LISA: Machine Description Language

Jared Wood

Outline

- LISA
  - Motivation
  - Description
  - Machine model
    - Requirements
    - Operation sequencer
  - L-charts

LISA: Motivation

- Accurate machine model
  - Bit- & cycle-/phase-accuracy
- Models:
  - ISA for compilers & instruction-set simulators
    - Too rough
  - HW design
    - Too detailed
- LISA: cover gap between models

LISA: Motivation

- Behavioral pipeline modeling
  - Pipeline controller for generic machine model
  - Parameterized by
    - Precedence constraints
    - Resource constraints

LISA: Description

- Operation-level description of pipeline
- Operations: register transfers during single control step
- Instructions: set of operations
- Control step:
  - Instruction-cycle
  - Clock-cycle
  - Phase-cycle

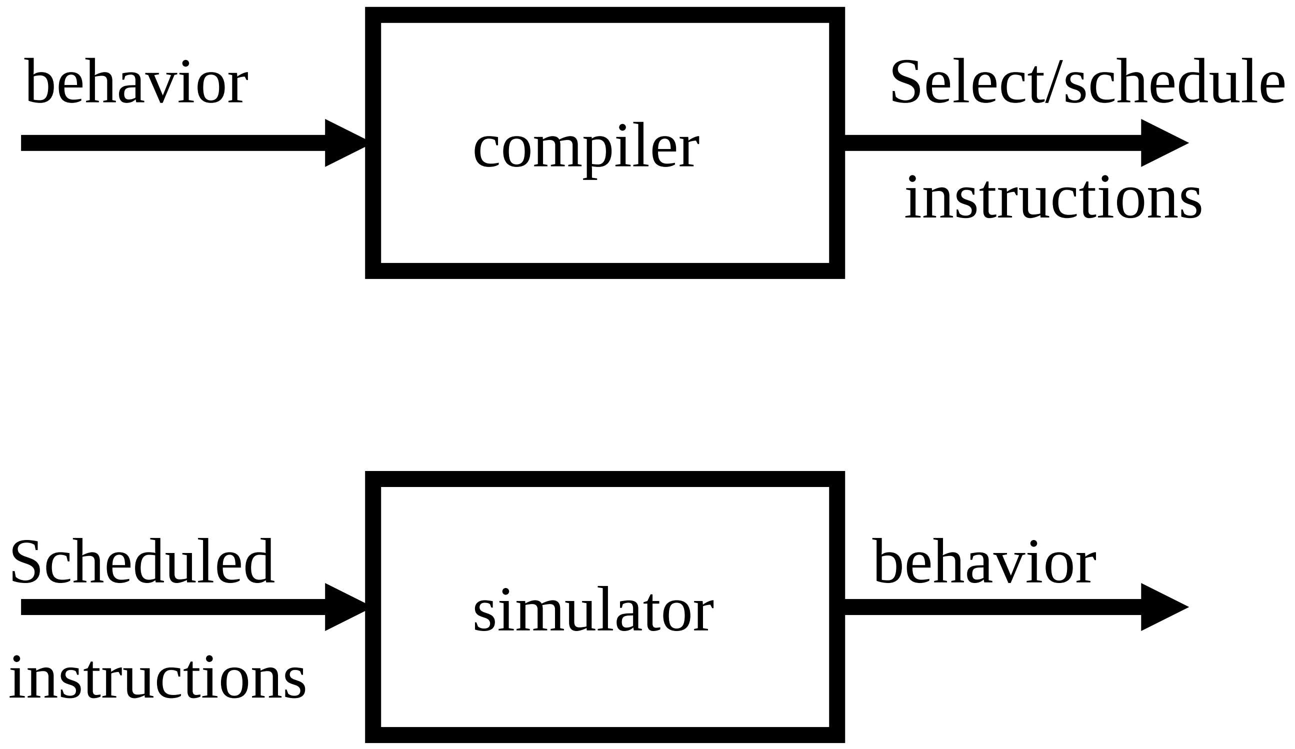
LISA: Description

- Operation scheduling→L-Charts
  - Specify:
    - Time and resource allocation
    - Operation sequencer w/ ASAP strategy
- Goal:
  - Single generic machine model
  - Single generic description language

LISA: Description

- Main applications so far:
  - Timed ISA simulation for HW/SW co-design
- Other possibility:
  - Compilation

LISA: Description



Machine Model: Requirements

- Application domain:
  - RT SW design
  - DSP/embedded system design
  - HW/SW co-verification
  - Architecture exploration

Machine Model: Requirements

- Processor class:
  - DSPs & microcontrollers
    - Low or medium complexity
    - Pipelined, VLIW, & RISC architectures

Machine Model: Requirements

- Model accuracy:
  - Timing:
    - Instruction, clock, or phase
  - Bit-accurate register transfers
  - Exact state modeling:
    - Pipeline, interrupt, & wait
  - Spatial accuracy:
    - SW-level: registers, memory
    - System-level: interrupts, peripherals
    - HW-level: pins
  - Control step state visibility

Machine Model: Operation Sequencer

- At control step t:
  - Admissible operations determined
    - Based on precedence & resource constraints
  - Transition function Ft formed
  - Ft applied to machine
  - Machine state changes

