Scalable Architecture of Constant Division on FPGA

AUTHORS

D. Gorodecky*^ *danila.gorodecky@gmail.com*

L. Sousa* *leonel.sousa@tecnico.ulisboa.pt*

PRESENTER

Ricardo Nobre*

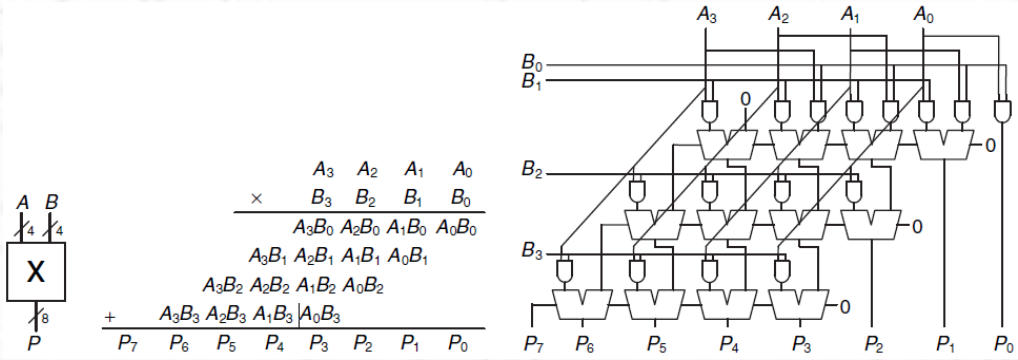
* INESC-ID, Instituto Superior Tecnico, Universidade de Lisboa, Portugal

^ EHU / EPAM School of Digital Engineering, Lithuania

Combinational logic

- multiplier, adder, and etc.

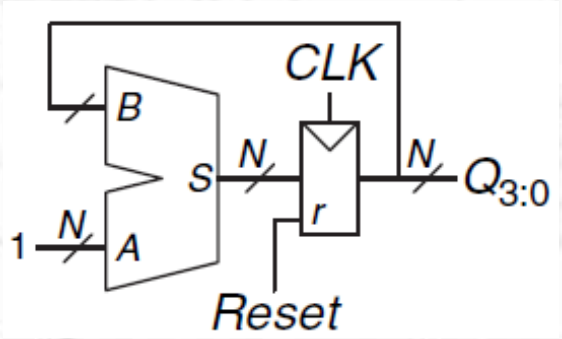
4x4->8 multiplier

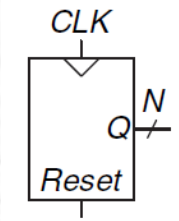


Memory-based

- counter, register, pipelining, and etc.

N-bit counter





Types of Divisors

Memory-based

- **restoring division**
- **non-restoring**

(store intermediate results, registers, pipelining, cycles, and etc.)

Combinational logic

adders, subtractors, logic gates, Look-Up tables, and etc.

The proposed approach:

- based on combinational logic;
- does not store intermediate results;
- uses adders and encoders, which represent systems of Boolean functions;
- scalable, i.e. the proposed architectures are suitable for arbitrary integer values, both for dividends and divisors, and output both the quotient and the residue

Hardware division: basic principles

1) $$2^q \cdot x = (x \underbrace{00\ldots0}_{q})$$

$$2^3 \cdot 9 = 2^3 \cdot (1001) = (1001000) = 72$$

2) $$X = (x_n, x_{n-1}, \ldots, x_1) =$$

$$\left(\underbrace{x_n x_{n-1} \ldots x_{n-k+1}}_{X_s} \ldots \underbrace{x_k x_{k-1} \ldots x_1}_{X_1}\right) =$$

$$(X_s, X_{s-1}, \ldots, X_1) =$$

$$X_1 + 2^k \cdot X_2 + 2^{2 \cdot k} \cdot X_3 + \cdots + 2^{(s-1) \cdot k} \cdot X_s$$

$$(100100101010) = \left(\underbrace{1001}_{X_3}\ \underbrace{0010}_{X_2}\ \underbrace{1010}_{X_1}\right) = X_1 + 2^4 \cdot X_2 + 2^8 \cdot X_3$$

Intermediate quotients and residues

$$X = (x_n, x_{n-1}, \dots, x_1) =$$
$$(X_s, X_{s-1}, \dots, X_1) =$$
$$X_1 + \underbrace{2^k \cdot X_2}_{X_{SC_2}} + \underbrace{2^{2\cdot k} \cdot X_3}_{X_{SC_3}} + \cdots + \underbrace{2^{(s-1)\cdot k} \cdot X_s}_{X_{SC_s}}$$

k – bit-range of the divisor *d*

QUOTIENTs RESIDUEs

$$\frac{X_{SC_2}}{d} = \{Q_2, R_2\}, \frac{X_{SC_3}}{d} = \{Q_3, R_3\}, \dots, \frac{X_{SC_s}}{d} = \{Q_s, R_s\}$$

$$X = (100100101010) \qquad d = 11$$

$$\mathrm{X} = (1010) + 2^4 \cdot (0010) + 2^8 \cdot (1001) = 10 + 2^4 \cdot 2 + 2^8 \cdot 9$$

$$\frac{X_{sc_2}}{11} = \frac{2^4 \cdot 2}{11} = \{2,10\} \qquad \frac{X_{sc_3}}{11} = \frac{2^8 \cdot 9}{11} = \{209,5\}$$

Quotients and Residues

$$\frac{X_1 + R_2 + R_3 + \cdots + R_s}{d} = \{QR, R\}$$

$$Q = Q_2 + Q_3 + \cdots + Q_s + QR$$

$$X = (100100101010) \qquad d = 11$$

$$X = (1010) + 2^4 \cdot (0010) + 2^8 \cdot (0010) = 10 + 2^4 \cdot 2 + 2^8 \cdot 9$$

$$\frac{2^4 \cdot 2}{11} = \{2,10\} = \{Q_2, R_2\} \qquad \frac{2^8 \cdot 9}{11} = \{209,5\} = \{Q_3, R_3\}$$

$$\frac{10 + 10 + 5}{11} = \{2,3\} = \{QR, R\}$$

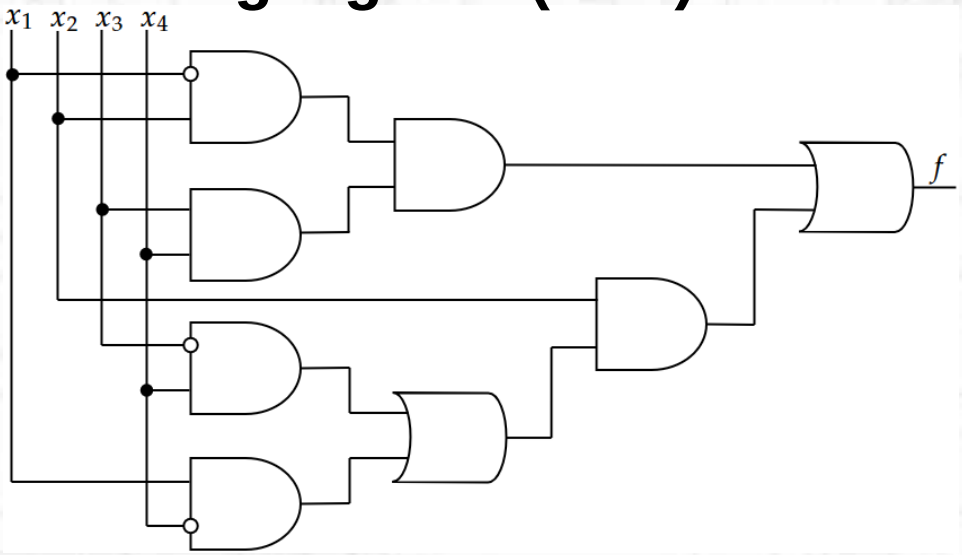
$$Q = 2 + 209 + 2 = 213$$

Boolean functions: truth table

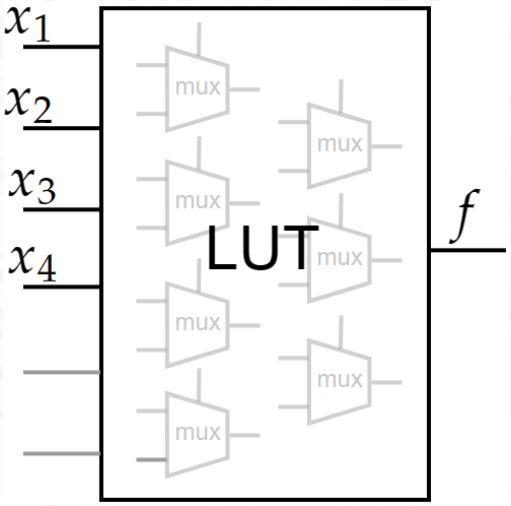
$$f = \overline{x_2}(\overline{x_3}x_4 \vee x_1\overline{x_4}) \vee \overline{x_1}x_2x_3x_4$$

x_1	x_2	x_3	x_4	f
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0

logic gates (RTL)



6-input LUT FPGA



Boolean functions: truth table

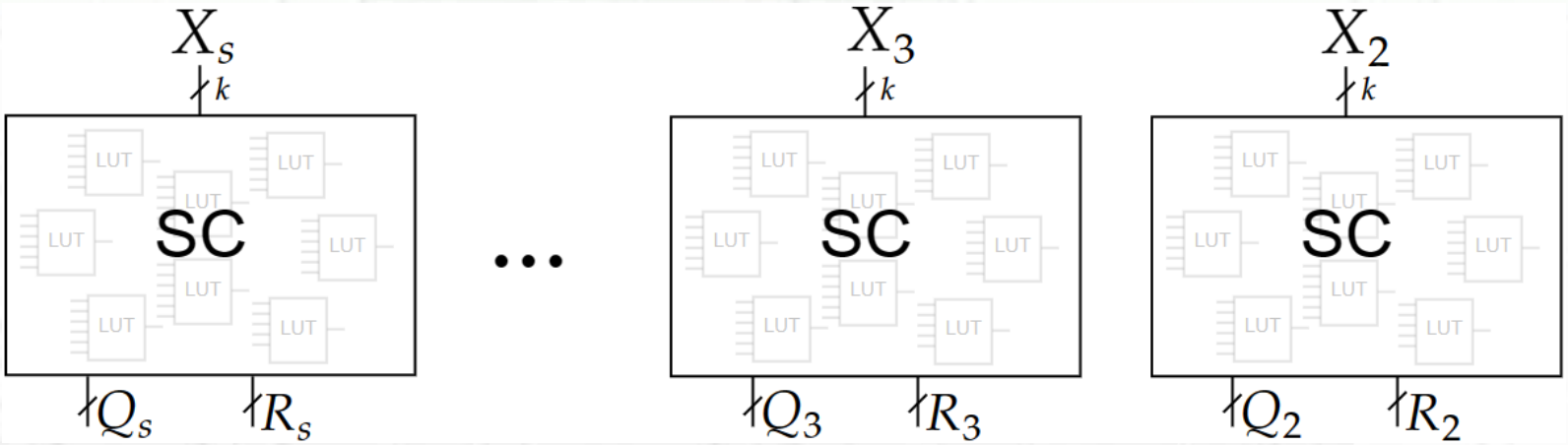
$$\frac{X_{SC_i}}{d} = \frac{2^i \cdot X_i}{d} = \{Q_i, R_i\}$$

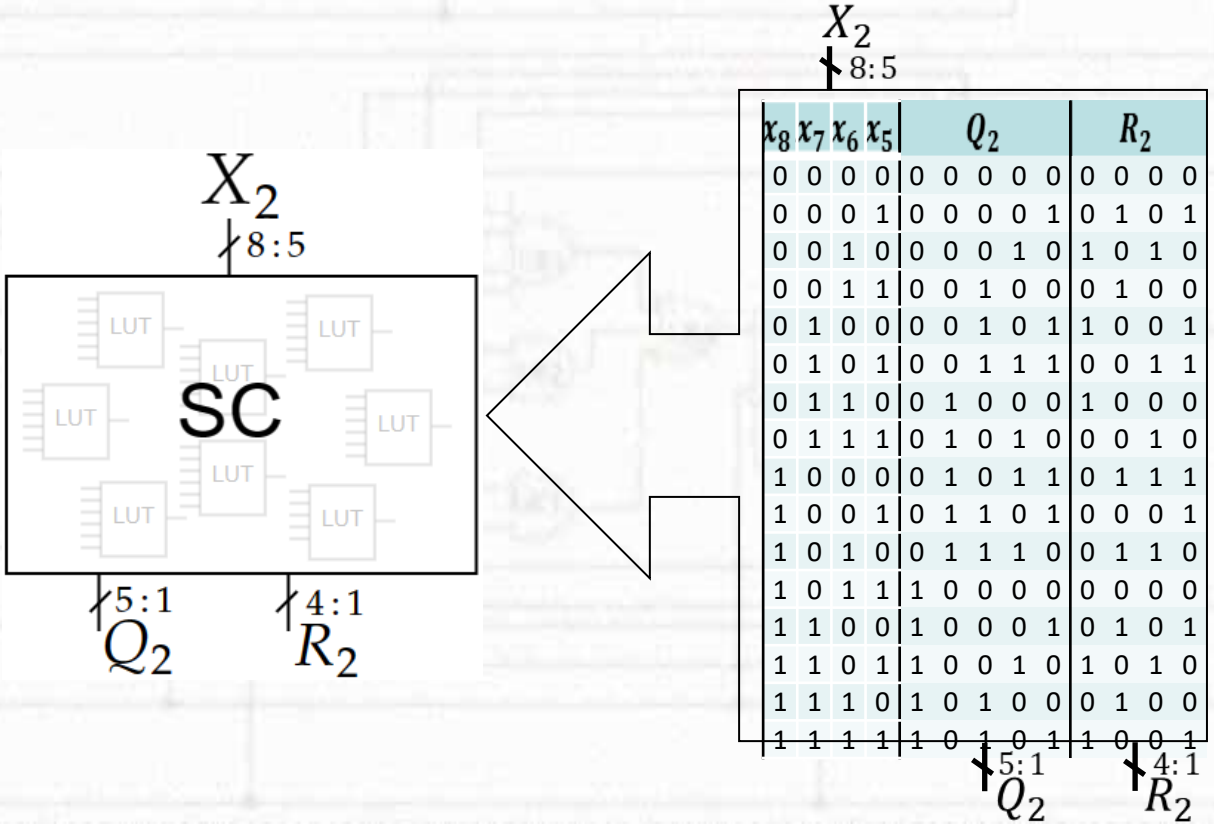
constant			$x_{i\cdot k+k}$	…	$x_{i\cdot k+2}$	$x_{i\cdot k+1}$	Q_i			R_i		
0	…	0	0	…	0	0	0	…	0	0	…	0
0	…	1	0	…	0	1	…	…	…	…	…	…
…	…	…	…	…	…	…	…	…	…	…	…	…
1	…	1	1	…	1	1	…	…	…	…	…	…

