CS184b: Computer Architecture [Single Threaded Architecture: abstractions, quantification, and optimizations]

Day9: February 1, 2000

SuperScalar

Costs and Opportunities

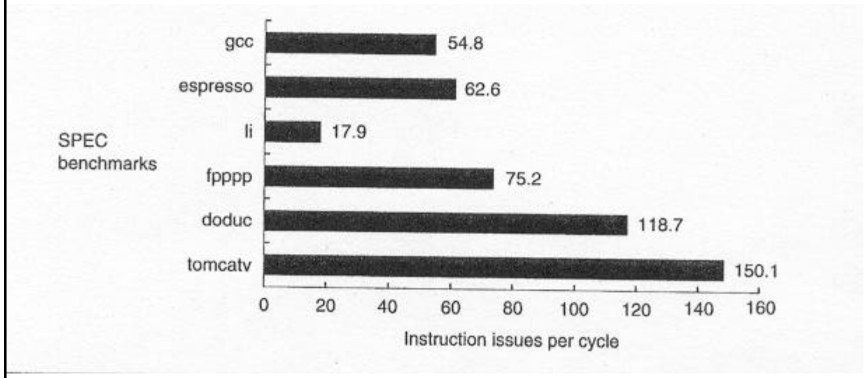
Today

- Issue Window
- Registers
- Bypass
- Ultrascalar

Limit Studies

- Goal: understand how far you can go
  - this case, how much ILP can find
- Remove current/artificial limits
  - do full renaming
  - arbitrary look ahead
  - perfect control prediction
- Careful with assumptions
  - can still be pessemistic
  - is there another way to do it?
  - Another way around the limitation?

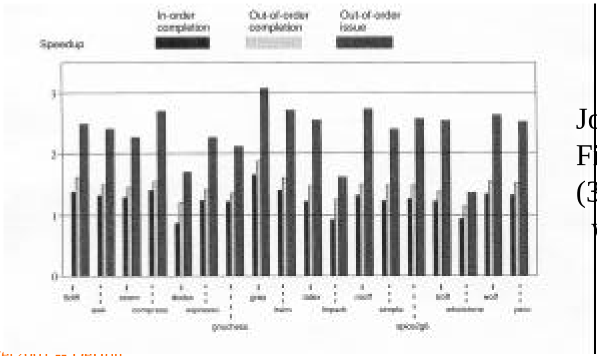
Available ILP



Hennessy and Patterson 4.38

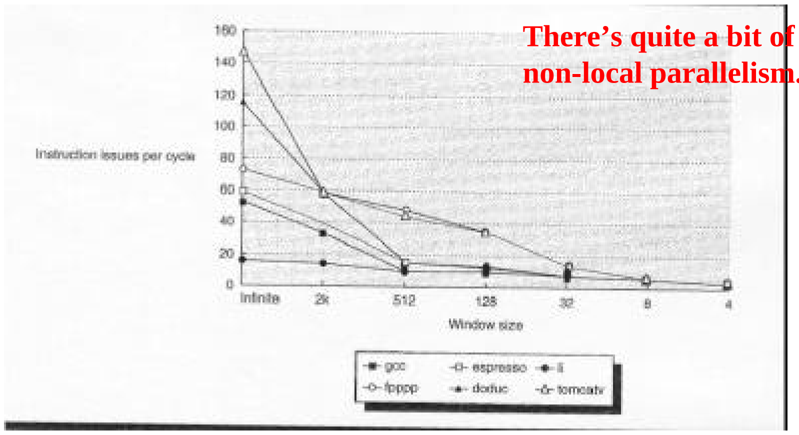
Window Size

- How many instructions forward do we look?
  - Only look at next = in-order issue



Johnson
Fig. 3.9
(32 issue
window?)

