Physical Design Space Exploration FPGA 2015

Ephrem Wu (Presenter), Inkeun Cho

Goal: Map an Arithmetic Function to an FPGA

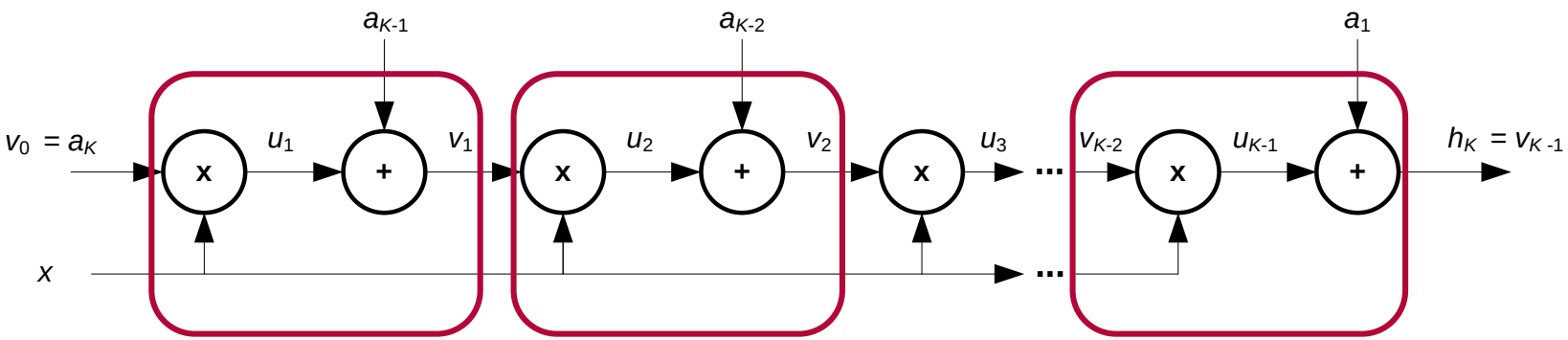
Target 500MHz with a 28nm low-speed-grade (-1) device

The polynomial $h_K = \sum_{k=1}^{K} a_k x^{k-1}$

Q0.17 Fixed Point

High-dynamic range with 17 significant bits

is expressed in the Horner form for stability and reducing resources



$K-1$ stages of multiply-add in a K^{th}-degree polynomial

XDR

Floating-Point Format for Two's Complement Integer Hardware

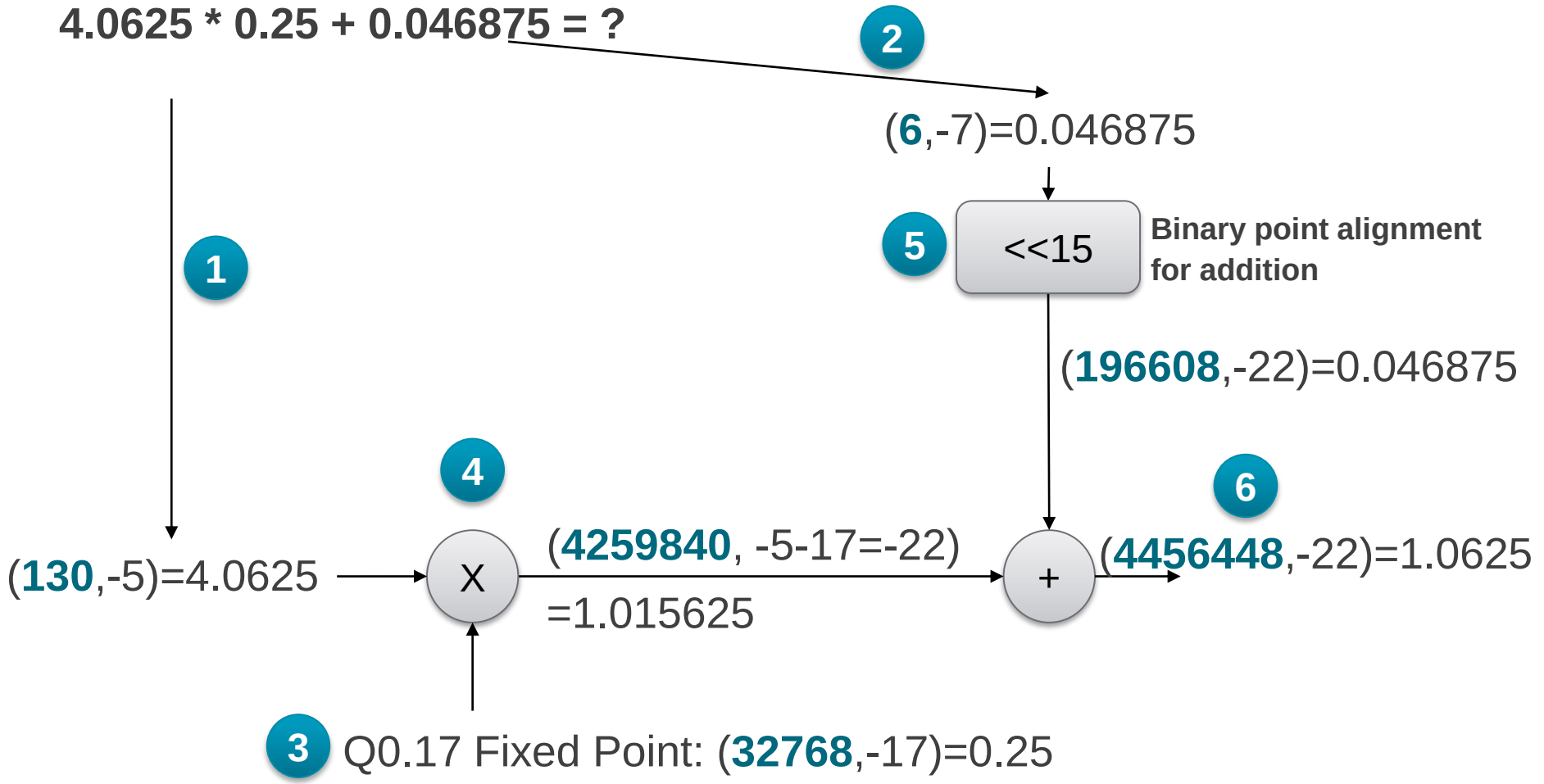
- **Every XDR$\langle L, E\rangle$ number x is a pair $x = (d, \varepsilon) = d \cdot 2^{\varepsilon}$, where**
  - d: ***significand*** of x, an L-bit two's complement integer
    - $|d| < 2^{L-1}$ to emulate sign-magnitude, or
    - $-2^{L-1} \leq |d| < 2^{L-1}$ if you know what you're doing to avoid overflow, and
  - ε: E-bit ***exponent*** of x, a two's complement integer. Normalization optional.

- **XDR is a custom floating point format.**
  - For block floating-point implementation with fixed-point hardware
  - Example: Multiple representations of 4.0625 in XDR$\langle 11,8\rangle$

    $x = (130, -5) = (260, -6) = (520, -7) = 4.0625.$

A Hybrid Fixed-Floating-Point Fused-Multiply-Add Example

Modeling intent: Put the XDR significand directly on the wires.



