Further Error-Correction Techniques

Or "How to be wrong but not *that* wrong"

Featuring:

- Realm Review
- Decreasing wrongness for less LUT space (ECALE method)
- Further error correction techniques.
- Matlab Guide
- (Potentially) Decreasing wrongness using (QSAM method)

↓

Absolute error

REALM

Low relative error & low area

(1% relative error with M = 16)

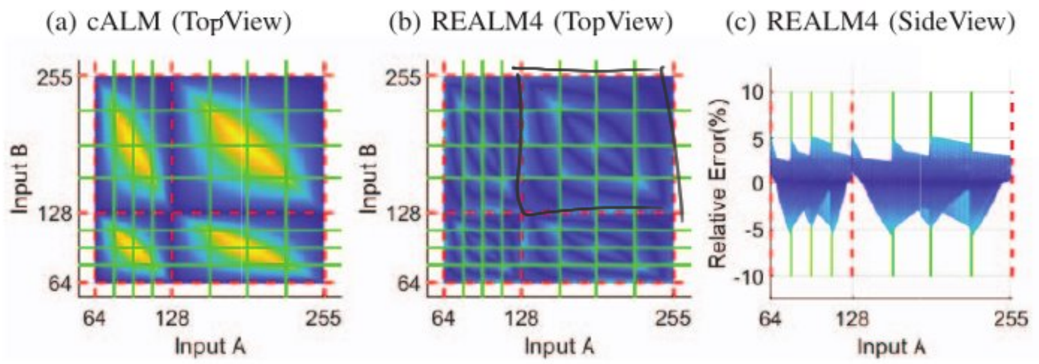
Short crit-path

Arbitrarily configurable Error at the cost of space

Error ∝ 1/M

Space ∝ M^2

M=4



Normal error profile

Each grid square shifted down…

…to be symmetric around zero

But at what cost?

For q = 8191:

Maximum product is 8191*8191 = 67 092 481

=> Need relative error of 0.003%

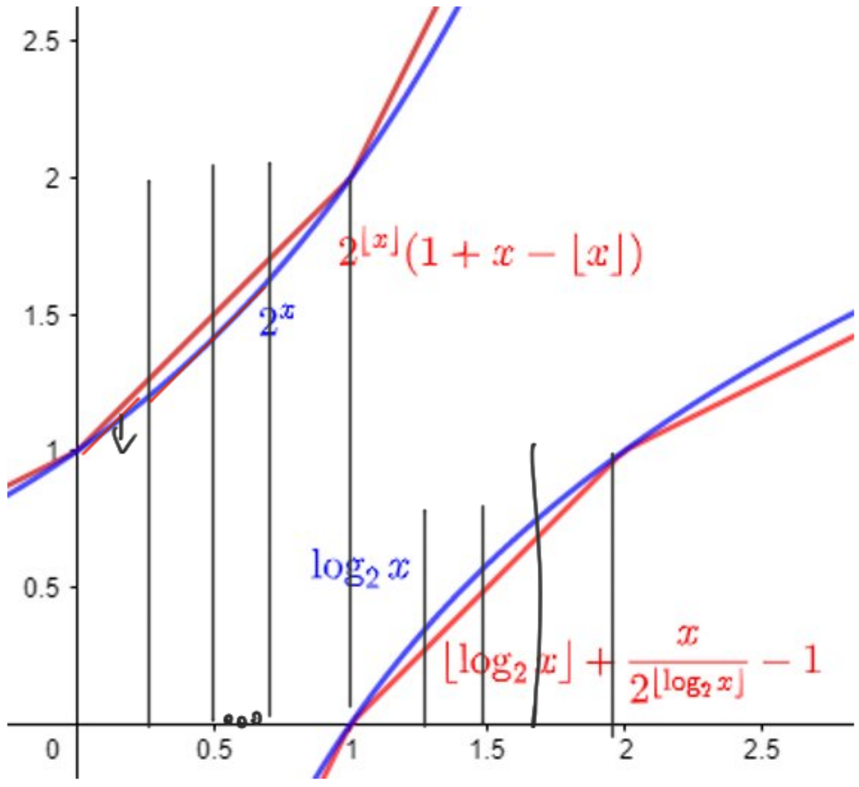
M=9600

=> ~9600^2 = 92 160 000 correction terms.

