2006

Policy-Driven Memory Protection for Reconfigurable Hardware [presentation]

Huffmire, Ted

Ted Huffmire, Shreyas Prasad, Tim Sherwood, and Ryan Kastner, Policy-Driven Memory Protection for Reconfigurable Hardware. Proceedings of the 11th European Symposium on

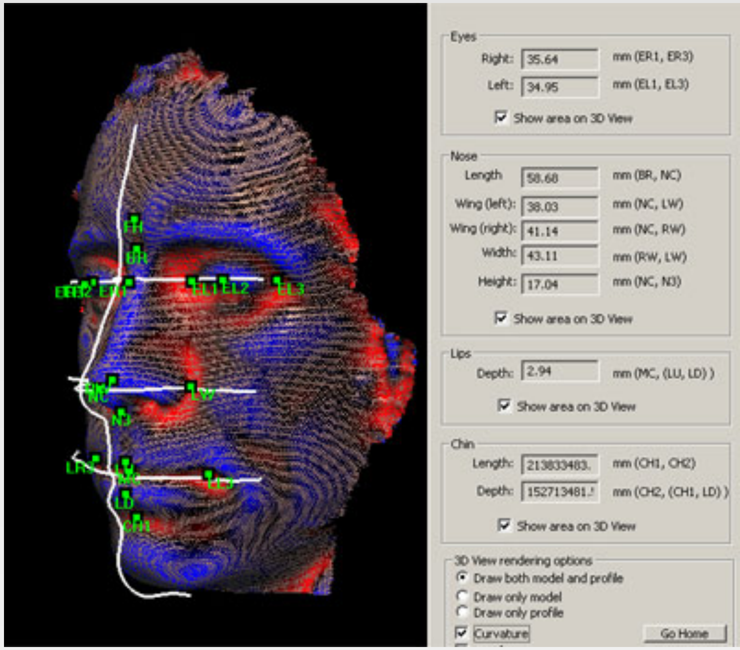
Policy-Driven Memory Protection for Reconfigurable Hardware

Ted Huffmire, Shreyas Prasad,
Tim Sherwood, and Ryan Kastner

www.cs.ucsb.edu/~arch/RCsec

reconfigurable security

FPGA Systems are ubiquitous



Eyes

Right: 35.64 mm (ER1, ER3)

Left: 34.95 mm (EL1, EL3)

☑ Show area on 3D View

Nose

Length 58.68 mm (BR, NC)

Wing (left): 38.03 mm (NC, LW)

Wing (right): 41.14 mm (NC, RW)

Width: 43.11 mm (RW, LW)

Height: 17.04 mm (NC, N3)

☑ Show area on 3D View

Lips

Depth: 2.94 mm (MC, (LU, LD))

☑ Show area on 3D View

Chin

Length: 213833483. mm (CH1, CH2)

Depth: 152713481.! mm (CH2, (CH1, LD))

