Design Virtualization for Mainstream Reconfigurable Computing

CHREC NSF Center for High-Performance Reconfigurable Computing

James Coole, David Wilson, Austin Baylis, and Greg Stitt

UF UNIVERSITY of FLORIDA

FPGA Development Woes

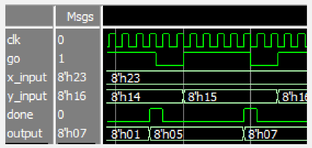
RTL Design

Requires designers with expertise in digital design and specialized languages

```
use ieee.std_logic_1164.all;
entity my_design is
port (
    clk : in std_logic;
```

Low-level Debugging

Involves lengthy verification and cycle-by-cycle analysis of signal waveforms



Limited Reuse

Device-specific bitstreams and vendor-specific tool flows inhibit design reuse

Lengthy Compilation

Large designs may take up to days/weeks to place-and-route on FPGAs

FPGA Overlays

Virtual coarse-grained architectures intended to alleviate FPGA productivity issues through abstraction

Application Layer

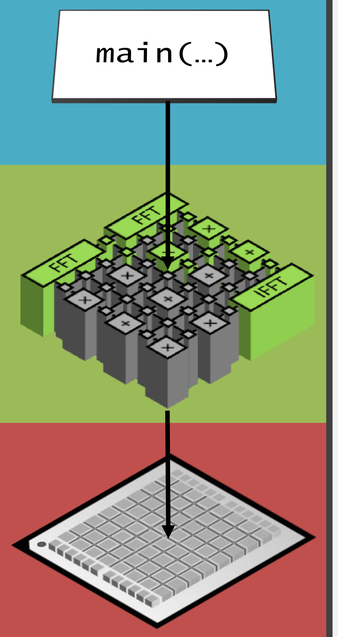
App users write applications in high-level languages

Overlay Abstraction Layer

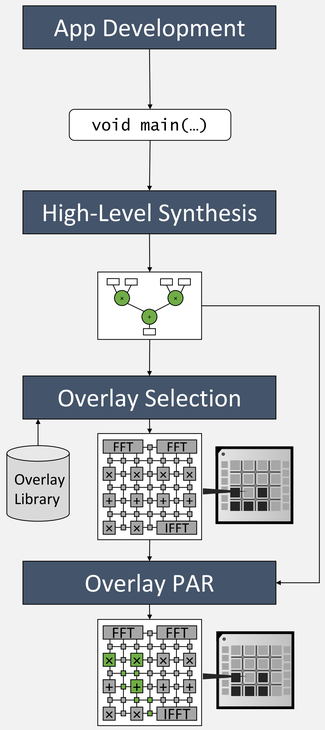
Overlay appears to app as a reconfigurable circuit of app-level resources

FPGA Layer

Overlay built on FPGA and abstracts FPGA resources



Overlay+HLS Design Flow



1. Application Development
User develops application in high-level language

2. High-level Synthesis
High-level synthesis (HLS) compiles app into an intermediate representation (IR)

3. Overlay Selection
HLS selects or synthesizes an overlay that can support IR's resource needs

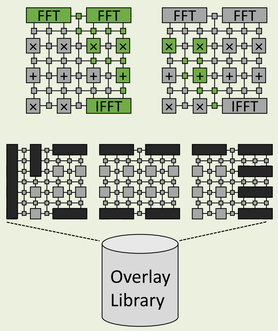
4. Overlay PAR
IR is virtually placed-and-routed (PAR) on overlay at runtime

Productivity Advantages

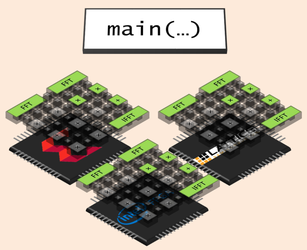
Fast Compilation

The overlay design process uses multiple-levels of reconfiguration to enable iterative design flows

- **Virtual Reconfiguration**:
  Overlays designed with innate reconfigurability to enable run-time changes
- **Context Reconfiguration**:
  Overlay library stores overlay instances with different resources to meet different demands
- **FPGA Reconfiguration**:
  Resort to synthesis only when there are no overlay instances that can meet requirements



