

# Using Sequencing to Trigger a Better Analysis

Nathan Coopriider

John Regehr

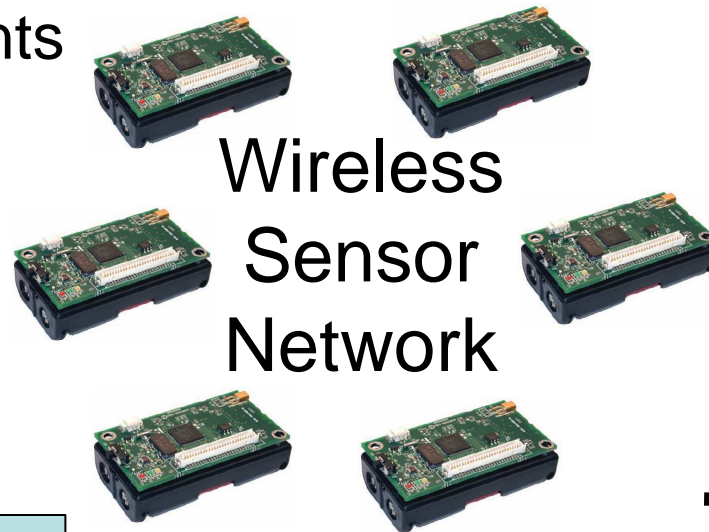
University of Utah  
School of Computing