XDR Arithmetic Operations

- **For some common exponent ε,**
  - Addition: $(d_1, \varepsilon_1) + (d_2, \varepsilon_2) = (d_1 \cdot 2^{\varepsilon_1 - \varepsilon} + d_2 \cdot 2^{\varepsilon_2 - \varepsilon}, \varepsilon)$
  - Multiplication: $(d_1, \varepsilon_1) \cdot (d_2, \varepsilon_2) = (d_1 d_2 \cdot 2^{\varepsilon_1 + \varepsilon_2 - \varepsilon}, \varepsilon)$
  - Fused multiply-add
    - No access to shift product. Must do all necessary shifting up-front.
    - $(d_1, \varepsilon_1) \cdot (d_2, \varepsilon_2) + (d_3, \varepsilon_3) = (d_1 d_2 \cdot 2^{\varepsilon_1 + \varepsilon_2 - \varepsilon} + d_3 \cdot 2^{\varepsilon_3 - \varepsilon}, \varepsilon)$
- **Finding the right common exponent ε is key to**
  - saving hardware and
  - maximizing dynamic range.
- **Absolute value for an L-bit two's complement value**

$$\boldsymbol{d} = \langle d_{L-1}, d_{L-2}, \cdots, d_0 \rangle$$

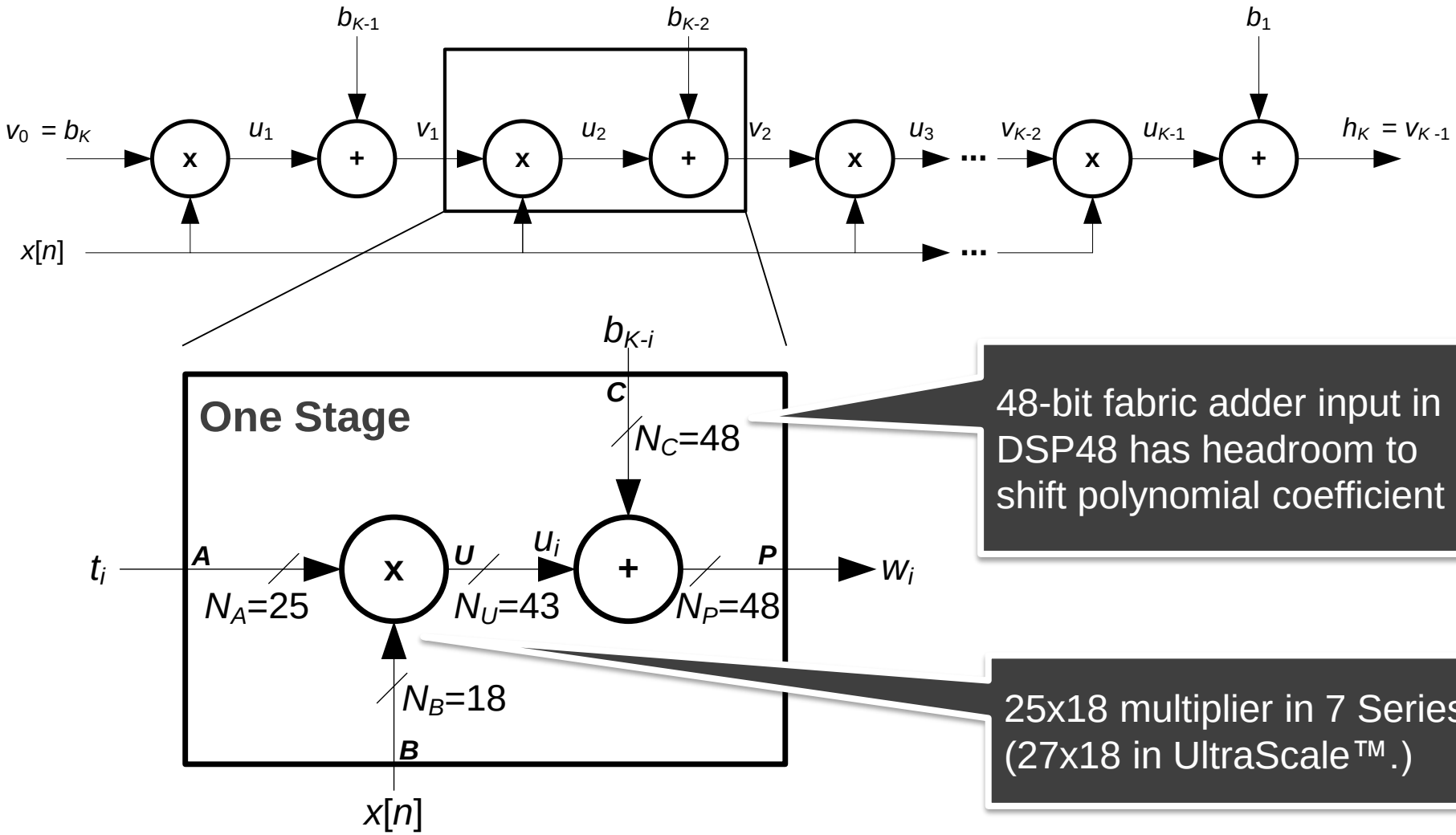
$$\text{abs}(\boldsymbol{d}) = \begin{cases} \boldsymbol{d}, & d_{L-1} = 0 \\ \neg \boldsymbol{d} + 1, & \Pi(\boldsymbol{d}) = 0 \\ \neg \boldsymbol{d}, & \Pi(\boldsymbol{d}) = 1 \end{cases}, \qquad \text{where } \Pi(\boldsymbol{d}) = \begin{cases} 1, & d = -2^{L-1} \\ 0, & \text{otherwise.} \end{cases}$$

Original exponent minus final exponent

One stage of the Horner polynomial accelerator

Test of most-negative two's complement integer

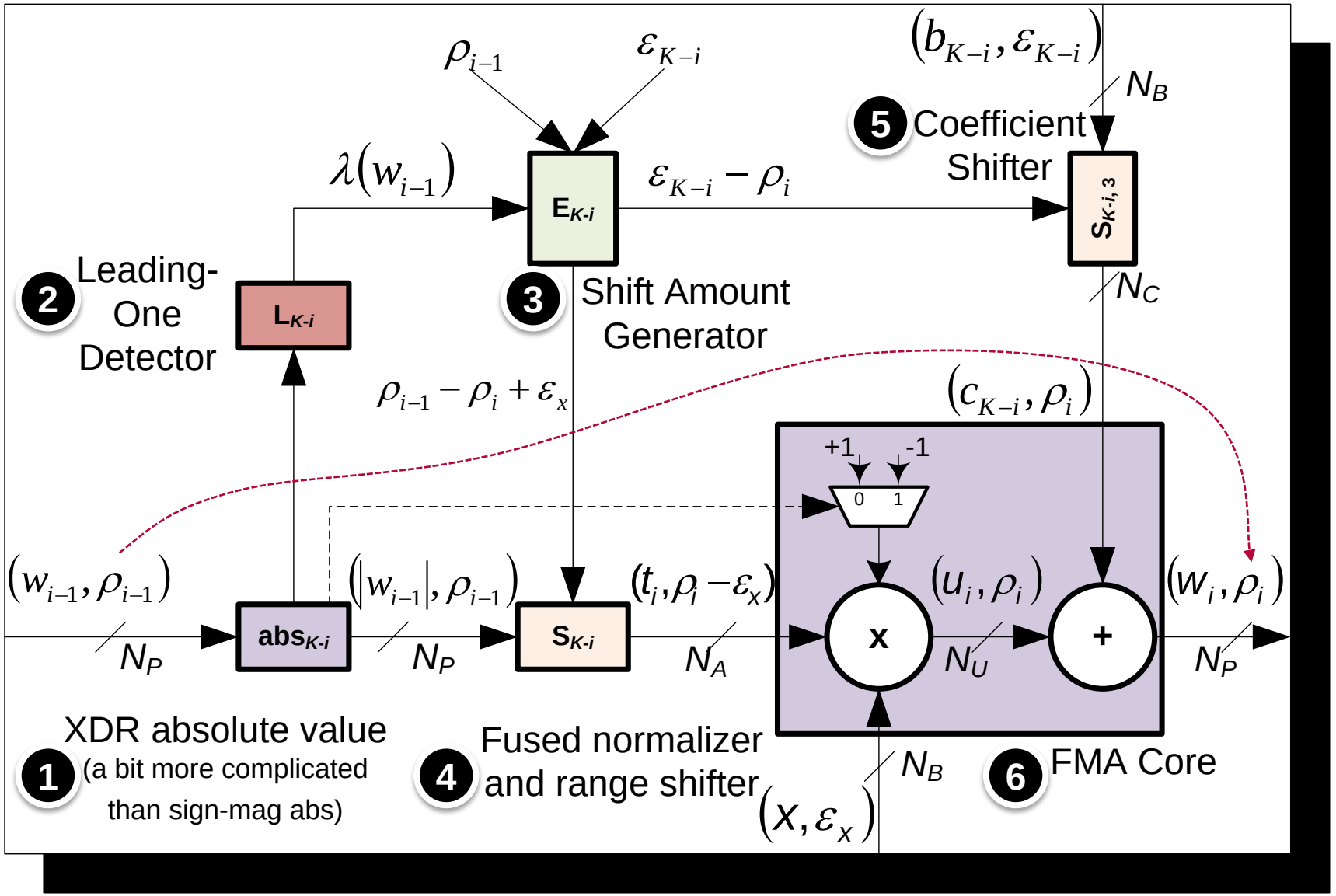
Fused Multiply-Add as Polynomial Systolic Cell



