# Assignment Report: Interface of 8x4 Register Bank to 16x4 Memory System

Bratin Mondal (21CS10016)
Somya Kumar (21CS30050)

September 11, 2023

## 1 Introduction

In this assignment, we are tasked with designing an interface between an 8x4 register bank and a 16x4 memory system. The primary goal is to create a Verilog module that enables various operations, including reading and writing data to both the register bank and memory, as well as displaying memory contents. This report provides an in-depth overview of the modeling, implementation, and testing of the system.

## 2 Problem Statement

The assignment requires the development of an interface for an 8x4 register bank to communicate with a 16x4 memory system. The following operations need to be supported:

1. **Read and Store (Opcode 0):** This operation involves writing a 4-bit data value into a specified memory location identified by a 4-bit address.
2. **Register to Memory Transfer (Opcode 1):** In this operation, data from a specific register (selected by a 3-bit register number) is transferred to a specified memory location defined by an address.
3. **Memory to Register Transfer (Opcode 2):** Similar to the previous operation, data is transferred, but this time from a memory location (specified by an address) to a selected register (determined by a 3-bit register number).
4. **Display Memory Content (Opcode 3):** This operation involves reading data from a memory location designated by an address and displaying the contents.

**Algorithm 1:** Switch-case Flow for Operations

**Data:** Opcode, Data, Address, Register Number
**Result:** Result (if applicable)

```
switch Opcode do
    case 0 do
        // Read a 4-bit data and store it into memory
        Write data to memory location specified by Address;
    case 1 do
        // Transfer data from register to memory
        Read data from Register[Register Number] and write it to
         memory location specified by Address;
    case 2 do
        // Transfer data from memory to register
        Read data from memory location specified by Address and store
         it in Register[Register Number];
    case 3 do
        // Display the contents of the memory location
        Read data from memory location specified by Address and store
         it in Result;
        Display Result;
```

# 3 Memory Modeling Approach

To model the 16x4 memory system, we have designed a Verilog module that includes address lines, data lines, write enable signals, and read data outputs. The memory system is implemented as a block memory generator with appropriate control signals for read and write operations.

# 4 Register Bank Modeling Approach

The 8x4 register bank is implemented as an array of eight 4-bit registers. Each register is addressed using a 3-bit register number. The register bank module includes the necessary control signals to enable writing data into registers and reading data from registers.

# 5 Operations

The operations specified in the problem statement are implemented within the Verilog code as follows:

1. **Read and Store (Opcode 0):** When the opcode is set to 0, the Verilog module performs a write operation. The data specified by the 'data_in'

input is written into the memory location indicated by the 'address' input. This operation effectively stores a 4-bit data value in memory.

2. **Register to Memory Transfer (Opcode 1):** With an opcode of 1, the module initiates a read operation from the register bank. The data stored in the specified register ('regNo') is transferred to the memory location identified by 'address.' In essence, this operation copies data from a register to memory.

3. **Memory to Register Transfer (Opcode 2):** When the opcode is set to 2, a write operation is performed to the register bank. The data stored in the memory location specified by 'address' is written into the register indicated by 'regNo.' This operation facilitates data transfer from memory to a register.

4. **Display Memory Content (Opcode 3):** With an opcode of 3, the Verilog module performs a read operation on the memory. The data residing in the memory location specified by 'address' is retrieved and assigned to the 'result' output. This operation allows for the display of memory contents.

# 6 Testbench Description

The testbench serves as a critical component in verifying the functionality of the implemented Verilog module. It generates a clock signal, initializes input values, and orchestrates a sequence of operations to thoroughly test the various functionalities of the module. The simulation is executed for a specified number of time units, and the results are displayed using the '$display' system task.