

3D-Flow: An innovative Parallel-Processing system architecture that allows execution in real-time of a programmable complete algorithm in each processor even if it requires an execution time longer than the time interval between two consecutive input data and allows neighboring (North, East, West, South) data exchange. It is not a hypercube, it is not targeted to execute long general purpose programs, but instead it has an instruction set optimized to execute fast pattern recognition algorithms on pixels of images acquired at high rate, or particle identification algorithms on data acquired at a very high rate from several detectors in High Energy Physics experiments.

The innovative 3D-Flow parallel-processing system architecture was conceived to solve the problem faced in a Trigger system (or “decision box unit”) that should be capable of executing a particle identification algorithm (or First-Level Trigger algorithm or pattern recognition algorithm on pixels of images) on data acquired at a very high rate whose quantity would be impossible to store. If all data from an HEP experiment had to be saved, it would fill up every hard drive on the planet in about one day. The Trigger system has the task of selecting in real-time the highest quality collision events (“good events”) out of many. For example, at LHC it has the task of selecting 100 event/sec out of over 1 million event/second (or about 52 trillion event/day) generated by the machine.

Before the 3D-Flow invention, the problem was solved by creating three or more levels of Trigger. Each level, starting from Level-1 analyzes with increased complexity if the “event candidate” might be a “good event” that will meet all criteria (the three levels of Trigger algorithms) to be accepted and stored on the hard drive. If the “candidate event” fails on some criteria at level 1 or level 2, the process is aborted, data related to the “candidate event” are discharged from the temporary buffer memories to make room for another possible “event candidate” as soon as possible.

Without the 3D-Flow invention, Level-1 trigger has to compromise between: a) complexity of the algorithm that is more or less capable of identifying the desired pattern (or event), b) sustaining an input data rate more or less high and analyzing all pictures (or events) or dropping many of them; c) cost of the electronic system that could be more or less high with a low development cost of low performance electronics or a very high development cost of Application Specific Integrated Circuits (ASIC) with nanometer technology running at GHz executing a fixed algorithm.

Typically the designer of the Level-1 Trigger system (or “decision box unit”) has the dilemma of favoring one requirement to the detriment of another. The invention of the 3D-Flow architecture creates a revolution in the way Triggers can be implemented, eliminating the constraint of designing different Trigger levels (for example Level-1 and Level-2 could be combined increasing the efficiency in capturing more “good candidate events”). The invention of the 3D-Flow architecture overcomes all the above mentioned limitations of current Trigger systems and allows executing a complex and most efficient algorithm in real time to be very selective in discriminating the background noise and identifying even the weak signals carrying the information of the “good event” (or object satisfying the pattern recognition algorithm). It accomplishes this using low-cost off-the-shelf components and technology and it can analyze accurately every picture for the existence of a “good event” (or object satisfying the pattern recognition algorithm) even if the algorithm takes a time for execution that is longer than the time interval between two consecutive pictures.

How is this done? The invention works like a simple game. The first picture “A” goes to a 3D-Flow parallel processing system (layer 1 of processors). The second picture “B” arrives and, because layer 1 is busy, a “bypass switch” forwards it to a register associated with each processor of layer 1 (“Bypass-Register-A”). The third picture “C” arrives, and because layer 1 is still busy, a “bypass switch” forwards it to “Bypass-Register-A”, while picture “B” that was in that register, at the same clock cycle is forwarded by a second ‘bypass switch’ to layer 2 of processors and so on. Processor layers are added in numbers that will allow finishing execution of the algorithm in layer 1 (or in any layer, even if it requires long execution time) even if complex, but that will provide the best chance to identify the desired object (or “good event”). In the event a physicist would like to increase the complexity of the algorithm because he trusts it will be more selective, then one layer (or more) is added to the system. In the event the data rate or the “luminosity at LHC” is increased providing more data, but also more noise and a more selective (complex) First-Level Trigger algorithm is needed, then one layer (or more) can be added to the system.

The beauty of this innovative 3D-Flow architecture is that it is not necessary to use the most expensive leading-edge technology (e.g. an expensive 40 nanometer, GHz technology), but the most cost-effective technology, even at low speed can solve the problem. Thus the innovative 3D-Flow system is technology-independent. When using a low cost, lower performance technology, one just needs a few more layers of 3D-Flow processors. However, if the volume of the 3D-Flow programmable processors increases to justify the development cost (not just of a “Hard Copy FPGA-Altera” process of a few hundred thousand dollars, but that of a “Gate Array” process, or that of a more expensive “Standard Cell” process, or that of even a more expensive development of several masks costing a million dollars in a “Full Custom” process), then the most cost-effective solution that considers the total cost of the system (cost of the development, plus cost of the component) will be chosen. In the latter case of a higher performance technology, the number of layers of processors needed to execute the same algorithm will be fewer because each processor, running faster, can accomplish more algorithm steps.