constant = 16					x_8	x_7	x_6	x_5	Q_2					R_2			
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1	0	1
1	0	0	0	0	0	0	1	0	0	0	0	1	0	1	0	1	0
1	0	0	0	0	0	0	1	1	0	0	1	0	0	0	1	0	0
1	0	0	0	0	0	1	0	0	0	0	1	0	1	1	0	0	1
1	0	0	0	0	0	1	0	1	0	0	1	1	1	0	0	1	1
1	0	0	0	0	0	1	1	0	0	1	0	0	0	1	0	0	0
1	0	0	0	0	0	1	1	1	0	1	0	1	0	0	0	1	0
1	0	0	0	0	1	0	0	0	0	1	0	1	1	0	1	1	1
1	0	0	0	0	1	0	0	1	0	1	1	0	1	0	0	0	1
1	0	0	0	0	1	0	1	0	0	1	1	1	0	0	1	1	0
1	0	0	0	0	1	0	1	1	0	1	1	1	1	0	1	1	0
1	0	0	0	0	1	1	0	0	1	0	0	0	0	0	0	0	0
1	0	0	0	0	1	1	0	1	1	0	0	0	1	0	1	0	1
1	0	0	0	0	1	1	1	0	1	0	0	1	0	1	0	1	0
1	0	0	0	0	1	1	1	1	1	0	1	0	1	1	0	0	1

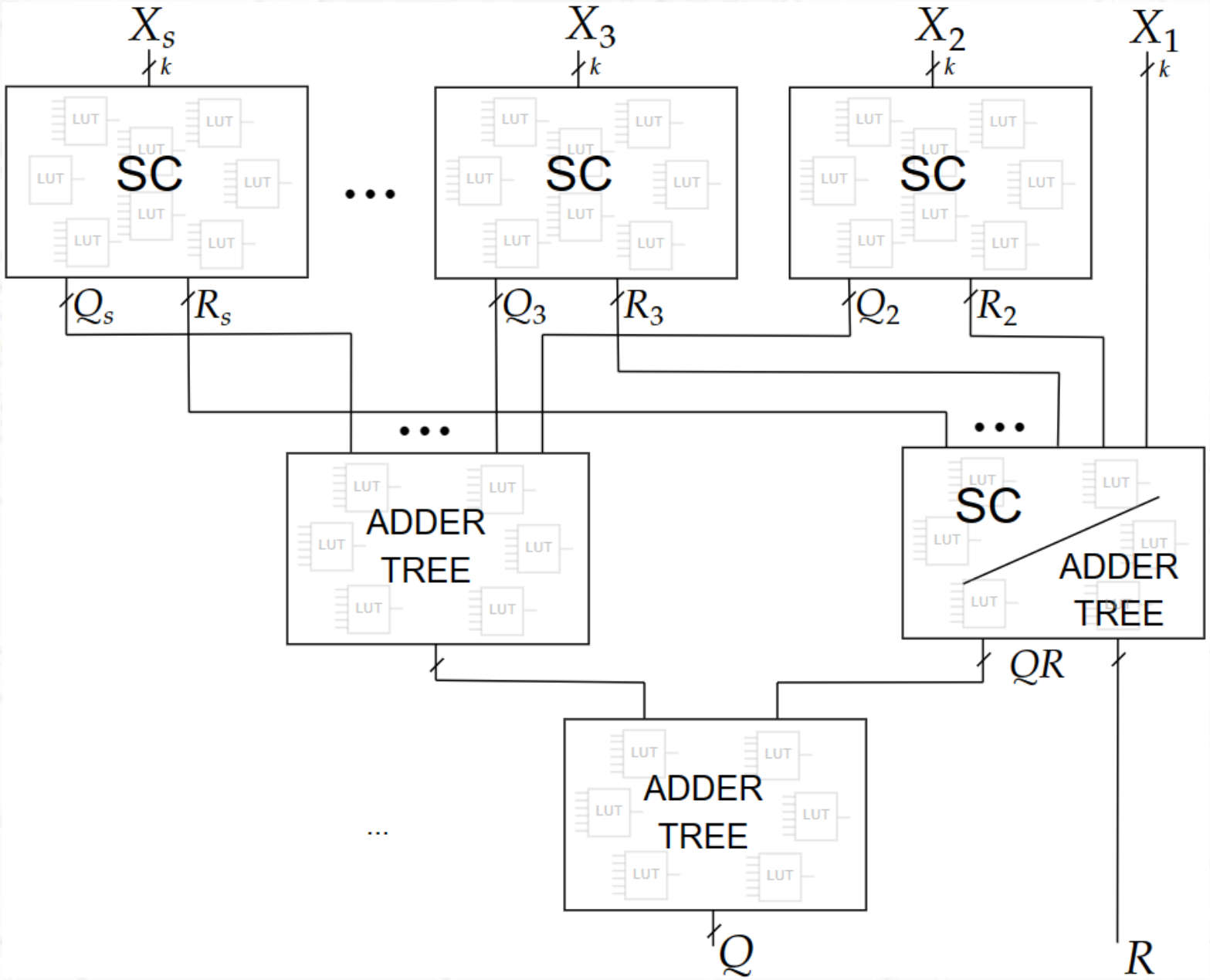
$$\frac{X_{sc_2}}{11} = \frac{2^4 \cdot X_2}{11} = \{Q_2, R_2\}$$