☑ Show area on 3D View

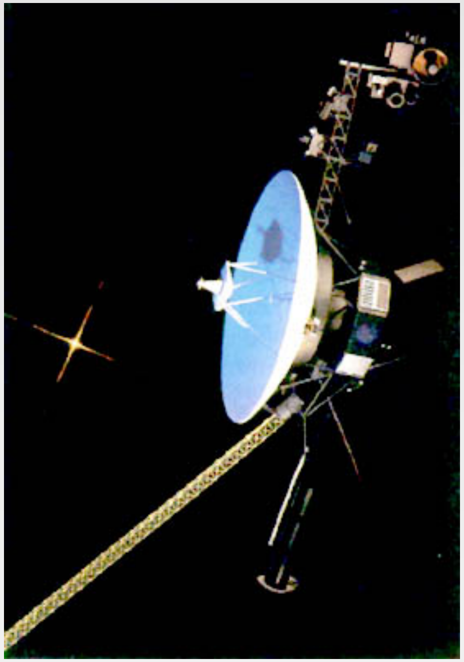
3D View rendering options

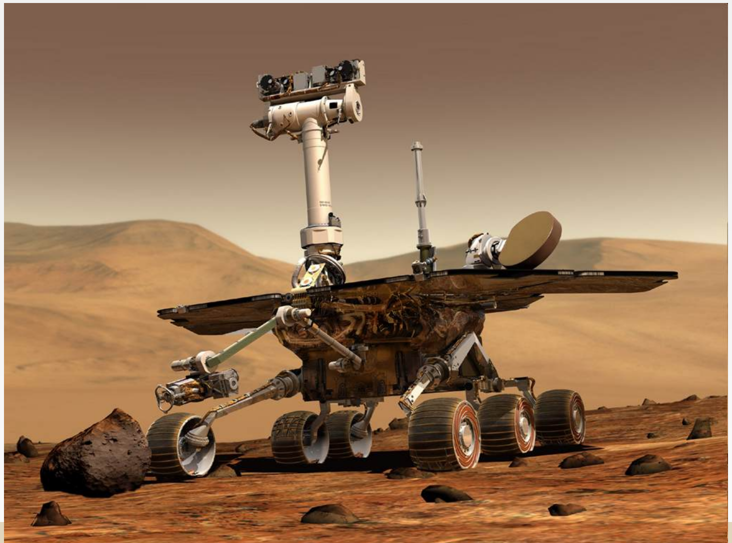
● Draw both model and profile

○ Draw only model

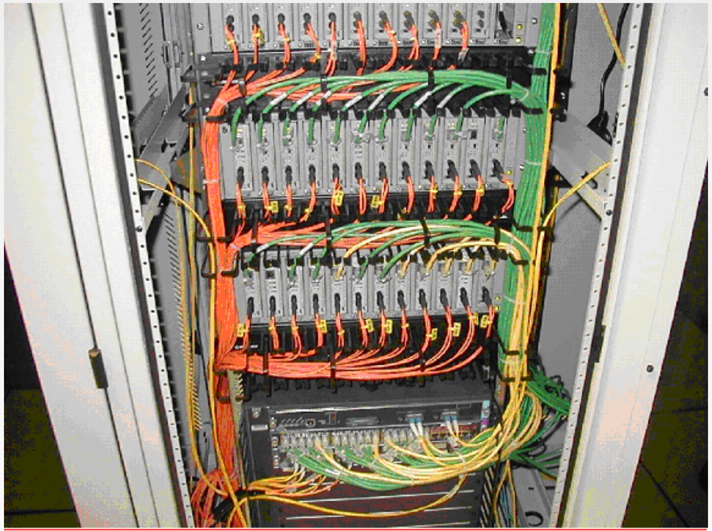
○ Draw only profile

☑ Curvature

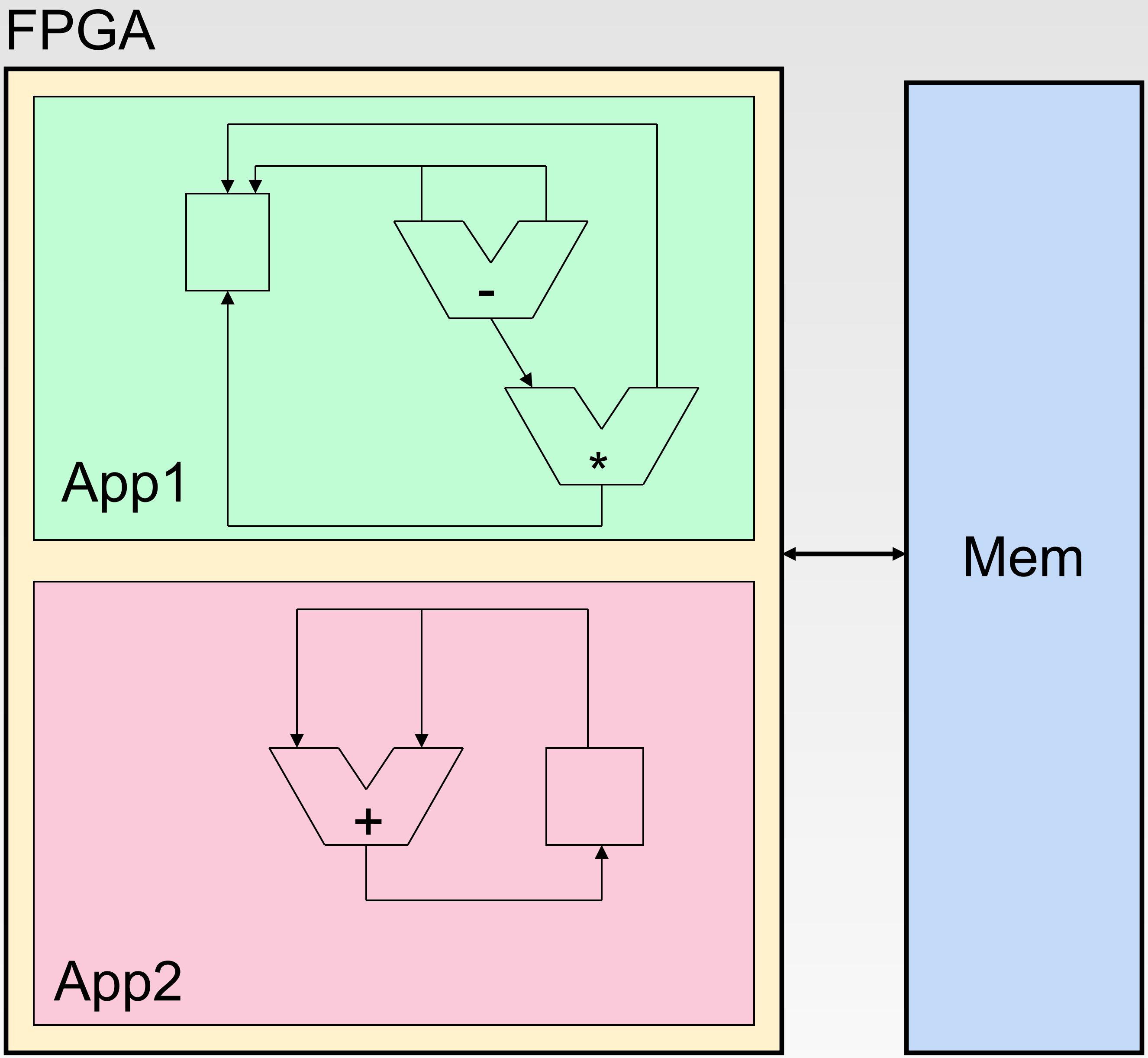




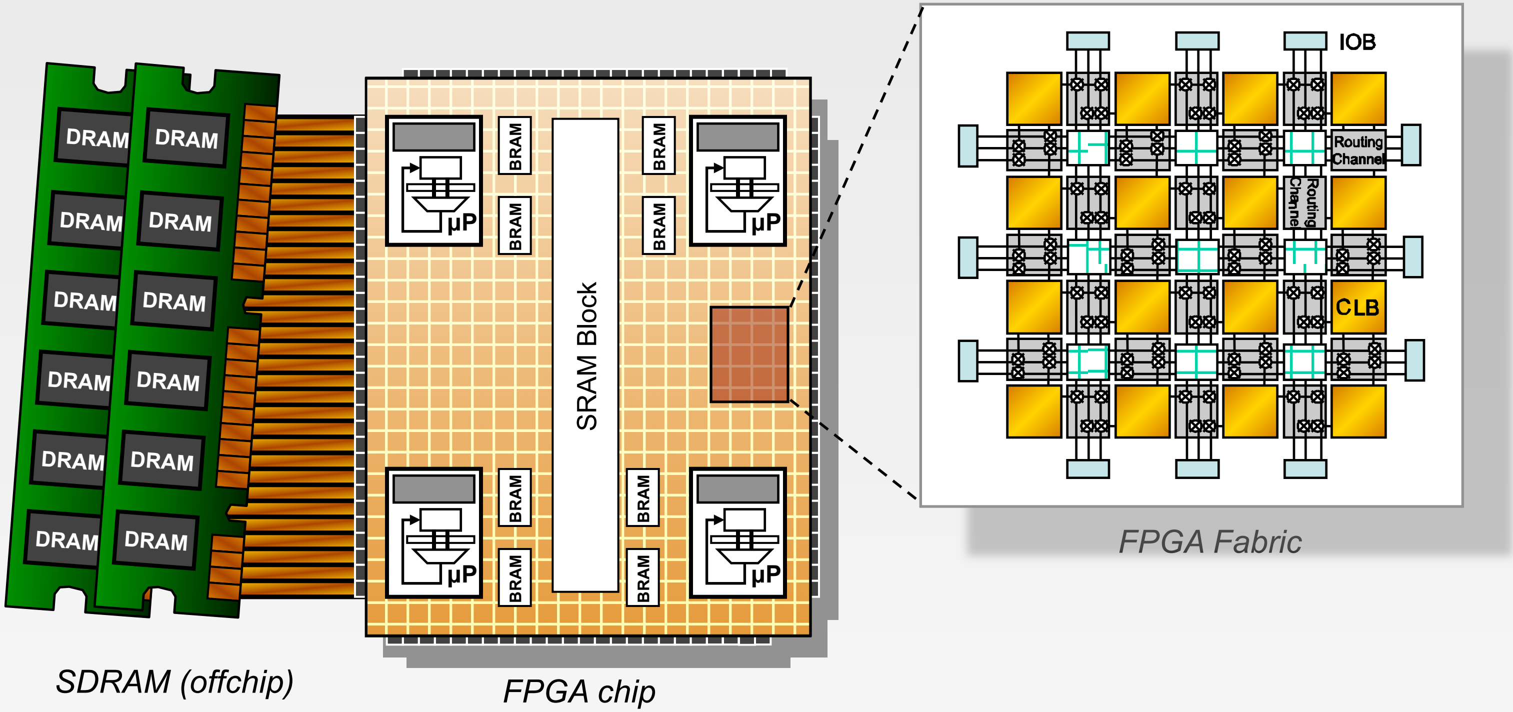
FPGA Systems are ubiquitous