# Static analysis helps

Deeply embedded

Severe constraints

- Power
- Size
- Features
- Price



Wireless  
Sensor  
Network

Static analysis

Verification

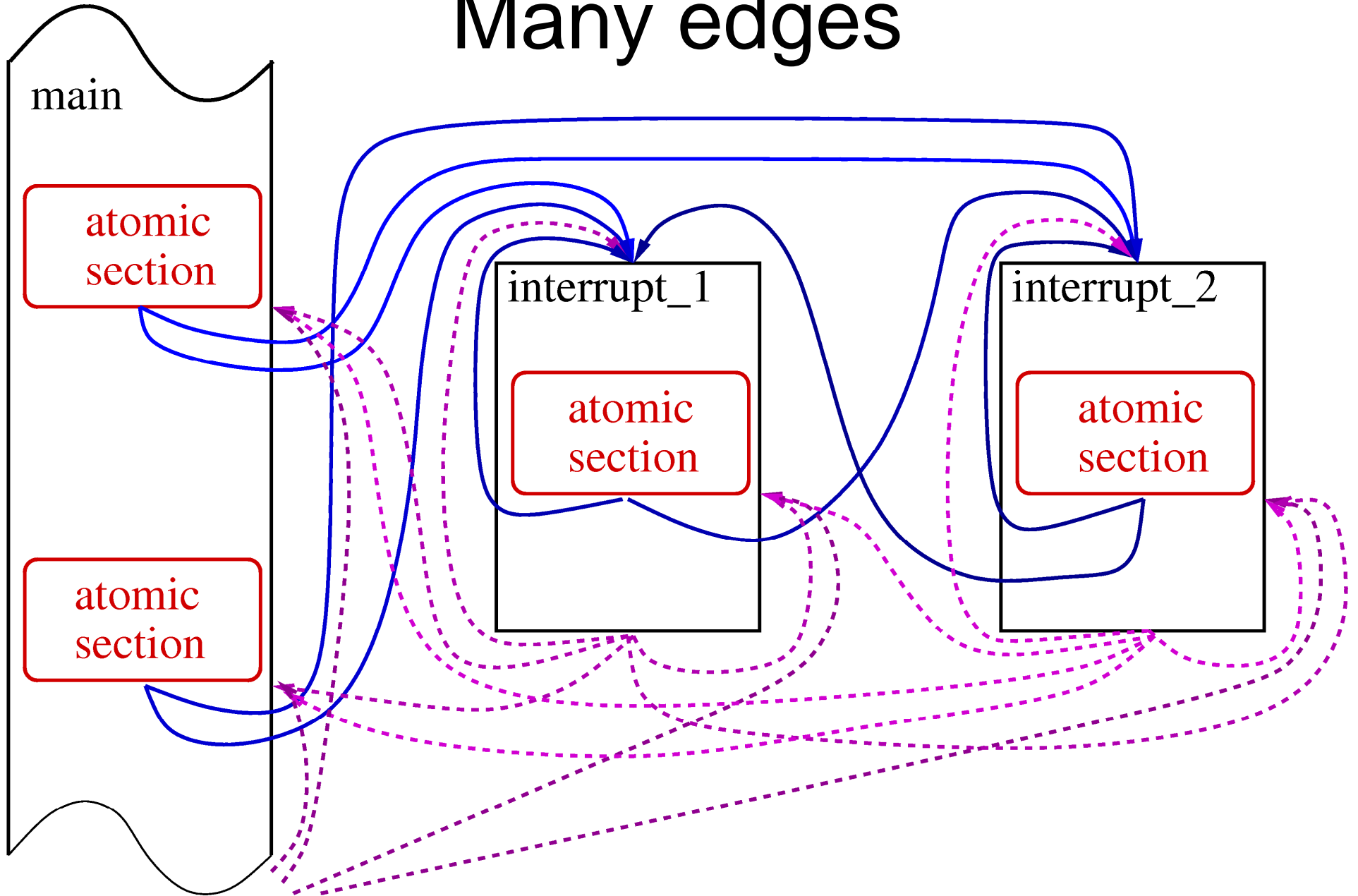
Optimization

cXprop

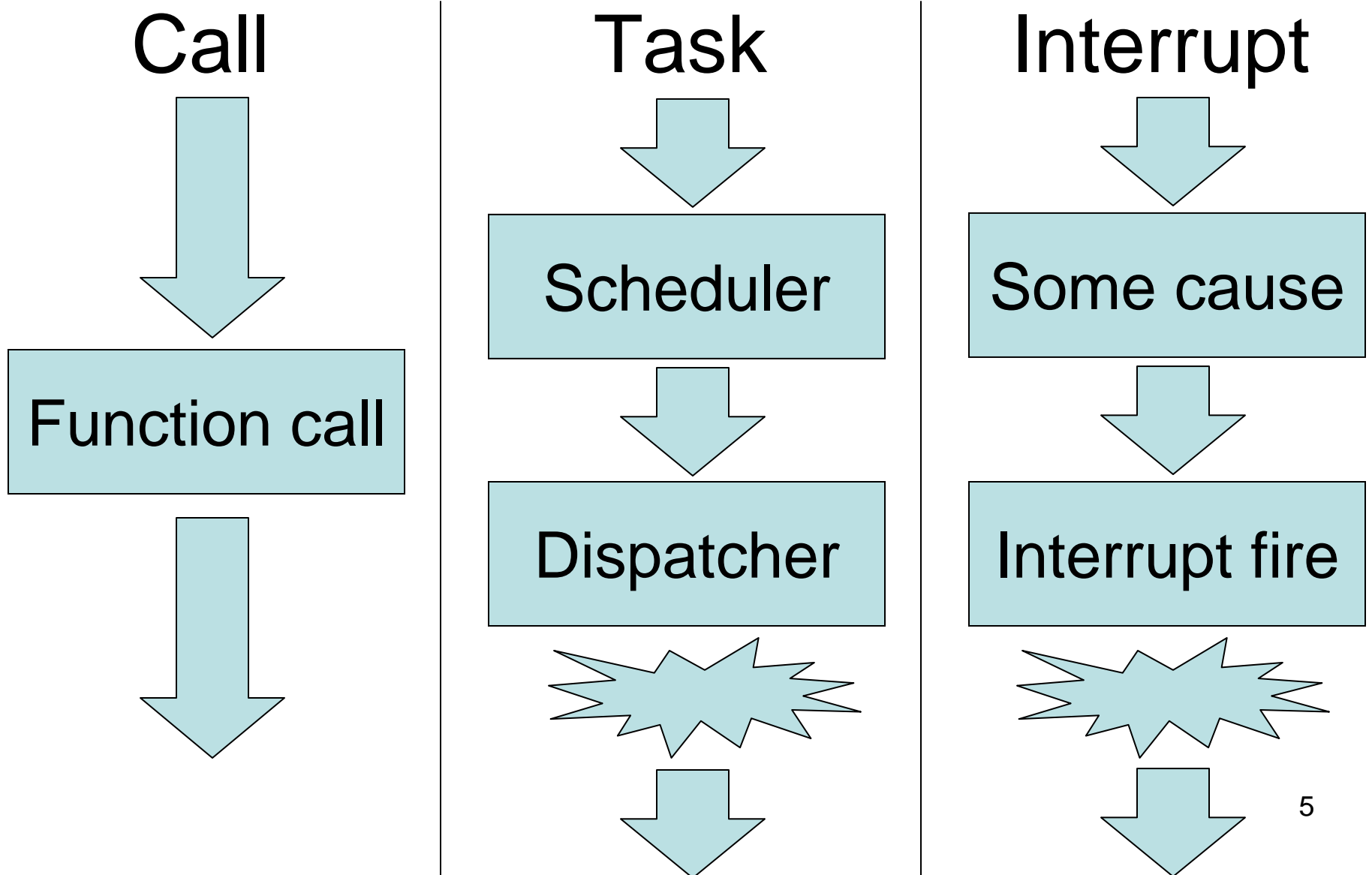
# Tricky Control-flow Cases

- Central to analysis
- Hidden dependencies exist
  - Task dispatch and concurrency
- Control-flow successors
  - Use non-local information
  - Analyzer must be conservative
- Goal: Capture these implicit dependencies
  - Better optimization and verification

