Will Instruction Sets be Important in Future Processors?

Guri Sohi

Computer Sciences Department
University of Wisconsin — Madison
URL: http://www.cs.wisc.edu/~sohi

What is an Instruction Set?

- A means for specifying how a desired computation should be carried out
- Software creates a static representation
  - uses an instruction set
  - contains the computation and sequencing information
- Hardware sequences through static representation
  - determines the operations to be carried out
  - carries out the desired computation

The Job of a Processor

- Start with a static representation of a program
- Sequence through static representation and generate dynamic sequence of operations/instructions
- Execute dynamic sequence at desired speed
  - How to schedule operations for execution?
  - How to communicate values from producer operation to consumer operation?

What role do instruction sets play in the above?

The Pre-RISC Era

Issues driving instruction set/processor design

- Minimizing number of instructions
  - instruction sets that closed the semantic gap
- Reducing fetch bandwidth demands of sequencers
  - compactly encoded instructions
- Ease of creating static representations
  - orthogonality, etc.

The RISC Era

- Streamlining the sequencing/decoding process
  - easy to decode instructions
- Facilitating the implementation of a (pipelined) schedule
  - simple instructions
  - fixed-length instructions
  - load/store, register-register instructions
- Facilitating inter-operation communication
  - more registers

Define a new instruction set to accomplish above

The Post-RISC Era

- Continued increase in power of sequencing/decoding process
- Facilitating the implementation of a (parallel-pipelined) schedule
- Facilitating inter-operation communication

Do we need to define new instruction sets?

