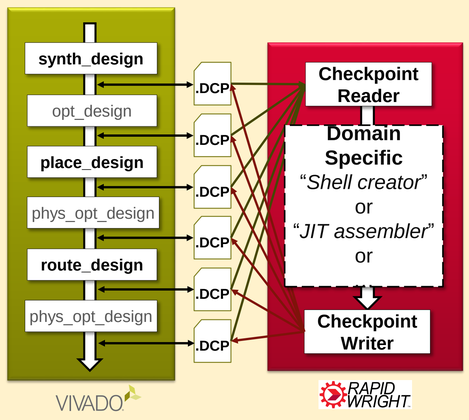
Customizing FPGA Implementation Flows for Domain-specific Applications

Chris Lavin and Alireza Kaviani
Xilinx Research Labs, San Jose, CA USA

What is RapidWright?

- **Companion framework for Vivado**
  - Fast, light-weight, open source
  - Communicates through Design CheckPoints[1] (DCPs)
  - Java code, Python scripting
- **Enables targeted solutions**
  - Reuse & relocate pre-implemented modules
  - Just-in-time implementations
  - Create shells & overlays
- **Power user ecosystem**
  - Academic algorithm validation
  - Rapid prototyping of CAD concepts

[1]DCP = netlist + P&R data + constraints



What Can it Do?

BUILD ROUTED CIRCUITS

FROM SCRATCH — GENERATORS

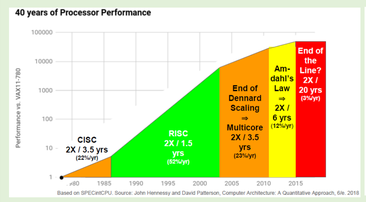
- Well-defined circuits in seconds
- Parameterizable library of generators

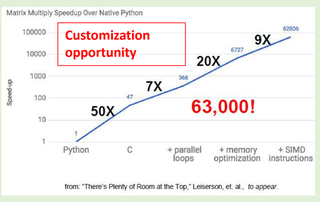
REUSE P&R CIRCUITS

FROM VIVADO — SHELLS & OVERLAYS

- Reuse/relocate P&R circuits from Vivado
- Combine P&R circuits together

Motivation





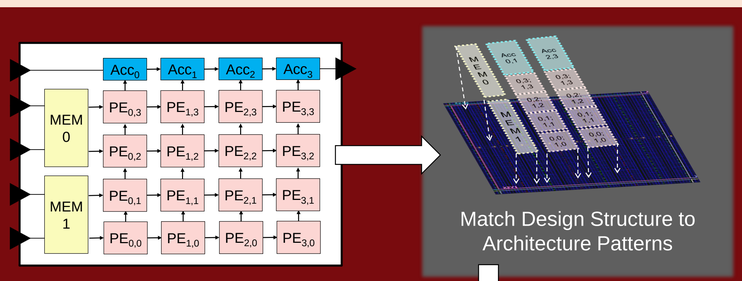
- New tools are required to enable domain-specific optimizations (not easily achievable with existing tools)
- Achieve higher efficiency by tailoring the architecture to characteristics of the domain
  - More effective parallelism for a specific domain, More effective use of memory bandwidth
  - Domain specific programming language

Source: A New Golden Age for Computer Architecture: (Domain-Specific Hardware/Software Co-Design)
John Hennessy and David Patterson
Stanford and UC Berkeley,13 June 2018

A Modular Pre-implemented Methodology

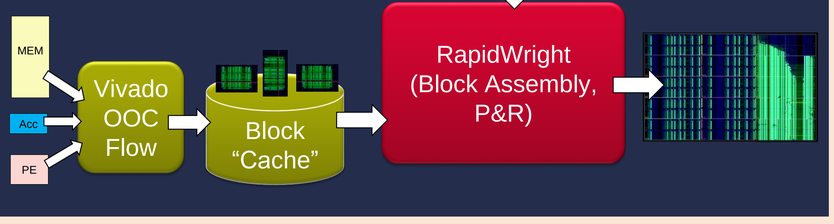
USER TASKS (MANUAL)

1. Design selection attributes:
   - Modular
   - Latency tolerant
   - Prefers replication
2. Placement planning



TOOL TASKS (AUTOMATED)