$$\frac{2^4 \cdot 2}{11} = \{2, 10\}$$

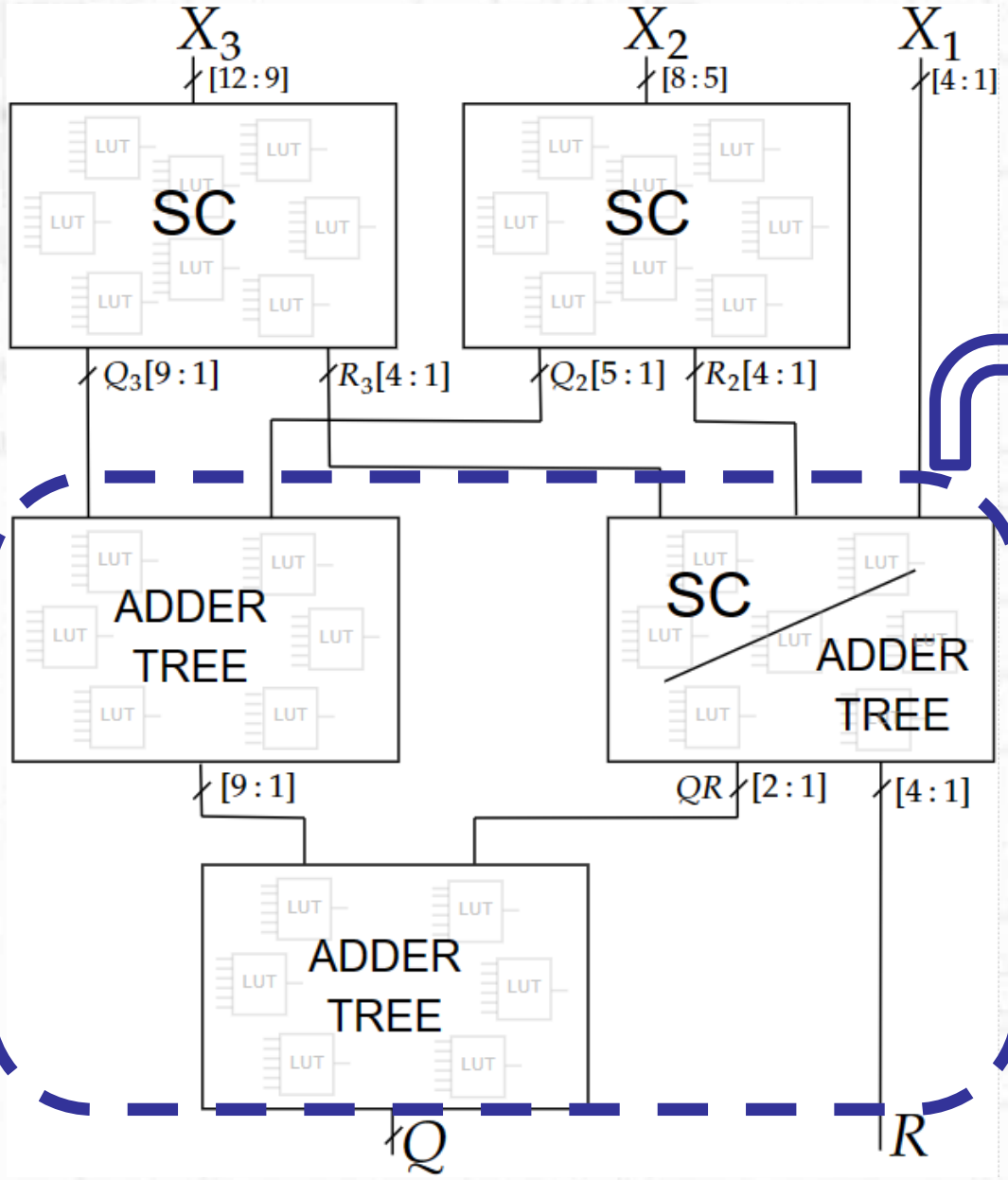

X_s
k
SC
LUT
Q_s
R_s
...
X_3
k
SC
LUT
Q_3
R_3
X_2
k
SC
LUT
Q_2
R_2


X_2
$8:5$
SC
LUT
$5:1$
Q_2
$4:1$
R_2
X_2
$8:5$
x_8 x_7 x_6 x_5 | Q_2 | R_2
0 0 0 0 | 0 0 0 0 0 | 0 0 0 0
0 0 0 1 | 0 0 0 0 1 | 0 1 0 1
0 0 1 0 | 0 0 0 1 0 | 1 0 1 0
0 0 1 1 | 0 0 1 0 0 | 0 1 0 0
0 1 0 0 | 0 0 1 0 1 | 1 0 0 1
0 1 0 1 | 0 0 1 1 1 | 0 0 1 1
0 1 1 0 | 0 1 0 0 0 | 1 0 0 0
0 1 1 1 | 0 1 0 1 0 | 0 0 1 0
1 0 0 0 | 0 1 0 1 1 | 0 1 1 1
1 0 0 1 | 0 1 1 0 1 | 0 0 0 1
1 0 1 0 | 0 1 1 1 0 | 0 1 1 0
1 0 1 1 | 1 0 0 0 0 | 0 0 0 0
1 1 0 0 | 1 0 0 0 1 | 0 1 0 1
1 1 0 1 | 1 0 0 1 0 | 1 0 1 0
1 1 1 0 | 1 0 1 0 0 | 0 1 0 0
1 1 1 1 | 1 0 1 0 1 | 1 0 0 1
$5:1$
Q_2
$4:1$
R_2

