On a principal difference between multicellular architecture and data-flow machines EDGE, CISC, RISC, VLIW-architectures and their modifications.

When we talk about multicellular architecture, we assert it as an essentially new trend in creation of processor devices, though our opponents claim it to be either one more implementation of DF or EDGE-architecture, something multi-core and so on. Let us try to give an answer to the question "what is a radical novelty of multicellular architecture and how does multicellular architecture differ from the existing ones"?

As we know, functioning of any processor includes the execution of an informationally combined sequence of commands. Informational cohesiveness, in its turn, presupposes existence of commands which are the producers of results and the receivers of these results.

There are 2 possible types of connection – indirect (mediated) and direct connection.

In case of indirect connection it is the storage medium (register, memory storage) which is used for transmission of data (the result) between a producer and a receiver. The result is stored and only thereafter it becomes available for other commands but as a value of an element of a storage medium. Naturally, the result is disposed, i.e. the connection with a producer of a command is lost. Indirect connection is used in Von Neumann processors and a key principal of creation of such processors which is a principle of sequential programmable control whereby solution of any task by a processor is achieved by step-wise task-oriented alteration of his state – direct consequence of application of this kind of connection. This process of alteration of states is a result of performance of a command sequence in an exactly set order. Performance order is set by a programmer in the process of programming and may be realized either by a forced or an unforced sample method.

A forced method does not set any limits for commands placement in memory, however it requires each word of a command to have a hot spot field, where there should be an address of a next performed command. Nowadays this method is almost non usable.

An unforced method is a polar opposite solution. With this method commands are placed in memory sequentially, one by one in an order which reflects the order of execution of these commands. Field of address of the next command is only used in control transfer instruction when it is necessary to jump from the execution of one sequence of commands to the execution of another one.

There are rather many variants of implementation of the Von Neumann model. Usually, all of them are aimed at solution of one and the same task, which is increase of processor throughput. However, they solve this problem by various methods.

For instance: pipeline processor solves this problem by means of execution of a stream of commands at one time while using one executive device. RISC processors solve it by means of decreasing the number of operations being executed and accordingly, decreasing the run-time of commands. CISC processors solve this problem by increasing the number of operations which are being executed by one command and accordingly by decreasing total number of the commands executed.

Superscalar, multicore and VLIW processors provide the increase of throughput by means of combination of commands execution in space, i.e. with the help of several executive devices. The purpose of distribution of a command stream to executive devices in superscalar processors is solved mostly by hardware, meanwhile in multicore and VLIW processors it is solved by software.

In certain executions these two types may be combined. RISC and CISC processors usually have a pipeline organisation, but differ with the length of a pipeline. A pipeline is also widely used for reduction of run-time of a stream of commands which is sent to an executive device in superscalar processors. The most advanced methods of pipeline organisation not only combine the performance of a stream of commands at one time, but also provide for rearrangement of commands in a pipeline for elimination of collisions on data between stages of a pipeline. In this case a command in a pipeline is being sent to be executed not according to the order, but according to readiness. There are 2 mostly known

algorithms, which such rearrangement of commands: algorithm of centralized instruction window and Tomasulo algorithm.

TRIPS project is characterized by combination of methods as well. The authors assign the architecture of this project to EDGE architecture, which is mere execution of a data-flow graph. A crystal has two functional blocks, each of which is practically a VLIW processor containing 16 (4x4) 64-bit arithmetic and logic unit with a floating point. These blocks works as a DF machine but differ from a traditional DF machine by the fact that commands are sent to all the arithmetic and logic units simultaneously, but the process of their execution is anisochronous as it is in DF machines.

It is necessary to mention that usage of methods of parallelizing in superscalar and VLIW processors and methods of rearrangement of commands in a pipeline (which change the order of command performance) as well as methods of command execution according to their readiness still require usage of a key principal of Von Neumann model – step-wise alteration of a processor's state. Regardless of order of command execution, the states of a processor which are formed by this order are realized in the sequence according to which they were placed inside of memory, but not executed. This principle limits possibilities of the abovementioned methods and increases costs for their realization as well as requires solution of a complicated task of parallelizing using more than one processor for program execution.

Direct data connection presupposes that the result is sent to receivers and then it becomes a part of these commands. This means that the result exists only at the moment of transmission and disappears after transmission is complete.

There are two possible ways of realization of direct connection. They are: address dispatch and broadcasting.

In the case of address dispatch the result is sent to a particular command-receiver. A producer of the result is an initiator of transmission.

In case of broadcasting all the results are sent to all command-receivers. In order to choose the required ones, the results are being named, i.e. each result receives an individual name which attaches it to a command-producer. A command-receiver carries out selection of required results among the stream of commands by their names.

Implementation of address dispatch is a key principal of creation of DF processors. Generally, each command word of such processors contains a code of an operation, argument fields, address of a command word, which receives the result of an executed operation, and an argument field where the received value will be put. As a result, the sequence of commands in a program is irregular and generally can be arranged in programs' memories in a random way. A command is selected and executed according to "the readiness of an operand", i.e. at the moment when the command has received all the results of execution of other commands which are necessary for execution of this very command. The principle of execution of a data-flow processor commands according to "the readiness of operands" allows realizing parallelism by a "natural" way (without solving the task of parallelizing but requires the usage of special programming language, single-assignment language).

Reduction processor model uses address dispatch as well, but it is specified in a command word with indication of address of a command. The result of execution of this command is used by the command as an operand. In this case the commands are selected from memory and performed according to "command enquiry". Such a model makes it possible to refuse from commands arrangement and thus they can be arranged in memory in a random way. The reduction model, as well as the stream one, provides for the "natural" organisation of parallelism, but requires coding of programs of special languages (functional programming language).

All potential advantages of DF (stream) and reduction processors are connected with usage of direct data connections, and all disadvantages are bound with usage of address dispatch. Multiple efforts of modification of address dispatch did not alter the sense of this method (for instance, tag input is a virtual address extension ("address" & "tag"), but not a change of a paradigm). But these efforts proved the significance of potential advantages. Nevertheless, all the efforts remained as research projects only.

Broadcasting is a fundamental principle of multicellular architecture. Realization of direct data connections using such dispatch makes it possible to preserve all the advantages connected with direct connections and eliminate all the disadvantages, which characterize address dispatch. Moreover, we can see new possibilities which were unavailable for either Von Neumann model or DF processors (see also web-site materials www.multiclet.com).