3. P&R modules cached:
   - Relocatable
   - Reusable
   - Timing predictable
4. Run implementation

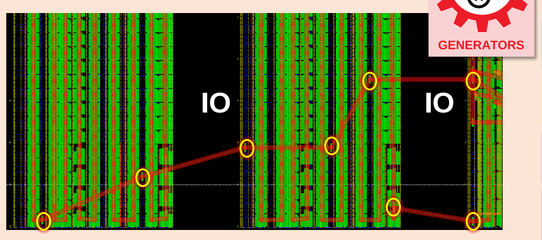


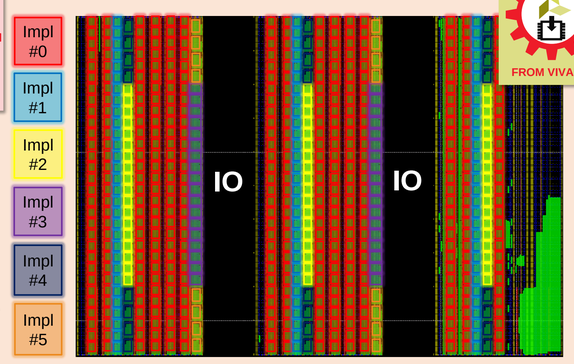
Design	Target Device	Baseline (initial design)	RapidWright[1] Flow	Gain	LUT	FF	DSP	BRAM
Seismic	KU040	270MHz	390MHz	41%	93%	5%	-	-
FMA	KU115	270MHz	417MHz	54%	25%	50%	97%	6%
GEMM	KU115	391MHz	462MHz	16%	19%	20%	87%	-
ML overlay	ZU9EG	368MHz	541MHz	50%	46%	29%	42%	96%

[1]RapidWright: Enabling Custom Crafted Implementations for FPGAs, FCCM 2018

Speed Grade: -2

- **FMA 1320 kernels**
- **97% DSP utilization**
- **4.4 TeraOp/s**
- **"Fabric discontinuites"**
  - SLR boundary
  - IO Columns
  - Laguna Tiles





- Exploiting latency-tolerance and architectural knowledge
  - Automatic insertion of latency blocks

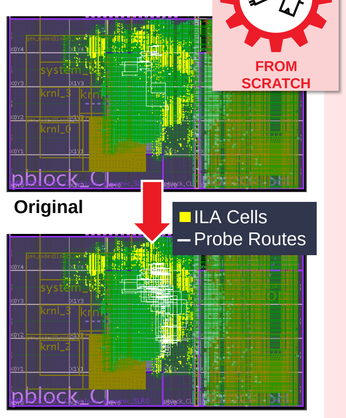
Use Case Examples

Customize ILA Probe Re-routing

FROM SCRATCH

- **RapidWright probe router enables higher productivity**
  - 21X more debug turns per day
  - Highest level of routing preservation possible
  - Future innovation:
    - iteration with extra probe inputs
    - Automatic insertion of pipeline flops to manage timing

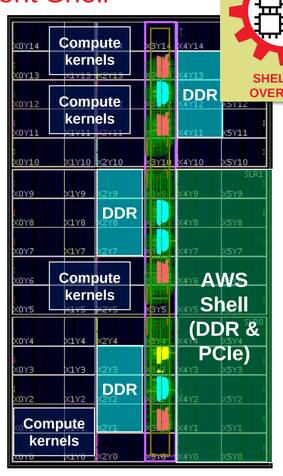
Vivado modify_debug_probes	RapidWright ProbeRouter	Δ
130 mins	6.3 mins	21X



Pre-implemented Data Movement Shell

SHELLS & OVERLAYS

- Goals
  - Minimize overhead of compute (and overlays)
  - Prove shell assembly model
- Build-to-order LinkBlaze[1] shell
  - 512 bit, bi-directional
  - RapidWright Pre-implemented modules

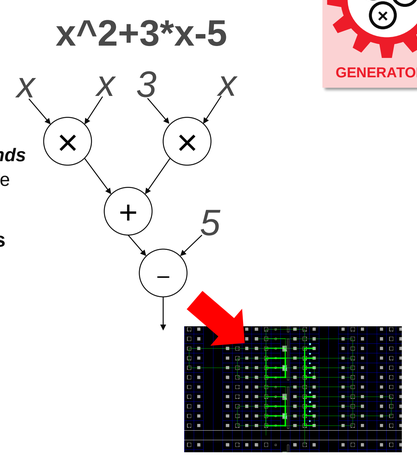


[1] LinkBlaze: Efficient global data movement for FPGAs (ReConFig 2017)

Just-in-time, Circuit Module Generators

GENERATORS

- **Build modules on-demand**
  - Placed and routed ***in seconds***
  - Reusable and compose-able
  - Target spec performance
- **Parameterizable Generators**
  - Adder
  - Subtractor
  - Multiplier
- **Expression Generator**
  - Invokes math generators
  - Built to spec: 775MHz



A Foundation for Community Tools

RAPID WRIGHT™

www.rapidwright.io

XILINX® RESEARCH LABS