Divider architecture

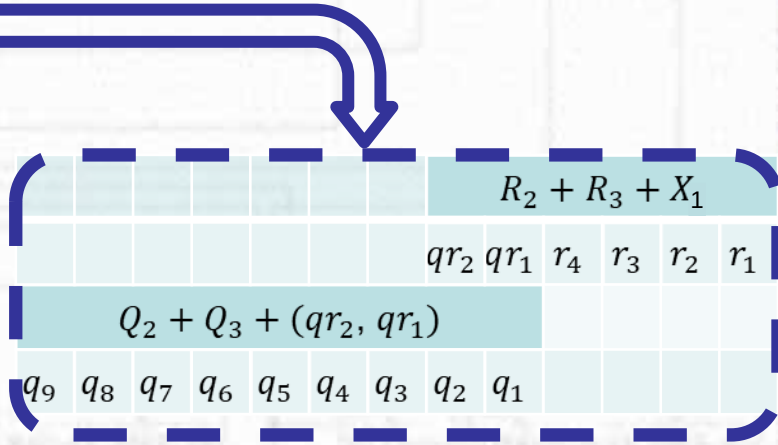


Divider architecture



X is 12-bit number

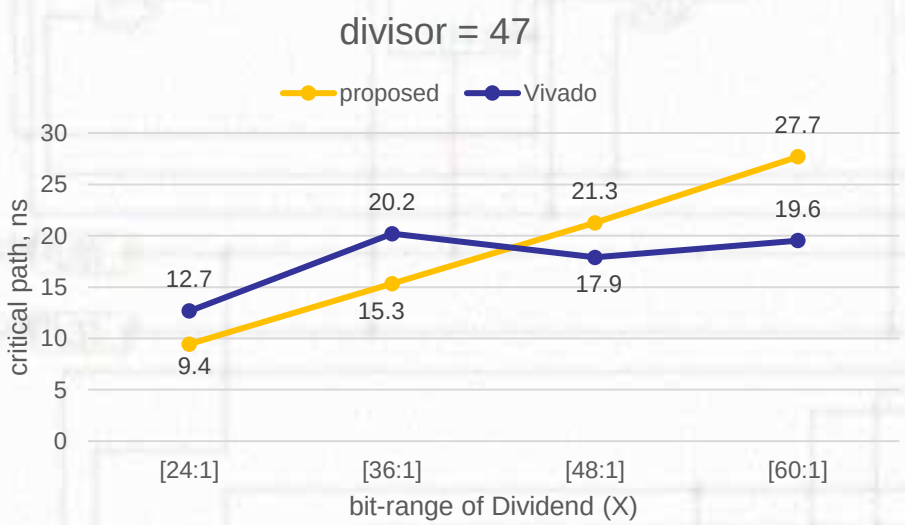
divisor $d = 11$

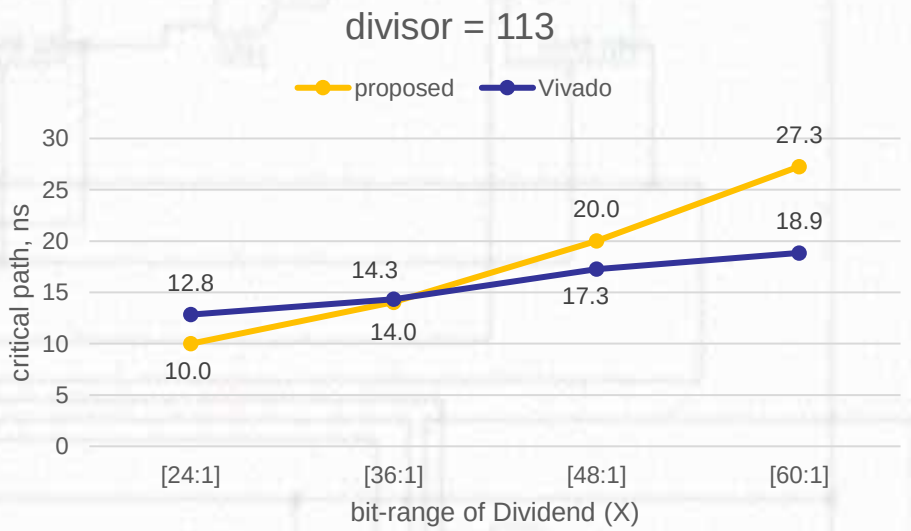


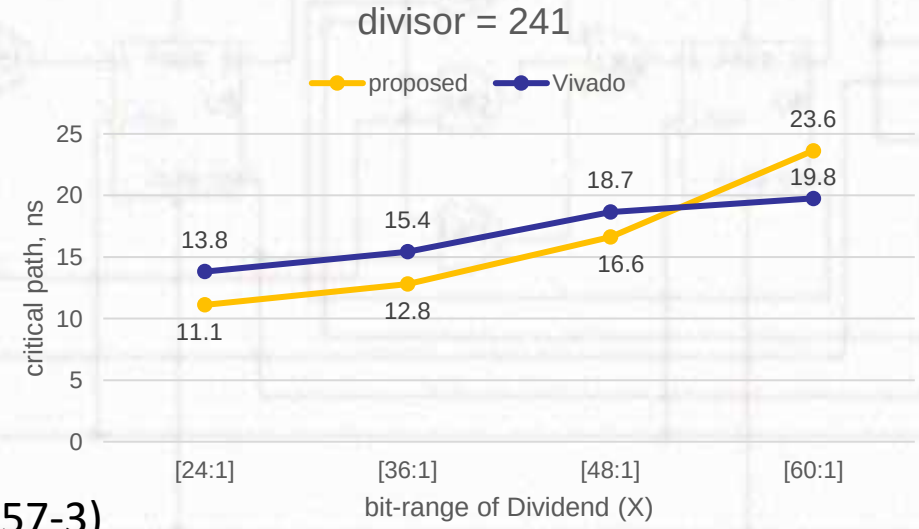
$QR = (qr_1, qr_1)$

Experiments*: approach vs. Vivado

quotient (*Q*) and residue (*R*) critical path, ns



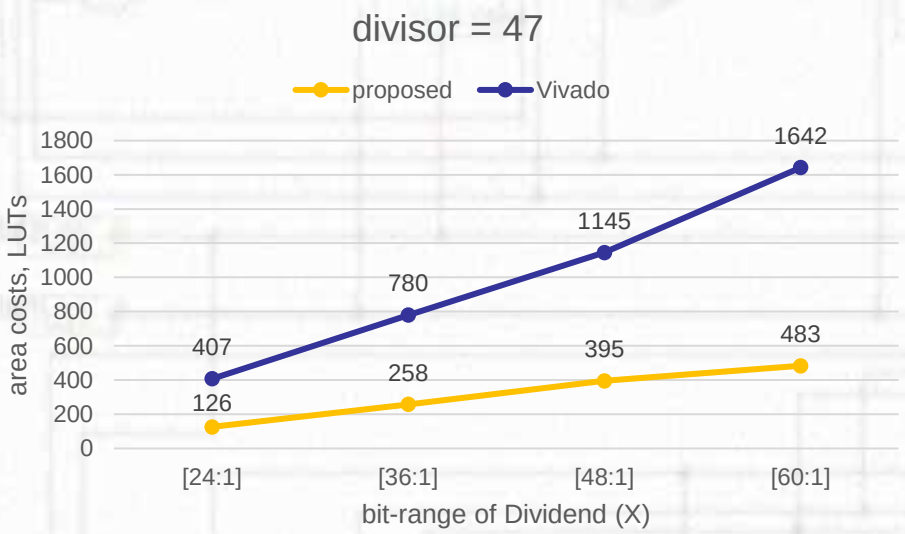


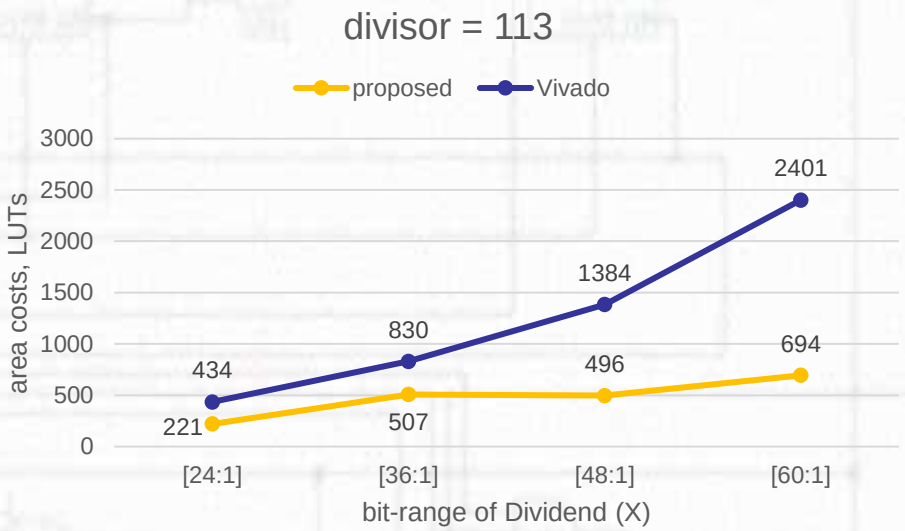


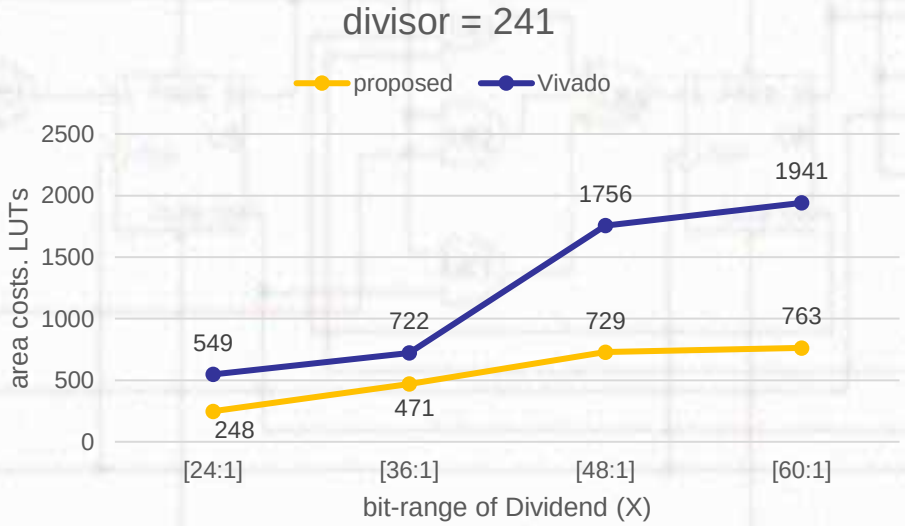
* Virtex-7 (xc7v585tffg1157-3)

