Intel Processor Implementations
CS 740

October 5, 1995

Topics

- **486**
  - Single pipeline
- **Pentium**
  - Superscalar
- **P6**
  - Decoupled superscalar

Comparisons

History

Processor	Year	Transistors	Frequency	SpecInt92	SpecFP92
i386	‘86	275K	16MHz		
i486	‘89	1.2M	33MHz		
i486DX2			66/33MHz	32	16
Pentium	‘93	3.1M	66MHz	67	61
P6	‘95?	5.5M	133MHz	200	200

Notes

- Year indicates when available commercially
- Clock frequency is for part at time of introduction

x86 ISA Characteristics

Multiple Data Sizes and Addressing Methods

- Recent generations optimized for 32-bit mode

Limited Number of Registers

- Stack-oriented procedure call and FP instructions
- Programs reference memory heavily (41%)

Variable Length Instructions

- First few bytes describe operation and operands
- Remaining ones give immediate data & address displacements
- Average is 2.5 bytes

486 Block Diagram

i486 Pipeline

Fetch

- Load 16-bytes of instruction into prefetch buffer

Decode1

- Determine instruction length, instruction type

Decode2

- Compute memory address
- Generate immediate operands

Execute

- Register Read
- ALU operation
- Memory read/write

Write-Back

- Update register file

Pipeline Stage Details

Fetch

- **Moves 16 bytes of instruction stream into code queue**
- **Not required every time**
  - About 5 instructions fetched at once
  - Only useful if don't branch
- **Avoids need for separate instruction cache**

D1

- **Determine total instruction length**
  - Signals code queue aligner where next instruction begins
- **May require two cycles**
  - When multiple operands must be decoded
  - About 6% of "typical" DOS program

Stage Details (Cont.)

D2

- **Extract memory displacements and immediate operands**
- **Compute memory addresses**
  - Add base register, and possibly scaled index register
- **May require two cycles**
  - If index register involved, or both address & immediate operand
  - Approx. 5% of executed instructions

EX

- **Read register operands**
- **Compute ALU function**
- **Read or write memory (data cache)**

WB

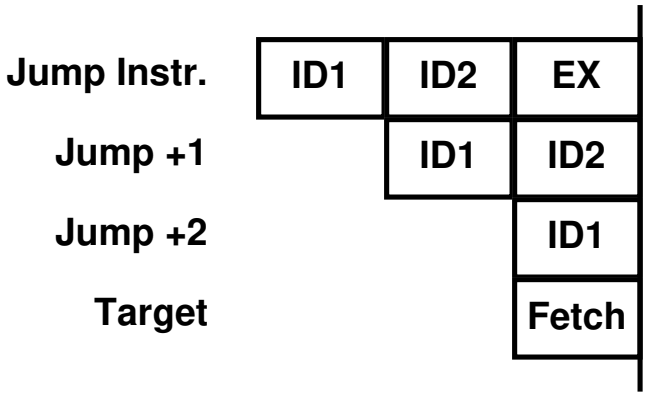
- **Update register result**

Data Hazards

Data Hazards

Generated	Used	Handling
ALU	ALU	EX–EX Forwarding
Load	ALU	EX–EX Forwarding
ALU	Store	EX–EX Forwarding
ALU	Eff. Address	(Stall) + EX–ID2 Forwarding

Control Hazards



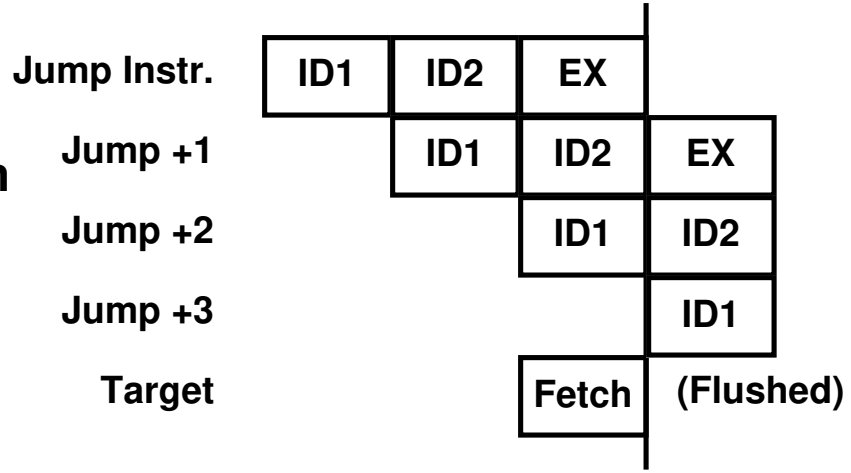
Jump Instruction Processsing

- Continue pipeline assuming branch not taken
- Resolve branch condition in EX stage
- Also speculatively fetch at target during EX stage

Control Hazards (Cont.)

