Automatic Generation of Complete Polynomial Interpolation Design Space for Hardware Architectures

Bryce Orloski
Samuel Coward
Theo Drane

GFx Numerical Hardware Group

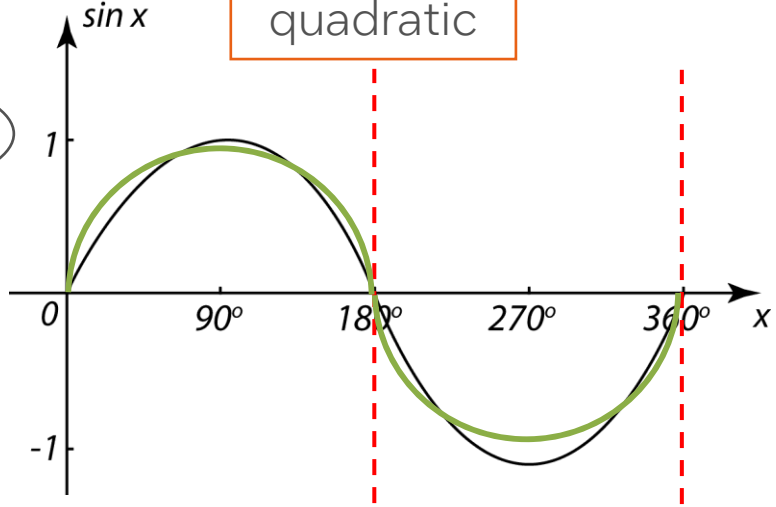
Thursday 19th January

Piecewise Polynomial Approximation

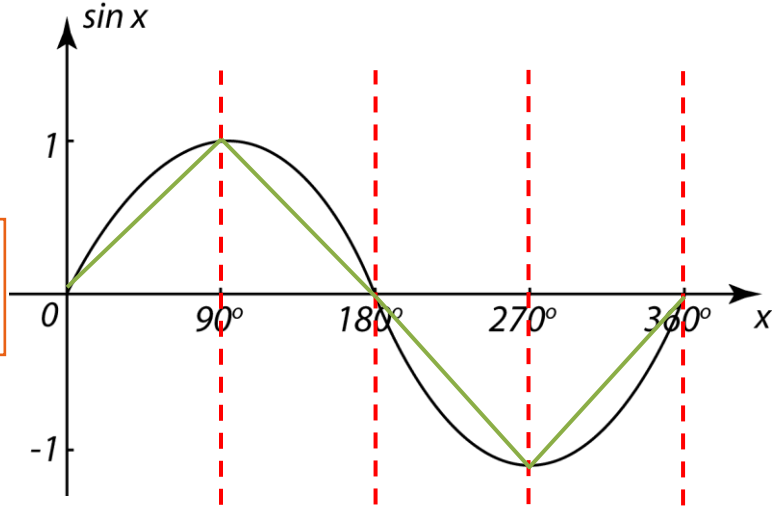
Math libraries
ML Activations
Graphics

How do I build HW for $\sin(x)$?

2 regions quadratic



4 regions linear



How many regions?

Linear or quadratic?

Equally sized regions?