Experiments*: approach vs. Vivado

quotient (*Q*) and residue (*R*) area costs, LUTs





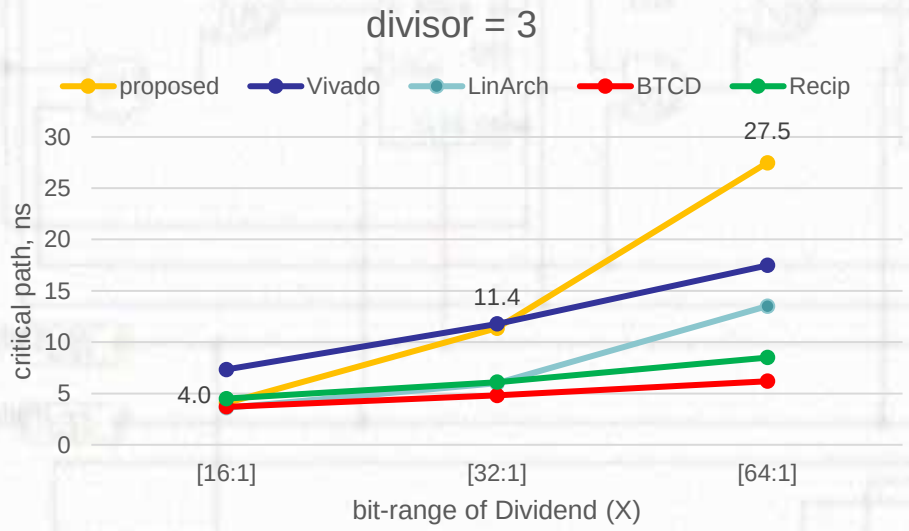


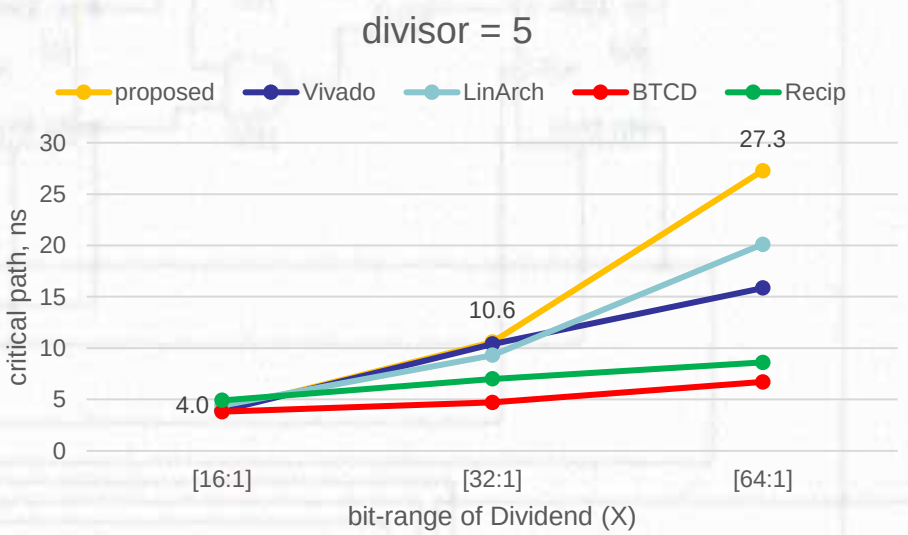
* Virtex-7 (xc7v585tffg1157-3)

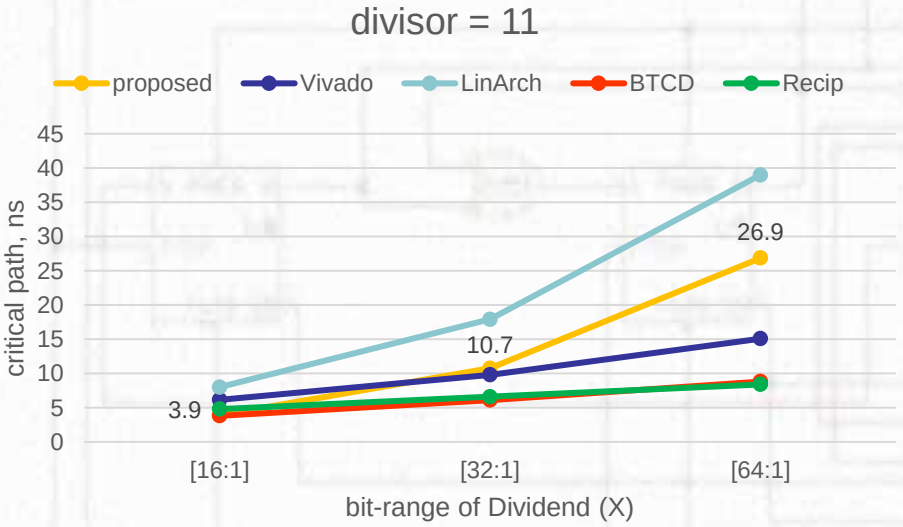
Experiments*: approach vs. Vivado vs. analogues^

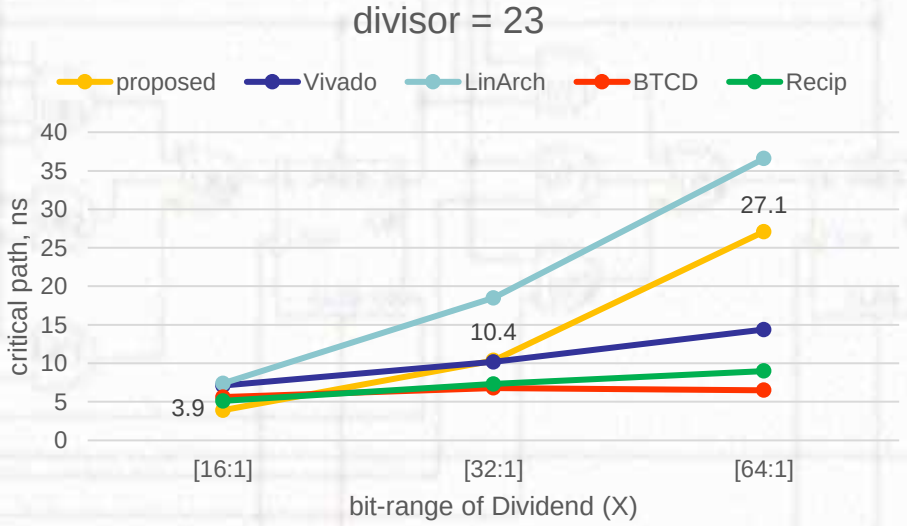
quotient (*Q*) and residue (*R*)

critical path, ns









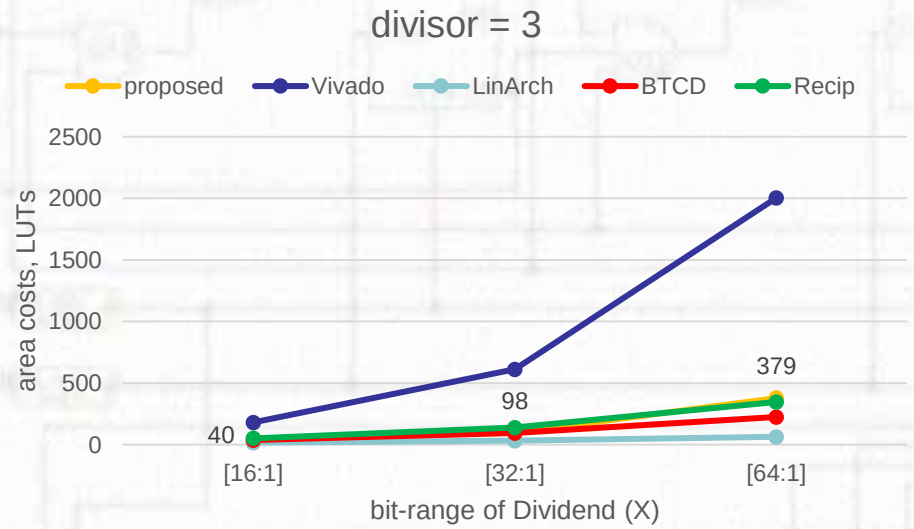
* Kintex-7

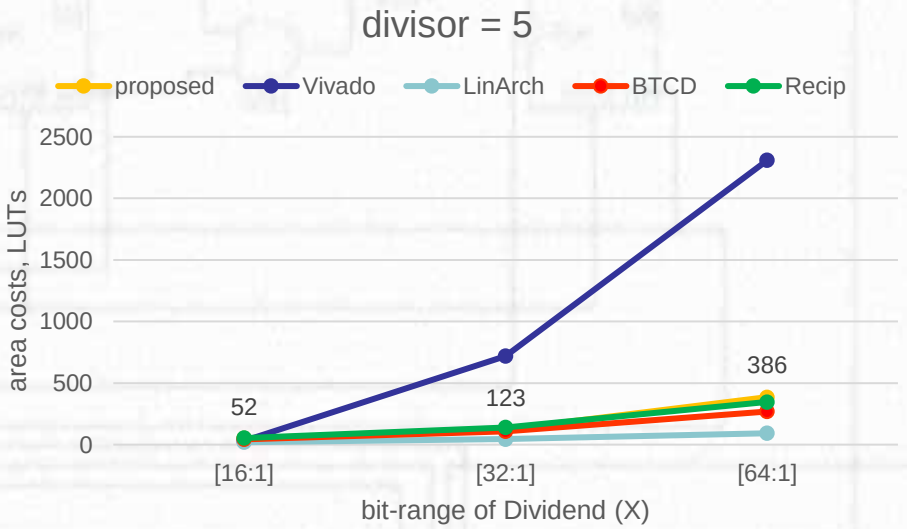
^ H. F. Ugurdag, F. de Dinechin, Y. S. Gener, S. Goren, L.-S. Didier,
"Hardware Division by Small Integer Constants", IEEE Transactions on Computers, Vol. 66, No. 12, Dec. 2017.

