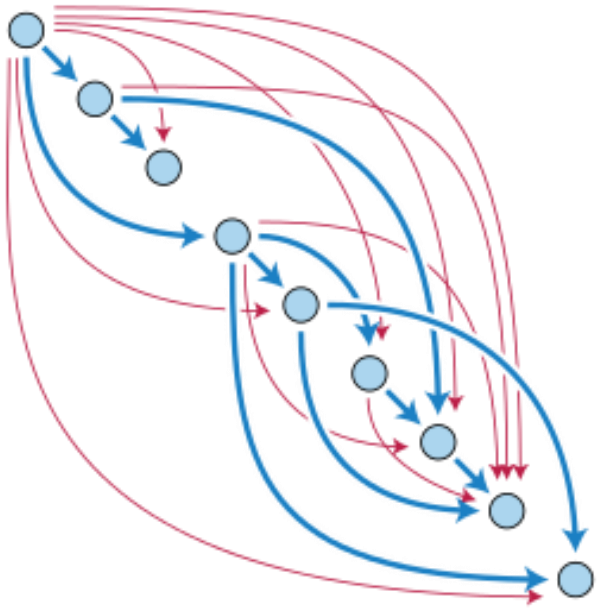
ASIC² Project:
Efficient Column Reuse in a Memristive Memory Processing Unit

Background: **Nowadays, the performance of computer systems is significantly limited by the speed of the memory. Data transportation between the memory and the processor is time consuming and wasteful in energy.**

One of the leading ideas for solving these issues is to transfer part of the processing capabilities of the processor into the memory itself. For data-intensive applications, this means a significant increase in computing processing power, while saving a significant amount of time and energy.

For achieving that, we developed a new computer architecture approach where a *memristive Memory Processing Unit* (mMPU) replaces the memory. This novel memory with processing capabilities is based on memristors. The memristor is a passive circuit element, predicted in 1971 by the circuit theorist Leon Chua. The first prototype of this element was unveiled in 2008 by HP labs. The device remembers its history, by varying its own resistance, so it can be used for memory applications. It also enables the formation of basic logic circuits, based on the MAGIC logic gate. The combine of memory with logic enables to perform logic operations within the memory itself, thus to explore advanced non-von Neumann architectures.

In previous work, we developed a tool which outputs the execution sequence of any desired logic operation. This tool assumes the structure of the logic circuit is a tree, whereas it is truly a DAG (Directed Acyclic Graph). Improving the tool to consider that will probably improve the latency of the execution sequence the tool produces.



Project Description:

- In this project, the students will improve the current Matlab code to consider the DAG-based circuit implications, by implementing the idea presented in [1].

Contacts info:

Rotem Ben Hur

rotembenhur@tx.Technion.ac.il

VLSI LAB

ASIC² ARCHITECTURES SYSTEMS INTELLIGENT COMPUTING INTEGRATED CIRCUITS

[1] R. Govindarajan, et al., "Minimum Register Instruction Sequencing to Reduce Register Spills in Out-of-Order Issue Superscalar Architectures".