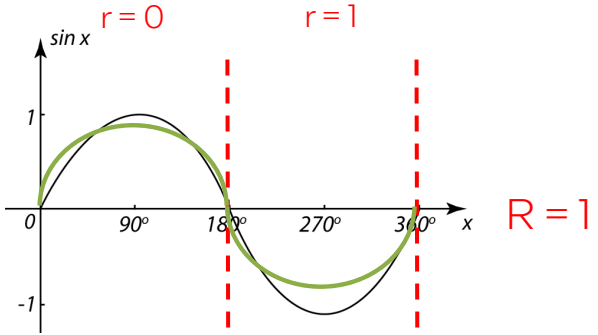
Parameterisable Hardware Implementation

Num Regions = 2^R

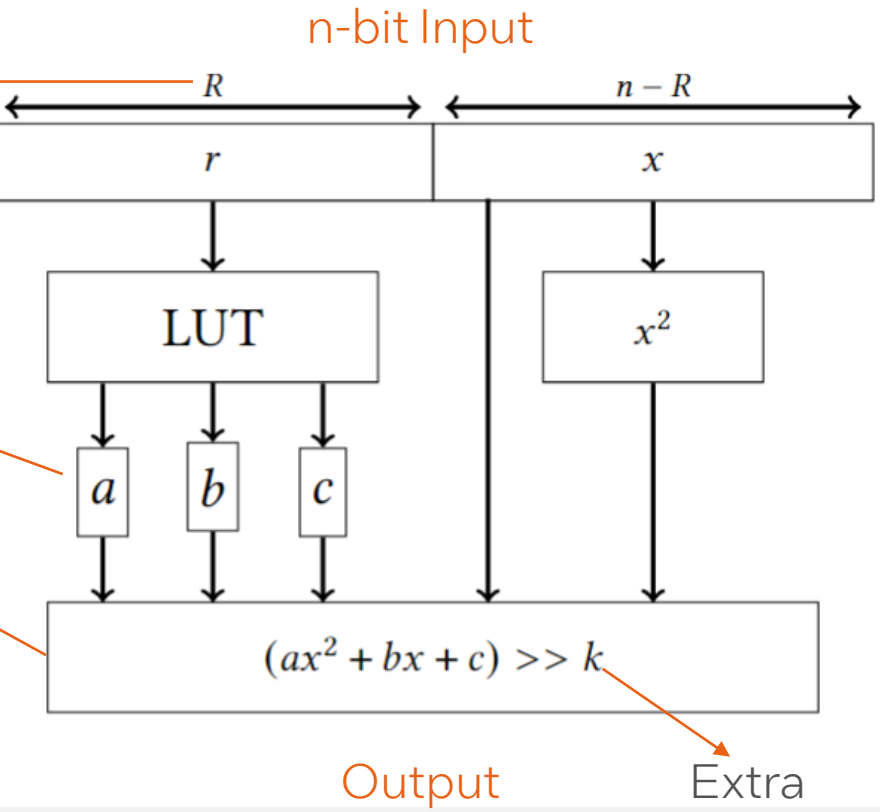
Determines region

Polynomial coefficients

Polynomial computed



R = 1



Problem Statement

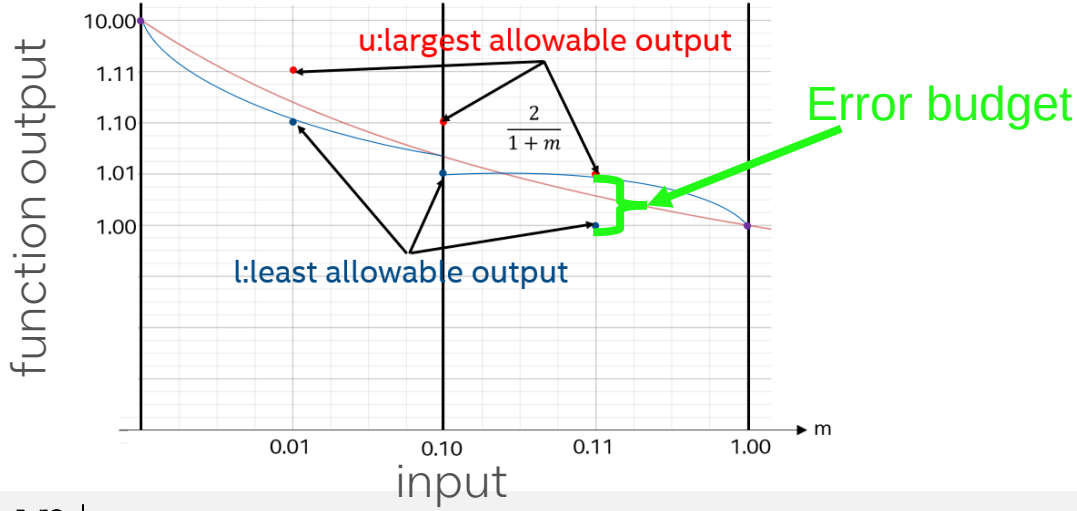
X_n = n-bit integers

Given $f: X_n \rightarrow \mathbb{R}$, *and upper/lower bound functions* $u, l: X_n \rightarrow X_m$, *encoding the acceptable approximation error:*