48-bit fabric adder input in DSP48 has headroom to shift polynomial coefficient

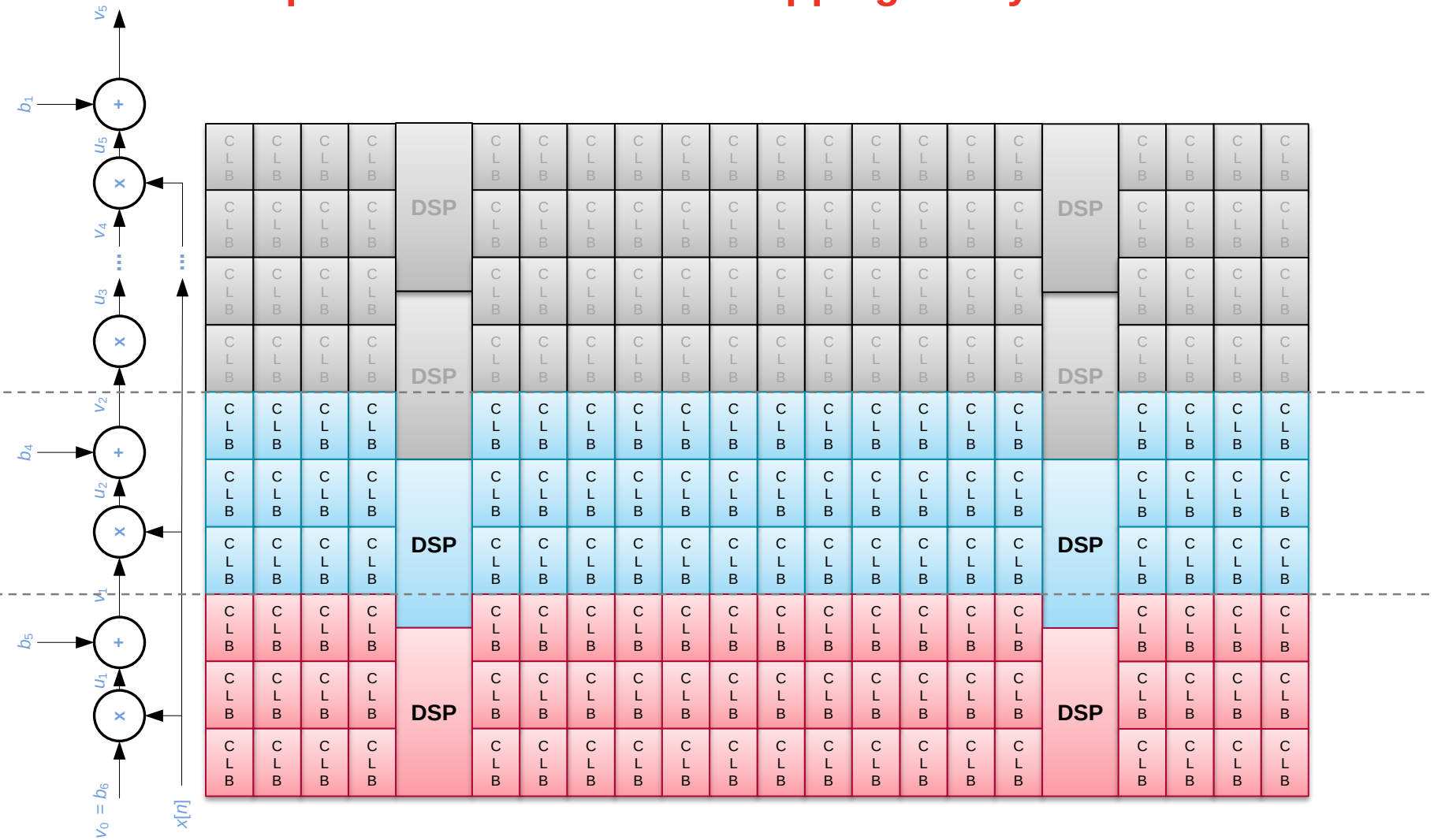
25x18 multiplier in 7 Series. (27x18 in UltraScale™.)

XDR Systolic Cell



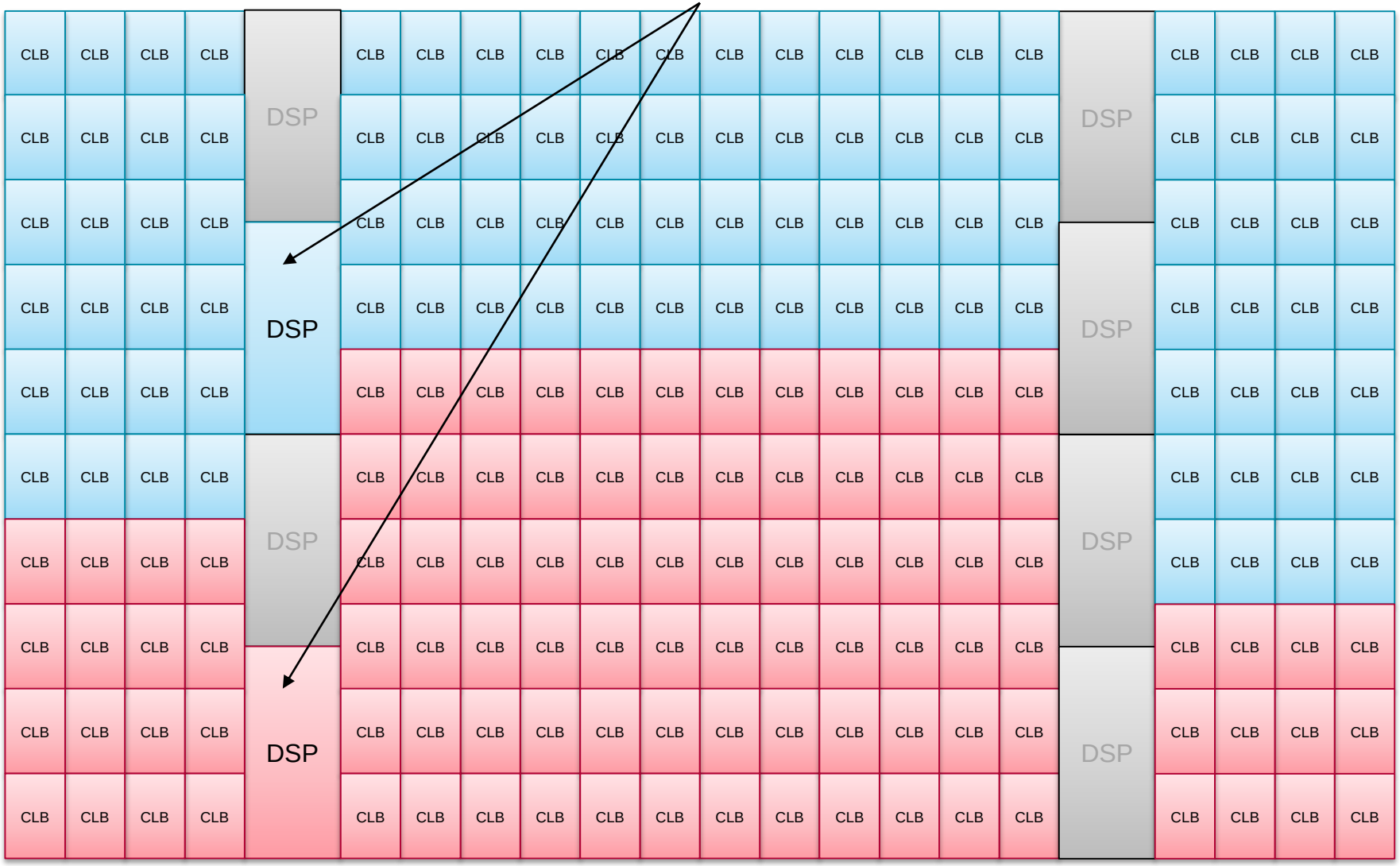
Strategy for a Scalable High-Performance Design

