Preclass Parity

• How many gates?

• Draw solutions

Preclass xor2 in nor2s

• $\text{xor2}(a,b) = /\text{xnor2}(a,b) = (/a*/b+a*b)$

Preclass xor3 in nor2s

• $\text{Xor3}(a,b,c)=\text{xor2}(\text{xor2}(a,b),c)$

Previously

• Pipelining – reuse in time for \textbf{same operation}
• Memory
• Memories pack state compactly
  – densely

What is Importance of Memory?

• \textbf{Radical Hypothesis:}
  – Memory is simply a very efficient organization which allows us to store data compactly
  \hspace{1em} (at least, in the technologies we’ve seen to date)
  – A great engineering trick to optimize resources
• \textbf{Alternative:}
  – memory is a \textbf{primary}
Today

- Operator Sharing (from Day 4)
- Datapath Operation
- Virtualization
- Memory
  - ...continue unpacking the role of memory...

Universal Sharing

Review

- Given a task: \( y = Ax^2 + Bx + C \)
- Saw how to share primitive operators
- Got down to one of each

Very naively

- Might seem we need one of each different type of operator

..But

- Doesn't fool us
- We already know that nand gate (and many other things—HW1.3)
  .... are universal
- So, we know, we can build a universal compute operator

Temporal Composition
**Temporal**

- Don’t have to implement all the gates *at once*
- Can *reuse* one gate over time

**Temporal Decomposition**

- Take Set of gates
- Sort topologically
  - All predecessors before successors
- Give a unique number to each gate
  - Hold value of its outputs
- Use a memory to hold the gate values
- Sequence through gates

**Example Logic**

![Example Logic Diagram]

**Numbered Gates**

![Numbered Gates Diagram]

**Preclass**

- Number gates

**nor2 Memory/Datapath**

![nor2 Memory/Datapath Diagram]
**Programming?**

- How do we program this netlist?

**Programming?**

- Program gates
  - Tell each gate where to get its input
    - Tell gate n where its two inputs come from
    - Specify the memory location for the output of the associated gate
  - Each gate operation specified with
    - two addresses (the input sources for gate)
    - This is the instruction for the gate

**nor2 Memory/Datapath**

- White Addr
- Left Read Addr
- Right Read Addr

Instruction

**Supply Instruction**

- How can we supply the sequence of instructions to program this operation?

**Simplest Programmable Control**

- Use a memory to “record” control instructions
- “Play” control with sequence

**Temporal Gate Architecture**

Program Counter (Address Pointer into Instruction Memory)
How program preclass computation?

- How would we program the preclass computation?
  - Complete the memory

Simulate the Logic

- For Preclass
  - Go around the room calling out:
    - Identify PC
    - Identify instruction
      - Perform nor2 on slot ___ and slot ___
      - Result is ___
      - Store into slot ___

What does this mean?

- With only one active component
  - nor gate
- Can implement any function
  - given appropriate
    - state (memory)
    - muxes (interconnect)
    - Control

Defining Terms

Fixed Function:
- Computes one function (e.g. FP- multiply, divider, DCT)
- Function defined at fabrication time

Programmable:
- Computes “any” computable function (e.g. Processor, DSPs, FPGAs)
- Function defined after fabrication

Result

- Can sequence together primitive operations in time
- Communicating state through memory
  - Memory as interconnect
- To perform “arbitrary” operations

“Any” Computation? (Universality)

- Any computation which can “fit” on the programmable substrate
- Limitations: hold entire computation and intermediate data
Temporal-Spatial Variation

- Can have any number of gates
  - Tradeoff Area for Reduce Time,…

Use of Memory?

- What did we use memory for here?
  - State
  - Instructions
  - Interconnect

“Stored Program” Computer/Processor

- Can build a datapath that can be programmed to perform any computation.
- Can be built with limited hardware that is reused in time.
- Historically: this was a key contribution from Penn’s Moore School
  - Computer Engineers: Eckert and Mauchly
  - ENIAC→EDVAC
  - (often credited to Von Neumann)

What have we done?

- Taken a computation: \( y = A x^2 + B x + C \)
- Turned it into operators and interconnect
- Decomposed operators into a basic primitive:
  - nor, adds

Virtualization

- We’ve virtualized the computation
- No longer need one physical compute unit for each operator in original computation
- Can suffice with:
  1. shared operator(s)
  2. a description of how each operator behaved
  3. a place to store the intermediate data between operators

What have we done?

- Said we can implement it on as few as one of compute unit (nor2)
- Added a unit for state
- Added an instruction to tell single, universal unit how to act as each operator in original graph
Virtualization

Why Interesting?
- Memory compactness
- This works and was interesting because
  - the area to describe a computation, its
    interconnect, and its state
  - is much smaller than the physical area to
    spatially implement the computation
- e.g. traded multiplier for
  - few memory slots to hold state
  - few memory slots to describe operation
  - time on a shared unit (adder, gate)

Temporal vs. Spatial
- Spatially Programmable
- Temporally Programmable

Compare Area
- Spatially Programmable
- Temporally Programmable

Simple Model
- Xbar area = area of the crosspoints
- Memory area = area of the memory bits
- Program Counter = ignore for now
- Areas:
  - Memory bit = 1
  - Xpoint = 2
  - Nor2 = 4
Spatial Area (100 gates)?

- Spatially Programmable
  - Estimate area for spatial
    - Number of crosspoints?
    - Total Area?

Temporal Area (100 gates)?

- Temporally Programmable
  - Bits in data memories?
  - Bits in instruction?
  - Total bits in instruction memory?
  - Total Area?

Temporal vs. Spatial

- How does area compare?
  - Discussion:
    - How will delay compare?

Big Ideas

[MSB Ideas]

- Memory: efficient way to hold state
  - ...and allows us to describe/implement computations of unbounded size
- State can be << computation [area]
- Resource sharing: key trick to reduce area
- Memory key tool for Area-Time tradeoffs
- “configuration” signals allow us to generalize the utility of a computational operator

Big Ideas

[MSB-1 Ideas]

- Different kind of programmable computing unit
- Two key functions of memory
  - retiming (interconnect in time)
  - instructions
  - description of computation
Admin

• HW4 due Wednesday
• No Office Hours Tomorrow
  – Piazza, email, appointment
• Reading for Wednesday on Canvas