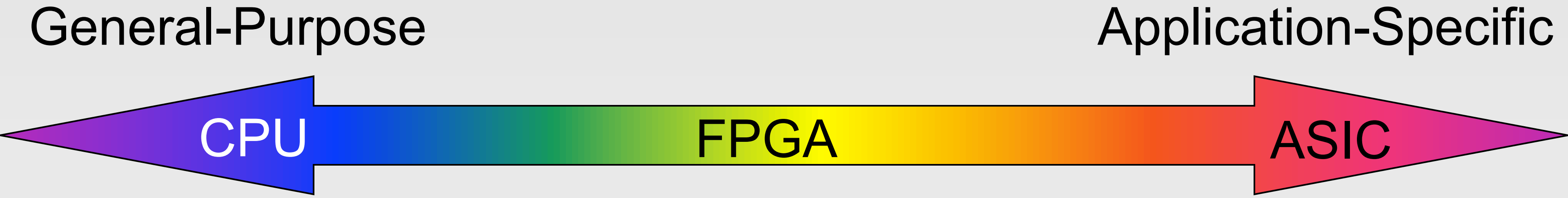
What is an FPGA?



What is an FPGA?



Why are FPGAs desirable?



- **Fabrication, Verification Cost**
- **IP is vulnerable during fabrication**
- **Parallelism → Throughput**
- **Updatable**

However…

- Security is an afterthought at best

- Fundamental security primitives do not yet exist

- Goal: Start building those primitives

- Opportunity to leverage the benefits of hardware