Machine Model: Operation Sequencer

Instruction n-1
{O1,O2,O3}

Instruction n
{P1,P2,P3}

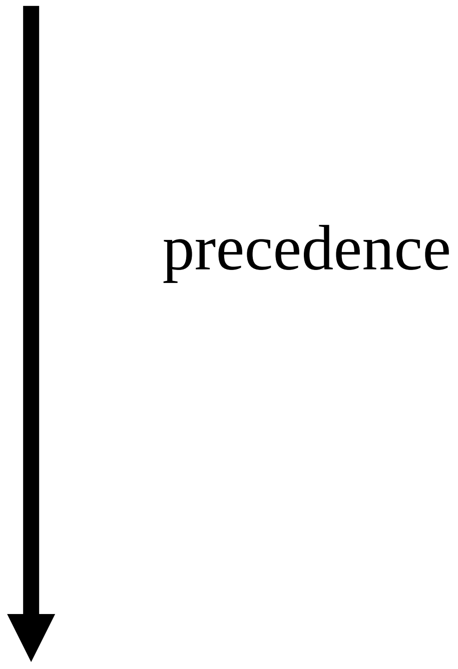
Instruction n+1
{Q1,Q2,Q3}

Machine Model: Operation Sequencer

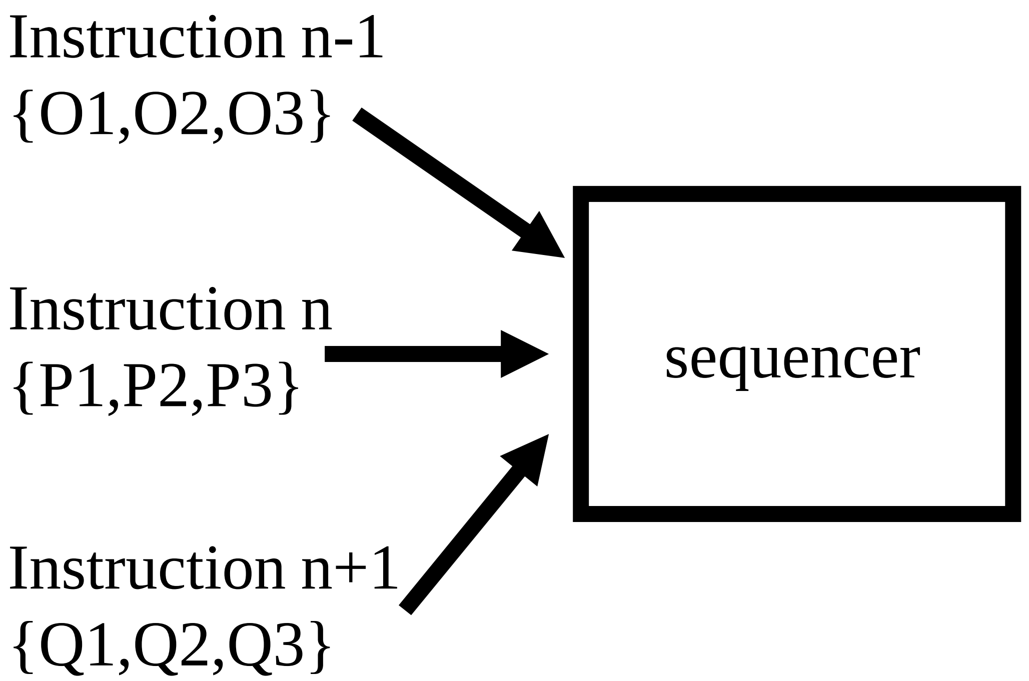
Instruction n-1