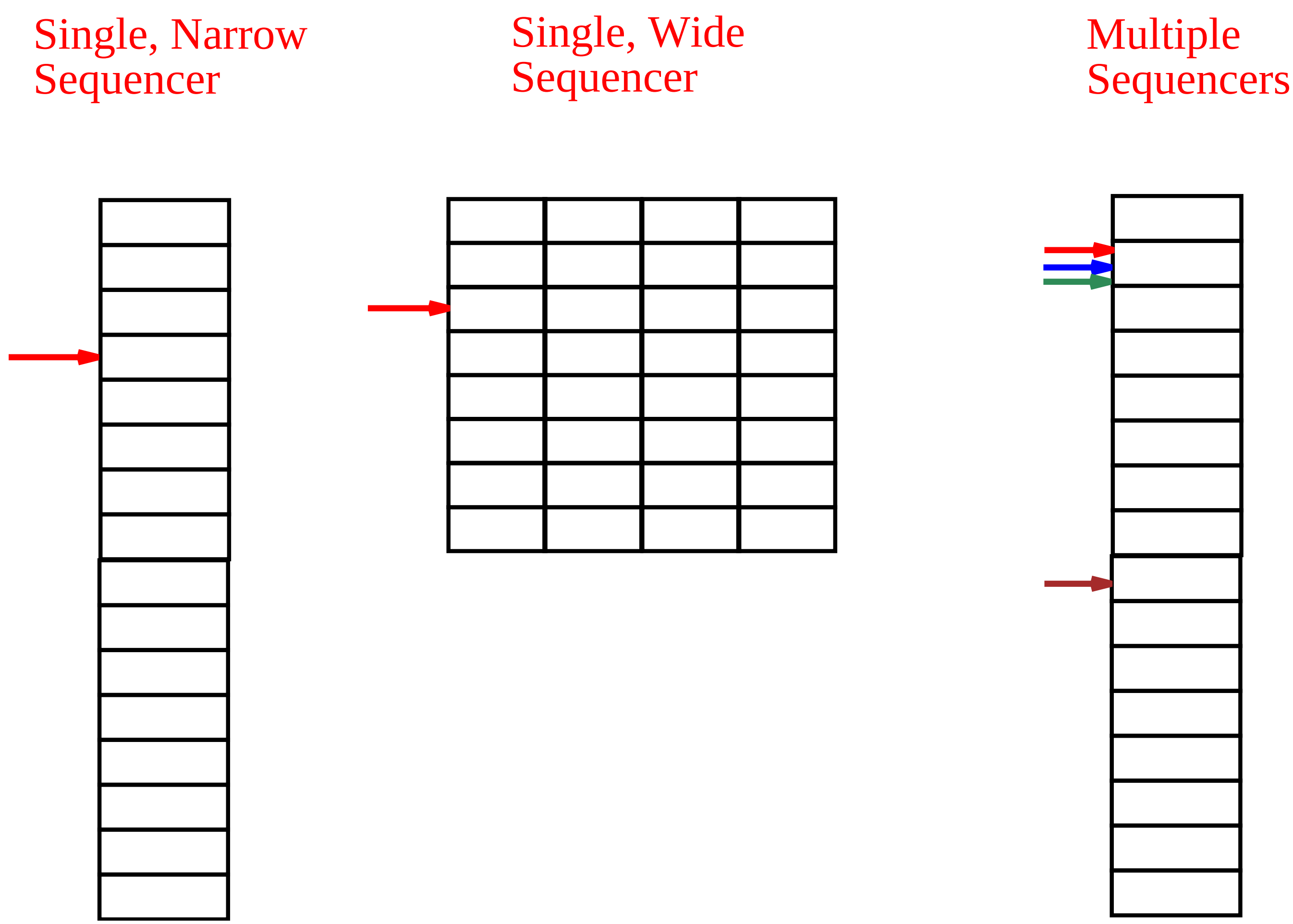
Outline

- Background and historical perspective
- Sequencing through a static representation
- Building an execution schedule
- Inter-operation communication
- Summary and future role of instruction sets

Sequencing Options

- Single or Multiple
  - One sequencer, or multiple sequencers used in the sequencing process?
- Narrow or Wide
  - Single instruction or multiple instructions per sequencing step?
- Uninformed or Informed
  - Knowledge about soon-to-be-encountered instructions available?

Sequencing Options



Static and Dynamic Sequencers

Static sequencing model: the sequencing model assumed when the static representation is created

- Single, narrow: Easy
- Single, wide: Hard
- Multiple: Harder

Dynamic sequencing model: model used (by processor) to sequence through the static representation

- Single, narrow: Easy
- Single, wide: Harder
- Multiple: Unexplored

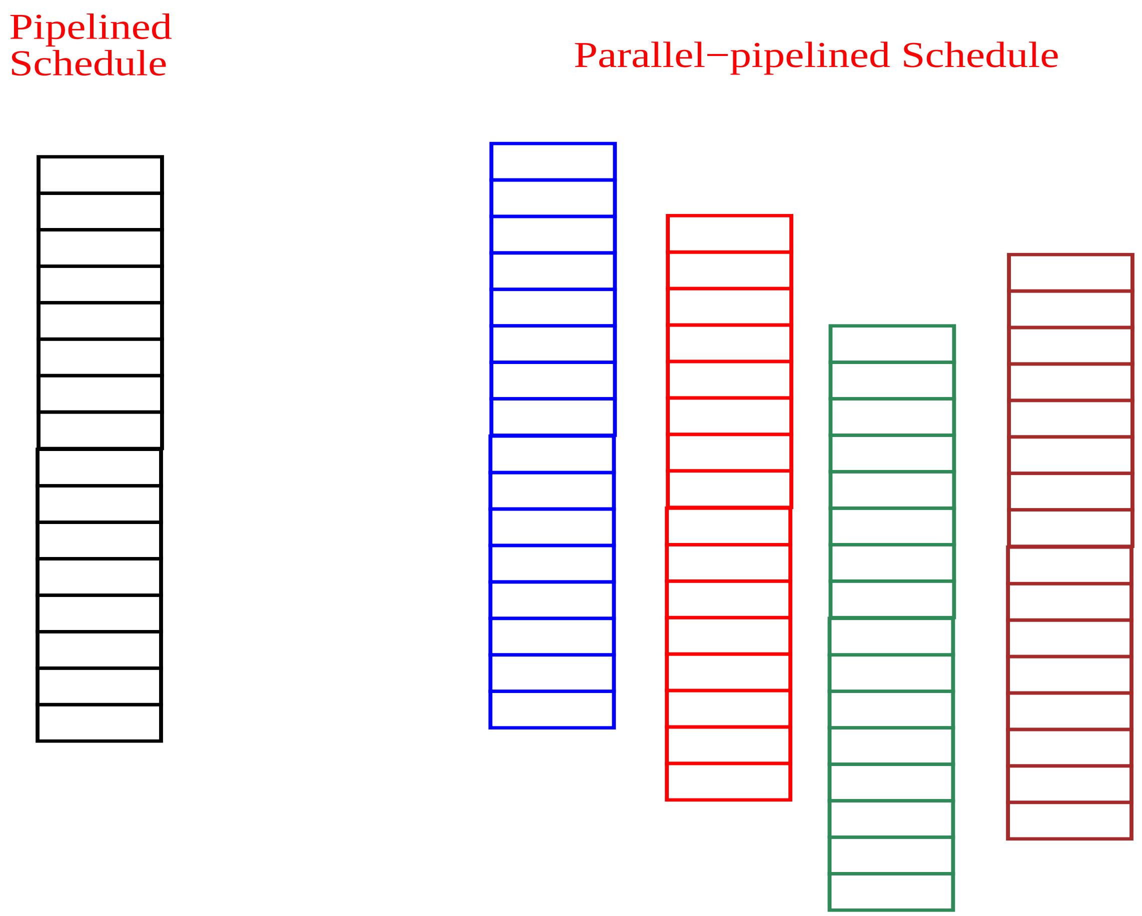
Taxonomy of Sequencing Methods

Dynamic Sequencing Model

Static Sequencing Model

	Single, Narrow	Single, Wide	Multiple
Single Narrow	Scalar	Superscalar	Multiscalar
Single Wide		VLIW	
Multiple			Multiprocessor Multithreaded

Scheduling Options: Schedule Type

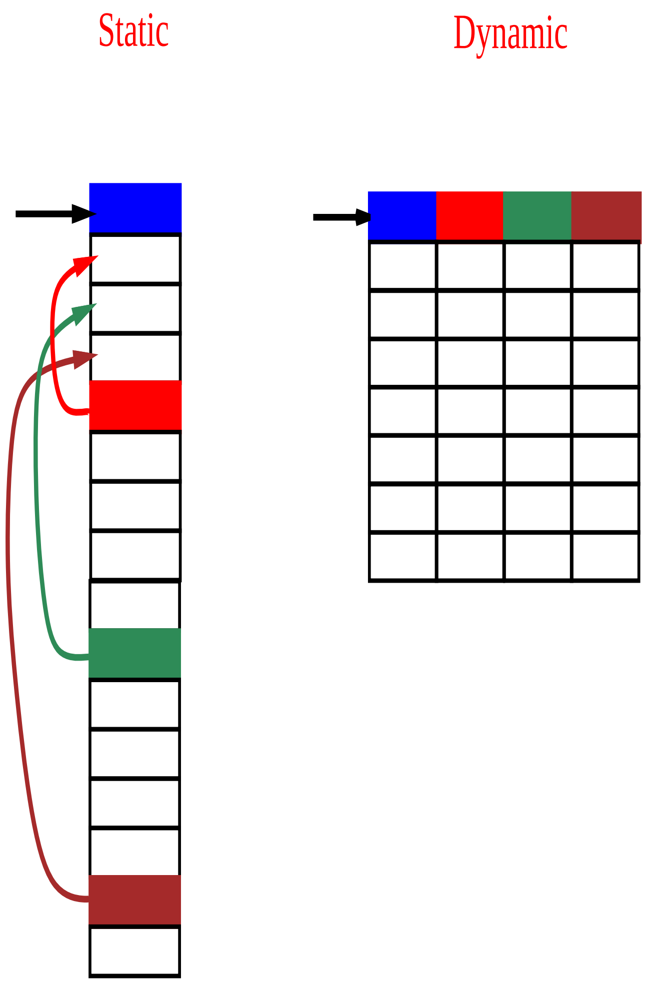


Schedule Orchestrating Options

- Statically
  - need to change static representation to get different schedule
- Dynamically (with adequate static help)
  - less need to change static representation to get different schedule

