Module-4
CSEN 3104
Lecture 30
17/10/2019

Dr. Debranjan Sarkar

VLIW Architecture

Very Large Instruction Word (VLIW): An introduction

- A processor that executes every instruction one after the other (non-pipelined scalar architecture)
  - may use processor resources very inefficiently, and
  - yield poor performance
- To improve performance in processors, many techniques are applied:
  - Divide instructions into sub-steps so that the instructions can be executed partly at the same time (pipelining)
  - Dispatch individual instructions to be executed independently, in different parts of the processor (superscalar architecture)
  - Execute instructions in an order different from that in which they occur in a program (out-of-order execution)

Very Large Instruction Word (VLIW): An introduction

- All these techniques require complicated hardware
  - larger circuits,
  - higher cost, and
  - higher energy use
- In contrast, the VLIW method depends on the software providing all the decisions regarding
  - which instructions to execute simultaneously, and
  - how to resolve conflicts
- So the compiler (software used to create the final programs) becomes far more complex, but the hardware is simpler than in many other means of parallelism

Superscalar versus VLIW architecture

- In superscalar design
  - Conventional instructions, conceived for sequential processors
  - The number of execution units is invisible to the instruction set
  - Each instruction encodes one operation only
  - For most superscalar designs, the instruction width is 32 bits or fewer
- In VLIW design
  - Long instruction words (64 to 1024 bits), each comprising a field (or opcode) for each execution unit
  - One instruction encodes multiple operations, at least one operation for each execution unit of a device
  - For example, if a VLIW device has five execution units, then a VLIW instruction for the device has five operation fields, each field specifying what operation should be done on that corresponding execution unit

Superscalar vs. VLIW architecture

- Superscalar
  - Instruction scheduling done dynamically at run-time by the hardware
  - Data dependency is checked and resolved in hardware
  - Need a look-ahead-hardware window for instruction fetch
- VLIW
  - Instruction scheduling done statically at compile time by the compiler
  - Data dependency is checked by compiler
  - In case of unfilled opcodes in a VLIW, memory space and instruction bandwidth are wasted

VLIW Architecture

- VLIW architecture is designed to take advantage of instruction level parallelism
- A typical VLIW machine has instruction words, hundreds of bits in length (Concept of horizontal microcoding)
- Multiple independent functional units are used concurrently in a VLIW processor
- Show Figure
- All functional units share the use of a common large register file
- The operations to be simultaneously executed by the functional units are synchronized in a VLIW instruction (256 or 1024 bits per instruction word)
- Instruction of a VLIW processor consists of multiple independent operations grouped together
- Different fields of the long instruction word carry the opcodes to be dispatched to different functional units

VLIW compiler

- Programs written in conventional short instruction words (e.g. 32 bits) must be compacted together to form the VLIW instructions
- This code compaction must be done by a compiler which can predict branch outcomes using elaborate heuristics or run-time statistics
- Compiler is responsible for static scheduling of instructions
- Compiler finds out which operations can be executed in parallel in the program
- It groups together these operations in single instruction which is the very large instruction word
- Compiler ensures that an operation is not issued before its operands are ready

VLIW instruction

- One VLIW instruction word encodes multiple operations which allows them to be initiated in a single clock cycle
- The operands and the operation to be performed by the various functional units are specified in the instruction itself
- One instruction encodes at least one operation for each execution unit of the device
- So, length of the instruction increases with the number of execution units
- To accommodate these operation fields, VLIW instructions are usually at least 64 bits wide, and on some architectures are much wider upto 1024 bits

Pipelining in VLIW Processors

- Each instruction specifies multiple operations (Show figure)
- The effective CPI becomes 0.33 in this particular example
- Instruction parallelism and data movement in a VLIW architecture are completely specified at compile time
- Run-time resource scheduling and synchronization are, in theory, completely eliminated
- One can view a VLIW processor as an extreme example of a superscalar processor in which all independent or unrelated operations are already synchronously compacted together in advance
- The CPI of a VLIW processor can be even lower than that of a superscalar processor (Example: Multiflow trace computer allows upto 7 operations to be executed concurrently with 256 bits per VLIW instruction)

Advantages of VLIW

- Dependencies are determined by the compiler and used to schedule according to function unit latencies
- Function units are assigned by compiler and correspond to the position within the instruction packet
- Reduces hardware complexity
  - Tasks such as decoding, data dependency detection, instruction issues etc. becoming simple
  - Ensures potentially higher clock rate
  - Ensures low power consumption

Disadvantages of VLIW

- Higher complexity of the compiler
- Compatibility across implementations: Compiler optimization needs to consider technology dependent parameters such as latencies and load-use time of cache
- Unscheduled events (e.g. cache miss) stall entire processor
- Code density: In case of unfilled opcodes in a VLIW, memory space and instruction bandwidth are wasted i.e. low slot utilization
- Code expansion: Causes high power consumption

Applications

- VLIW architecture is suitable for Digital Signal Processing applications
- Processing of media data like compression/ decompression of image and speech data

Thank you