FPGA-Floorplan-Aware Resource Mapping for Systolic Cells



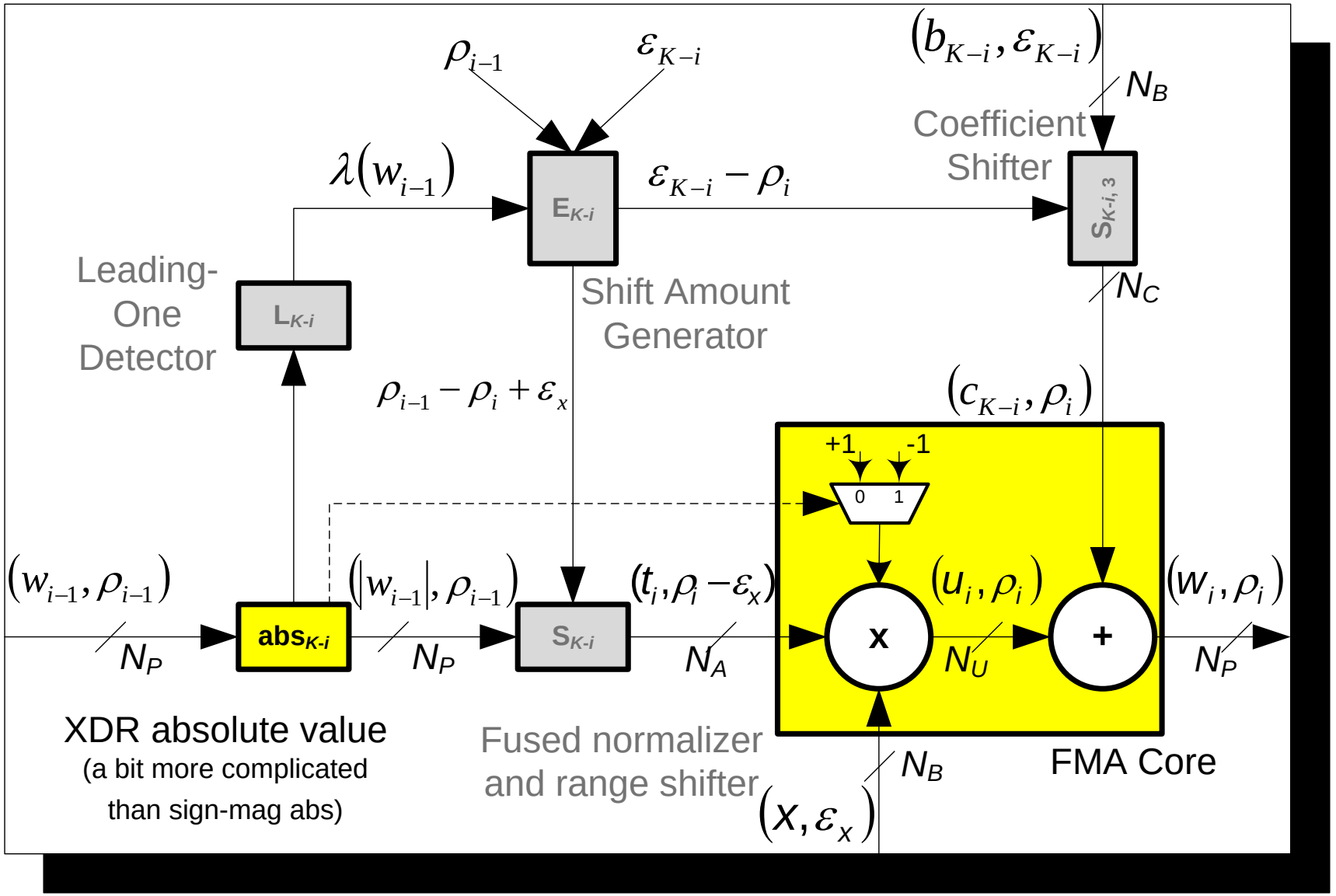
LUT-to-DSP Ratio Matters to Resource Mapping

DSPs assigned to FMA but not ABS results in DSP underutilization



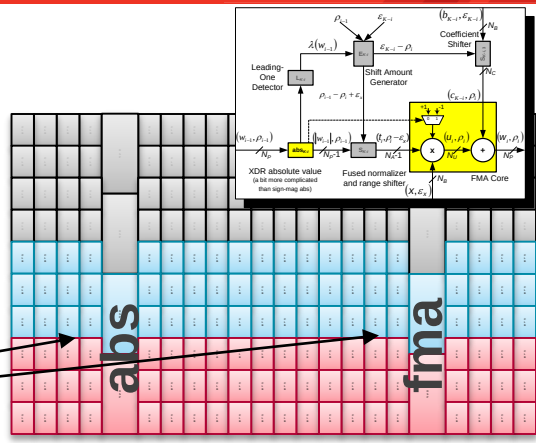
Mapped Both ABS and FMA to DSP48E1 on K-7

Two DSP48s Per Cell to Match LUT-to-DSP Ratio with FPGA Region



Mapping Choices

- **Two DSP48E1s instantiated for each systolic cell**
  - One column for the ABS.
  - Another column for the FMA.
  - ~300:1 LUT-to-DSP48 ratio in each stage to achieve balance for P&R



- **XDR ABS a tad more than sign-mag ABS**

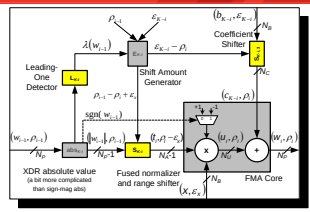
$$\text{abs}(\boldsymbol{d}) = \begin{cases} \boldsymbol{d}, & d_{L-1} = 0 \\ \neg\boldsymbol{d} + 1, & \Pi(\boldsymbol{d}) = 0 \\ \neg\boldsymbol{d}, & \Pi(\boldsymbol{d}) = 1\,. \end{cases}$$

2s complement negation

Saturation for most-negative 2s complement number (1000…0).

- **XDR abs as a DSP48E1 by controlling CARRYIN and ALUMODE.**
  - CARRYIN = $\text{sign\_bit}(\boldsymbol{d}) \,\&\, \sim\Pi(\boldsymbol{d})$
  - ALUMODE[0] = $\text{sign\_bit}(\boldsymbol{d})$

Leading-One Detect and Arithmetic Shift



Leading-One Detector

- Chosen over leading-one anticipator (LOA) to reduce area
- Tried many versions, including recursive code
- Current version is a pipelined CARRY4-based priority encoder
- Formally verified against a for-loop RTL description

Arithmetic Shifter

- Parameters: width and pipe depth
- Fused shifter for normalization and binary point adjustment for FMA
- Right shifter only. Up to 71 bits to emulate bidirectional shift.
- Formally verified against Verilog “a >>> b”

