Processors

Young W. Lim

May 12, 2016

Based on

Computer System Design : System-on-Chip
by M.J. Flynn and W. Luk

Processor Architecture

- Instruction Set
- Microarchitecture - different implementation details
- Synthesis result with area time power tradeoff

Instruction Set

- A Register Set to hold operands and addresses
- Floating Point Registers
- A Register for Program Status Word
  - including Condition Codes (CC)
- Types
  - Load-Store (L/S)
  - Register-Memory (R/M)

Load Store Architecture

- arguments must be in registers before execution
- ALU instructions have source and destination registers
- regularity of execution
- ease of instruction decode
- ease of timing requirements
- RISC microprocessors

Register Memory Architecture

- Operands in registers
- One operand in memory
- ALU instructions can have a operand in memory
- simple program representation
  - fewer instructions with variable size (complex) instruction types
- complex instruction decoding and timing
- IBM mainframe, Intel x86 series

Branches

- Branches (jumps) handles program control flow
- Unconditional BR
- Conditional BC
  - check the status of CC
  - CC is set by an ALU instructions
    - a positive result
    - a negative result
    - a zero result
    - an overflow

Interrupt and Exceptions

- User Requested vs Coerced
- Maskable vs Nonmaskable
- Terminate vs Resume
- Asynchronous vs Synchronous
- Between vs Within Instructions

MicroArchitecture

- an instruction execution pipeline
- issue one instruction for each cycle
  - many embedded and signal processors
- issue many instructions for each cycle
  - moder desktop, laptop, server systems
- Components
  - Memory System
  - Execution Unit
  - Instruction Unit

Pipeline Delays

- Data Conflicts - Unavailability of a source operand
  - the needed operand is the result of a preceding uncompleted instruction
- Resource Contention
  - multiple successive instructions requires the same resource
- Run-On Delays (In Order Execution Only)
  - when instructions must complete the WB in program order
- Branches
  - branch resolution
  - delay in fetching the branch targer instruction

Instruction Unit

- Instruction Register
- Instruction Buffer
  - for fast instruction decode
- Instruction Decoder
  - controlling the cache, ALU, registers...
  - I-Unit : FSM (hardware)
  - E-Unit : micro-prammed control, micro-instruction
- Interlock Unit
  - the concurrent execution of multiple instructions
  - must have the same result when serially executed

Instruction Decoder

- Instruction decoder provide
  - control and sequencing information
  - ensure proper execution (dependency exists between instructions)
- schedules the current instruction
  - delayed : AG (Address Generate) Cycle
- schedules the subsequent instructions
  - delayed to preserve in-order execution
- selects (predicts) the branch path

Data Interlocks

- may be part of I-Unit
- determines register dependencies
- schedules the AG and EX units
- ensures the current instruction does not use a result of a previous instruction until that result is available
- as an instruction is decoded, its source registers are must be checked
- they are compared against the destination registers previously issued instruction
- because uncompleted instructions may cause dependencies and additional delay must be added

Execution Unit

- Integer Core Processor
- Floating-point Unit
- Arithmetic Algorithms

Buffers

- change the way instruction timing events occur
- decouping the event occurring time and the data utilizing time
- allows some additional delays without affecting the performance
- latency tolerance
  - buffers hold the data awaiting entry into a stage

Branches

- reduce significantly performance
- conditional branch instruction (BC) tests the CC
- a number of cycles between decoding the BC and setting the CC
- the simple approache
  - do nothing but wait for the CC
  - defer the decoding of BC
  - if the branch is taken
    - the target is fetched during the allotted time for data fetch
  - simple to implement and minimizes the amount of excess memory traffic

Reducing the branch cost

- Simple Approaches
  - Branch Elimination
    - for certain cases, it is possible to replace the branch with other instruction sets
  - Simple Branch Speedup
    - reduces the time required for target instruction fetch and CC determination
- Complex Approaches
  - Brach Target Capture
    - keep the target instruction and address in a table for a later use to avoid branch delay
  - Branch Prediction
    - predict the branch result and begin processing on the predicted path

Branch Target Buffers (BTBs)

- stores the target instruction of the previous execution of the branch
- each entry has
  - the current instruction address
  - branch target address
  - the most recent target instruction
- operation
  - each instruction fetch indexes the BTB
  - if the instruction matches, a prediction is made (taken or not)
  - for a branch taken prediction, the target instruction is used
  - during the actual resolution at the execution stage, the BTB is updated

BTB Effectiveness

- BTBs are used with I-cache
- The IF is made to the BTB & I-cache
- if it hits in the BTB, the stored target instruction is used without memory accessing delays
- both the target instruction and new PC address are provided
- no branch delay for the taken branch that was correctly predicted
- the branch instruction itself must be fetched from I-cache
  - if the AG result and the CC result is not as expected
  - all instruction in the target path must be aborted
- effectiveness depends on its hit ratio

Branch Prediction

- guessing whether or not a branch will be taken
- a static strategy
  - based on the type of branch instruction
- a dynamic strategy
  - based on the recent history of branch history

Concurrent Processors

- more than one CPI (cycle per instruction)
- multiple instructions a the same time
- simultaneous accesses to the instruction and data memory
- simultaneous execution of multiple operations
- instruction level concurrency
- uniprocessors : special case
  - only one program stream
  - a single instruction counter (PC)
  - the original instructions are significantly rearranged
  - compiler, execution resources, memory system
    - Vector Processors
    - VLIW (Very Long Instruction Word)
    - Superscalar

Vector Data Structure

- Vectors - derived from large data arrays
  - conventional data cache cannot handle efficiently
  - strided access exhibits little temporal locality
  - no reuse of the localities before the items must be replaced
- Vector Registers
  - decouples arithmetic operations from accessing memory
  - source and destination vector register sets
  - independent of data cache
  - data cache contains only scalar data objects

Vector Processors

1. reduce the I-bandwidth
   reduce the number of instructions of a program
2. reorganize data into regular sequences
3. simple loop constructs
   removing the control overhead

- extensions
  - the instruction set
  - the function units
  - the register sets
  - the memory

ILP (Instruction Level Parallelism)

- multiple-issue machines
- combination of
  - statically scheduling
  - dynamical analysis
- to execute concurrently many instructions as possible
- the actual evaluation phase of several different operations
- execution rate : more than one operation per cycle

Pipelined Processor

- simple pipelined processor
- only one operation in each phase at any given time
  - IF (instruction fetch)
  - ID (instruction decode)
  - AG (address generation)
  - DF (data fetch)
  - EX (execution)
  - WB (write back)

VLIW Processors

- multiple-issue machines
- group instructions according to dependencies
  - statically scheduled
  - dynamically scheduled

Multiple Issue Machines

- Superscalar
  - dynamically examines the instruction stream
  - find out independent and concurrently executable instructions
- VLIW
  - depends on the compiler
  - analyze the available operations (OP)
  - schedule independent operations into wide instruction words

VLIW Processors

- multiple-issue machines
- group instructions according to dependencies
  - statically scheduled
  - dynamically scheduled

VLIW Processors

- multiple-issue machines
- group instructions according to dependencies
  - statically scheduled
  - dynamically scheduled

VLIW Processors

- multiple-issue machines
- group instructions according to dependencies
  - statically scheduled
  - dynamically scheduled

VLIW Processors

- multiple-issue machines
- group instructions according to dependencies
  - statically scheduled
  - dynamically scheduled

Reference

[1] M.J. Flynn and W. Luk, "Computer System Design : System-on-Chip", Wiley, 2011