  - Low-overhead stateful reference monitors

- Separation: a very important primitive

Separation

- Multiple Cores on one chip

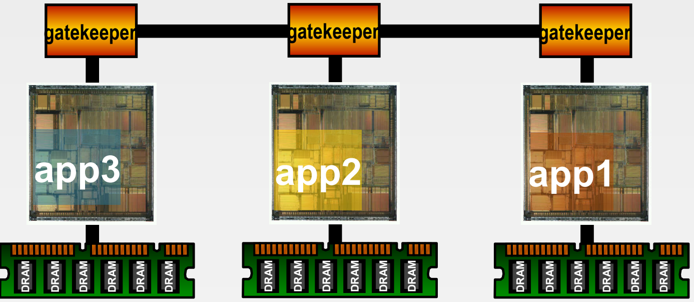
- Cores may have different trust levels and clearance levels

- Cores share resources
  - Logic
  - Memory

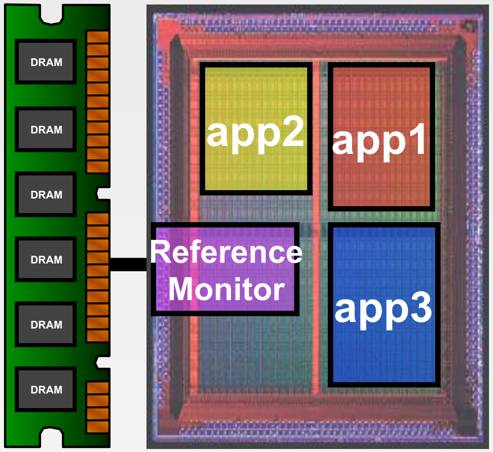
- Separation: controlled sharing of memory between cores