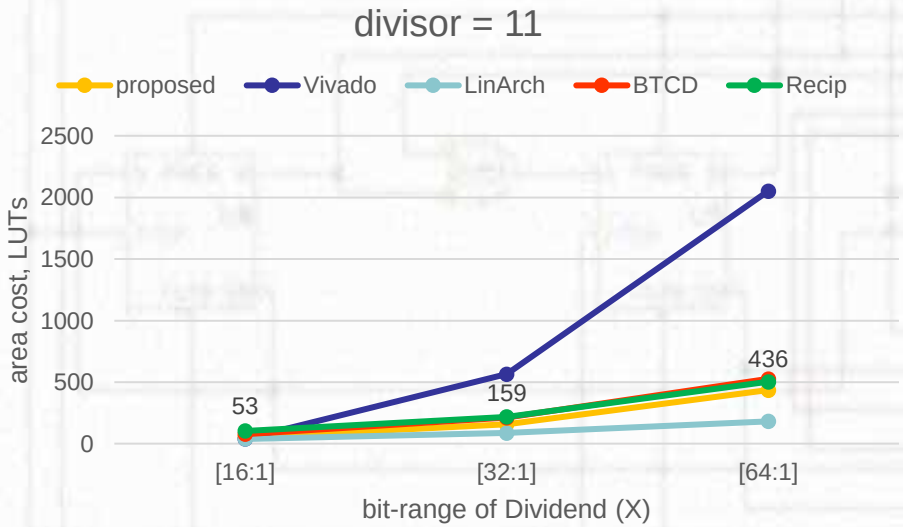
Experiments*: approach vs. Vivado vs. analogues^

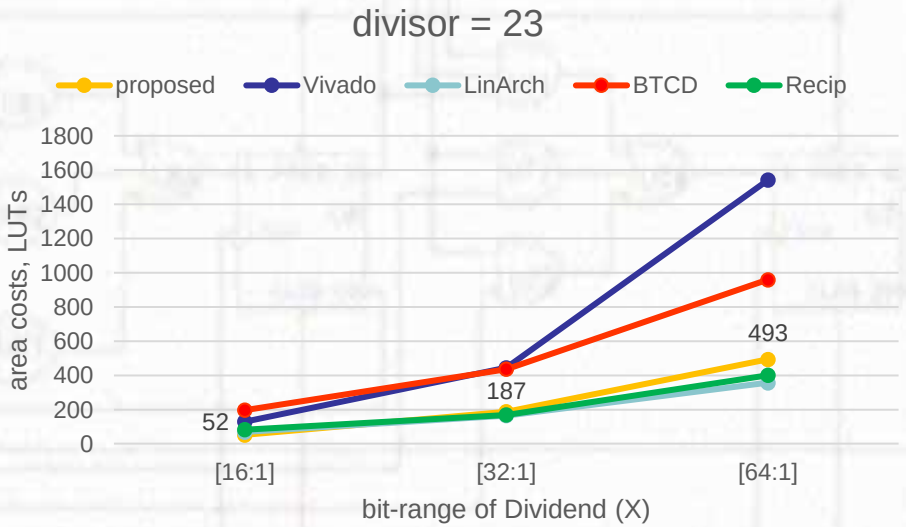
quotient (*Q*) and residue (*R*)

