Advanced processor architectures (APAZ_02.ppt)

Karel Vlček,
vlcek@fai.utb.cz
Institute of Applied Informatics FAI, UTB in Zlin
Superscalar processors

- Superscalar processors are characteristic by parallel issue of instructions and by execution several instructions during one instruction cycle.
- In the present days the superscalar processors represent the main direction of development in spite of that processors CISC must have the high complexity of instruction decoding.
Generic superscalar processor

- Typical property is **preliminary decoding** which is organised by the **issue window**
- During this process, it is tested data **independency** of issued instructions
- After preliminary decoding, the instructions are shifted from the issue window into the **queue**
- There exists another **possibility to execute instructions without the queue**
Dynamic planning of instructions

- Inserting of instructions into the queue can be realised into **more the one queue**, which are organised by functional units as so called **reservation stations**
- Timing of execution is realised during execution of **programme – dynamically** (IBM360/91 in 1967)
- The principal of **dynamic planning** was used in microprocessors 25 years later
Recognition and elimination of data dependences

- **Reservation stations** represent solution of partial queue for individual function units.
- To **five instructions** is sent in one cycle.
- Another way of solution is to use **central reservation station**.
- Enhancement of computational power can be reached by **execution of instructions without the queue**.

Karel Vlček  
Advanced processor architectures
Reservation stations

- Central reservation stations can be solved *without values of operands*, for this case, it is used *table of score of pre-processed operands*

- Reservation stations *with values of operands* allowed to *cancel of relation between* names of variables in the programme, and their values in registers
RS and Common Data Bus (CDB)

- **Reservation stations (RS)**, which cooperate are connected by the **Common Data Bus (CDB)**
- It is optimal, if it is used this number of **CDB**, how many is **functional units**
- Every **RS** must do several associative look ups for all flags of results, which are generated on the start
Tolerating of latency of jumps

- Jump instruction can be issued with limited number of further instructions only
- Control dependency represent the main obstacle for computing power of processors
- The unit called „Branch Unit“ offers solution, in which are realised loading and prediction of jumps independently to other units of processor
Speculative processing of instructions

Speculative processing of instructions contains:

- Conditional stop of the loops
- Other conditional jumps (are executed in 50% cases)
- Non-conditional jumps including jumps into the sub-programme and returns
Prediction of jumps

- Fix prediction
- Real prediction, if it depends only to programme code, it is called **static prediction**
- Dynamic prediction processes the table of history of jumps
- Processing of so called "jumps in the last time" with destination addresses of jumps enhances computational power
Overlapping of memory operations

- The memory consistency is described as the co-operation of ALU, and MMU.
- Other models, which are not so strictly formulated, allowed, that the processing of instructions will continue, after non-finished memory operation with long latency.
- The models, which are strictly formulated, use so called prevention of writing.
Prefetching of data

- **Data pre-fetching** can be used as the tolerating of latency of memory operating in two versions: with the *software*, as well as with the *hardware* support.
- The *compiler* co-operate on the data pre-fetching.
- The effective **data pre-fetching** is this one, which *brings operands just-in-time*.
Limitations of superscalar processors

- The processed problems ask no more than four instructions in 90% cases.
- An acceleration can reach limit value 2.5 times higher number of issued instructions in a cycle.
- The processor PicoJava represents a successful solution with memory LIFO (last in first out) organisation.
Advantages of PicoJava processor

- **FIFO memory** can contain several instructions with various length with various number of variables
- The **instruction decoder can decode to two instructions** in the cycle
- Acceleration is supported by the method **instruction folding**
- **Threads synchronisation** and **garbage collection** is realised
Advantages of PicoJava-II processor

It is supported **object oriented** OO programming:

- **Methods calling**
- **Hidden loading** of local variables
- **The link of instructions** can manipulate with six levels
Superscalar processors – overview (1)

The **basic idea** can be expressed as follows:

- The solution of execution units use the **principle of pipelining**, the new **instructions** are called in the following machine cycle.
- It is enhanced **the number of instructions in actual machine cycle**.
- It must be done **addressability of memory for data**, which are processed by instructions.
Superscalar processors – overview (2)

- It is necessary, for the practical solution, to be used specialised cache, for the reading and writing of operands for the execution of operation by every instruction of programme independently, if it is possible.

- Superscalar processors are done for today actual universal conception, which is used by microprocessor systems.
Realised superscalar processors

- The computer systems with high computational power are for example: Control Data 6600, and 7600, Cray-1 as well as „top-of-the-range“ IBM mainframe computers
- There are used hardware solutions for execution variable types of instructions
- The first example is introduced execution of fixed point, and floating point calculations
Fixed, and floating point (1)

- More execution units – example

Diagram:
- Instruction Loading
- Instruction Decoding
- Floating Point Calculation
- Memory
- Bus
- Fixed Point Calculation
Fixed, and floating point (2)

- Basic conception of superscalar processors uses **differentiation of execution units for different operations**
- It is supposed, that **execution of operations consume equivalent time period**
- This **assumption is not possible to fulfil, in praxis**
- The calculation, moreover, **require changes of order of executed instructions**
Superscalar conception (1)
Superscalar conception (2)

- The acceleration is possible only if there were changed the order of executed instructions by this way to be reached data independency
- This assumption can be fulfil, but it is necessary to resolve data dependency
- There exists three ways of solution of conceptions of superscalar processors
Superscalar conception (3)

- **Static conception** – these processors **issue** instructions by the programme, and **execute** instructions **in original order**

- **Dynamic conception** – instructions are **issued** in order by programme, but the order of execution is **optimised**

- **Dynamic conception with speculation** – for the case of **branching** are **executed the both branches** of possible executions
Superscalar conception (4)

Terminology of conceptions:

- **Scheduling** – definition of the order of instruction execution (static/dynamic conception)
- **Issue** – the moment of fetching of operands for instruction, if the order of execution is by programme
- **Execution** – in-order / out-of-order
- **Speculation and Committal** – about the result
Superscalar conception (5)

Terminology of conceptions (continuation):

- **Write Back** – In real execution is not write in every cases, but only if, the data validity is done, to this moment is the result loaded in auxiliary register.

- **Retirement or Completion** – the instruction is finished, and operands leave the registers of link of operation execution.
Data dependences, and hazards (1)

- An example of changed order instruction
Data dependences, and hazards (2)

- The output order must be changed, if the results of the two instructions must be loaded into the same register.
- As an "Output Dependency" is called situation, if the first result can rewrite the second one, because the second was finished in shorter time, then it was expected.
- As an "Anti-dependency" is called situation, in which the valid data are in opposed order.
Data dependences, and hazards (3)

- The solving of problems of data dependences is done in implementation **duly number of registers** for order of instructions for the algorithm or in this, that the instructions are executed in the original order of programme.
- In this case cannot bring the time advantages.
- Superscalar architecture **needs the hardware support by registers** for its implementation.
Superscalar processors computational power

There exist limitations, which must be considered:

- **Availability of registers**, which are usable in following operations WAW a WAR (it can be enhanced by using of register aliasing)

- **Foretell of branching** – it is possible do on the probability basis

- **Re-addressing of registers** – is based on their aliasing
Literature: