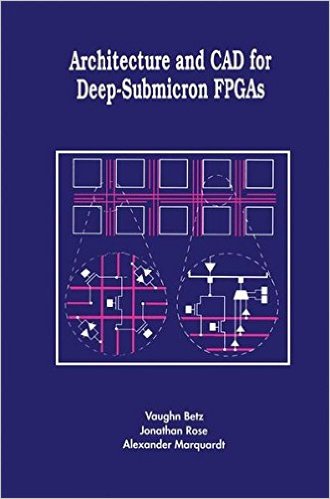
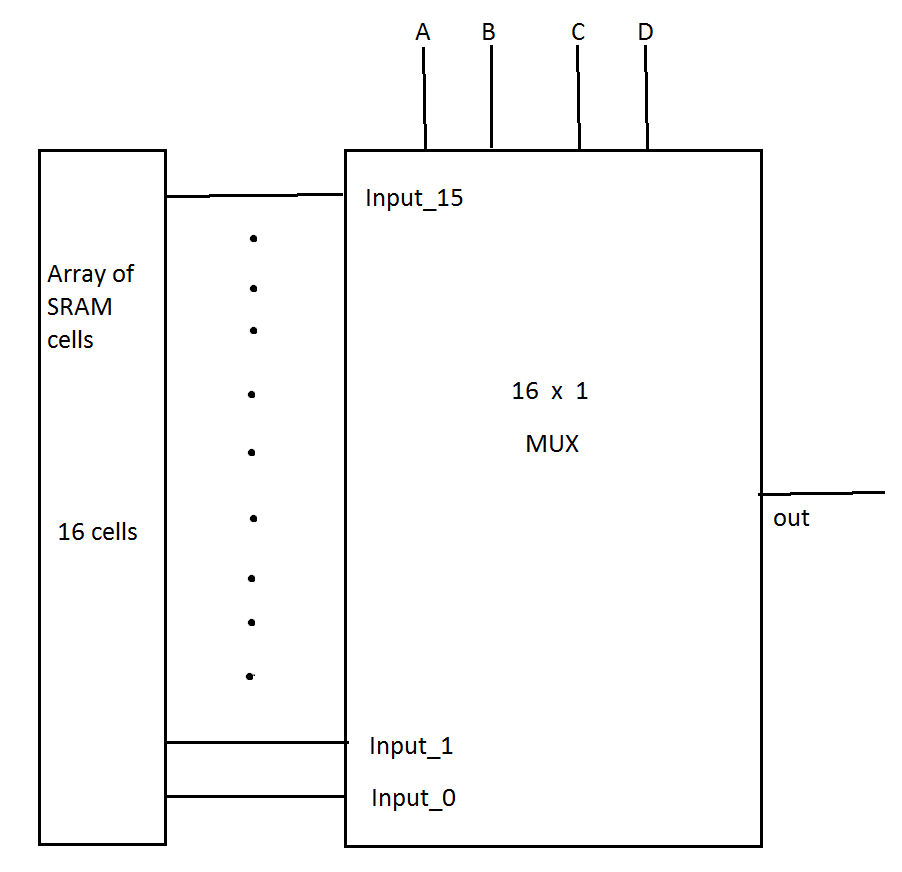
**Yang Azevedo Tavares – 6/13/2016**

**FPGA Project Description**

The architecture that is about to be presented is based on the book “Architecture and CAD for Deep-Submicron FPGAs”, also on several discussions and available time to be accomplished.