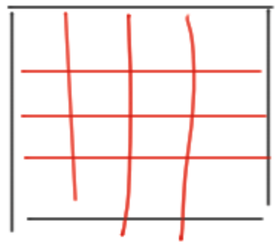
Can't fit in all block ram & distributed ram & LUT's combined

(Derivation Later for those interested)

Looking elsewhere…



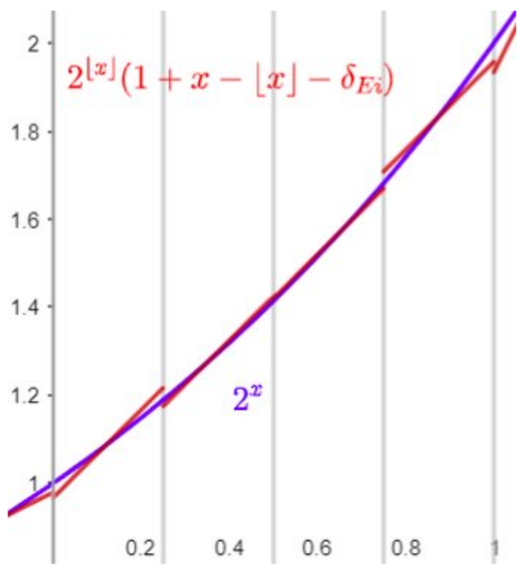
Bel



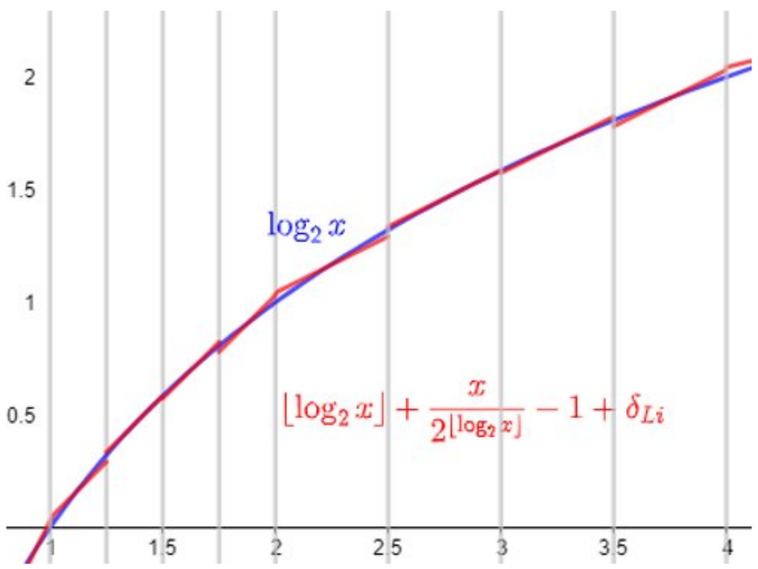
Error corrected Log and exponential (ECALE)