{O1,O2,O3}

Instruction n
{P1,P2,P3}

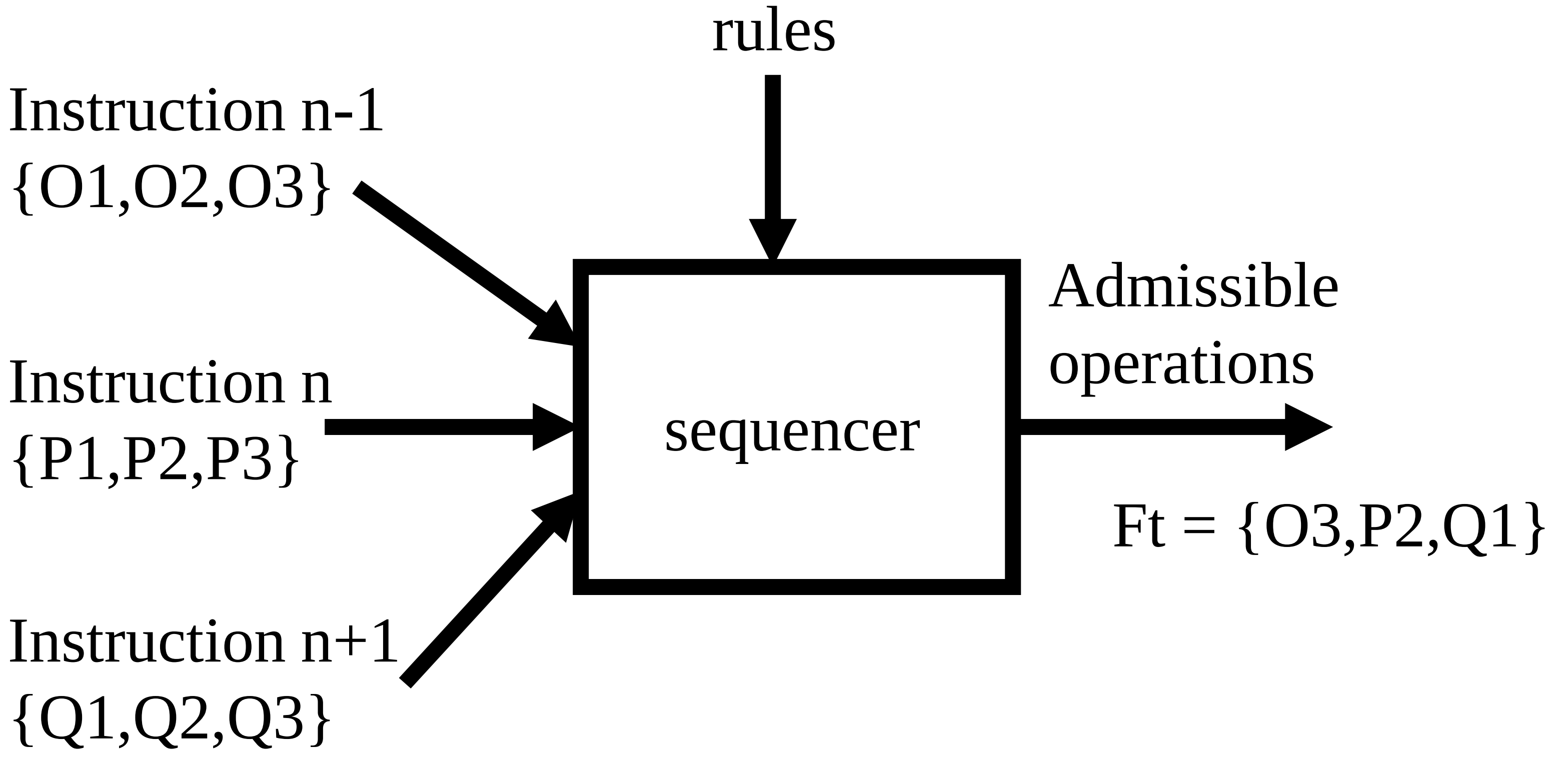
Instruction n+1
{Q1,Q2,Q3}



Machine Model: Operation Sequencer



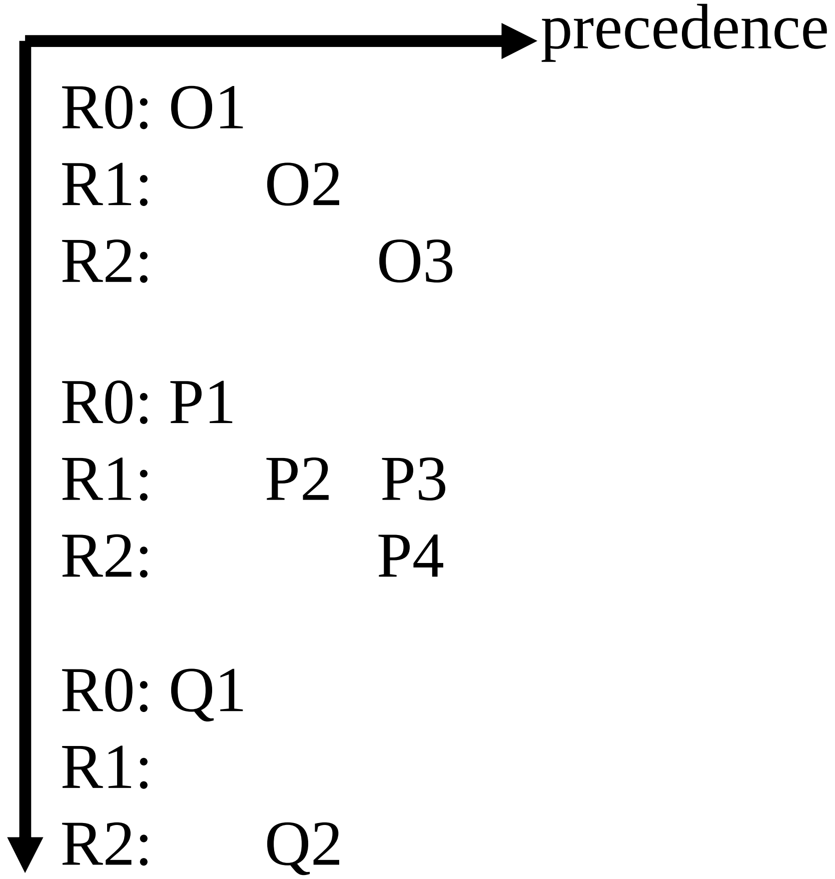
Machine Model: Operation Sequencer



L-Charts

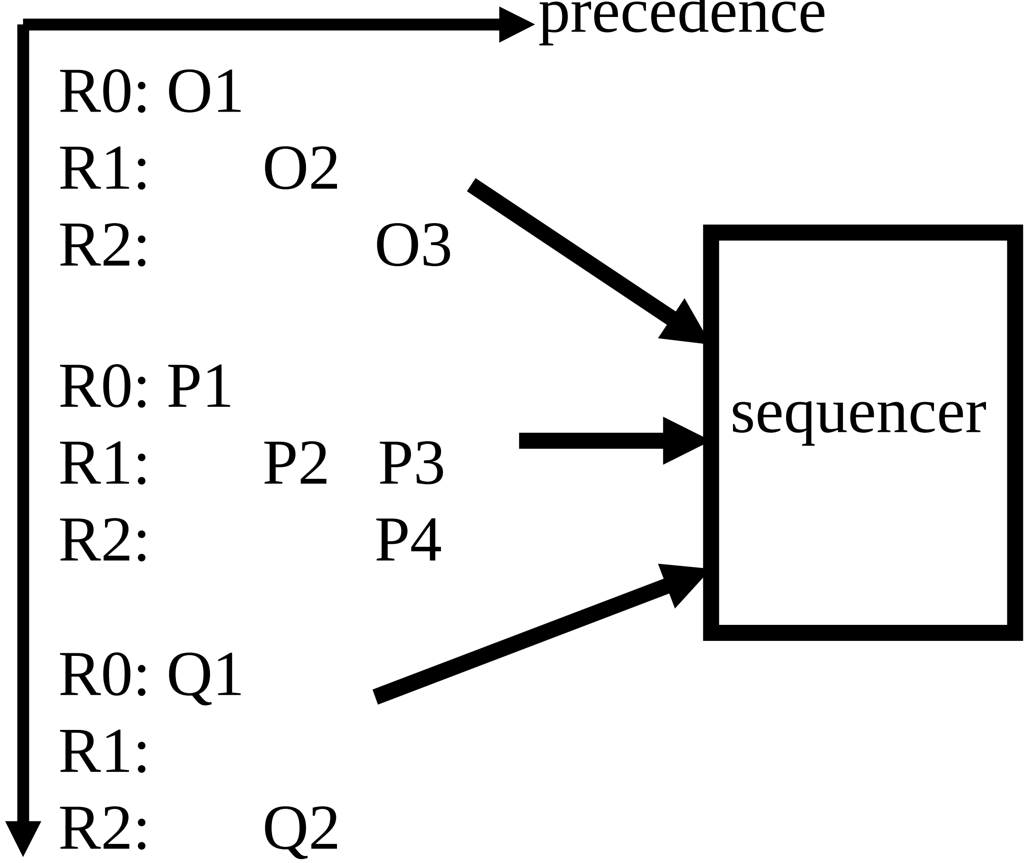