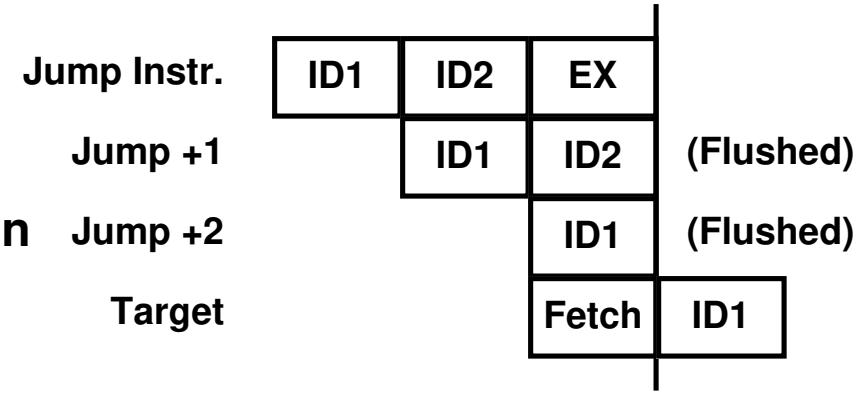
Branch Not Taken

- Allow pipeline to continue.
- Total of 1 cycle for instruction



Branch taken

- Flush instructions in pipe
- Begin ID1 at target.
- Total of 3 cycles for instruction



Comparison to MIPS

Two Decoding Stages

- **Harder to decode CISC instructions**
- **Effective address calculation in D2**

Multicycle Decoding Stages

- **For more difficult decodings**
- **Stalls incoming instructions**

Combined Mem/EX Stage

- **Avoids load stall without load delay slot**
  - But introduces stall for address computation