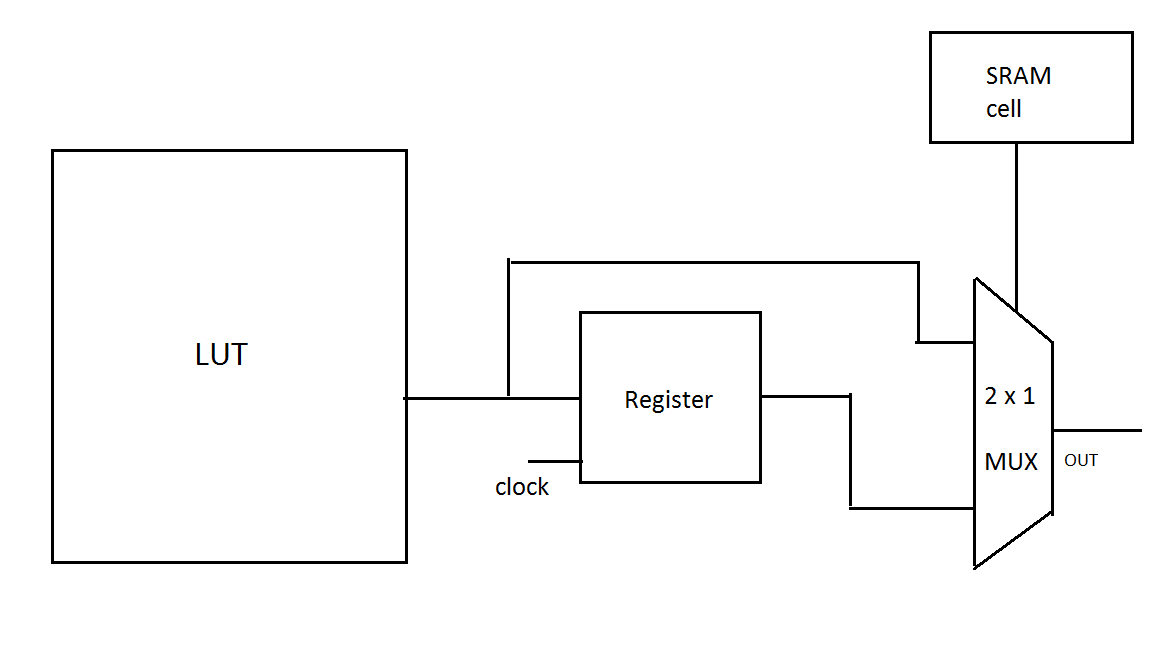


We are going to implement an FPGA with a single type of a Tile, which will have four LUTs (Look-Up-Table) inside and each LUT will have four inputs, each output of the LUT will be connected to a D flip flop and after, a mux can choose between the D flip flop output or directly from the LUT. Studies presented on the book have shown that LUTs with 4-inputs leads to FPGAs with the highest area-efficiency, and most commercial FPGAs are based on 4-input LUTs, also, Tiles consisting of multiple LUTs will have a better area efficiency.



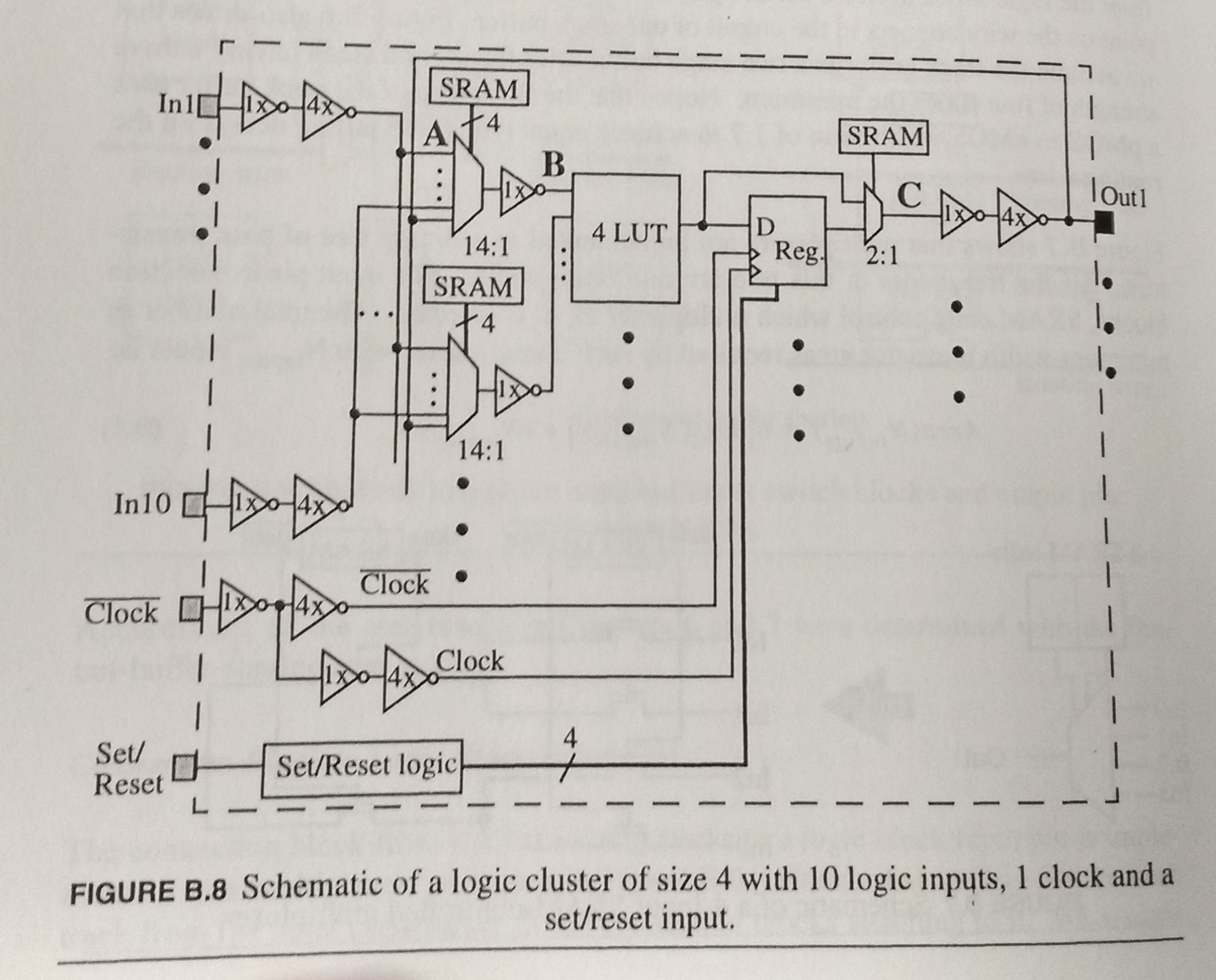
(LUT schematic)