- Extended Gantt chart
  - Change time axis to precedence axis

L-Charts

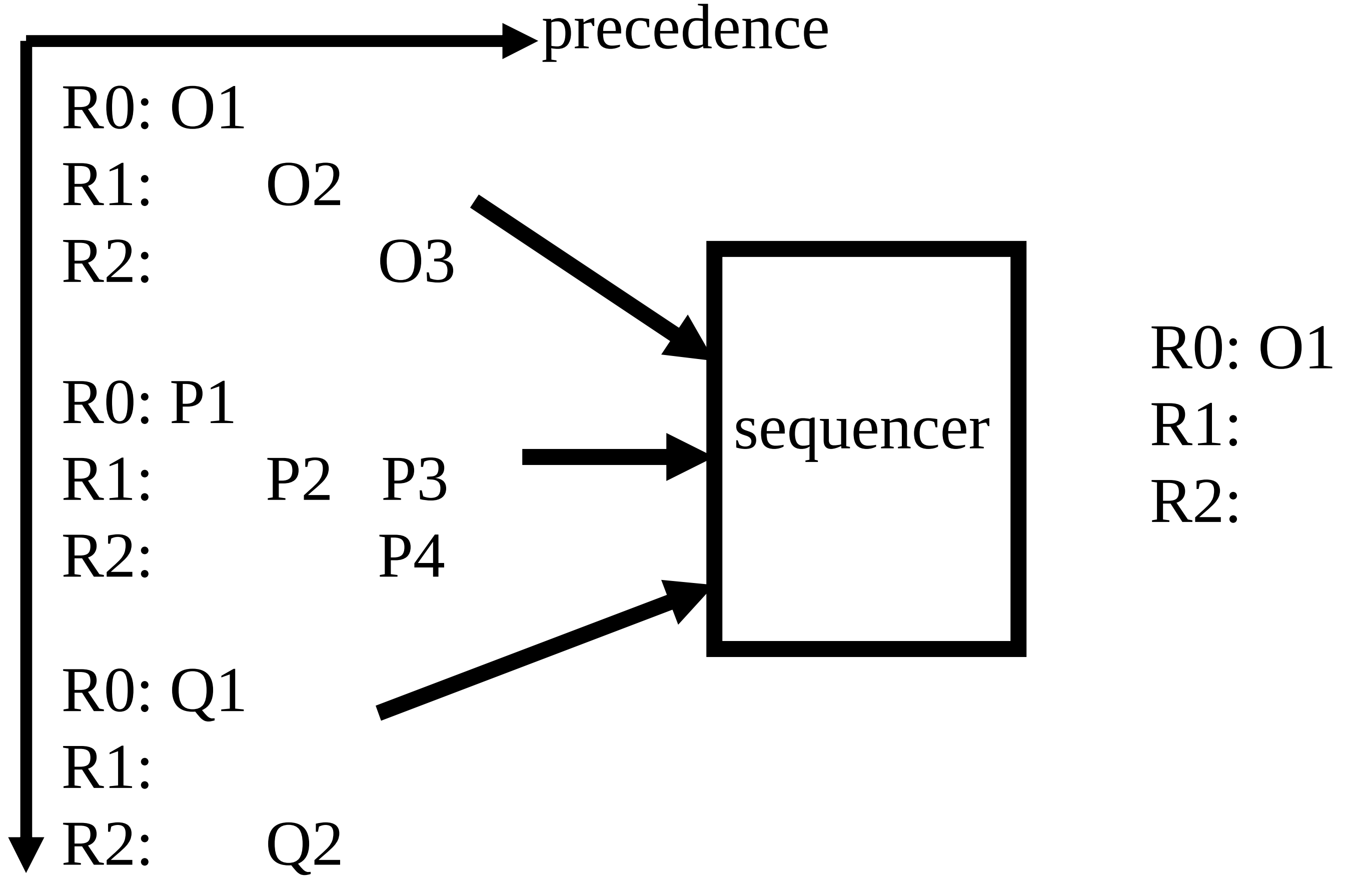


```
R0: O1
R1:       O2
R2:             O3

R0: P1
R1:       P2    P3
R2:             P4

R0: Q1
R1:
R2:       Q2
```

