**QuickSilicon**

# Address Target Buffer

**Remote Hardware Internship Take Home Problem**

## OVERVIEW

As part of this exercise you would be designing and verifying an address target buffer (ATB). The ATB holds the addresses of the target of the branch instruction and uses it if the current instruction is a branch instruction. The idea for this exercise is to test you on your hands-on RTL Design and Microarchitectural skills. You are free to use any resource necessary to meet the project specifications listed below.

## REQUIREMENTS

1. The design must be in SystemVerilog and must be sythesizable
2. The design must follow all the specifications listed below

## SPECIFICATIONS

1. The size of the target buffer must be parameterizable (assume the parameter as N)
2. Assume that the value of N wouldn't exceed 256 entries
3. Assume that after reset the ATB is empty and doesn't contain any valid information
4. The ATB would be written using the retire interface (describe below)
5. For every branch instruction the ATB must be accessed and the target address must be fed from the ATB if it contains a valid data for that entry
6. Assume that the ATB is a flop-array and hence the write would take a cycle while the read can get the data in the same cycle

## INTERFACE SPECIFICATIONS

The following section describes the input and output interface for the ATB:

```
input   logic          clk
input   logic          reset_n

// Read Interface
input   logic          is_branch_i       // The current instruction is a branch
input   logic [31:0]   pc_i              // The current program counter
// Write Interface
input   logic          retire_valid_i    // Branch Instruction got retired
input   logic [31:0]   retire_pc_i       // Retire PC
input   logic [31:0]   retire_tgt_pc_i   // Retire Target PC

// Output Interface
output  logic          atb_valid_o       // ATB output is valid
output  logic [31:0]   atb_tgt_pc_o      // ATB Target PC
```

1. Assume that the reset is an active low, asynchronous reset
2. The read and write interfaces are completely independent of each other. This implies that the read interface can be active while the write interface is also active hence would need to be handled accordingly

You are free to make any assumptions while coding the design as long as those don't violate the above specifications.

The deadline to submit this project is **Sunday, 18th Aug, 11.59PM IST**. Please submit the SystemVerilog code and the microarchitectural for your design.

All the best!