The figure shown above is the drawing of the LUT configuration; it is a MUX that chooses between 16 signals over 16 SRAM cells, so it can be programmed to be any logic gate or combination of logic gates that has 4 inputs and one output.



(BLE schematic)

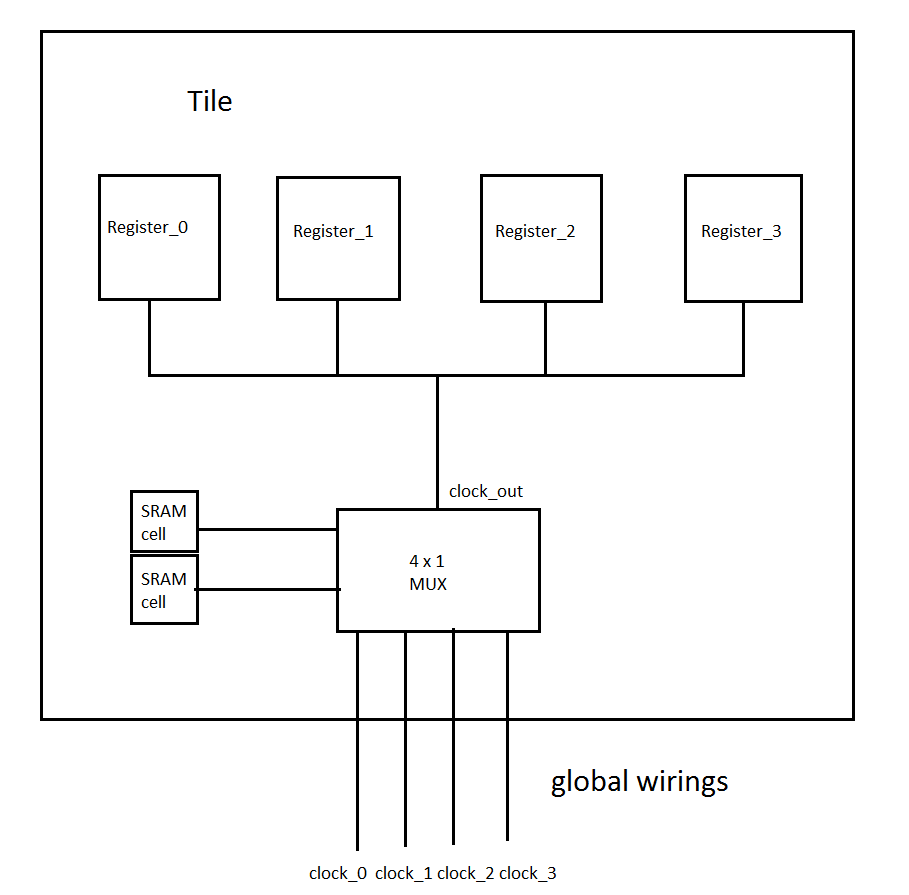
The figure above shows the BLE schematic, it is a MUX that can choose between the direct output of the LUT or the registered version controlled by a clock; it is useful for creating circuits that can be synchronous or asynchronous.