Approximate

$M_E = 4$



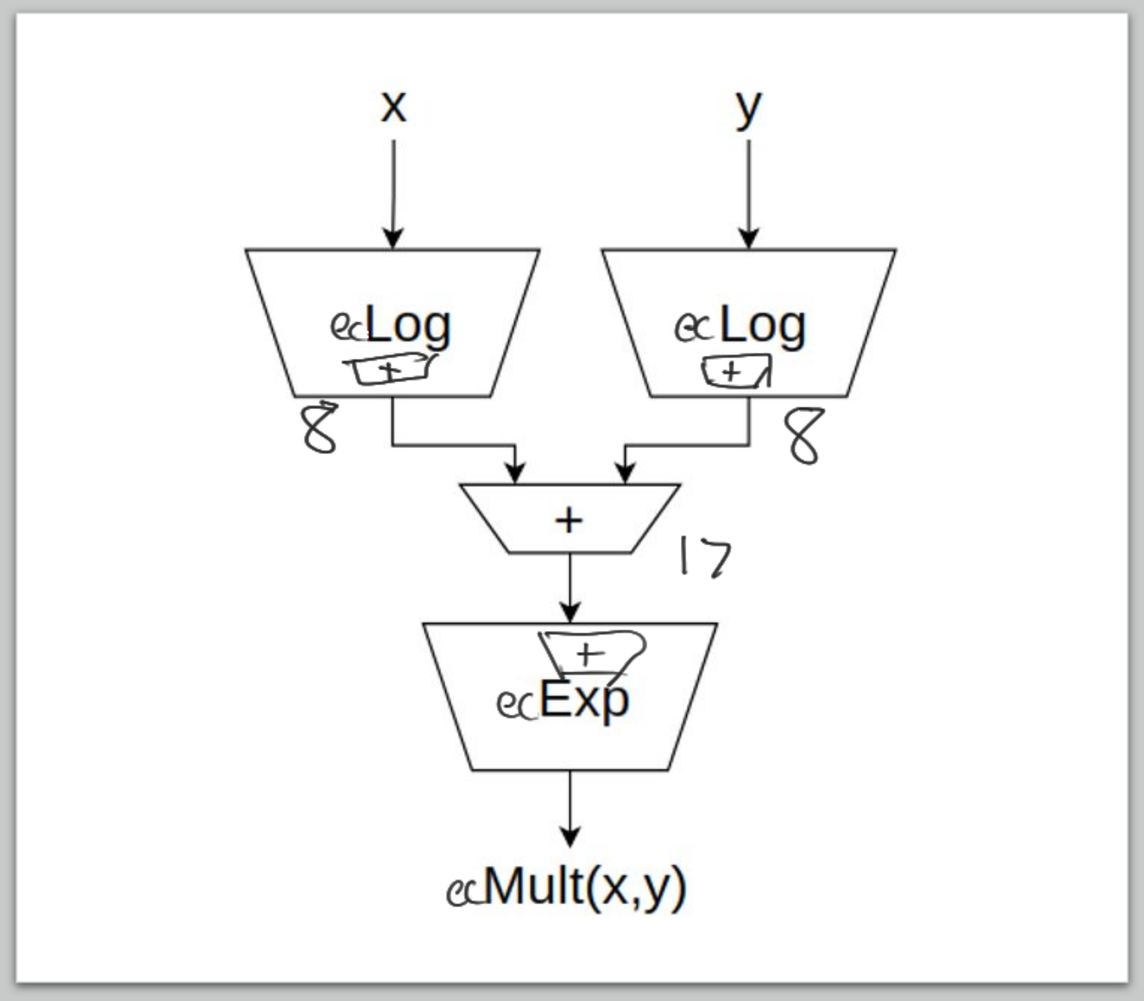
$M_L = 4$



Multiplication

