Announcements

- Schedule for the rest of the semester

- April 20: Milestone II presentations
  - Same format as last time

- April 27: Oral Exam
  - 30 minutes per person; in my office; closed book/notes
  - All content covered could be part of the exam

- May 6: Project poster session
  - HH 1112, 2-6pm

- May 10: Project report due
Reviews and Reading List

- Due Today (April 11), before class

- Upcoming Topics (we will not cover all of them)
  - Shared Resource Management
  - Memory Consistency
  - Synchronization
  - Main Memory
  - Architectural Support for Debugging
  - Parallel Architecture Case Studies
Can we get the best of both worlds?

Two possibilities

- Model 1: Keep control flow at the ISA level, do dataflow underneath, preserving sequential semantics
- Model 2: Keep dataflow model, but incorporate control flow at the ISA level to improve efficiency, exploit locality, and ease resource management
  - Incorporate threads into dataflow: statically ordered instructions; when the first instruction is fired, the remaining instructions execute without interruption
Systolic Arrays
Why Systolic Architectures?

- Idea: Data flows from the computer memory in a rhythmic fashion, passing through many processing elements before it returns to memory.

- Similar to an assembly line:
  - Different people work on the same car
  - Many cars are assembled simultaneously
  - Different: Systolic arch. Can be non-linear and multi-dimensional

- Why? Special purpose accelerators/architectures need:
  - Simple, regular designs (keep # of unique parts small and regular)
  - High concurrency → high performance
  - Balanced computation and I/O (memory access)
Systolic Architectures


**Diagram:**

- Memory: heart
- PEs: cells
- Memory pulses data through cells

**Figure 1. Basic principle of a systolic system.**

Instead of:

- Memory: 100 ns, 5 million operations per second at most

We have:

- Memory: 100 ns, 30 MOPS possible
- Cells: PE PE PE PE PE PE PE

The systolic array
Systolic Architectures

- Basic principle: Replace a single PE with a regular array of PEs and carefully orchestrate flow of data between the PEs to achieve high throughput w/o increasing memory bandwidth requirements.

- Differences from pipelining:
  - Array structure can be non-linear and multi-dimensional
  - PE connections can be multidirectional (and different speed)
  - PEs can have local memory and execute kernels (rather than a piece of the instruction)
Systolic Computation Example

- Convolution
  - Used in filtering, pattern matching, correlation, polynomial evaluation, etc ...
  - Many image processing tasks

Given the sequence of weights \( \{w_1, w_2, \ldots, w_k\} \) and the input sequence \( \{x_1, x_2, \ldots, x_n\} \),

**compute** the result sequence \( \{y_1, y_2, \ldots, y_{n+1-k}\} \) defined by

\[
y_i = w_1 x_i + w_2 x_{i+1} + \cdots + w_k x_{i+k-1}
\]
Systolic Computation Example: Convolution

- \( y_1 = w_1x_1 + w_2x_2 + w_3x_3 \)
- \( y_2 = w_1x_2 + w_2x_3 + w_3x_4 \)
- \( y_3 = w_1x_3 + w_2x_4 + w_3x_5 \)

Figure 8. Design W1: systolic convolution array (a) and cell (b) where \( w_i \)'s stay and \( x_i \)'s and \( y_i \)'s move systolically in opposite directions.
Systolic Computation Example: Convolution

- Worthwhile to implement adder and multiplier separately to allow overlapping of add/mul executions

Figure 10. Overlapping the executions of multiply and add in design W1.
Other Examples: Broadcast Based

Figure 3. Design B1: systolic convolution array (a) and cell (b) where $x_i$’s are broadcast, $w_i$’s stay, and $y_i$’s move systolically.

Figure 4. Design B2: systolic convolution array (a) and cell (b) where $x_i$’s are broadcast, $y_i$’s stay, and $w_i$’s move systolically.
Other Examples: Broadcast Based

Figure 5. Design F: systolic convolution array (a) and cell (b) where \(w_i\)'s stay, \(x_i\)'s move systolically, and \(y_i\)'s are formed through the fan-in of results from all the cells.
Other Examples: No Broadcast

![Diagram of systolic convolution array](image)

Figure 6. Design R1: systolic convolution array (a) and cell (b) where $y_i$'s stay and $x_i$'s and $y_i$'s move in opposite directions systolically.
Other Examples: No Broadcast

Figure 7. Design R2: systolic convolution array (a) and cell (b) where $y_i$'s stay and $x_i$'s and $w_i$'s both move in the same direction but at different speeds.
Figure 8. Design W1: systolic convolution array (a) and cell (b) where $w_i$’s stay and $x_i$’s and $y_i$’s move systolically in opposite directions.

Figure 9. Design W2: systolic convolution array (a) and cell (b) where $w_i$’s stay and $x_i$’s and $y_i$’s both move systolically in the same direction but at different speeds.
Figure 11. Two-dimensional systolic arrays: (a) type R, (b) type H, and (c) type T.
Systolic Array

- Advantages
  - Makes multiple uses of each data item → reduced need for fetching/refetching
  - High concurrency
  - Regular design (both data and control flow)

- Disadvantages
  - Not good at exploiting irregular parallelism
  - Relatively special purpose → need software, programmer support to be a general purpose model
    - Simplicity versus flexibility tradeoff in each cell
Each PE in a systolic array
- Can store multiple “weights”
- Weights can be selected on the fly
- Eases implementation of, e.g., adaptive filtering

Taken further
- Each PE can have its own data and instruction memory
- Data memory to store partial/temporary results, constants
- Leads to **stream processing, pipeline parallelism**
  - More generally, **staged execution**
Pipeline Parallelism

Figure 1. (a) The code of a loop, (b) Each iteration is split into 3 pipeline stages: A, B, and C. Iteration i comprises Ai, Bi, Ci. (c) Sequential execution of 4 iterations. (d) Parallel execution of 6 iterations using pipeline parallelism on a three-core machine. Each stage executes on one core.
File Compression Example

Figure 3. File compression algorithm executed using pipeline parallelism
Pipeline Parallelism: Readings


The WARP Computer

- HT Kung, CMU, 1984-1988

- Linear array of 10 cells, each cell a 10 Mflop programmable processor
- Attached to a general purpose host machine
- HLL and optimizing compiler to program the systolic array
- Used extensively to accelerate vision and robotics tasks

The WARP Computer

Figure 1: Warp system overview
Systolic Arrays vs. SIMD

- Food for thought...