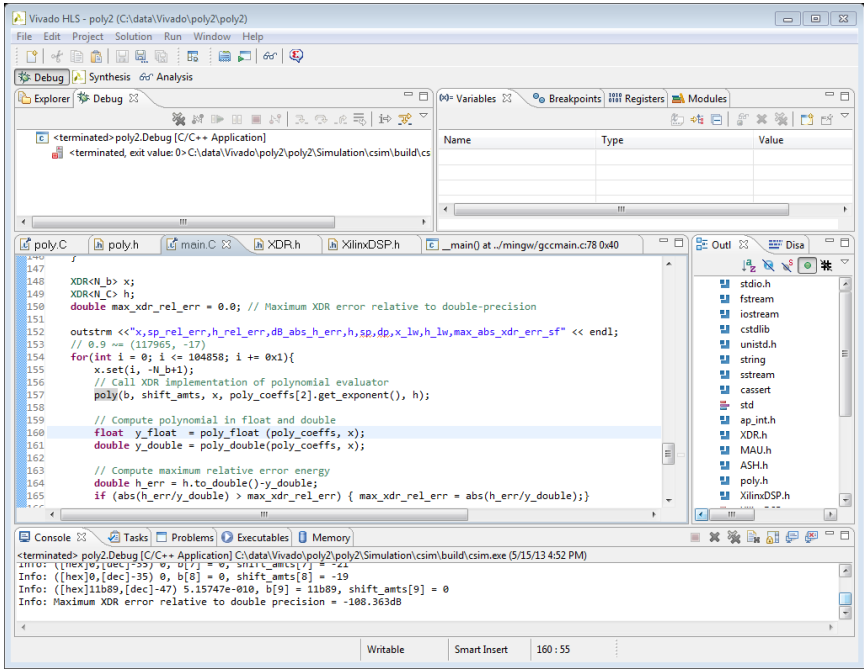
Vivado HLS C++ Used for Numerical Analysis

- **The Vivado C++ arbitrary-precision template classes**
  - ap_int<>
  - ap_fixed<>

Well-defined and handy C++ classes from HLS!

- **XDR C++ template class models high-dynamic-range numbers of the form $b = (c, \varepsilon) = c \cdot 2^{\varepsilon}$.**

- **Round-towards-zero chosen for lower HW costs.**
  - Doesn't have to use RTZ.
  - Thanks to ap_fixed<>, easy to change rounding mode