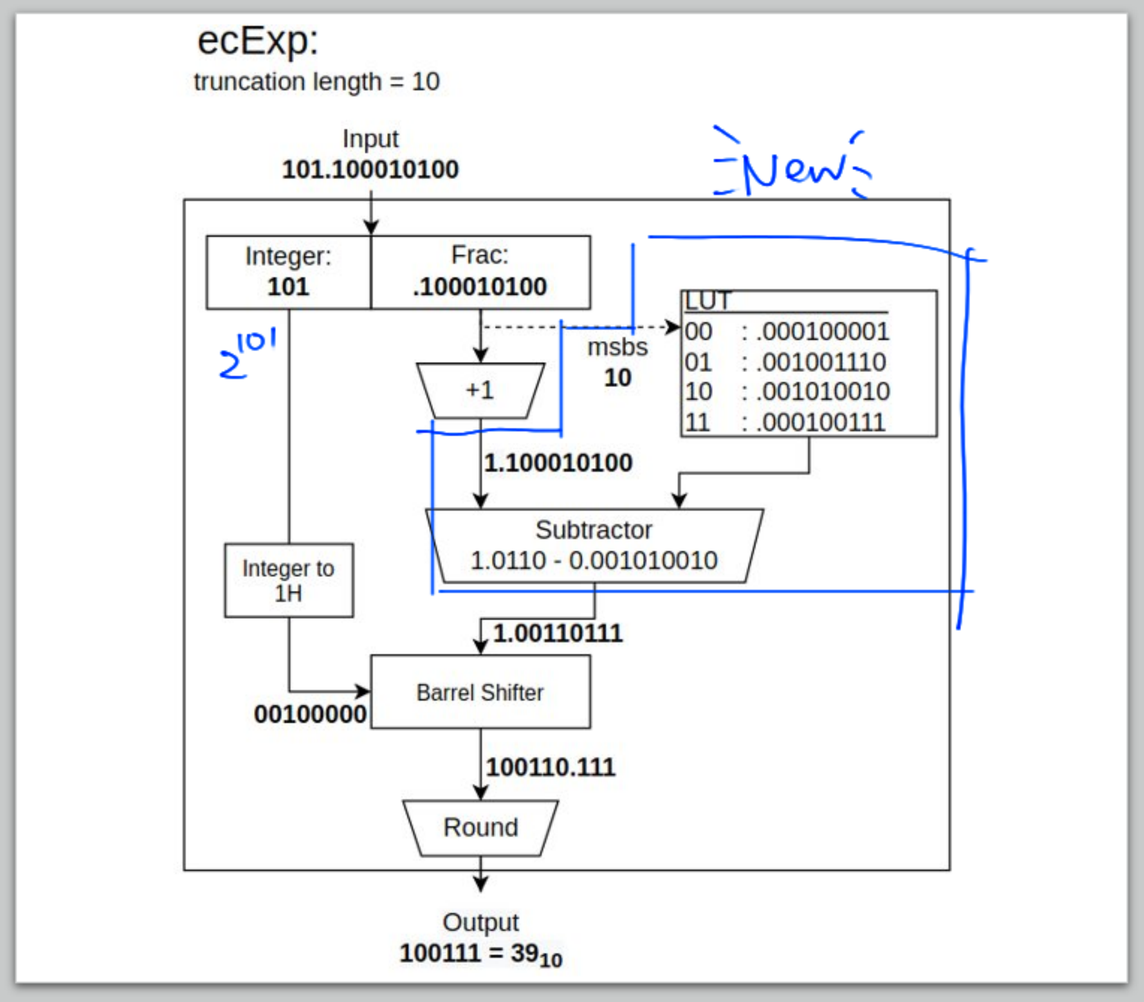
As easy as $2^{\log(x) + \log(y)} = xy$

ecLog: "error corrected Log"



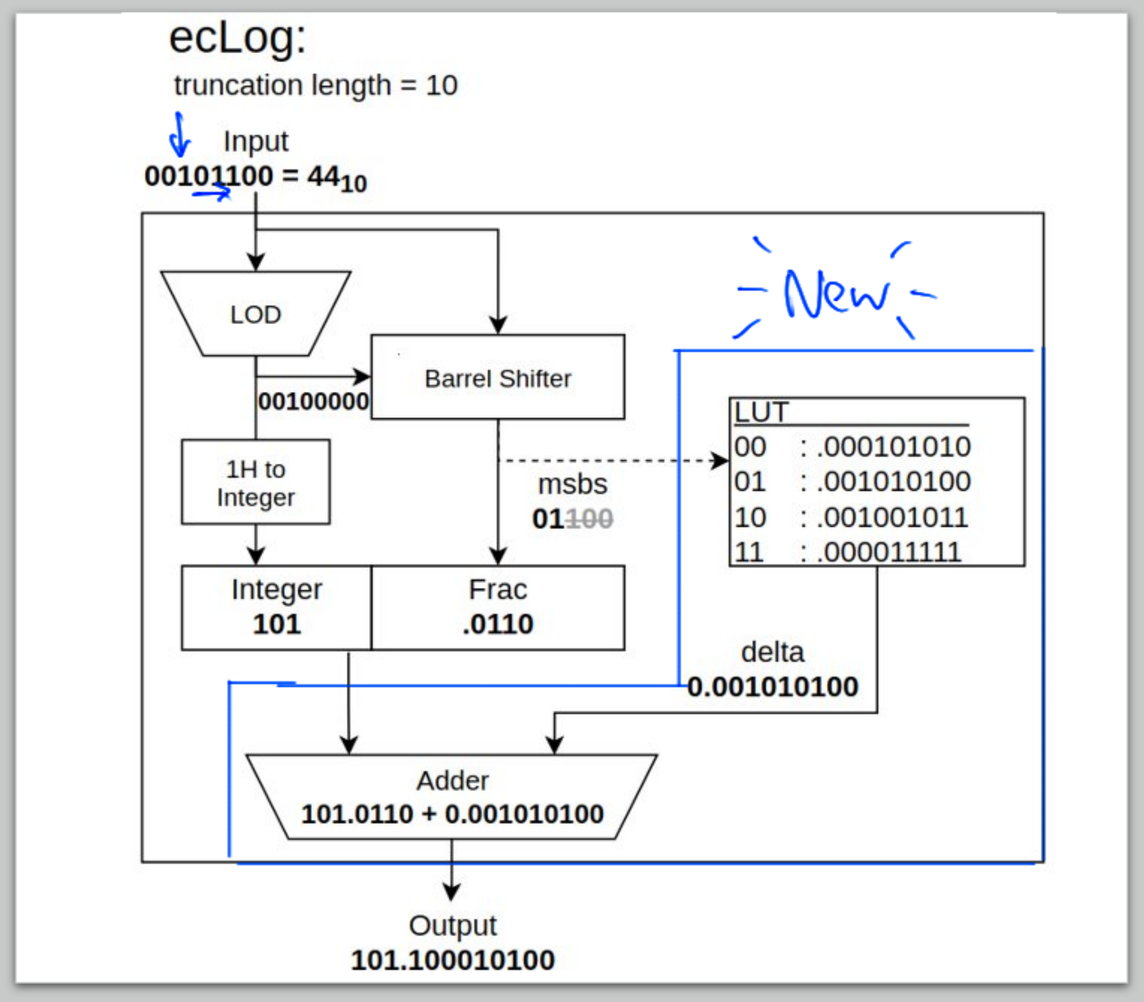
ECALE 2^x (ecExp)

- Input is fixed point binary.
- Truncation length is
  - = Length of fractional part.
  - = Length of correction values
  - > Maximum integer part number. For max precision.
    - > bitlen×2+1



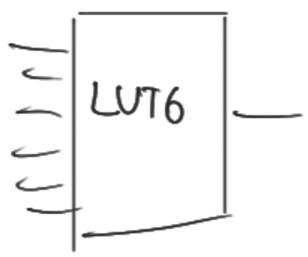
ECALE $\log_2 x$ (ecLog)