Separation Alternatives

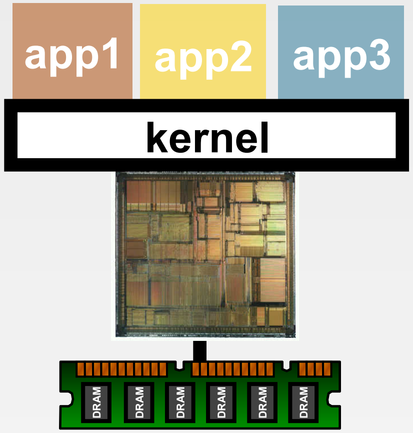
Separate Processors

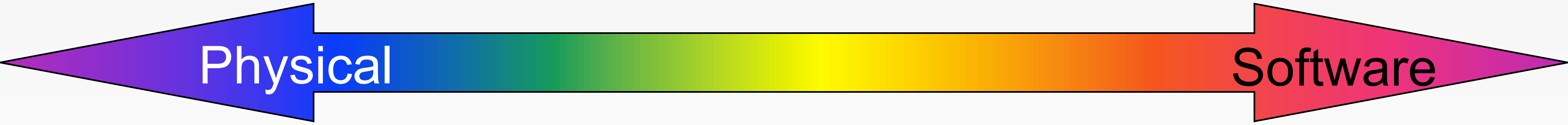


Reconfigurable Separation



Separation Kernels





Why reference monitors?

- Provides a well-understood foundation for controlled sharing [Anderson 72]

- Standard memory protection does not make sense for FPGA systems

- Separation kernels [Irvine et al. 04] are a software-based scheme that won't work for embedded applications that lack code

- Modern processors have more state in the hardware, making kernel development harder

- Need to protect the integrity of the reference monitor

A Memory Protection Language

- Exploit the fine-grained reprogrammable nature of FPGAs

- All modules on chip must obey a *memory access policy*
  - Ensured via the architecture
  - Formal, mathematically precise

- Memory protection policies are expressed in the language
  - Formal Top Level Specification (FTLS)

- Compiler translates the policy FTLS to a circuit

Formal Memory Protection Specifications

- A precise language of legal accesses
  - Subjects (Modules)
  - Access Rights
  - Objects (Memory Ranges)
- Fixed (Stateless) Models
  - e.g., B&L, Biba
- Transitional (Stateful) Models
  - e.g., Chinese Wall, high water mark