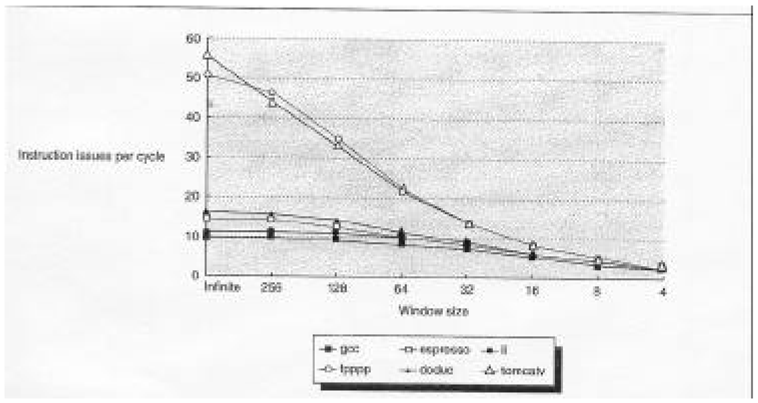
Window Size



There's quite a bit of non-local parallelism.

[Hennessy and Patterson 4.39]

Window Size



[64-issues Hennessy and Patterson 4.47]

Operation Organization

- Consider Tree-structured calculation
  - freedom in ordering
  - consider:
    - post-order traversal
    - by levels from leaves
  - where is parallelism?
  - Storage cost?

Cost?

- $Rsrc_i \neq Rdst_{i-1}$; $Rsrc_i \neq Rdst_{i-2}$;…
- $O(WS^2)$ comparisons

Cost?

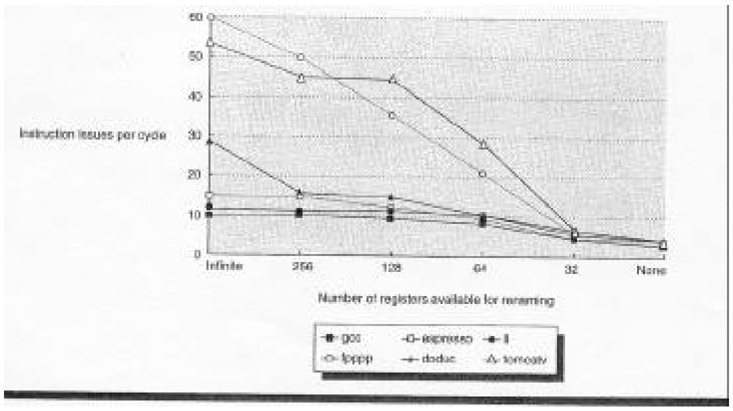
- Anecdotal [Farrell, Fischer JSSC v33n5]
  - DEC 20-instruction queue
  - 4 instruction issue
  - (80 physical registers)
  - $10mm^2$ in $0.35\mu m$ ($300M\lambda^2$+)
  - 600 MHz = 1.6ns

Costs?

- Both DEC and “Quantifying” (also DEC)
  - appear to use a scoreboarded scheme to avoid
  - accept not issue until result computed?
- “Quantifyng” suggests:
  - wakeup time $\propto IW^2 \times WS^2$
    - but assuming quadratic wire delay in length
    - (never buffer wire)
  - but WS=F(IW)
  - certainly faster than linear time
  - $A \propto IW \times WS$

Registers

- How many virtual registers needed?



[Hennessy and Patterson 4.43]

Register Costs?

- First Order
  - area linear in number of registers
  - delay linear in number of registers
- Bank RF
  - maybe sublinear delay
  - at least square root number of registers
    - wire delay sqrt of area

RF and IW interaction

- Larger Issue (Decode)
  - want to read/retire more registers per cycle
  - RF ports = 3 IW
  - $A \propto$ ports * number
  - …and number of registers = F(IW)
- RF grows faster than linear

Bypass: Control

- Control comparison
  - every functional input (2 IW)
  - get input from
    - every pipestage (d) from issue produce to wb
    - for every result producer (IW)
- Total comparisons: $d \times IW^2$

Bypass: Interconnect

- Linear layout
  - bypass span functional units and RF
  - physical RF grows with IW
    - read/write ports
    - more physical registers to support IW
  - FU bypass muxes grows with IW
- Consequently
  - height grows with IW   (IW^2 ?)
  - cycle grow with IW?