$$l(x) \leq f(x) \leq u(x)$$

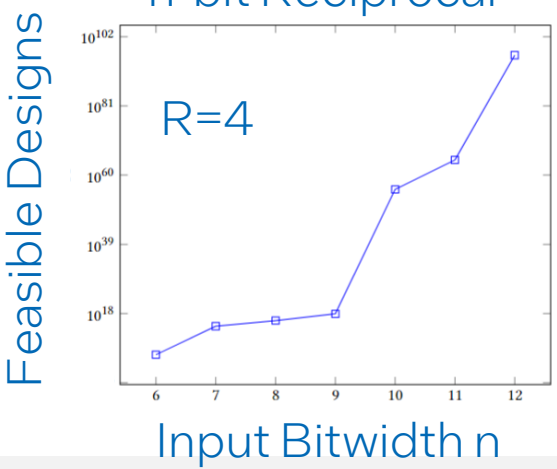
Find all piecewise polynomial HW implementations P *satisfying:*

$$l(x) \leq P(x) \leq u(x)$$



Why?

n-bit Reciprocal



Feasible Design Space – fixed R

$$M(t) = \max_{x<y, x+y=t} \frac{u(x)+1-l(y)}{x-y}$$

$$m(t) = \min_{x<y, x+y=t} \frac{u(y)+1-l(x)}{y-x}$$

For each region $\mathrm{r} = 0 \ldots 2^R - 1 \Rightarrow$ find a polynomial satisfying:

$$\forall x \in X_{n-R} \quad l_r(x) \le \left\lfloor \frac{ax^2+bx+c}{2^k} \right\rfloor \le u_r(x)$$

Re-arranging equations…

Hardware result

Determines existence

1. Find a and k:

$$\max_{t<s} \frac{M(s)-m(t)}{s-t} < \frac{a}{2^k} < \min_{t<s} \frac{m(s)-M(t)}{s-t}$$

2. Find b:

$$\max_{t} 2^k M(t) - at < b < \min_{t} 2^k m(t) - at$$

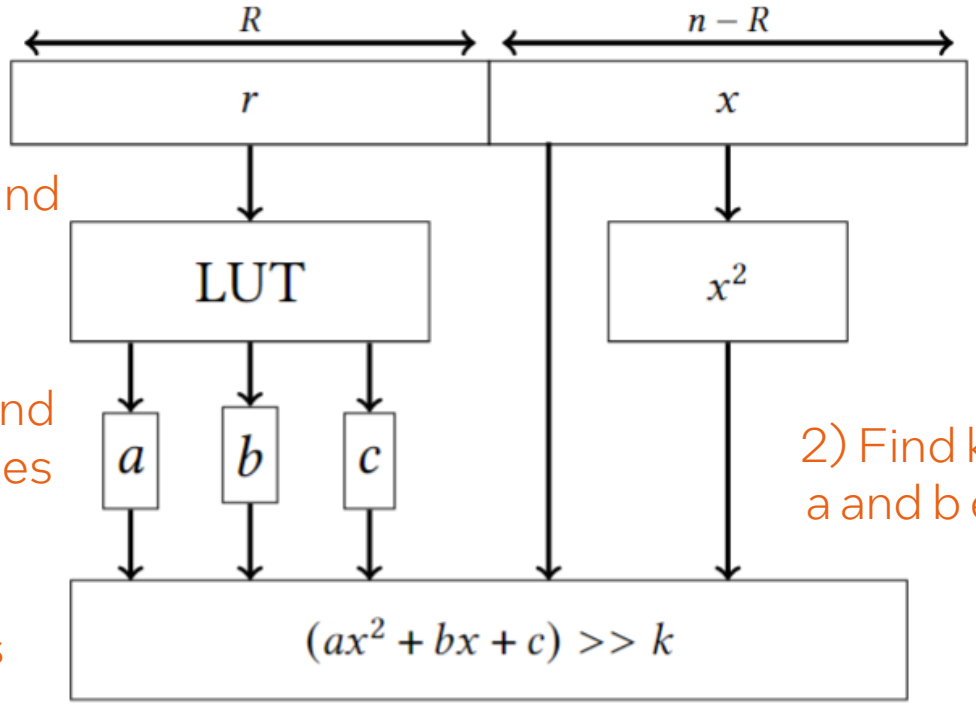
3. Find c:

$$\max_{x} 2^k l(x) - ax^2 - bx \le c < \min_{x} 2^k (u(x)+1) - ax^2 - bx$$

Complete Design Space Generation

Output dictionary:
$r \rightarrow a \rightarrow b \rightarrow c$

1) Calculate minimum possible R



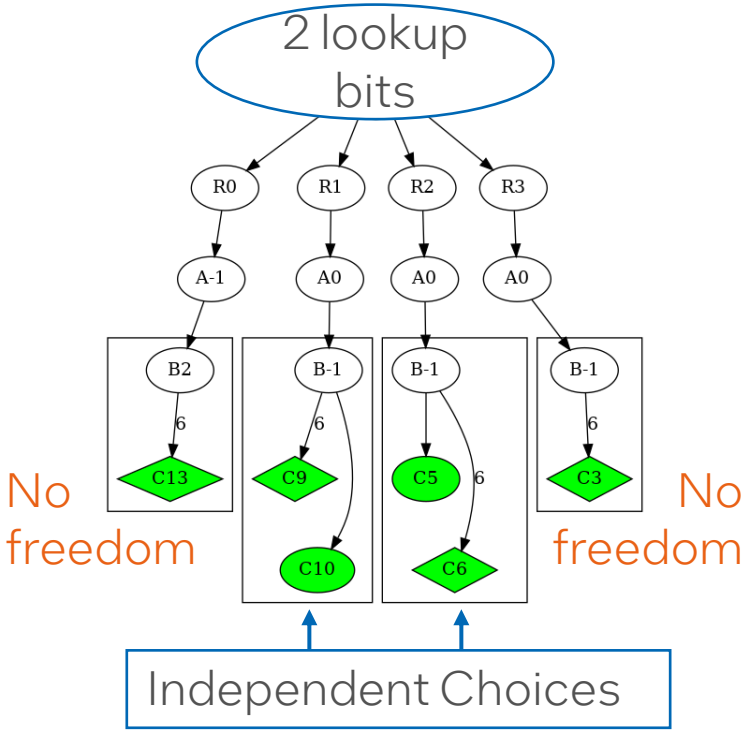
3) For each region find all possible a values

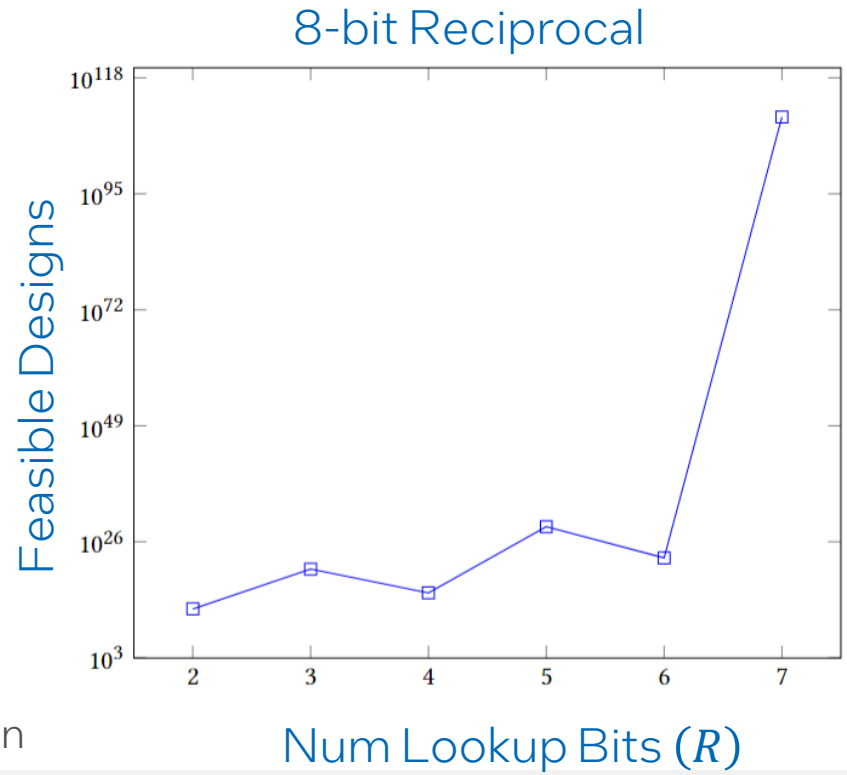
4) For each region and each a, find all b values

5) For each region, a & b find all c values

2) Find k such that feasible a and b exist for all regions

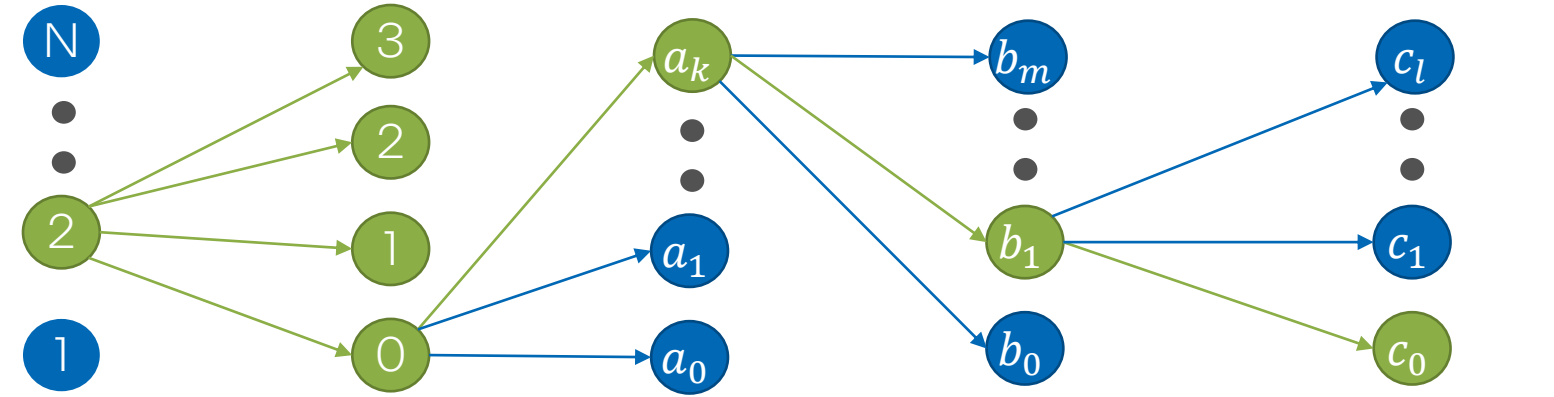
Design Space Evaluation





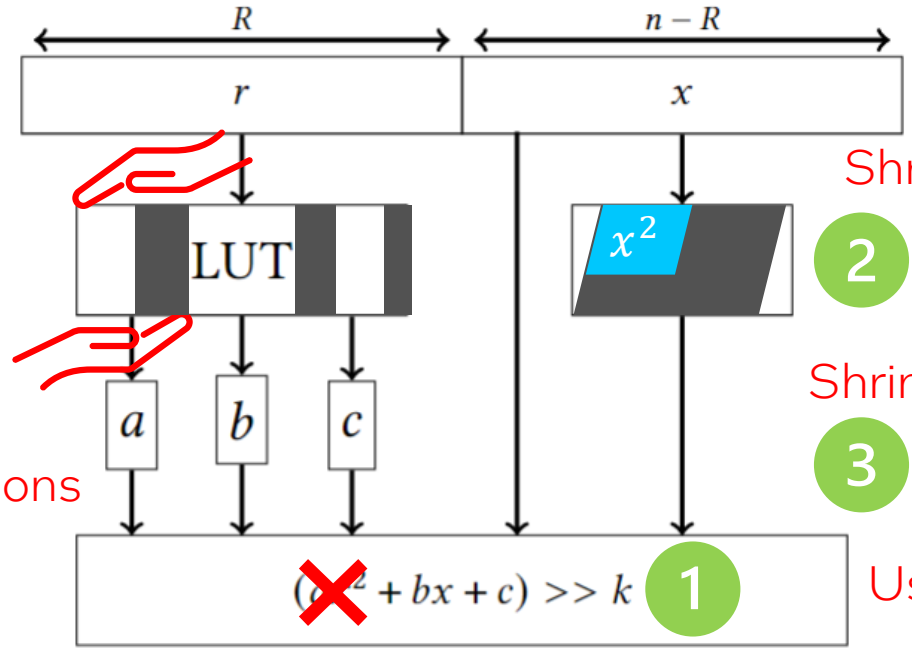
A valid design selects one leaf for each region