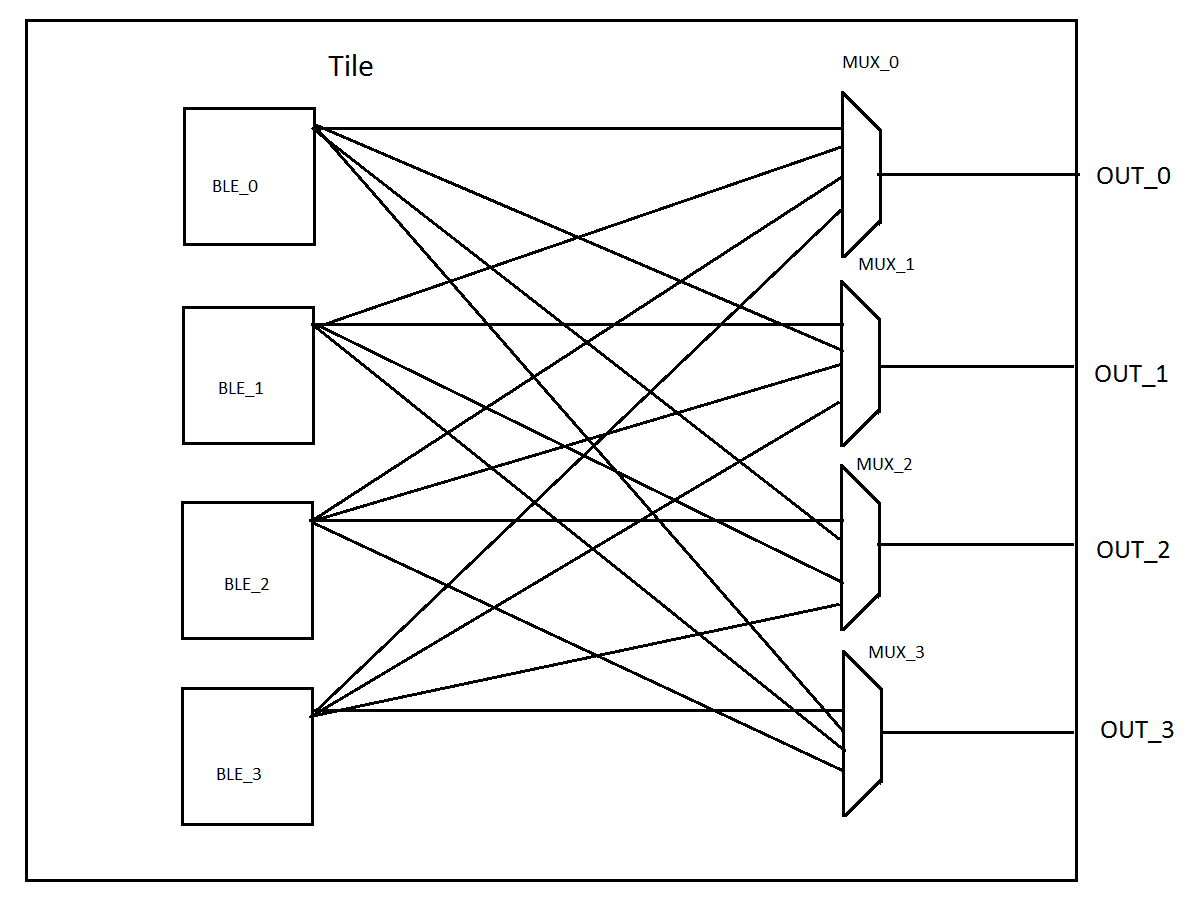
(Fully connected cluster. Figure from the book)

The inputs of the LUTs will have Muxes for each possible Input to the Tile or Output from the LUTs, which means that it is going to be a fully connected cluster. The number of inputs to the Tile will be based on researches from the book that says “2\*N+2”, where N is the number of LUTs on the Tile, resulting in 10 inputs per Tile.

The positioning of the pins around the Tile will be a full-perimeter configuration, which means it will be evenly distributed around the square to achieve the best area-efficiency. Each Tile will have Global wirings to program the SRAMs cells and to control the four different possible clocks to the registers, each Tile can only have one clock.