area costs, LUTs







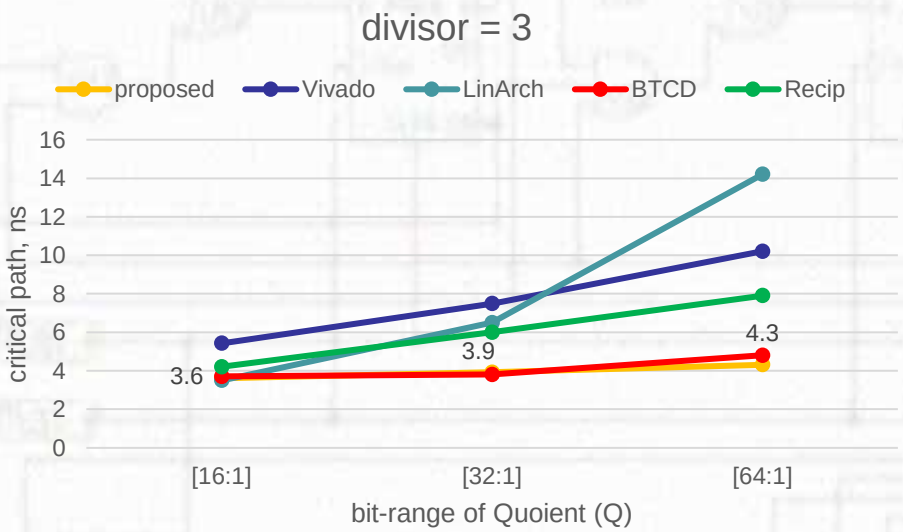


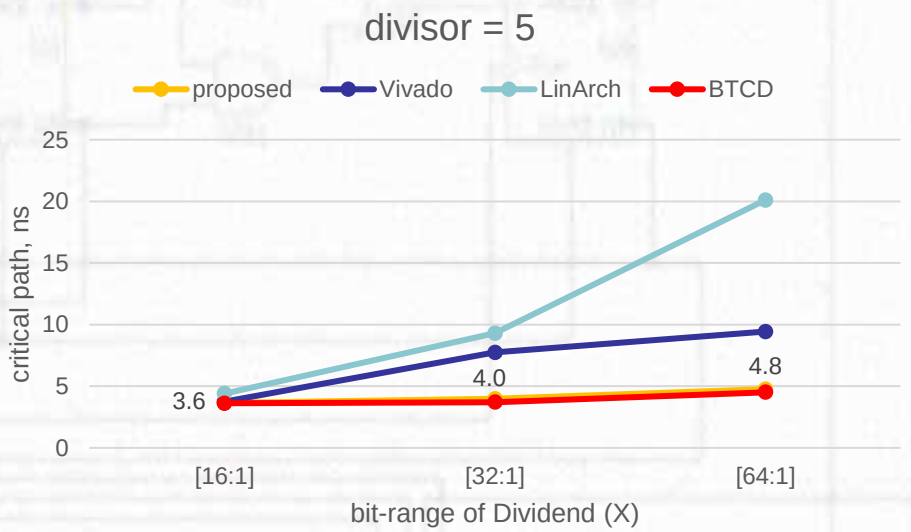
* Kintex-7

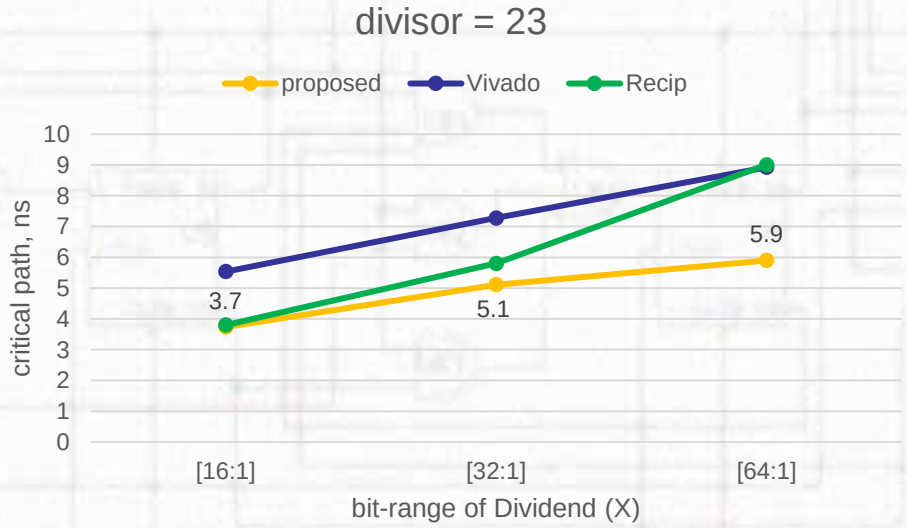
^ H. F. Ugurdag, F. de Dinechin, Y. S. Gener, S. Goren, L.-S. Didier,
"Hardware Division by Small Integer Constants", IEEE Transactions on Computers, Vol. 66, No. 12, Dec. 2017.

Experiments*: approach vs. Vivado vs. analogues^

residue (*R*) critical path, ns





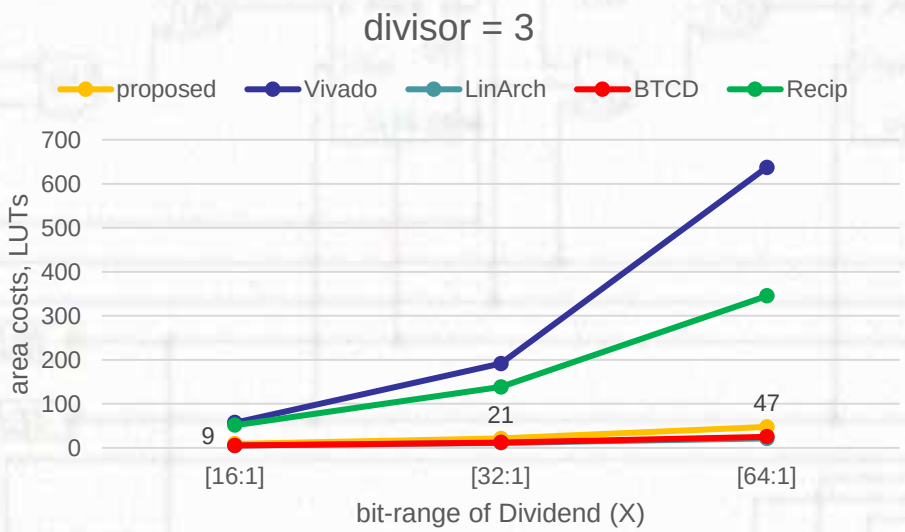


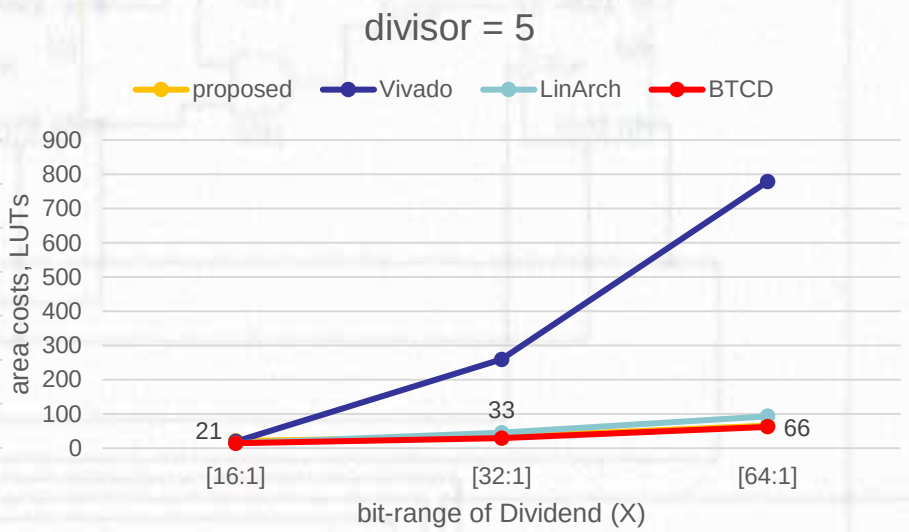
* Kintex-7

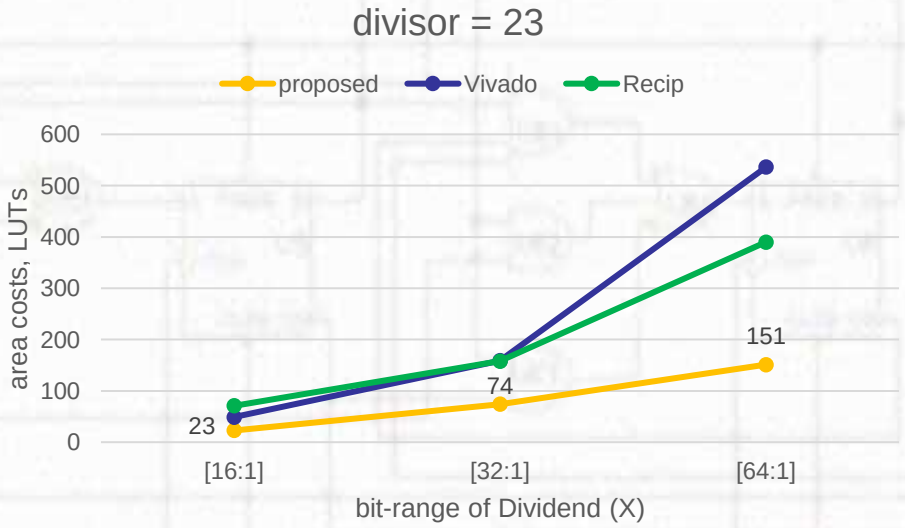
^ H. F. Ugurdag, F. de Dinechin, Y. S. Gener, S. Goren, L.-S. Didier,
"Hardware Division by Small Integer Constants", IEEE Transactions on Computers, Vol. 66, No. 12, Dec. 2017.

Experiments*: approach vs. Vivado vs. analogues^

residue (*R*) area cost, LUTs







* Kintex-7

^ H. F. Ugurdag, F. de Dinechin, Y. S. Gener, S. Goren, L.-S. Didier,
"Hardware Division by Small Integer Constants", IEEE Transactions on Computers, Vol. 66, No. 12, Dec. 2017.

CONCLUSIONS

- Combination logic, adders, and sub-coders, representing systems of Boolean functions
- Scalable for an arbitrary value of the dividend (X) and an arbitrary value of the divisor (d)
- The proposed division approach compared with embedded algorithms in Xilinx
  - shows a reduction on the number of LUTs up to 3.5x for the considered dividers d = 47, 113, and 241;
  - performance (i.e. critical path) improves by up to 25%
- Represents a trade-off in area costs and critical path comparing with SoA approaches

FURTHER RESEARCH

- Reduce area costs and critical path by optimizing the adder tree
- Extend the approach from division by a constant to the general division