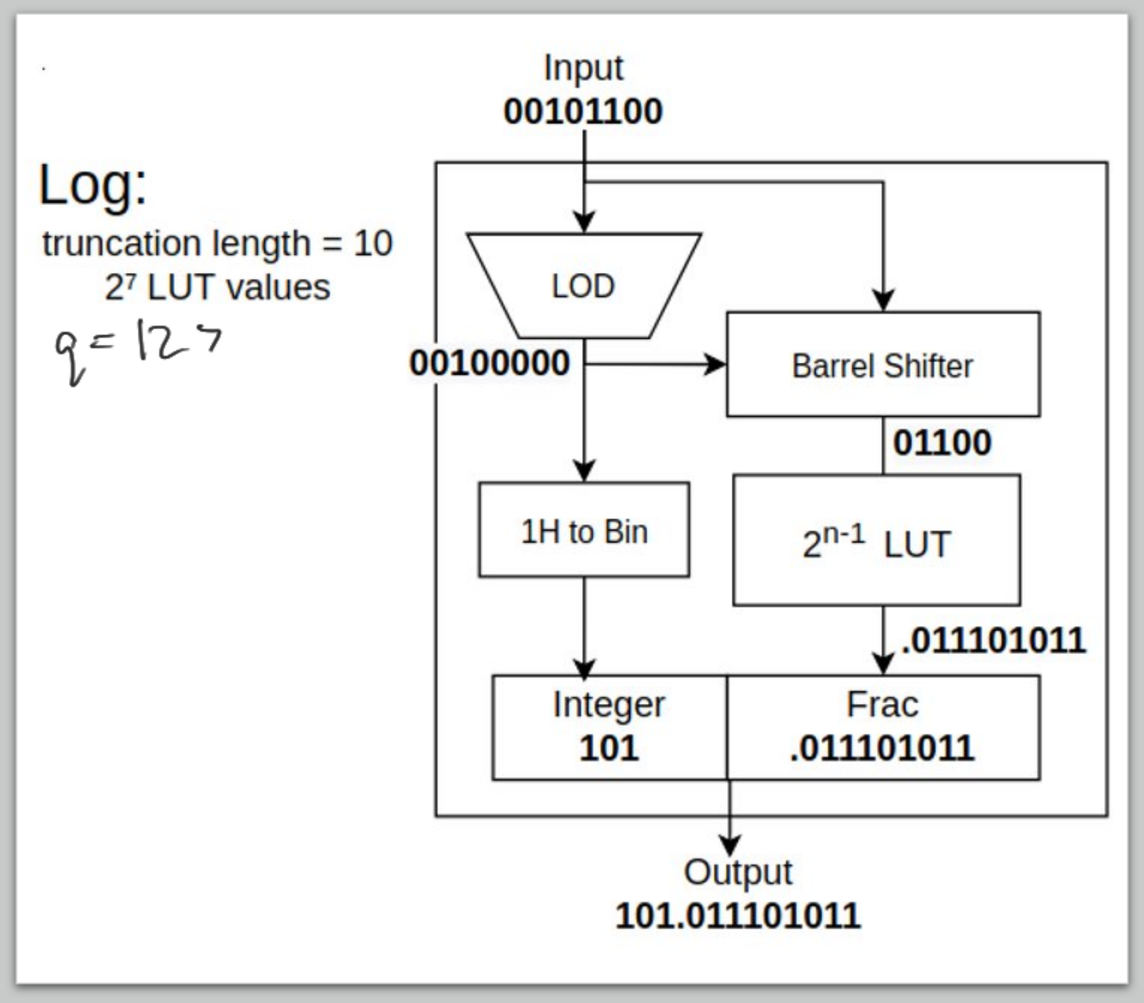
- Should have same truncation length as ecExp
- Can have a different number of LUT values to ecExp.



Precise LUT $\log_2 x$

- *"Let's memorize every possible value of log"*
- Use if need $M_L > q/2$ (Quite probable)
- LUT's on the Kintex-7 Are 6-bit => 64 possible values





ECALE Summary

Realm
space $\propto M^2$
ECALE
Space $\propto M_L + M_E$

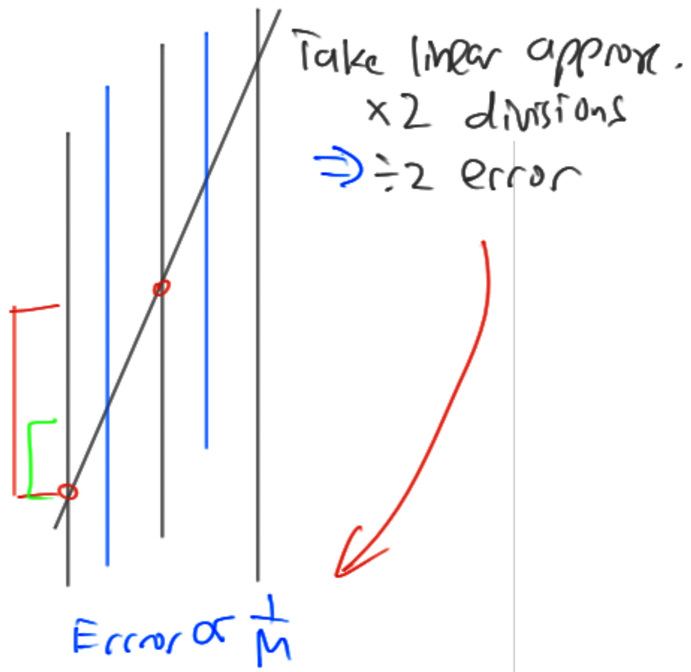
Pros:

- Significantly decreased error vs space complexity

Cons:

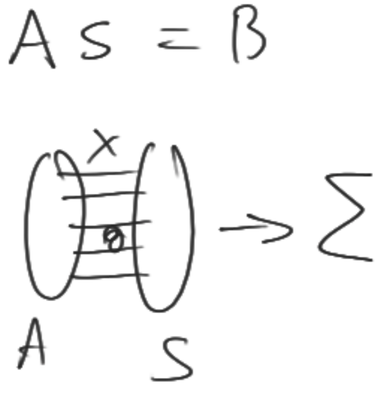
- Triples number of LUT's needed compared to realm
- Increases critical path by one LUT, and one adder
- Possible to implement, but still not space efficient for LWE

Excels when low range relative error rates are needed.



Question/Break time!

Further considerations:



- LUT Sharing
  - One look up table to rule them all?
  - How can multiple resources share the same table?
- LUT Delay
  - Block ram → clocking
  - Pipelining
- Truncation length
  - 2*bitlen + 1 for max precision

Bit splitting

$q+1 = 2^{13}$, $q = 2^{13} - 1$, $2^{13} \equiv 1 \bmod 8191$

$m = 6, n = 7$

12

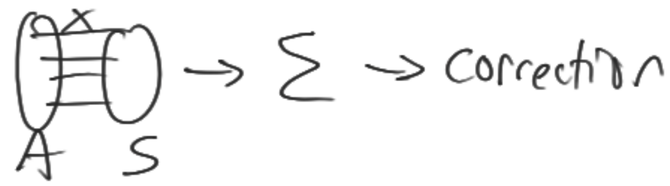
010101101001 -> 010101 | 101001

- $(a2^n + b)(c2^m + d) = ac2^{n+m} + ad2^n + bc2^m + bd$
- Precise logarithms achieved earlier
- Take advantage of modulus operation to reduce 2^{n+m} term
- Mixing in precise / approximate multiplication

$$ad2^n + bc2^m + bd + ac$$

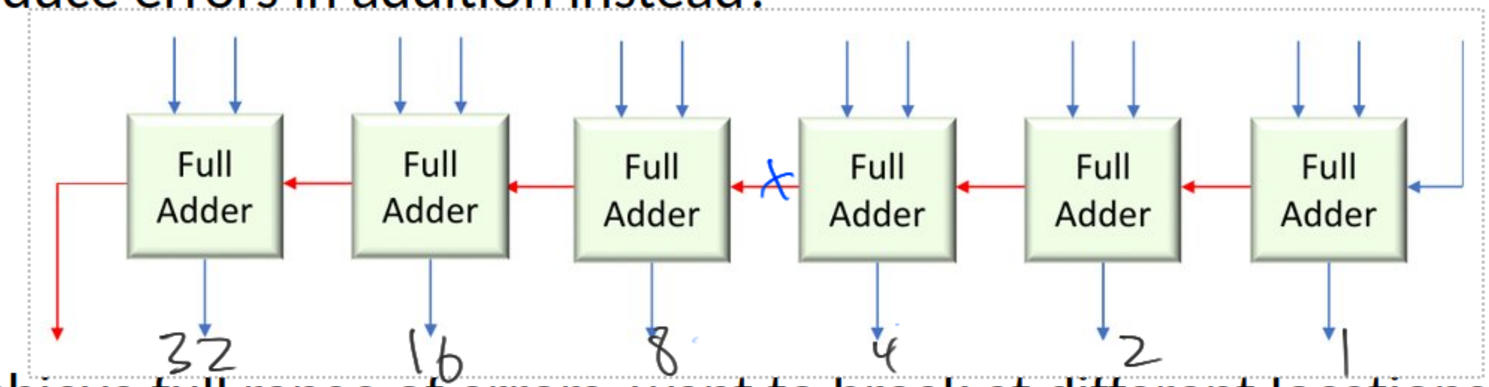
precise

Approximate addition



Multipliers on the Kintex-7 are fast → 6.5ns dsp units

Introduce errors in addition instead?