Bypass: Interconnect

- "Quantifying"
  - quadratic wire delay
  - (but asymptotically, we can buffer)
  - largest delay component calculated
    - (>1ns for IW=8)
    - IW=8 about 5-6 times IW=4

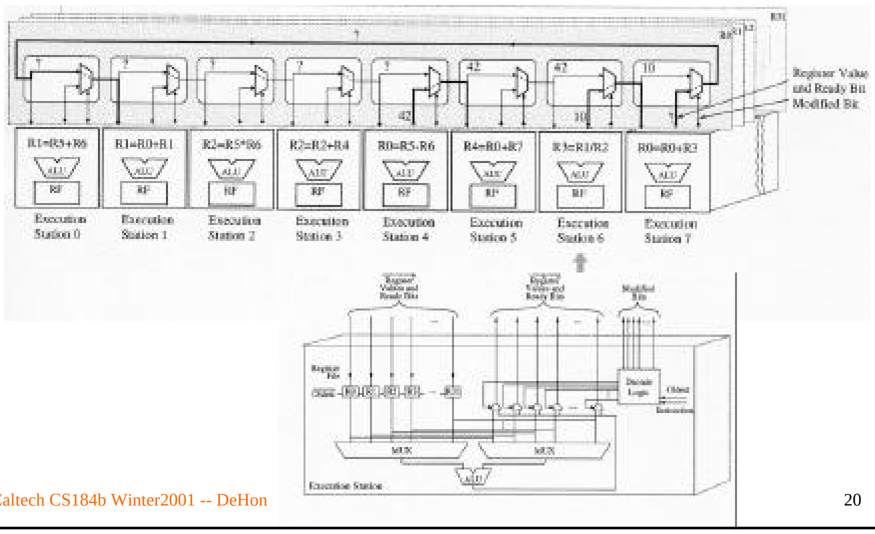
Different Solution

- These assume Number of Regs > IW
- If IW>R, different approach…
- From Henry, Kuszmaul, et. Al.
  - ARVLSI'99
  - SPAA'99
  - ISCA'00

Consider Machine

- Each FU has a full RF
- Build network between FUs
  - use network to connect produce/consume
  - user register names to configure interconnect
- Signal data ready along network

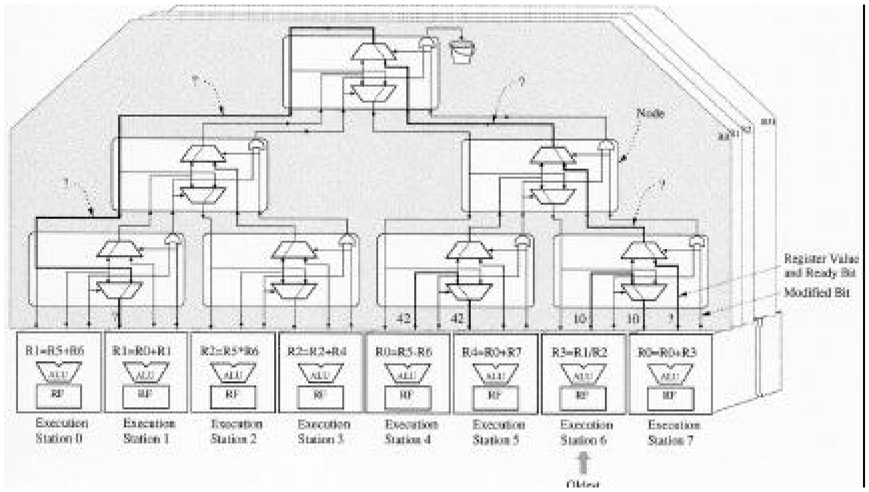
Ultrascalar: concept model



Ultrascalar concept

- Linear delay
- O(1) register cost / FU
- Complete renaming at each FU
  - – different set of registers
  - – so when say complete reg at each FU, that's only the logical registers

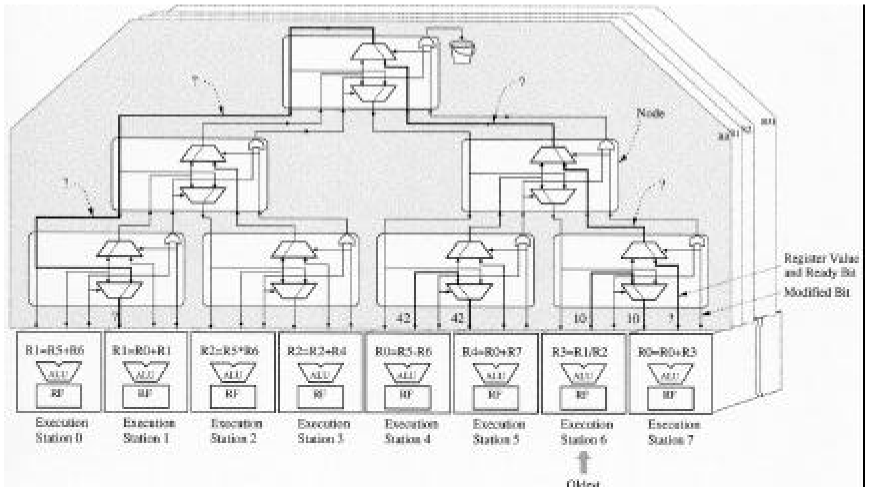
Ultrascalar: cyclic prefix



Parallel Prefix

- Basic idea is one we saw with adders
- An FU will either
  - produce a register (generate)
  - or transmit a register (propagate)
  - can do tree combining
    - pair of FUs will either both propogate or will generate
    - compute function by pair in one stage
    - recurse to next stage
    - get log-depth tree network connecting producer and consumer

Ultrascalar: cyclic prefix



Cyclic Prefix

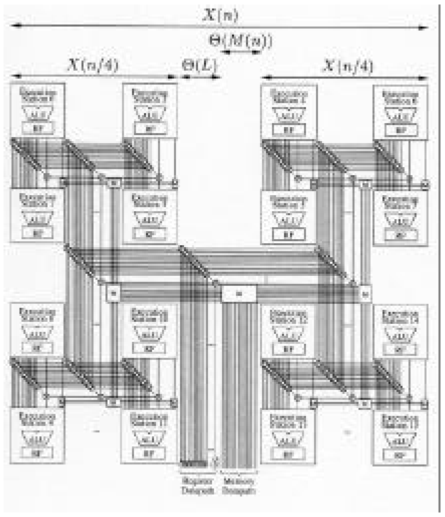
- Gets delay down to log(WS)
  - w/ linear layout, delay still linear
- Issue into, retire from Window in order
  - serves
    - rename
    - shared RF
    - issue
    - bypass
    - reorder

Ultrascalar: layout

Register paths not growing. Wide, but constant width

If Memory width <√n area goes as n

wire goes as √n



Ultrascalar: asymptotics

- Assume $M(n)<O(\sqrt{n})$
  - Area ~ $n \times R^2$
  - Delay ~ $(\sqrt{n}) \times R$
- Claim can do
  - Area ~ $n \times R$
  - Delay ~ $\sqrt{(n \times R)}$
- Memory grows faster, will dominate interconnect growth, hence area and delay
  - get extra term for memory growth (like Rent's Rule)

UltraScalar:

- 0.25 µm
- 128-window, 32 logical regs
- 64b ops ?
- 4-issue
- delays <2ns
  - comit, wakeup, schedule
  - wire delay dominate logic
- area ~$1G\lambda^2$ (? Includes all datapath)

Solution for:

- Object/binary compatibility is paramount
- Performance is King
- Recompilation not an option
- Cost (area, energy) is no object

Next Week

- …an alternative way to exploit ILP
- rely on compiler and feedback

- Tuesday: VLIW/Fisher

- Thursday: EPIC

(Semi?) Big Ideas

- Balance
- Size Matters
- Interconnect delay dominate
- As parameters grow
  - watch tradeoffs
  - widely different solutions prevail in different points in space (different asymptotes)