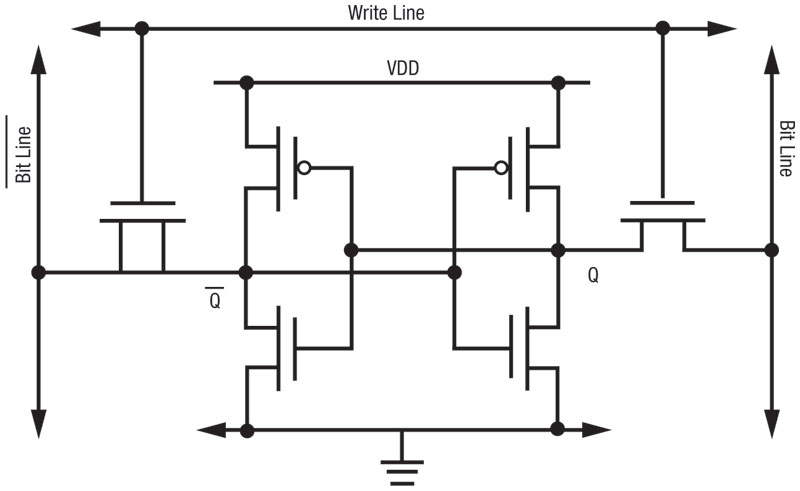


(Clock schematic for Tile registers)



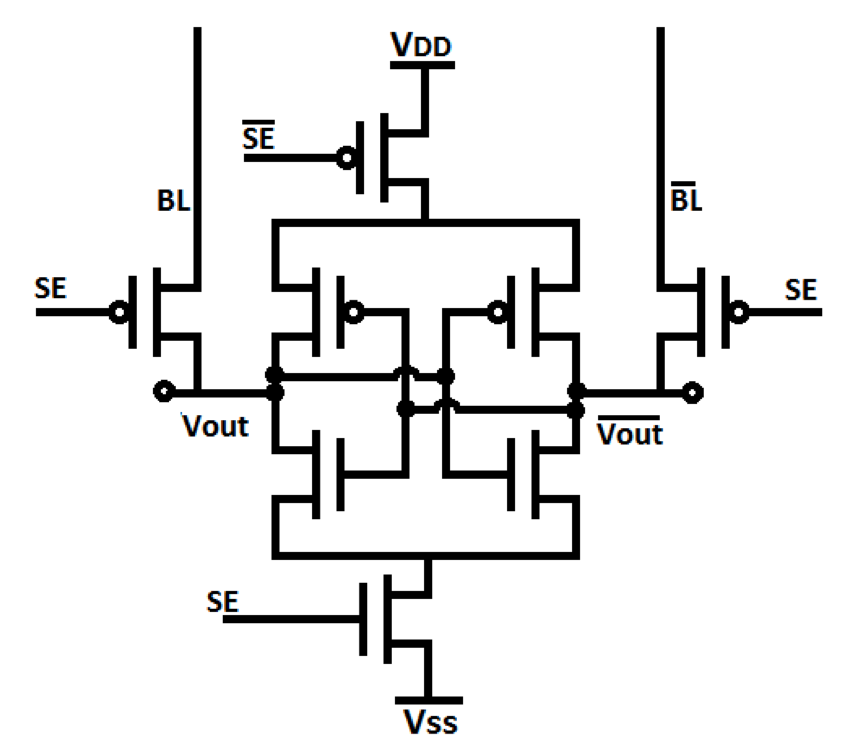
In addition, each output of the Tile will have a MUX that can choose between the 4 different outputs from the LUTs, so the inputs and outputs of the Tile will be logically equivalent.

To control several components such as LUTs, Muxes and routing switches, we will need SRAM cells. The sizing of each transistor on the SRAM is very important so it can hold the information and show it when doing common operations. There is a section on the book “CMOS VLSI Design a Circuits and Systems Perspective” which describes the proper sizing of a SRAM cell; it worked well for our simulations. 16 bits will program the SRAM cells on the Tile each time.