To achieve full range of errors, want to break at different locations.

Other points

- Sacrifice Security
  - Decrease size of #samples or q to reduce error
- Do more research!
  - Tons more designs exist in the wild.

Q Sam!

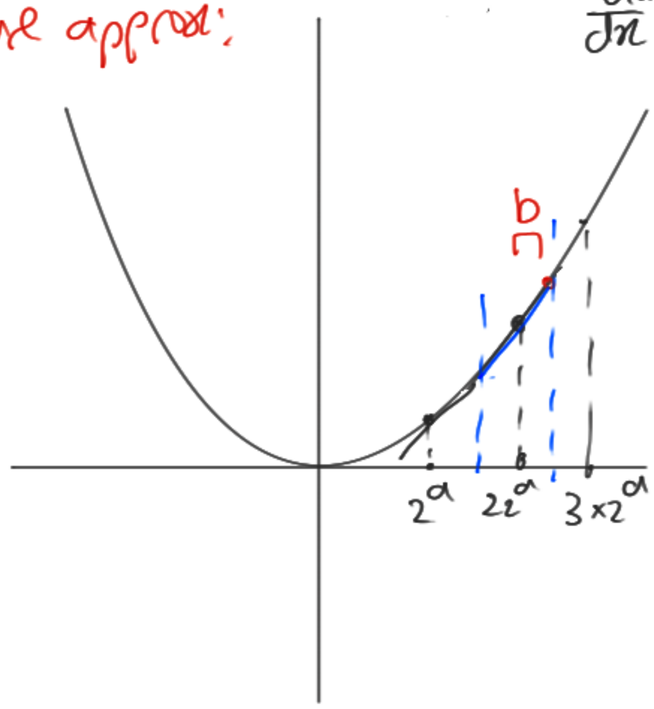
$$\frac{g(x+y) - g(x-y)}{4} \quad f(x,y)$$

$$\frac{(x+y)^2 - (x-y)^2}{4} = xy$$

Max → abs error is $\frac{2 b_{max}^2}{4}$

⇒ Want to approximate squares.

Square approx:



$\frac{dx^2}{dx} = 2x$ → take linear approx. $b < 2^{a-1}$

$$(n \times 2^a \pm b)^2 = \underbrace{n^2 2^{2a} + b 2 n 2^a}_{\text{approx}} + \underbrace{b^2}_{\text{error}} < 2^{2a-2}$$

LUT, Mult

Absolute error!! ∴

$a=3$	$b<4$ → easy
$a=4$	$b<8$ → okay
→ $a=5$	$b<16$ → Too hard: needs full multiplier.

Max viable reduction is 2^4. ∴

- Round x to nearest 2^a. eg: $a=3$
  → $x = 2^a \pm b$.
  $x = 4025$
  $x = 4024 + 1$
  $y = 4021$
  $= 4024 - 3$
  $z = 1010 \mid 10$
  $z = 11000 - (\sim 10 + 1)$

Derivations of REALM ballpark

For q = 8191:
Maximum product is 8191*8191 = 67 092 481

Want absolute error to be much less than q/4
∴ Relative error < $(q/4)/q^2$ = 0.003%.
Compare against Realm's 1.8% max error @ M = 16. Recall Error x M ∝ 1
∴ Need M x 0.003% > 1.8% x 16
∴ M >9600
Recall realm #Terms = M^2
∴ 9600^2 correction terms
Can't fit in all block ram & distributed ram & LUT's combined