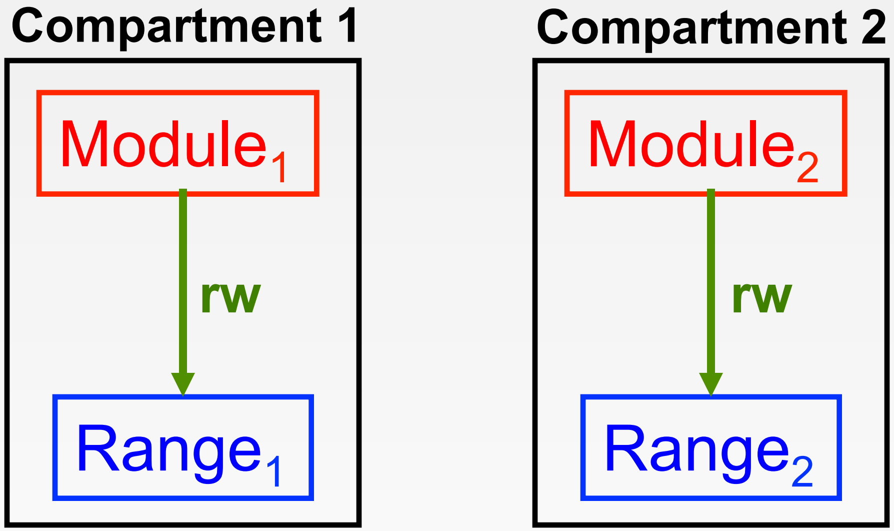
Isolation Example

- A fixed (stateless) model

- Each core is restricted to a fixed range (or set of ranges) of memory

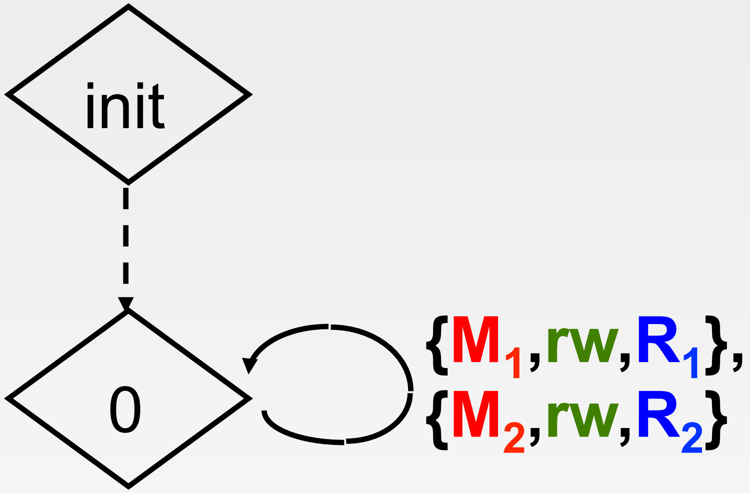
- Each range can only be assigned to one core

Access→{Module$_1$,rw,Range$_1$} | {Module$_2$,rw,Range$_2$};
Policy→(Access)*;



Policy Compiler

1. Policy FTLS:
   - Access→{Module$_1$,rw,Range$_1$} | {Module$_2$,rw,Range$_2$};
   - Policy→(Access)*;
2. Regular Expression:
   - ({Module$_1$,rw,Range$_1$} | {Module$_2$,rw,Range$_2$})*
3. Minimized DFA:



4. Verilog HDL:
   - case({module_id,op,r1,r2})
     - 9’ b011110: //Module$_1$,rw,Range$_1$
       - state=s0;
     - 9’ b101101: //Module$_2$,rw,Range$_2$
       - state=s0;
     - default:
       - state=s1; //reject
   - endcase

What we have done

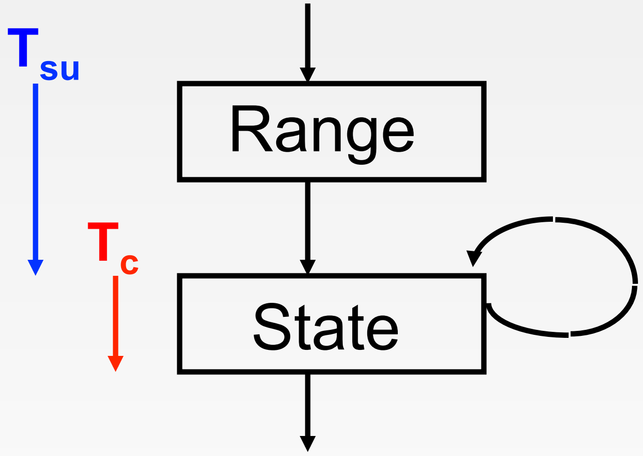
- Automated design flow from FTLS to synthesized circuit

- Language has a well-defined grammar

- Powerful enough to express a variety of policies that we have compiled and tested
  - Chinese Wall
  - Redaction
  - Access Control List
  - Secure Hand-off

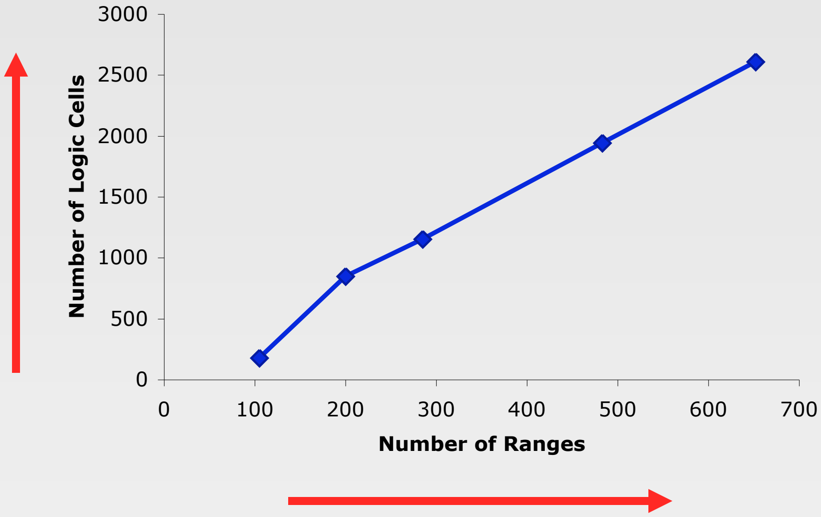
Methodology

- Constructed several isolation policies
  - Varied the number of ranges
- Used Quartus to synthesize
- Measured:
  - Area (Logic Cells)
  - Setup Time
  - Cycle Time

