and Boolean Satisfiability for Automatic Debugging and Fixing of Complex Multipliers. In Proc. of ISVLSI'18, pages 351–356, IEEE, 2018.

[MahzoonGroßeSchollDrechsler DATE'20] A. Mahzoon, D. Große, Christoph Scholl and R. Drechsler. Towards Formal Verification of Optimized and Industrial Multipliers. In Proc. of DATE'20, pages 544–549, DATE, 2020.

[RaoOndricekKallaEnescu VLSI-SoC'21] D. Ritirc, A. Biere and M. Kauers. Algebraic Techniques for Rectification of Finite Field Circuits. In Proc. of VLSI-SoC'21, pages 1–6, IEEE, 2021.

[RevSCA-2.0] A. Mahzoon, D. Große and R. Drechsler. RevSCA-2.0. `https://github.com/amahzoon/RevSCA-2.0`

[RitircBiereKauers DATE'18] D. Ritirc, A. Biere and M. Kauers. Improving and Extending the Algebraic Approach for Verifying Gate-Level Multipliers. In Proc. of DATE'18, pages 1556–1561, IEEE, 2018.

[RitircBiereKauers FMCAD'17] D. Ritirc, A. Biere and M. Kauers. Column-Wise Verification of Multipliers Using Computer Algebra. In Proc. of FMCAD'17, pages 23–30, IEEE, 2017.

References V

[RitircBiereKauers SC2'18] D. Ritirc, A. Biere and M. Kauers. A Practical Polynomial Calculus for Arithmetic Circuit Verification. In Proc. of SC'2, pages 61–76, CEUR-WS, 2018.

[SabbaghAlizadeh ETS'21] N. A. Sabbagh and B. Alizadeh. Arithmetic Circuit Correction by Adding Optimized Correctors Based on Groebner Basis Computation. In Proc. of ETS'21, pages 1–6, IEEE, 2021.

[SayedGroßeKühneSoekenDrechsler DATE'16] A. Sayed-Ahmed, D. Große, U. Kühne, M. Soeken, and R. Drechsler. Formal verification of integer multipliers by combining Gröbner basis with logic reduction. In Proc. of DATE'16, pages 1048–1053, IEEE, 2016.

[SayedGroßeSoekenDrechsler FMCAD'16] A. Sayed-Ahmed, D. Große, M. Soeken and R. Drechsler. Equivalence checking using Gröbner bases. In Proc. of FMCAD'16, pages 169–176, IEEE, 2016.

[SchollKonradMahzoonGroßeDrechsler DATE'21] C. Scholl, A. Konrad, A. Mahzoon, D. Große and R. Drechsler. Verifying Dividers Using Symbolic Computer Algebra and Don't Care Optimization. In Proc. of DATE'21, pages 1110–1115, IEEE, 2021.

[SchollKonrad DAC'20] C. Scholl and A. Konrad. Symbolic Computer Algebra and SAT Based Information Forwarding for Fully Automatic Divider Verification. In Proc. of DAC'20, pages 1–6, IEEE, 2020.

References VI

[SuYasinYuCiesielski ISCAS'18] T. Su, A. Yasin, C. Yu and M. Ciesielski. Computer Algebraic Approach to Verification and Debugging of Galois Field Multipliers. In Proc. of ISCAS'18, pages 1–5, IEEE, 2018.

[TemelSlobodovaHunt CAV'20] M. Temel, A. Slobodova and W. Hunt. Automated and Scalable Verification of Integer Multipliers. In Proc. of CAV'20, pages 485–507, Springer, 2020.

[YasinSuPillementCiesielski ISVLSI'19] A. Yasin, T. Su, S. Pillement and M. Ciesielski. Formal Verification of Integer Dividers:Division by a Constant. In Proc. of ISVLSI'19, pages 76–81, IEEE, 2019.