AEDVICES Consulting
166B Rue du Rocher de Lorzier
38430 Moirans– France
trainings@aedvices.com
http://www.aedvices.com

Training on
Functional Verification Methodology
Using UVM

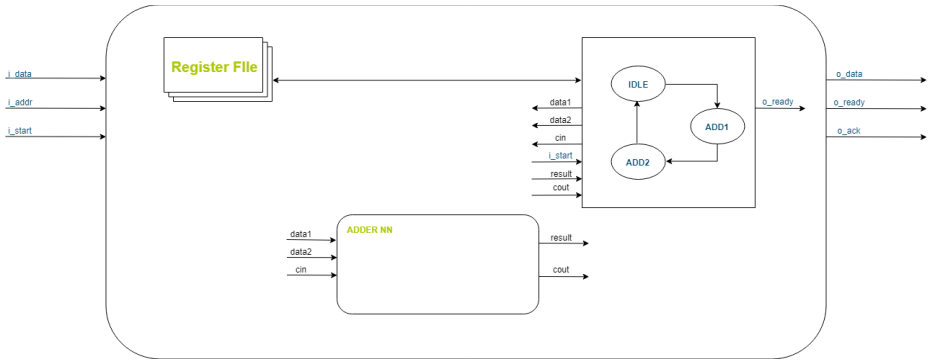
LAB
SystemVerilog Basics

Objectives

This lab goes through the main programming concepts of SystemVerilog language. It shows how to use SystemVerilog language to create programs, manipulate data and the main control flow of SystemVerilog language constructs.

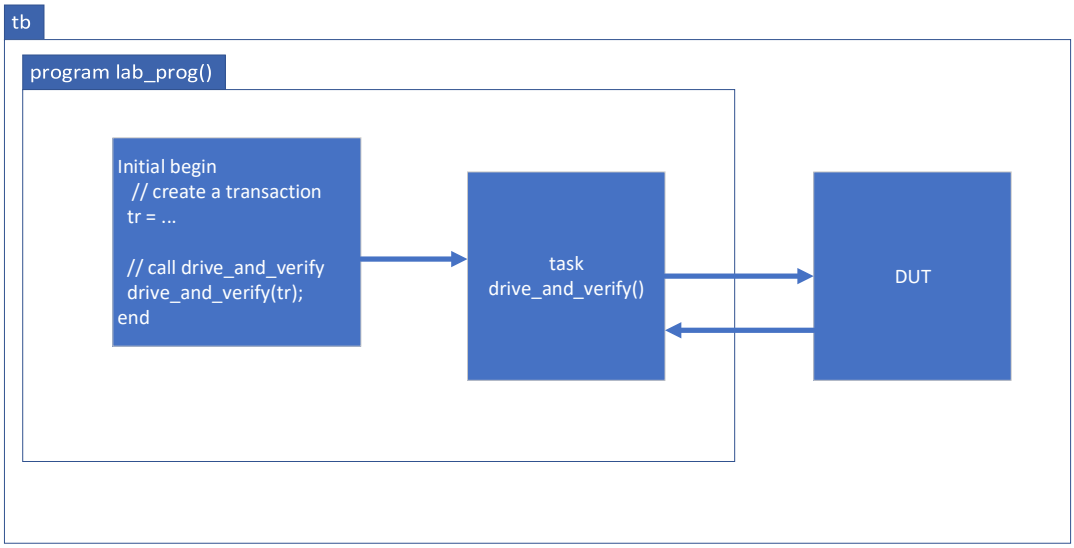
Global Explanation

In this lab we will implement tests to verify a extension of the previous adder, which includes additional registers.



It is not necessary to know the new DUT in details to follow this lab or the next ones, since all the necessary information will be exposed in each lab document. However, if you have the curiosity to know this new design an explanatory document called *AEDV-FA01* in the instructions directory.

The provided testbench “tb” instantiates the design under test and a SystemVerilog program “lab_prog” as followed:



The task drive_and_verify() is already implemented and provide a Bus Functional Model (BFM) to drive the design.

It receives a transaction as an argument a variable of the type trans_adder_s defined in the file “lab_utils_pkg” .

Therefore in this lab, we will create variables of the type trans_adder_s and we will then call this task to drive our design.
This will be done in the initial block of the “lab_prog” program (in the file lab_prog.sv)

Notes:
At the beginning of the programs, the following arrays, dynamic arrays and queues are instantiated and will be used in the lab:

```
trans_adder_t single_trans;        ///< defines a single transaction
trans_adder_t trans_sa[12];        ///< defines a static array
trans_adder_t trans_dl[];          ///< defines a dynamic list
trans_adder_t trans_q[$];          ///< defines a queue
trans_adder_t trans_aa[string];    ///< defines a associative array / map / keyed list
```

Get Started

To launch the simulation,

```
cd simulation/<SIMULATOR>
./runsim.sh
```

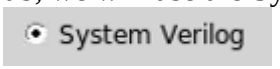
Note: in order to launch using LSF, you must first setup the variable LAB_LAUNCHER

```
setenv LAB_LAUNCER 'bsub -I -q gui -P mcdverif -R "select[rh60]"'
```

Alternatively you can run the command:

```
bsub -I -q gui -P mcdverif -R "select[rh60]" ./runsim.sh
```

In this lab, we will use the SystemVerilog mode. So the following checkbox should be set:



Browse and load the labs/NN-LABNAME/lab.sv file

Instructions

- Open the file <SANDBOX>/labs-Xdays/labNN-systemverilog_basic/tb.sv" in your favourite editor.
- Search for "LAB-TODO"
- Instantiates the program "lab_prog"

- Open the file <SANDBOX>/labs-Xdays/labNN-systemverilog_basic/lab_prog.sv" in your favourite editor.
- Search for "LAB-TODO"
- Follow the instructions present in this file.
- Compile, load and run.
- Don't hesitate to rerun on each step to understand your changes.

Questions:
STEP1 – Why changing single_trans content does not affect values in the list ?

STEP5 & 6: What is the difference between pop_back / pop_front ?

STEP9 – What happens if you try to read a key which does not exist directly without checking its existence before?