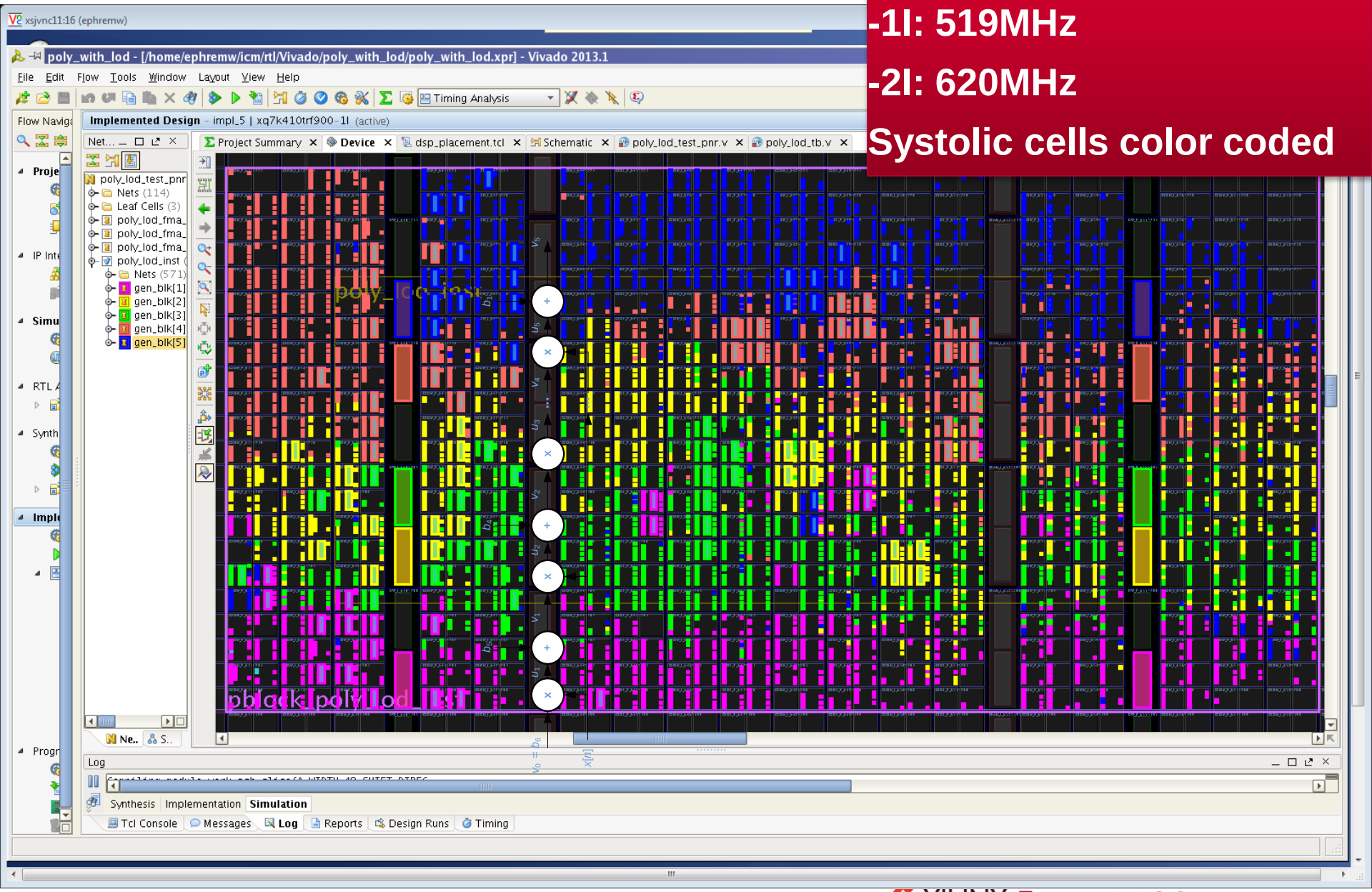


Kintex-7 410T Place-and-Route Results

-1I: 519MHz

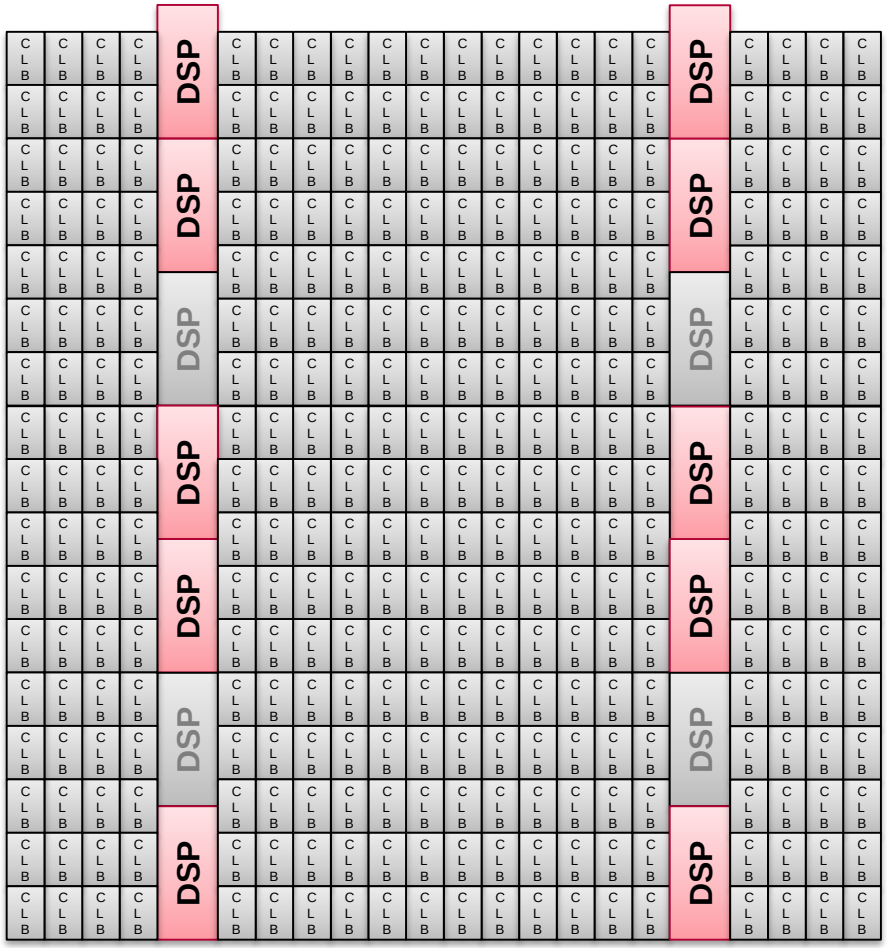
-2I: 620MHz

Systolic cells color coded



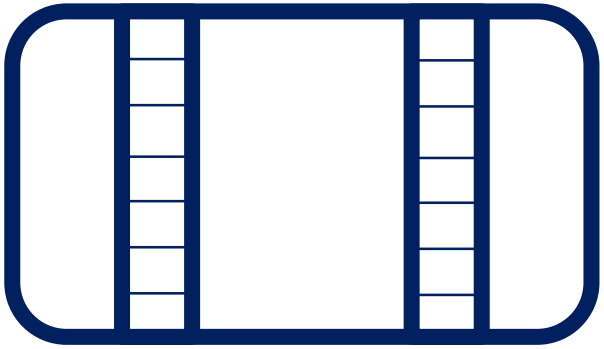
How Good is the Manual DSP Placement?

Design Space Exploration with Vivado



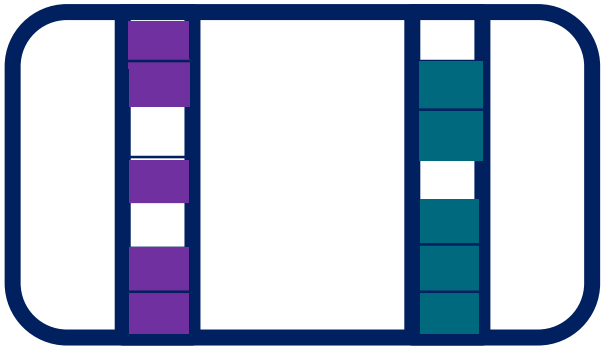
- Use 10 out of 14 DSP slots to match LUT-DSP ratio
  - $\binom{14}{10}10! = {}^{14!}/_{4!}$ unique DSP placements
  - If ~10 mins per P&R run, >69,110 years of run time
- Heuristically prune search space
- Plot fmax histogram

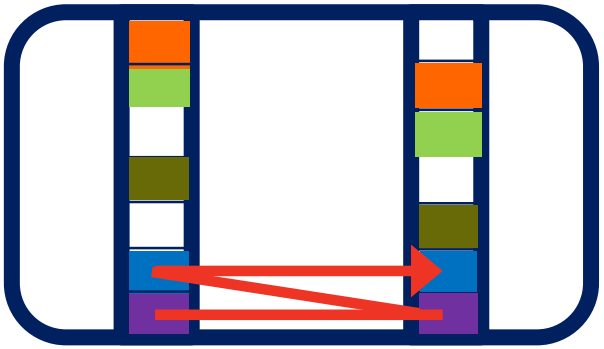
DSP48E1 Placement Exploration Pruning



Pick 5 DSP positions among 7 in each column

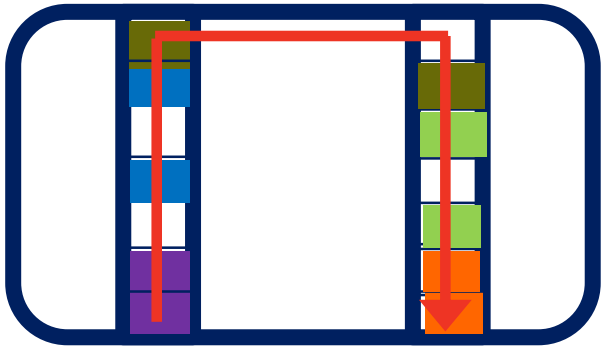
$_7C_5 \times {}_7C_5 = 441$





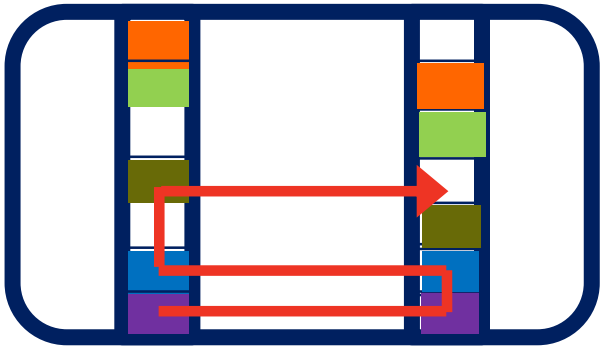
Case 1.

Ping-pong placement



Case 2.

U-shaped placement



Case 3.