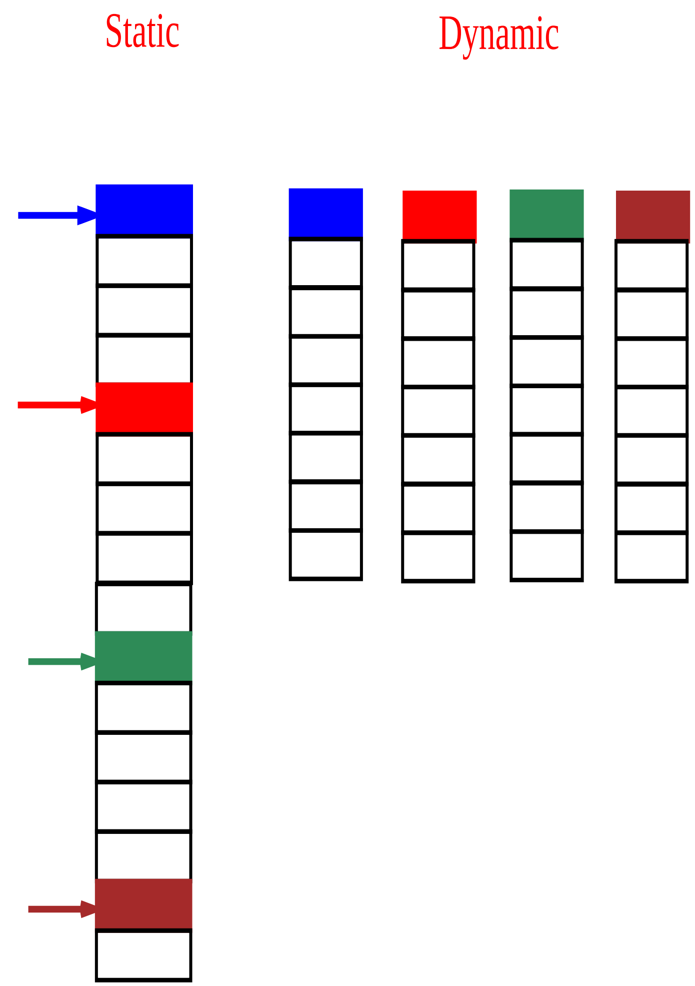
Sequencer/Schedule Interplay

- Single sequencer
  - Schedule is facilitated by placing instructions in static representation
    - move instructions “up”
  - statically orchestrating schedule very important



Sequencer/Schedule Interplay

- Multiple sequencers
  - Schedule is a ensemble of pipelined schedules
  - moving instructions “up” less important
  - statically orchestrating schedule less important



Artifacts of a Single Sequencer

- Wide instructions: to sequence through (and schedule) more than one operation at a time
- Guarded instructions: work around branches in scheduling
  - poor man's way of getting "multiple" flows of control
- Non-trapping instructions: to allow "early" placement of high-latency instructions
- Software Pipelining: to allow overlapped execution of multiple loop iterations

All become less important with multiple sequencers

Inter-Operation Communication

- Passing values from one operation to another
  - Memory
    - high latency and hard to increase bandwidth
  - Registers
    - low latency and easier to increase bandwidth
- Need to provide high-bandwidth, and low-latency storage for inter-operation communication
  - rely on registers
- Increasing the rate of operation execution
  - increases the required size of this store
  - increases the bandwidth demanded of this store

Inter-Op Communication Evolution

- Few architectural registers
- More architectural registers
- More physical registers, but same number of architectural registers
- What comes next?
  - single, large register set?
  - multiple, smaller register sets?
    - logically and physically distinct?
    - physically distinct, but logically same?

Register Requirements

Dynamic Sequencing Model

Static Sequencing Model

	Single, Narrow	Single, Wide	Multiple
Single Narrow	Single register set	Renamed registers	Multiple physical, single logical register set
Single Wide		Various forms	
Multiple			Multiple, small, register sets

With multiple register sets, need to increase the size of each set becomes less important

Where Are We Headed?

- Multiple hardware sequencers
  - sequence with multiple (narrow) sequencers
  - overall schedule is ensemble of pipelined schedules
  - multiple register sets (logical or physical)

Need to make dramatic changes in instruction sets to improve sequencing/scheduling/inter-operation communication becomes less important

Summary

- Compelling technical reason to dramatically change instruction sets is not apparent
- No obvious advantages for sequencing/scheduling/inter-operation communication with new instruction sets
  - microarchitectural techniques can achieve similar effect
- Implications of changing the static representation for each processor generation are obvious
  - remember binary compatibility
- Emphasis in future processor designs should be on novel microarchitectures, not on instruction sets

Assisting a Microarchitecture

- What info can we provide in the static representation?
- Assisting Sequencing
  - Information about what is ‘‘coming down the road’’; take blinders off the sequencer
  - E.g., distance until next (non-local) control flow change
- Assisting Scheduling
  - Information about dependence (data and control) relationships for soon-to-be-encountered instructions, e.g., Tera
  - Processor uses info to schedule available resources
- Assisting Inter-operation Communication
  - Compound operations (e.g., MUL-ADD, reg+reg addressing)