Design Space Exploration



Select parameters corresponding to delay optimal HW

Optimizations



Shrink precision of a, b or c — 4

Shrink input to the square — 2

Shrink input to the linear term — 3

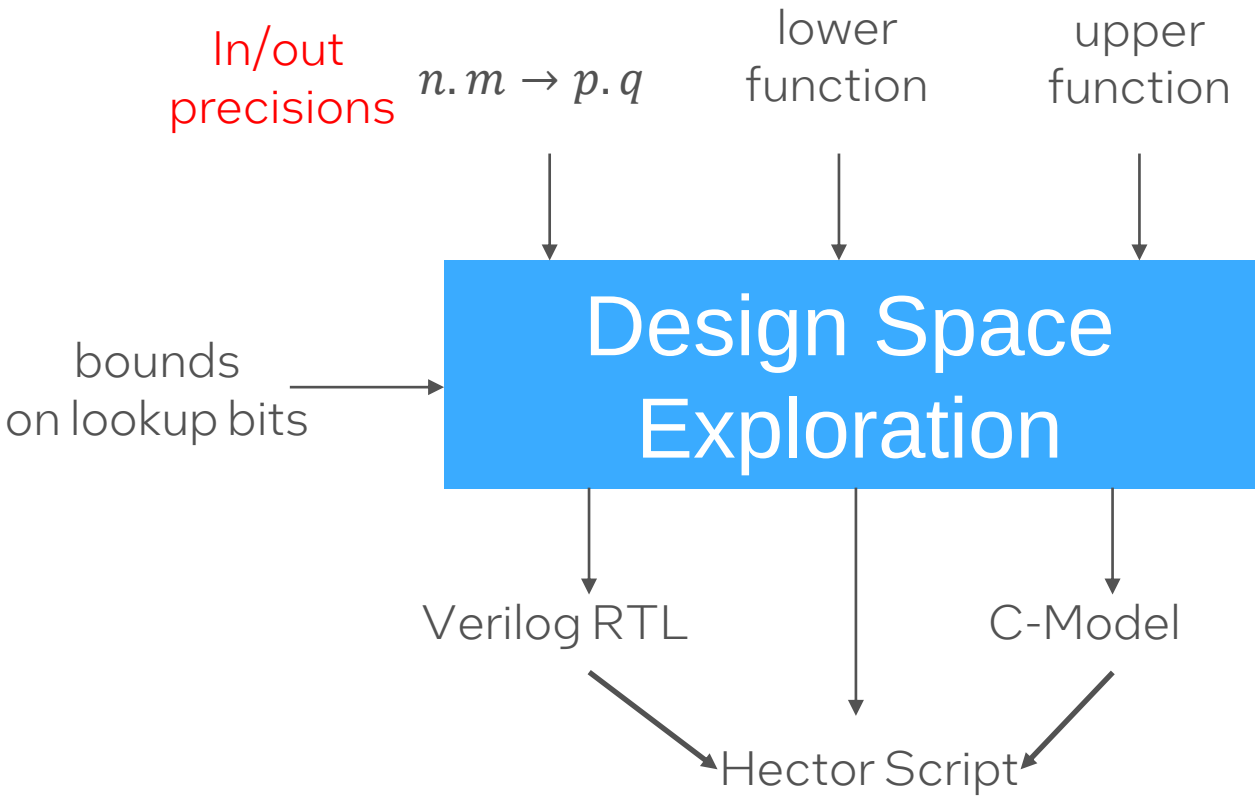
Merge adjacent regions using same poly

Use linear poly where possible — 1

Decision Procedure

```
for num_lookup_bits in 0..N:
    1. Minimize k – polynomial evaluation precision
    2. Maximize square input truncation
    3. Maximize linear input truncation
    4. Minimize a, then b, then c precisions
```