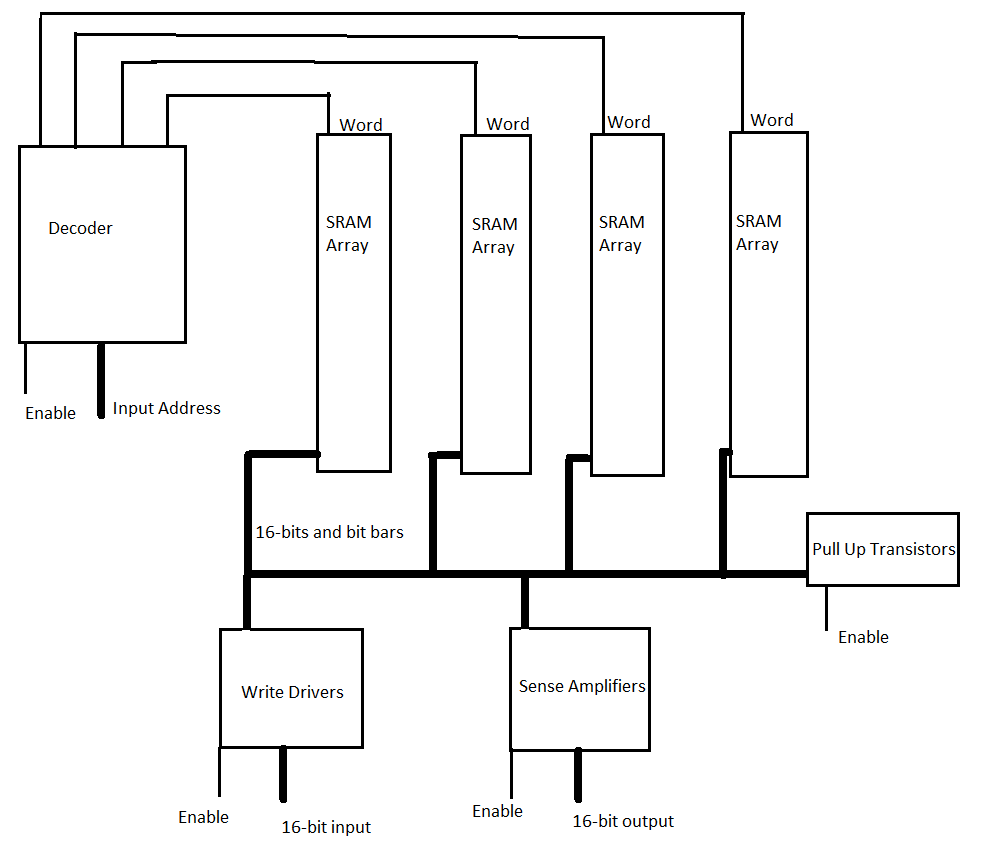
(Basic Ram cell schematic)

“Q” represents the bit stored in the SRAM cell on the image above. To change the value of the “Q”, it is needed to enable the write line (also called word line) with a logical 1 and drive the Bit line with the proper value and the Bit Line Bar with inverted value, so it will be electrically forced to change.