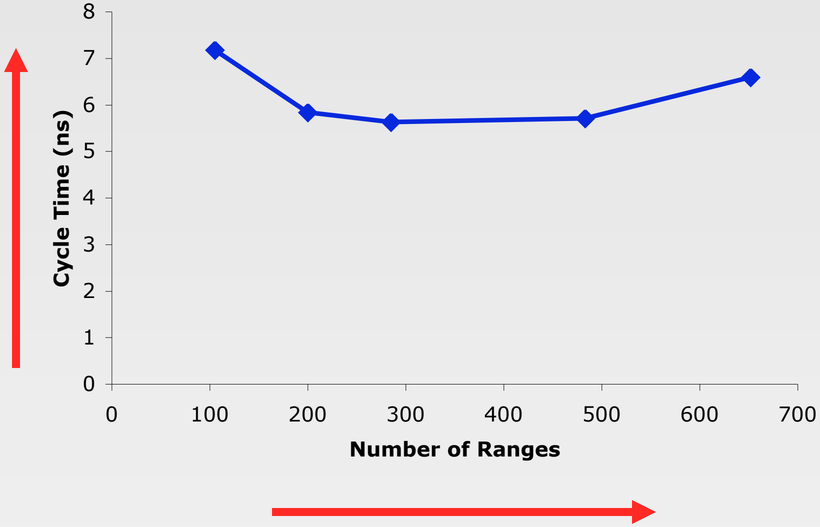


Synthesis Results

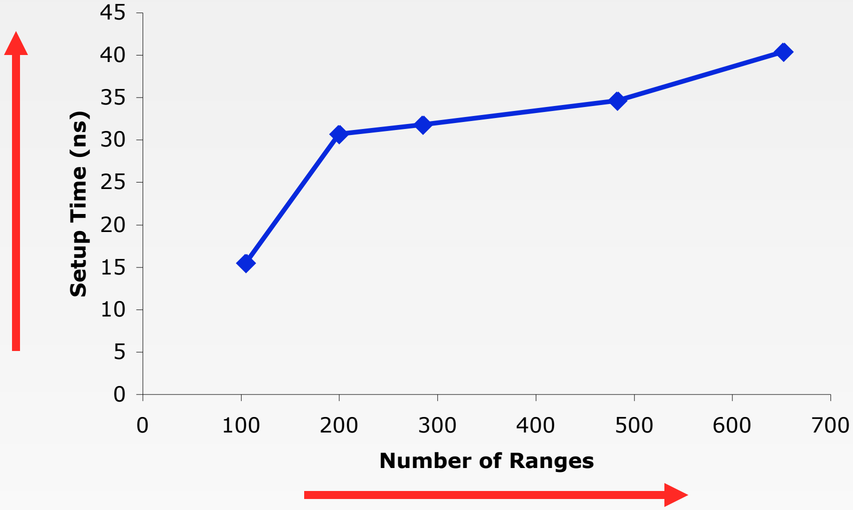
Circuit Area vs. Number of Ranges



Cycle Time vs. Number of Ranges



Setup Time vs Number of Ranges



Future Work

- A higher level language
  - Abstract formal security policy model
- Verify correctness of automatic translation
  - Model - FTLS - Verilog - circuit
  - Verify the model and FTLS using formal methods
- Information flow policies
- Dynamic policies
- Evaluate on a realistic embedded application

Acknowledgements

- NPS CISR

- NSF Grant CNS-0524771, Adaptive Security and Separation in Reconfigurable Hardware

- Andrei Paun and Jason Smith of Louisiana Tech University

Questions?

- huffmire@cs.ucsb.edu

- www.cs.ucsb.edu/~arch/RCsec