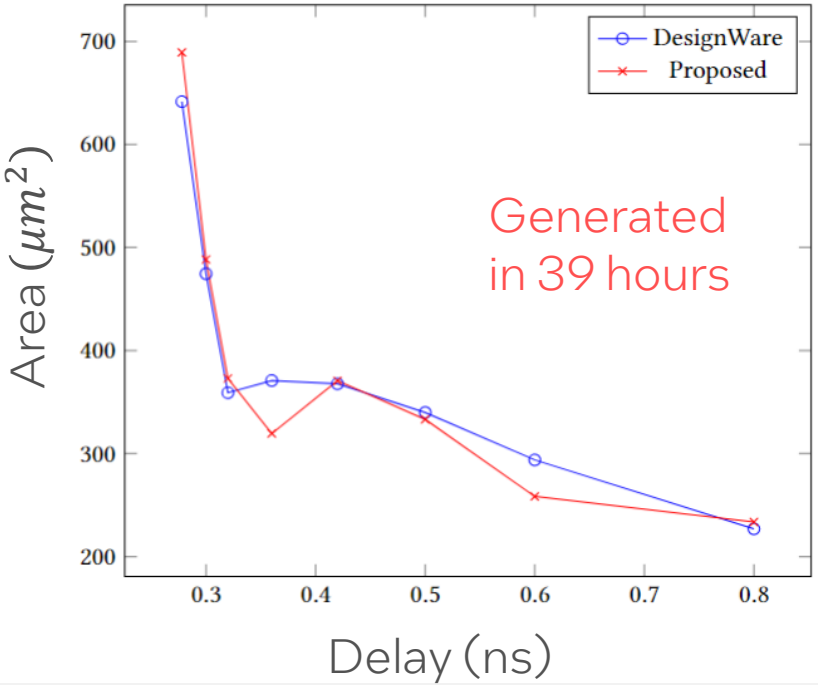
Tool Overview



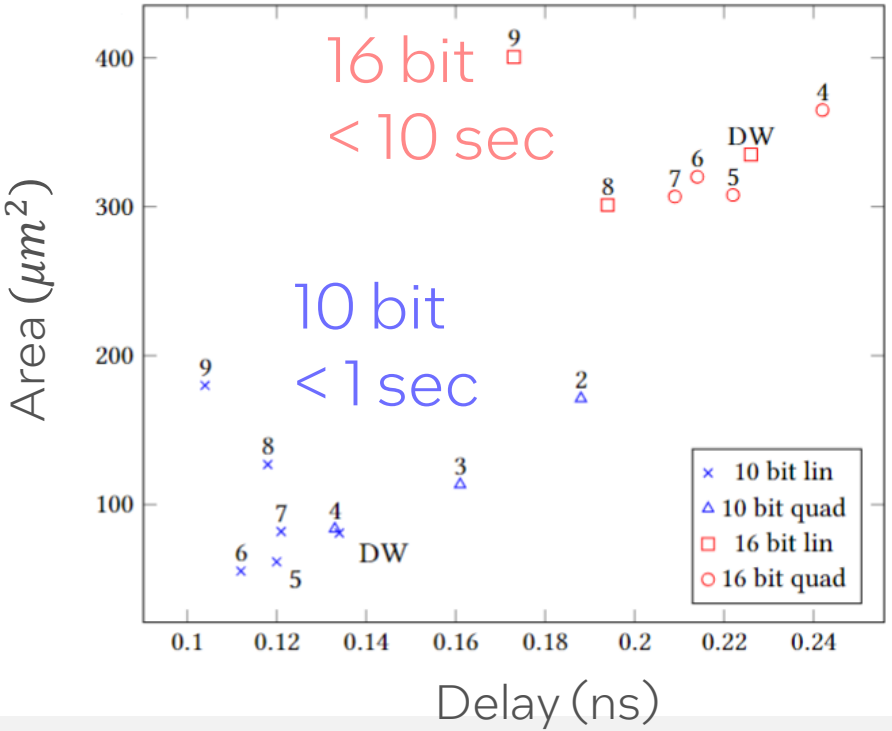
Results – 1 ulp reciprocal

Paper contains:
$\log_2$ and e^x designs

23 bit



10/16 bit



Competition

FloPoCo – arithmetic core generator for FPGAs

- Remez algorithm implemented in Sollya
- Runs in seconds

Function	Bitwidth	FloPoCo LUT	Proposed LUT
Recip	23	[10,18,26] = 54	[8,17,37] = 62
Log_2	16	[8,15,20] = 43	[4,14,20] = 38
Exp	10	[6,11,14] = 31	[2, 9,15] = 26

15% wider table

20% smaller a prec

12% smaller

16% smaller

[a prec, b prec, c prec] = total width

Conclusion

- Method to generate complete polynomial interpolation design space
- RTL generated for arbitrary precision elementary functions
- Easy re-targeting & exploration
- Decision procedure proposed

Future Work

- Improved design space exploration – optimal lookup bit selection
- Scalability concerns