S-shaped placement

441 x 3 x 10 (minutes) ➔ 3 days with 3 machines

Vivado™ Default vs. Expanded Placements

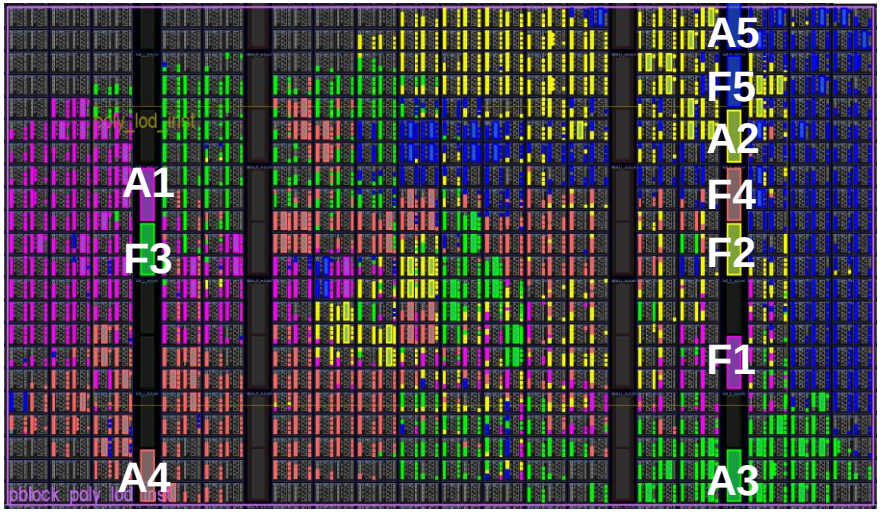


Fig 1. Default Placement

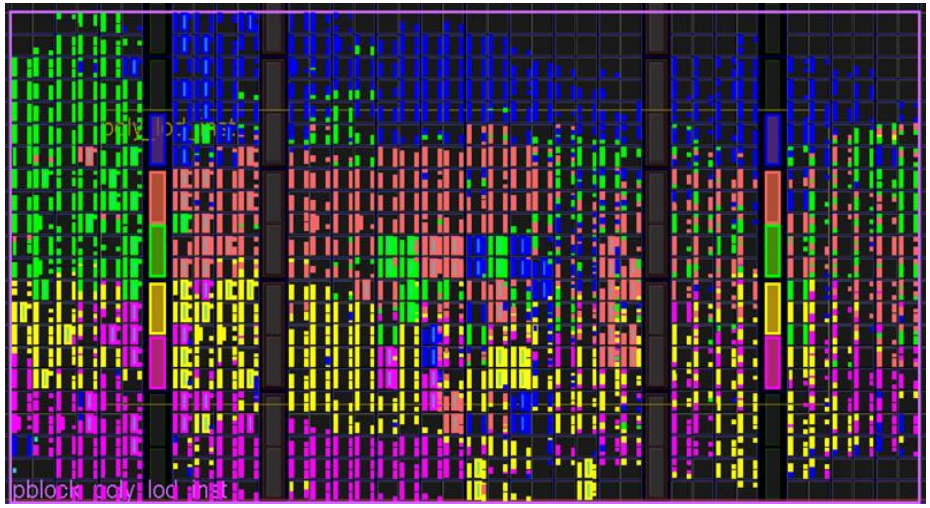


Fig 2. Ping-pong DSP Placement

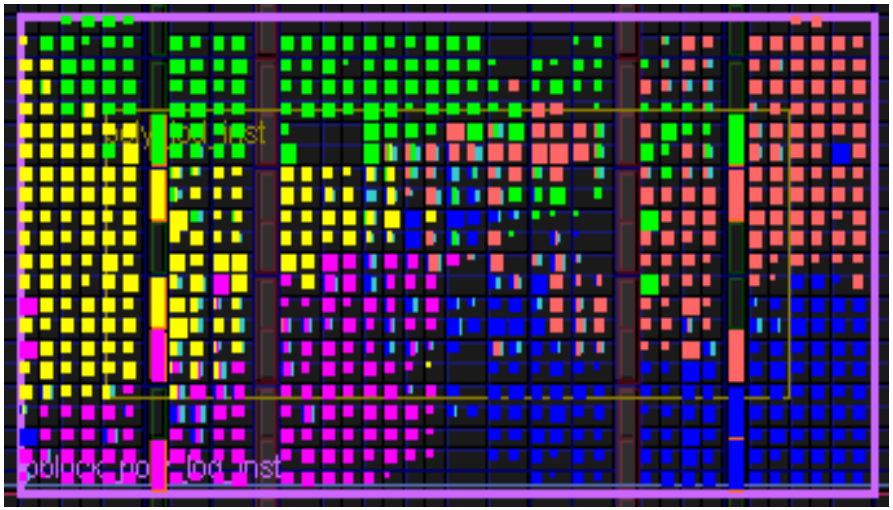


Fig 3. U-Shaped DSP Placement

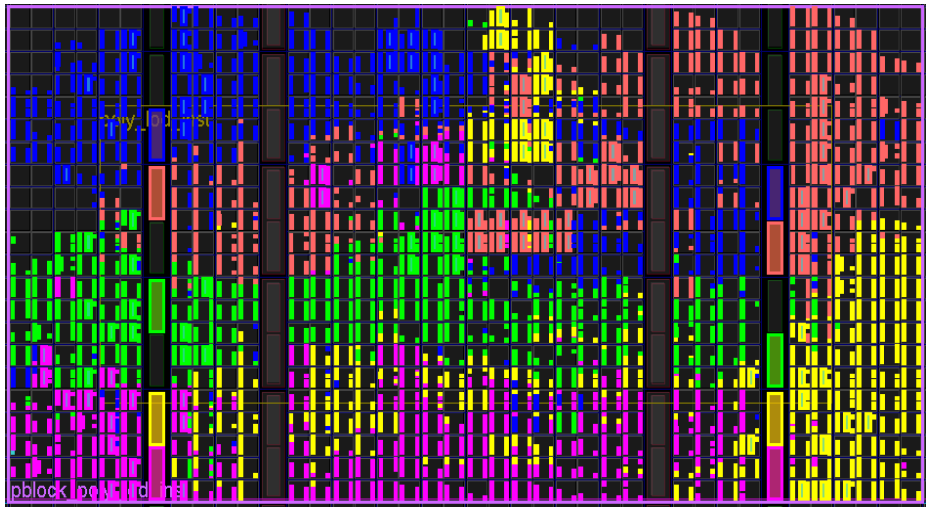
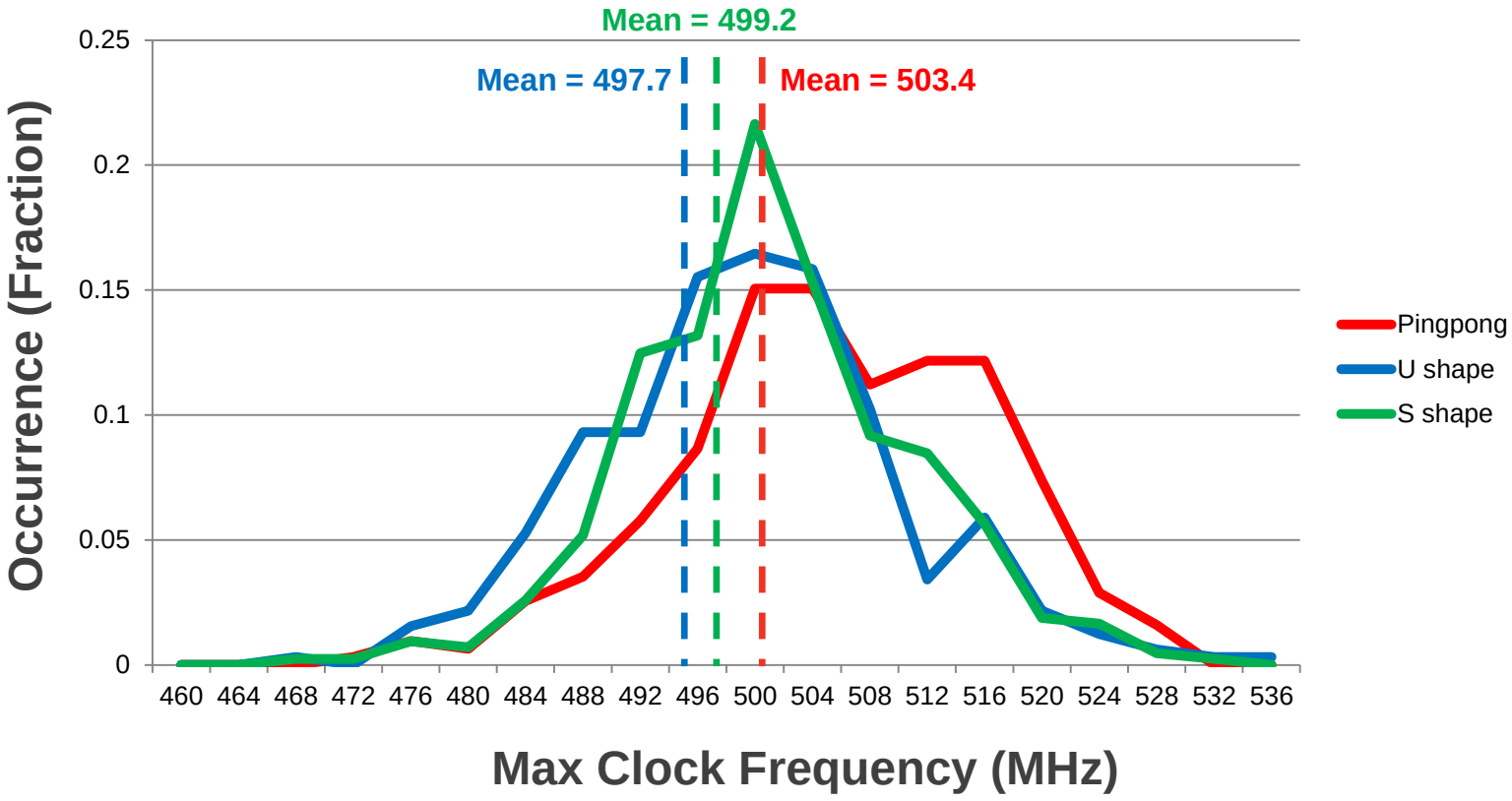


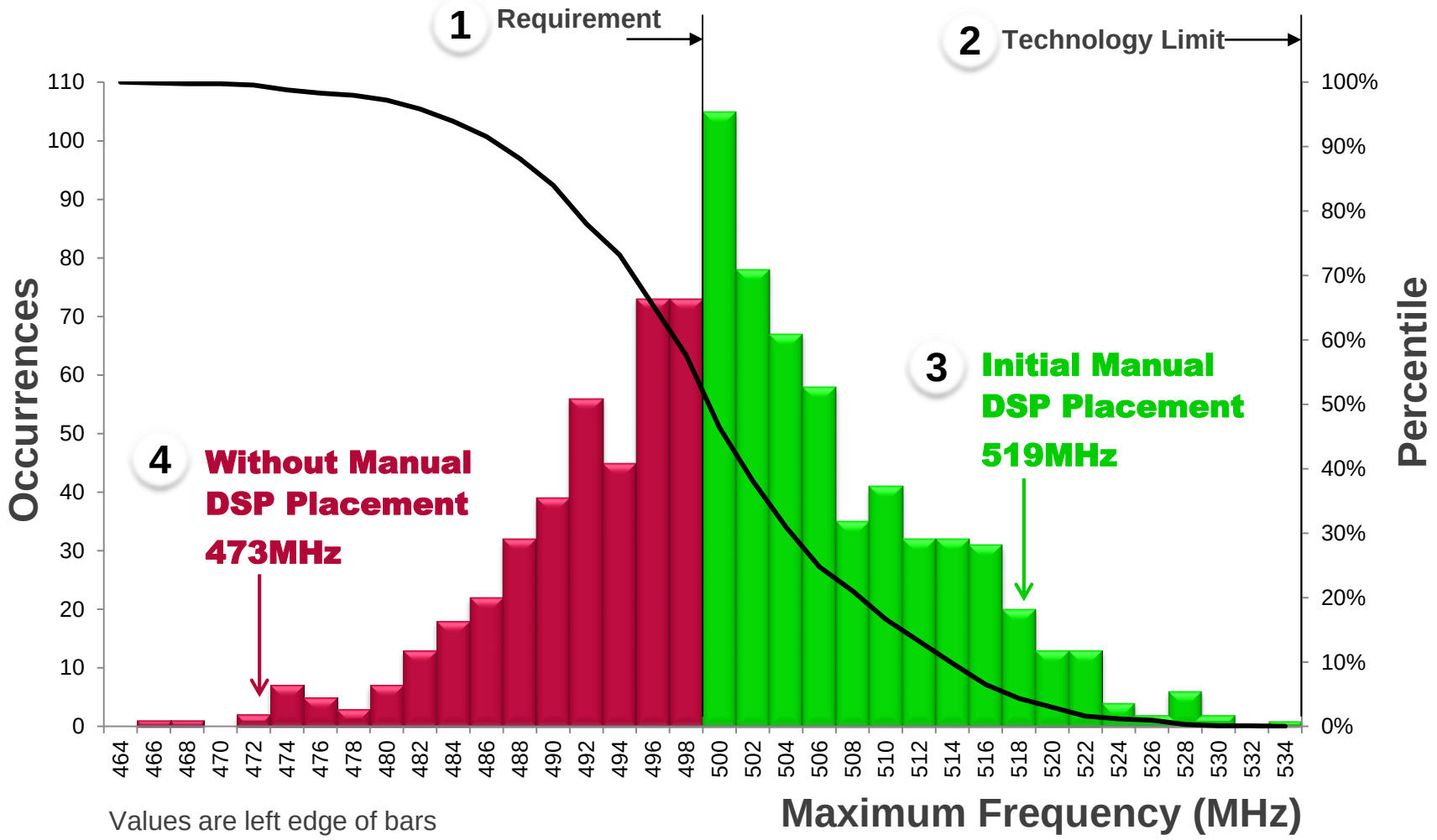
Fig 4. S-Shaped DSP Placement

Fmax Distributions

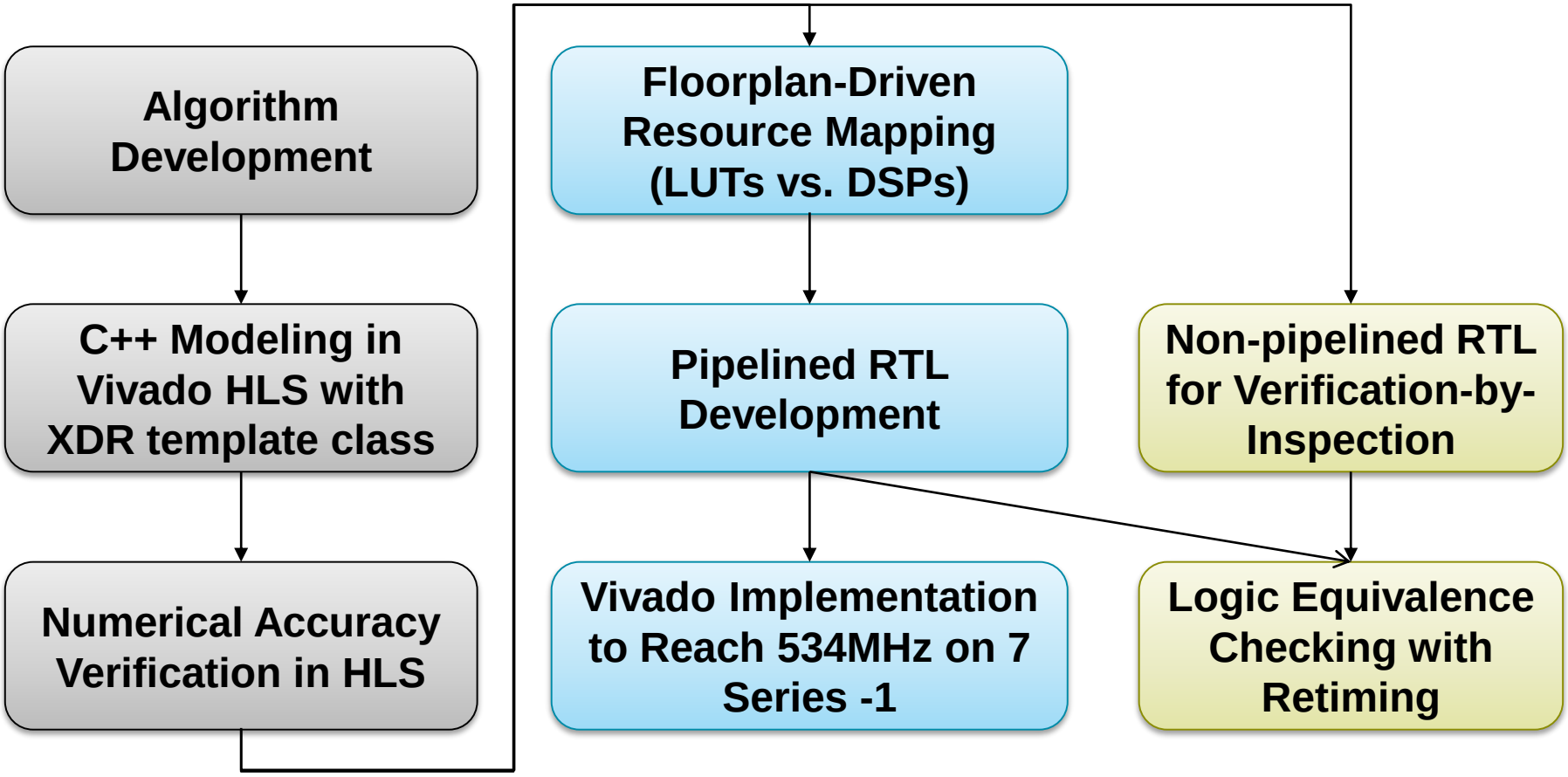


Performance Distribution

Top 46.4% of Results Meet 500MHz at -1



Work Flow



Summary

- **Developed XDR as an analytic tool to**
  - ease reasoning of custom-floating-point designs …
  - … for two's complement integer hardware.
- **Exceeded Fmax target with Vivado™ and guided placement**

Kintex -1 Fmax	Kintex -2 Fmax
519 MHz	620 MHz

- **Design space exploration revealed faster designs at -1 (534MHz)**
- **Divide and Conquer with Floorplan-Aware Resource Mapping**
  - Restrict RTL pipeline hacking to low-level blocks. A library of by-products …
  - Manually balance LUT and DSP resource use by recoding RTL
  - The only physical constraints used: pblock and manual DSP placements
  - Turns out initial manual DSP placement works pretty well

Thank you