Poorer Branch Performance

- **Can't make use of a delay slot**
- **Asymmetric performance**

Comparison to 386

Cycles Per Instruction

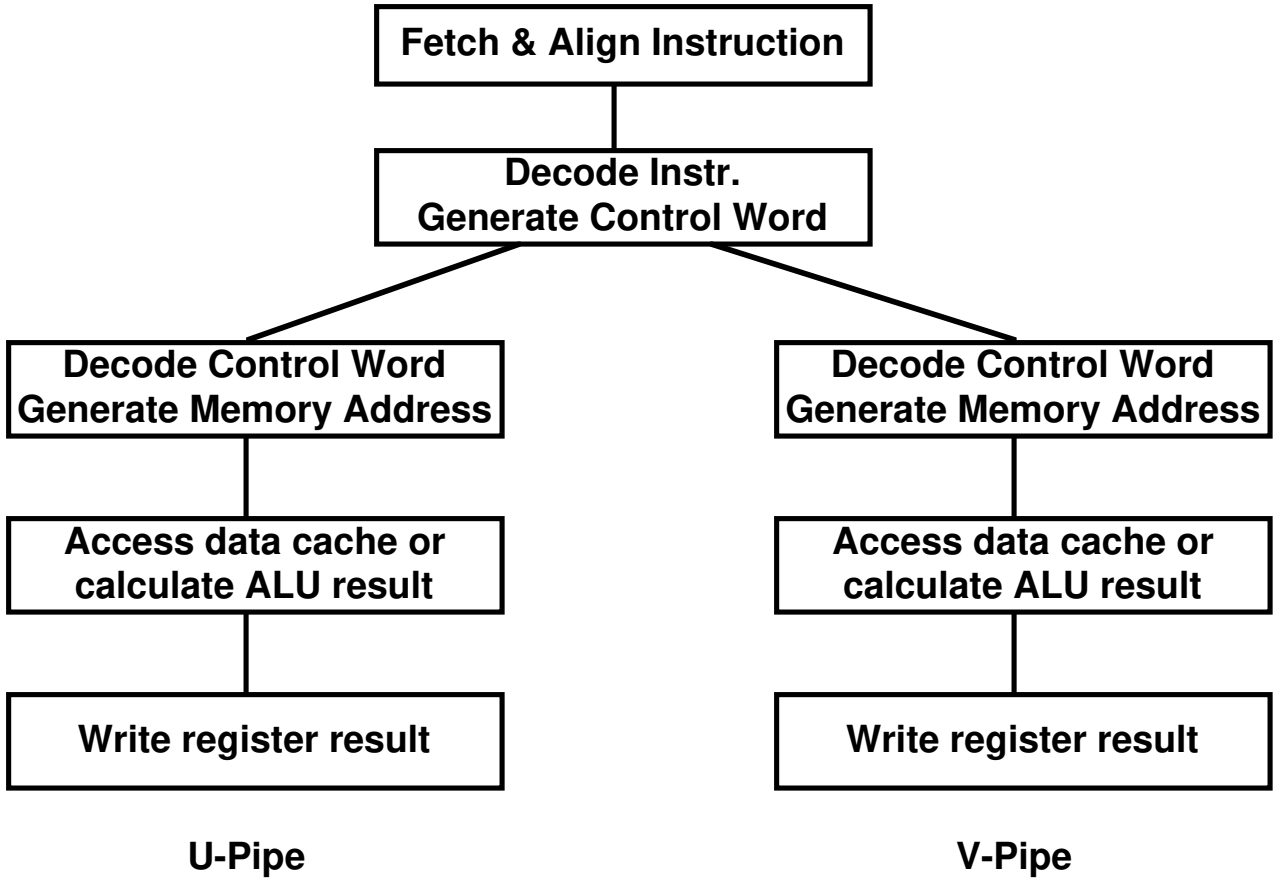
Instruction Type	386 Cycles	486 Cycles
Load	4	1
Store	2	1
ALU	2	1
Jump taken	9	3
Jump not taken	3	1
Call	9	3

Reasons for Improvement

- **On chip cache**
  - Faster loads & stores
- **More pipelining**

Pentium Block Diagram

Pentium Pipeline



Superscalar Execution

Can Execute Instructions I1 & I2 in Parallel if:

- **Both are "simple" instructions**
  - Don't require microcode sequencing
  - Some operations require U-pipe resources
  - 90% of SpecInt instructions
- **I1 is not a jump**
- **Destination of I1 not source of I2**
  - But can handle I1 setting CC and I2 being cond. jump
- **Destination of I1 not destination of I2**

If Conditions Don't Hold

- **Issue I1 to U Pipe**
- **I2 issued on next cycle**
  - Possibly paired with following instruction

Branch Prediction

Branch Target Buffer

- **Stores information about previously executed branches**
  - Indexed by instruction address
  - Specifies branch destination + whether or not taken
- **256 entries**

Branch Processing

- **Look for instruction in BTB**
- **If found, start fetching at destination**
- **Branch condition resolved early in WB**
  - If prediction correct, no branch penalty
  - If prediction incorrect, lose ~3 cycles
    - » Which corresponds to > 3 instructions
- **Update BTB**

P6

History

- **Announced in Feb. '95**
- **Expected to ship in 3Q95**

Features

- **Dynamically translates instructions to more regular format**
  - Very wide RISC instructions
- **Executes operations in parallel**
  - Up to 5 at once
- **Very deep pipeline**
  - 12–18 cycle latency

P6 Block Diagram

P6 Operation

Translates instructions dynamically into "Uops"

- **118 bits wide**
- **Holds operation, two sources, and destination**

Executes Uops with "Out of Order" engine

- **Uop executed when**
  - Operands available
  - Functional unit available
- **Execution controlled by "Reservation Stations"**
  - Keeps track of data dependencies between uops
  - Allocates resources

Branch Prediction

Critical to Performance

- **11–15 cycle penalty for misprediction**

Branch Target Buffer

- **512 entries**
- **4 bits of history**
- **Adaptive algorithm**
  - Can recognize repeated patterns, e.g., alternating taken–not taken

Handling BTB misses

- **Detect in cycle 6**
- **Predict taken for negative offset, not taken for positive**
  - Loops vs. conditionals