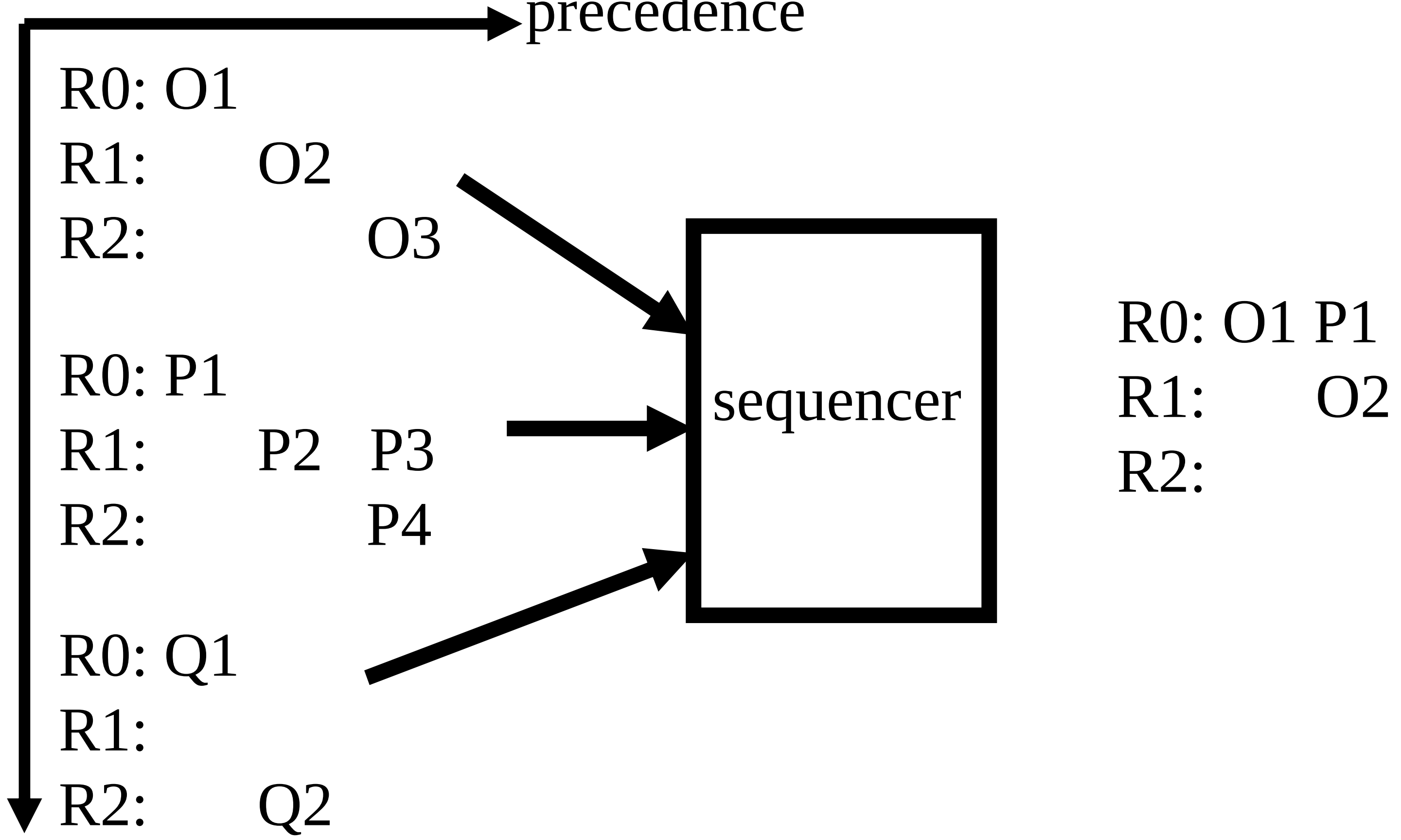
L-Charts


precedence
R0: O1
R1: O2
R2: O3
R0: P1
R1: P2 P3
R2: P4
R0: Q1
R1:
R2: Q2
sequencer