# Many edges



# Control-flow dependencies

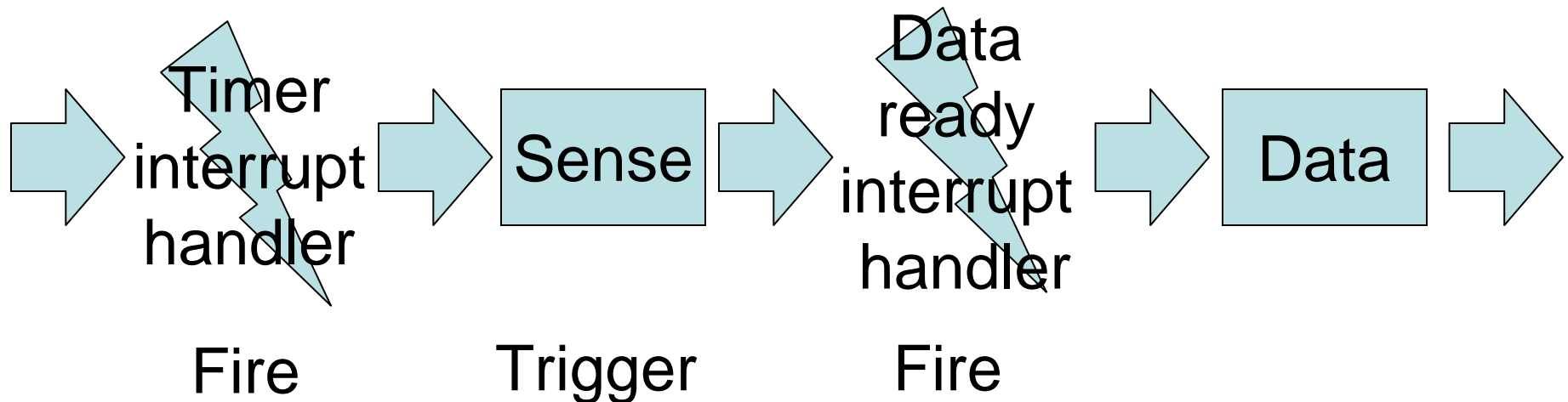


# Triggers

- Causes called *triggers*
- Implicit control-flow hidden from analysis
- Understanding triggers
  - Tasks
    - Scheduled before dispatched
  - Interrupts
    - Sometimes applications asks for them
    - Determined by modeling devices or annotations

# Example

Control-flow

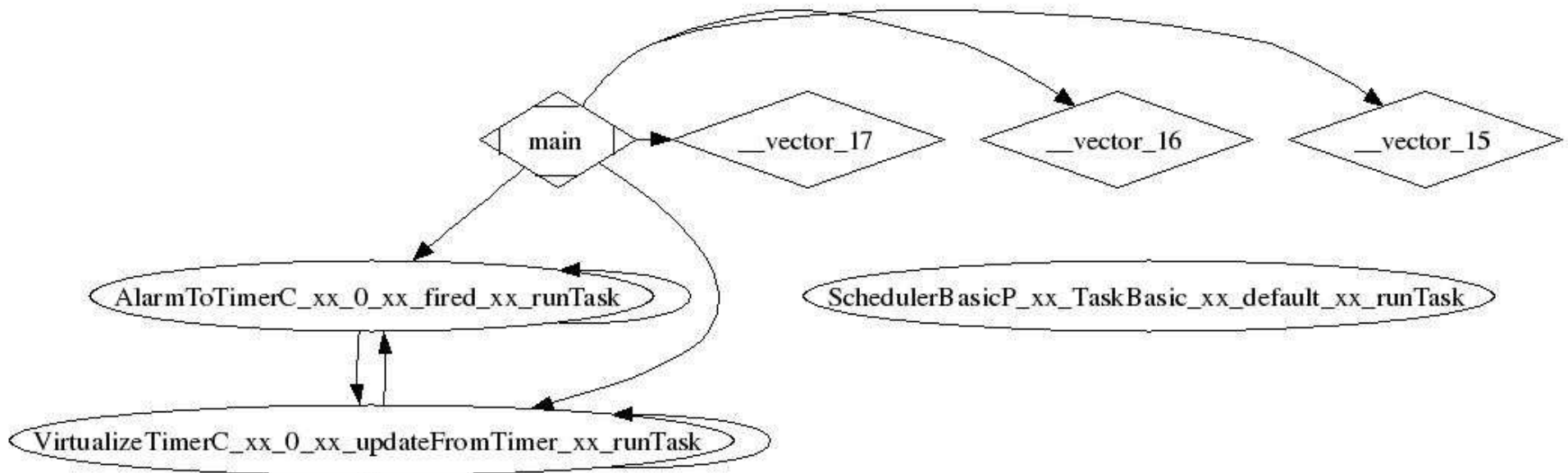


Timer trigger: On

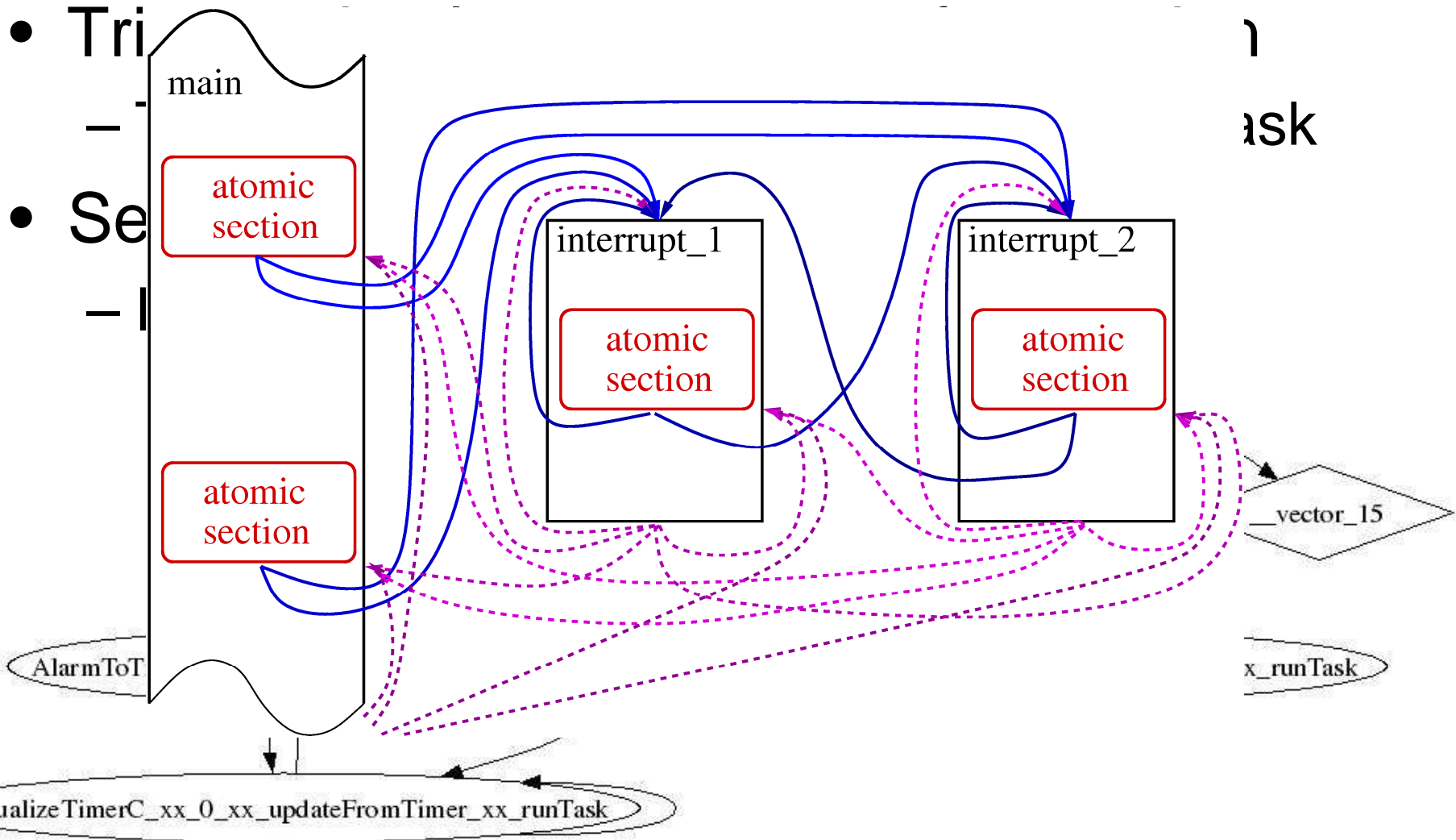
Data ready trigger: Off

# Sequencing

- Triggers imply a *sequence* of execution
  - Task triggers an interrupt that triggers a task
- Sequences form a graph
  - Remove unnecessary CFG edges



# Sequencing



# Summary

- Triggers identify implicit control-flow
  - Tasks and interrupts
- A chain of triggers forms a sequence
- Sequencing analysis augments the CFG
- More precise static analysis

<http://www.cs.utah.edu/~coop/research/cxprop>