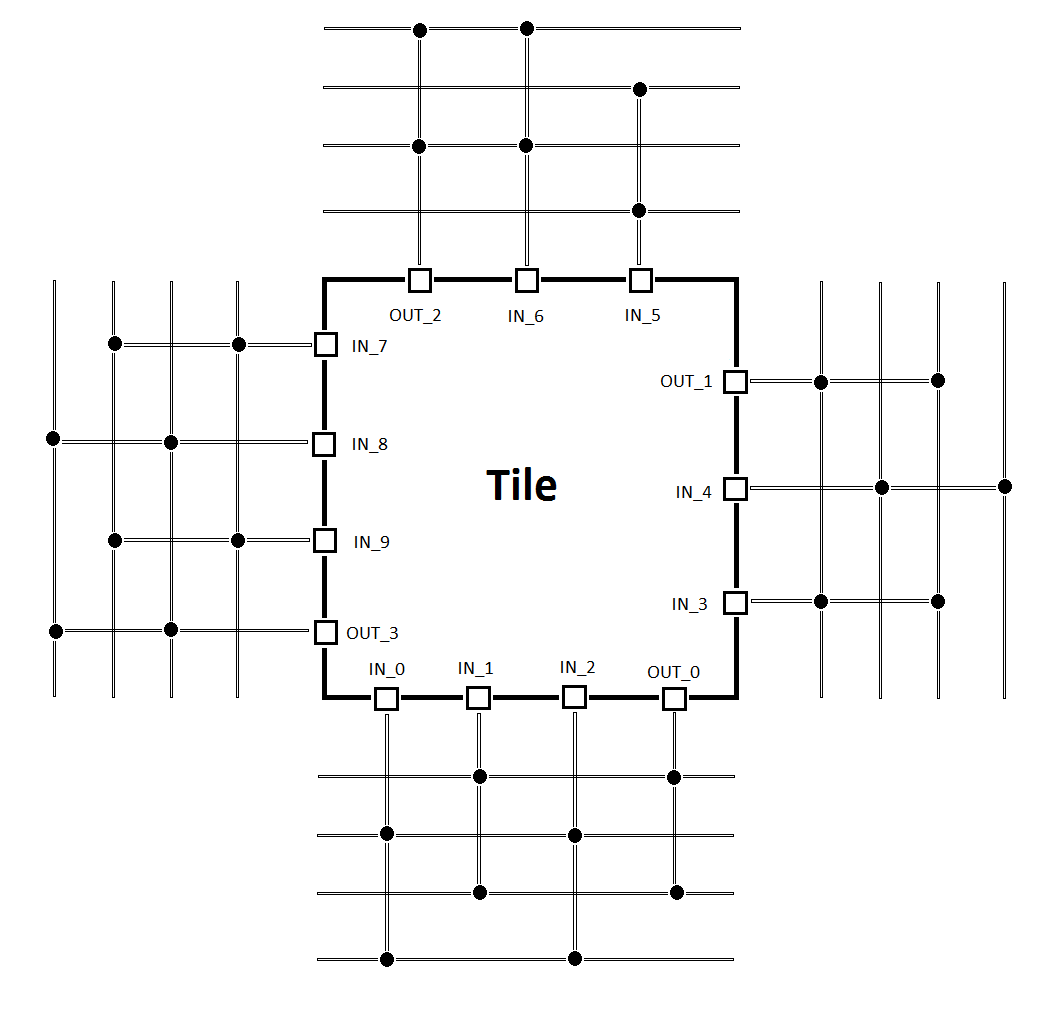


(Sense Amplifier Schematic)

The figure shown above is a Sense Amplifier which is used to read the current data stored in the SRAM cell, to do so, both Bit Line and Bit Line Bar needs to be pulled high and them tri-stated, after this the Sense will be enabled to read by pulling the SE down, it will be able to sense which of the bit lines have higher voltage and them amplify it.

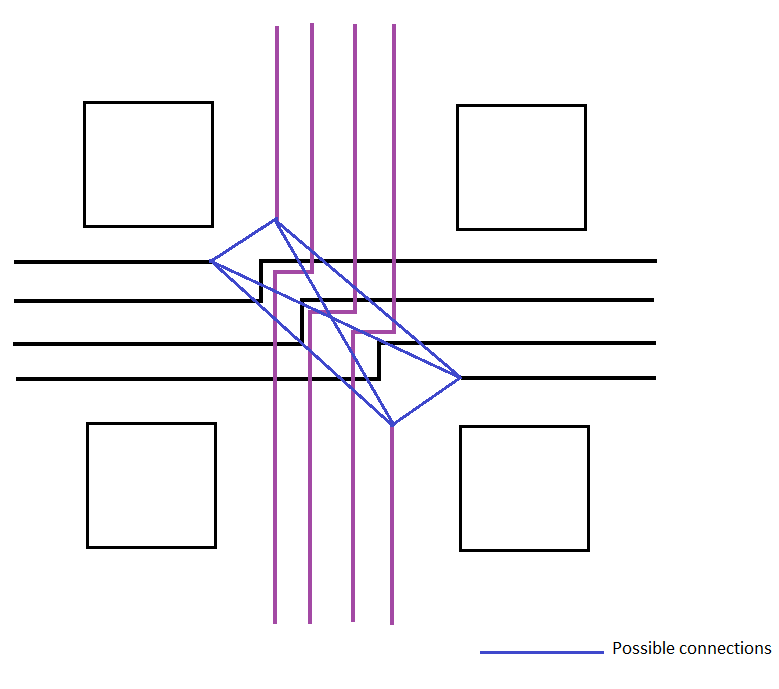


To program the SRAM cells on the tile, there will be a decoder choosing each single array of 16 cells per time, it is needed to manage the operations properly so the data on the cells will be written when needed. Each time you use one device on the figure, its needed to enable it and disable all others except the Decoder enable, it is only needed to disable the Decoder when the Address needs to be changed.