Design Reuse

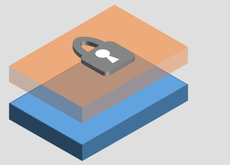


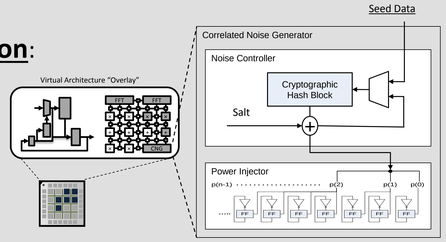
- **Design Portability**:
  Virtual bitstreams are portable to any FPGA that supports its respective overlay instance
- **Compilation Reuse**:
  Reuse overlay compilation runs for different applications with similar resource needs

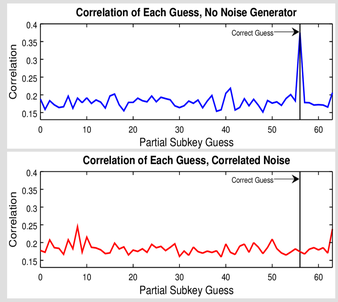
Other Features

Hardware Security

- Attackers must extract both overlay and FPGA bitstreams to steal IP
  - Overlay instance may include countermeasures to protect overlay bitstream
- **Correlated Noise Generation**:
  - Correlated noise generation protects overlay bitstream from side channel analysis



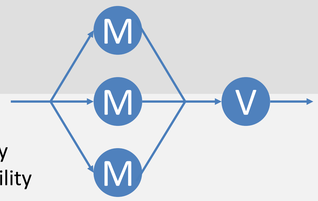




- Correlated Noise Generation (CNG) successfully mitigates CPA attack
- With CNG, no significant correlation forms
- Highest correlation is at incorrect subkey guess

Reliability

- Use overlay PAR to dynamically and transparently apply redundancy to application circuit for reliability



Results

Kernel Averages	Compilation Time	Supernets Clock (Nominal)	Supernets Area (% LUTs)	Intermediate Fabrics Desktop Time	Intermediate Fabrics Area (% LUTs)	Selectively Enabled (Direct) Compilation Time	Selectively Enabled (Direct) Clock	Selectively Enabled (Direct) Area (% LUTs)	Speedup Compilation Time	Overhead Clock	Overhead Area
FIR 16, Gaussian 4x4, Sobel 3x3	0.23s	260MHz	1.5%	0.27s	13.4%	187s	410MHz	0.9%	845×	36.6%	1.6×
Context 1 total	0.68s			0.82s		561s			826×		
Bilinear, Mean 4x4, Threshold 4x4	0.04s	276MHz	1.9%	0.12s	10.8%	1671s	337MHz	2.8%	62,030×	17.7%	0.7×
Context 2 total	0.11s			0.36s		5013s			43,619×		
Max 4x4, Min 4x4, Normalize 3x3, SAD 3x3	0.07s	266MHz	5.1%	0.12s	16.2%	615s	346MHz	5.9%	7,077×	18.5%	0.9×
Context 3 total	0.30s			0.48s		2460s			8,208×		
FIR 16 tap, Gaussian 4x4, Sobel 3x3 FLT	0.26s	315MHz	15.3%	0.18s	40.4%	566s	356MHz	27.9%	2,160×	11.6%	0.5×
Context 4 total	0.79s			0.55s		1699s			2,152×		
Bilinear Mean 4x4, Threshold 4x4, Max 4x4, Min 4x4, Normalize 3x3, SAD 3x3 FLT	0.08s	329MHz	14.6%	0.12s	48.0%	443s	356MHz	48.1%	6,674×	7.5%	0.3×
Context 5 total	0.55s			0.86s		3102s			5,610×		
Per-kernel average	0.12s		7.7%	0.15s	25.8%	642s		17.1%	13,506×	16.2%	0.4×
System total	2.44s		38.4%	3.06s	128.8%	12835s		85.7%	5,267×		

Experiments

Evaluation of OpenCL HLS onto overlays compared to RTL synthesis onto FPGAs using several computer-vision OpenCL kernels

- **Between Overlays:**
  Large design space between flexibility and overhead in overlay interconnect
  - Island-style interconnect in **Intermediate Fabrics (IFs)** enables flexible kernel mappings
  - Tailored interconnect in **Supernets** are up to 8.9x smaller than IFs
- **Overlays vs. RTL:**
  Overlay compilation orders of magnitude faster than RTL at reasonable overhead
  - Up to 13Kx faster compiles than direct-to-FPGA compilation @ 0.15s per kernel
  - 2.5s for system as a whole vs 3.6 hours
  - Up to 70% lower area vs. separate datapaths (60% for system)