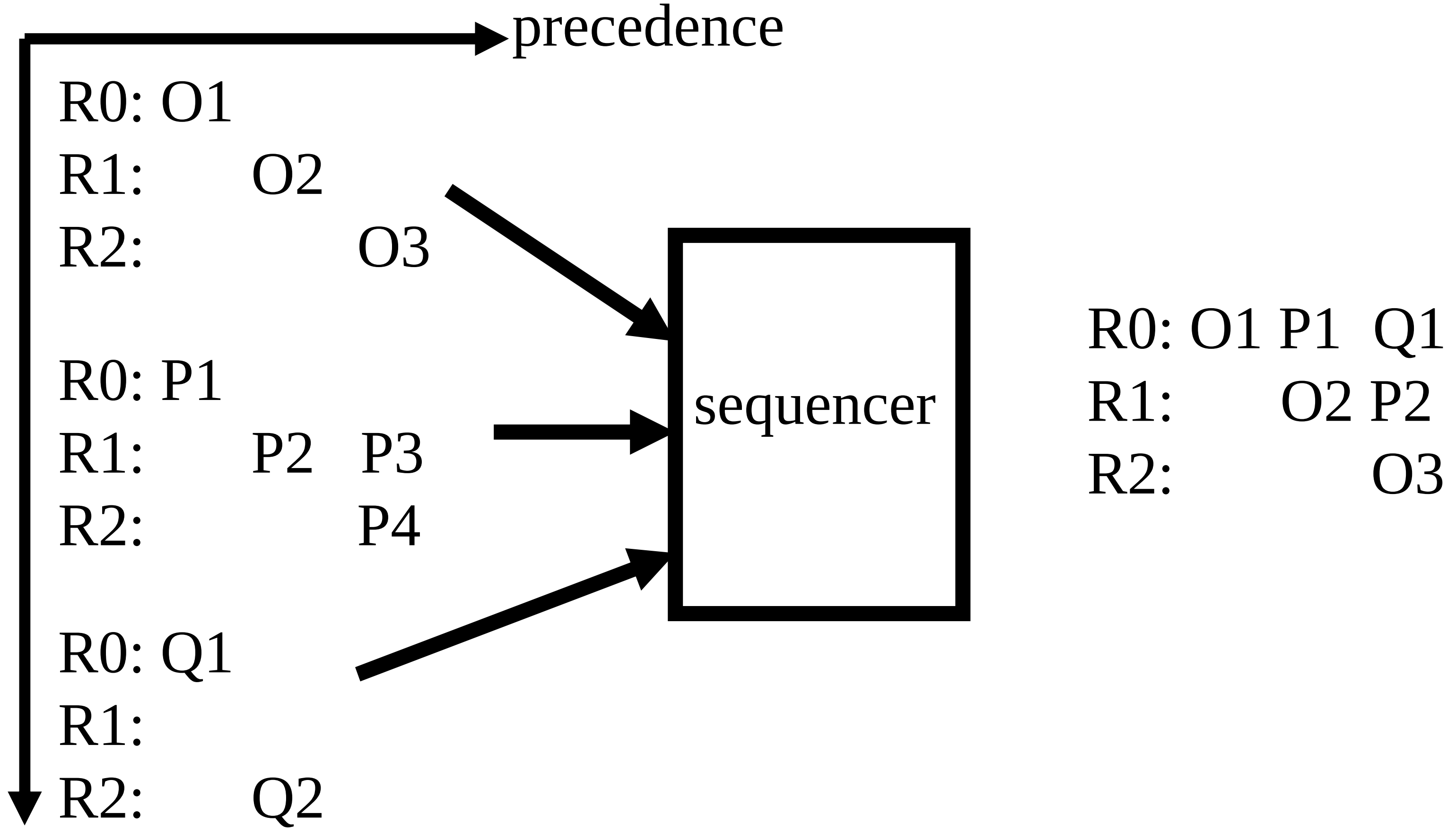
L-Charts


precedence
R0: O1
R1: O2
R2: O3
R0: P1
R1: P2 P3
R2: P4
R0: Q1
R1:
R2: Q2
sequencer
R0: O1
R1:
R2:

L-Charts

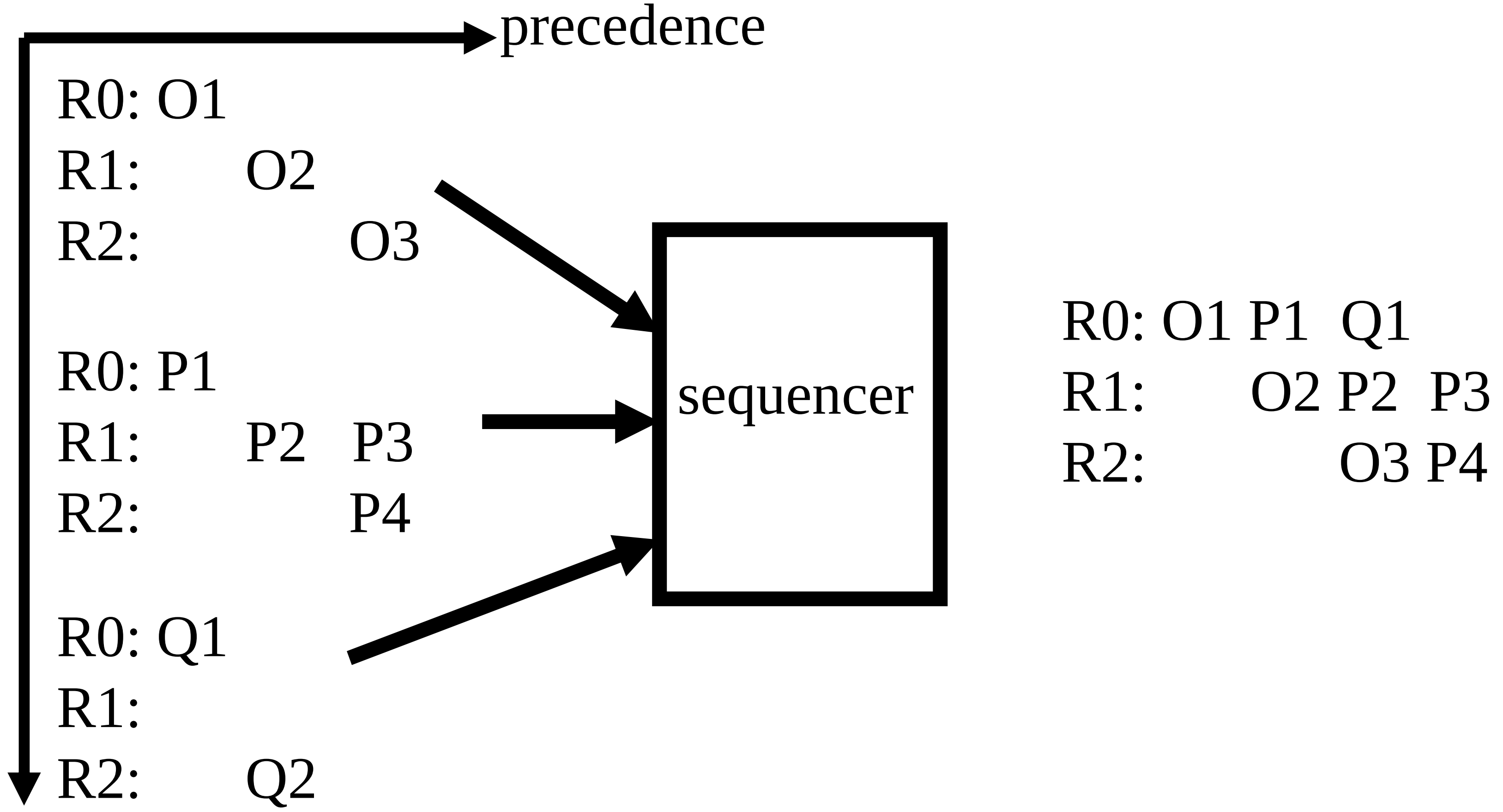

precedence
R0: O1
R1: O2
R2: O3
R0: P1
R1: P2 P3
R2: P4
R0: Q1
R1:
R2: Q2
sequencer
R0: O1 P1
R1: O2
R2:

L-Charts

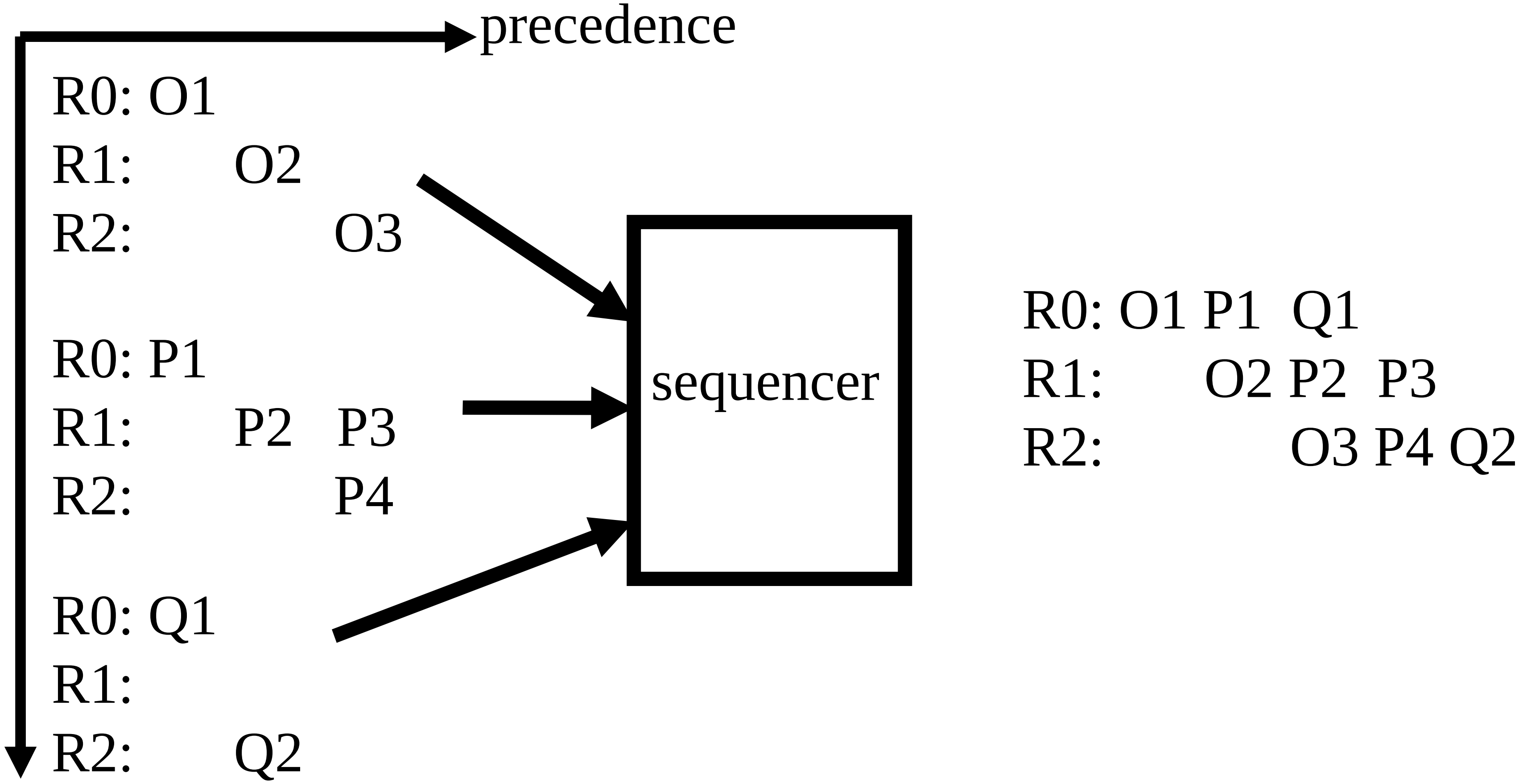


precedence
R0: O1
R1: O2
R2: O3
R0: P1
R1: P2 P3
R2: P4
R0: Q1
R1:
R2: Q2
sequencer
R0: O1 P1 Q1
R1: O2 P2
R2: O3

L-Charts


precedence
R0: O1
R1: O2
R2: O3
R0: P1
R1: P2 P3
R2: P4
R0: Q1
R1:
R2: Q2
sequencer
R0: O1 P1 Q1
R1: O2 P2 P3
R2: O3 P4

L-Charts


precedence
R0: O1
R1: O2
R2: O3
R0: P1
R1: P2 P3
R2: P4
R0: Q1
R1:
R2: Q2
sequencer
R0: O1 P1 Q1
R1: O2 P2 P3
R2: O3 P4 Q2

L-Charts

- LISA expressed more compactly

O1(R1) | O2(R2) | O3(R2) | O4(R3),O5(R4) | O6(R4)

L-Charts

- LISA expressed more compactly

O1(R1) | O2(R2) | O3(R2) | O4(R3),O5(R4) | O6(R4)

| $\rightarrow$ precedence

, $\rightarrow$ parallelism

( ) $\rightarrow$ resource

L-Charts

- LISA expressed more compactly

$$\text{O1(R1) | O2(R2) | O3(R2) | }\underbrace{\text{O4(R3),O5(R4)}}_{\text{No precedence}}\text{ | O6(R4)}$$

| → precedence

, → parallelism

( ) → resource

L-Charts

- Pipelined architecture
  - 3 types of hazards
    - Structural: resource conflicts
    - Data: data grabbed before update
    - Control: conflict assigning proper control step
  - Must be detected & resolved
    - Gantt → naturally covers structural
    - Operations accessing resource must specify R/W
    - Access must be announced in advance

L-Charts

- Additional extension
- Hazard scenario

Instruction 1: IF | ID(!w:R0) | IA | ID(w:R0) |

Instruction 2: IF | ID(r:R0) | IA | IE

L-Charts

- Additional extension
- Hazard scenario

Instruction 1: IF | ID(!w:R0) | IA | ID(w:R0) |

Instruction 2: IF | ID(r:R0) | IA | IE

Data hazard not admissible

L-Charts

- Additional extension
- Hazard scenario

```
Instruction 1:    IF | ID(!w:R0) | IA  | ID(w:R0) |
Instruction 2:          IF              | nop | nop        | ID(r:R0) | IA | IE
```

$$\longrightarrow$$

L-Charts

- Pipeline flow delayed only for resource conflicts
- Processors w/ out-of-order executions excluded
  - No superscalar processors
  - Instuction n checked with instruction n-1

Conclusion

- Main contribution of LISA
  - L-charts
    - Extend Gantt charts to handle data/control hazards
- Mainly used in simulation
- Capable to use in compilation
- Aimed at
  - low/medium complexity machine
    - DSP/embedded system
  - HW/SW co-design