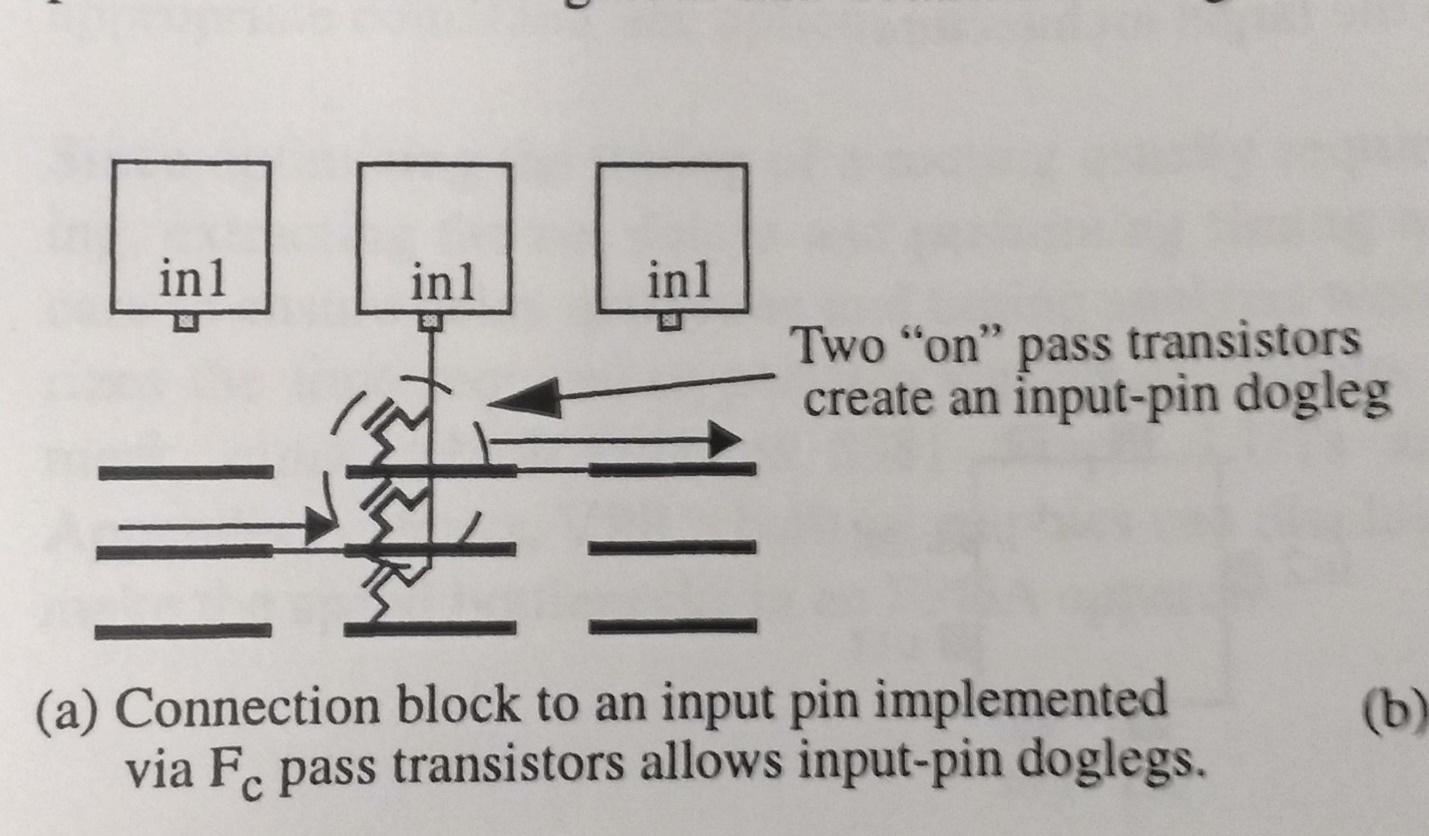
(Full perimeter configuration tile with 50% Fc)

Based on resulting placement files that are available on the University of Toronto Website, we have decided that an FPGA with 128x128 Tiles will be sufficient for our purposes. The number of tracks between the Tiles for routing will be /\*32 (45)\*/ and they will have the “segment start point adjustment” technique which makes routing uniform for entire FPGA. The population for connection between the pins of the Tile and wires will be FCinput 50% and FCoutput 50%. The wires will have a length of 4 logic blocks.



(“segment start point adjustment” configuration)

//The number of I/O pads will be four for the size of each Tile?



(Input-pin doglegs configuration)

//Switches made by a mix of pass transistors and buffers, the switch blocks will be on disjoint //configuration and the inputs, and output pins will be connected by the input-dogleg pin configuration.