Verification of NXP SAC57D54H Local Memory Controller (LMEM) module using UVM

Ghodekar, Akash (MS Electrical Engineering) Ubale, Omkar (MS Electrical Engineering)

Introduction
Modern computers have seen a significant increase in cache sizes in recent years. These large caches need complex controller modules for proper operation. Logical or design faults in the controllers lead to failure of the entire computing system. Functional verification of the design using test benches can accurately locate faults and also speed up the overall development process. This project verifies the cache controller module of a popular NXP SoC using UVM test bench. The cache controller (LMEM) module provides a local memory interface to the on-board ARM Cortex processor. A fault in the LMEM module can severely disturb the SoC functioning and thus, needs proper verification.

UVM provides reusable test bench components that simplify the verification of complex digital designs. The scoreboards and sequences of the test bench will act as a reusable IP for various other controllers along with the targeted module. Objective of this project is to verify the functionality of the LMEM module along with the state of the cache controllers. By verifying the cache functionality, one can determine whether the average time required to access the memory is reduced which would result in high performance.

Methodology
All of the functionalities of the LMEM module, as well as the states of cache lines, will be checked using the UVM testbench. The Universal Verification Methodology (UVM) employs the System Verilog programming language, which is comprised of a series of base classes. UVM is used to construct a verification environment that can be updated and repeated. The UVM is made up of modules that are organized in a hierarchical structure and execute various functions. The LMEM controller, the pin-cache controller, and the SRAM controller are all checked using UVM. The cache registers will be checked to verify the cache controller. With register will be examined in accordance with its bits. If the bits in the cache registers are incorrectly configured, faults may be detected using verification checks. The master and slave ports can be checked to see if the SRAM controller is running properly. Check if no garbage data is coming in, and that the master is reading the slave at a real-time interval. Check if the master appends read and writes signals to the slave. The operating protocol of the PC and PS buses will be tested. As a result, the Local Memory Controller will be checked by checking cache registers, master and slave ports, and bus protocols.

For the PC and PS buses, this system has been enhanced with closely coupled memories. RAMs and caches are among the memories.

An updated 32-bit Harvard bus architecture is used in the Cortex-M4 processor. Low-order addresses (0x0000 0000 to 0x1FFF FFFF) use the Processor Code (PC) bus, while high-order addresses (0x2000 0000 to 0x3FFF FFFF) use the Processor System (PS) bus in a 32-bit address space. Standard operation involves as the bus names say, the PC bus is for code access and the PS bus is for data access [5]. These local memories offer access to RAM and cacheable address spaces in a zero-wait state. Four memory controllers and their associated memories make up the local memory controller.

- PC bus through Cache memory controller.
- PS bus through Cache memory controller.
- PS bus through SRAM lower.
- PS bus through SRAM upper.

A local memory controller (LMEM) is a circuit that can be connected to a processor or built into the Processor core. It is being used to increase the data transfer rate between the memory and the processor. It would have the logic to read and write into and out of the memory of a machine. The memory controller allows even when the processor is switched off, and data is stored in the Random Access Memory (RAM). By integrating LMEM with the CPU’s core, the core will operate at the required wavelength, removing any latency issues. As a result, faster data speeds between the processor and memory enhance the overall speed of the processor. The LMEM hosts the SRAM and cache controllers, these controllers, however, use external single-port synchronous RAM arrays. For the PC and PS buses, the LMEM includes address decode logic. This reasoning is used to route the core’s access to the different computer services.

TESTBENCH ARCHITECTURE
- The LMEM module has an initialization protocol which sets all registers to all-zeros. A data read from the register bank should output all-zeros after a reset is set high regardless of the values of other inputs.
- A common approach to verifying a control section that is configurable using a register bank is to verify the register bank first. Faulty design of the registers can lead to improper functionality of the entire DUT, thereby invalidating further tests. This report performs certain read write operations on the register bank to confirm the correct data storage inside them.
- Each of the 8 registers in the programmers model has a specific functionality. Values inside the register influence the output of the LMEM module. In this step, the functionality of each register is verified.
- Each cache line in the controller is influenced by a set of bits in the register bank. Verification of each line is necessary.

Analysis and Results
- The sequencer generated a set of random sequences was for driving the register bank of the DUT.
- An output monitor extracted the outputs from the transactions with the DUT and displayed it on the terminal for debugging.
- The sequencer also randomized the Reset, Qwite, Select, and Addresses as well. The Synopsis VCS tool compiles the testbench.
- It provides 3 types of messaging, namely UVM_INFO, UVM_WARNING, UVM_ERROR. UVM_ERRORS are critical errors and have to be resolved on priority.
- UVM_WARNING are unsolved issues that do not break the functionality of the module but is recommended to be resolved.
- UVM_INFO are informational messages used for probing and debugging. The compilation of the testbench starts from the top-level module.
- The top module invokes the run_test command. This initiates the compilation of the LMEM_test. The LMEM_test builds and connects the LMEM_environment.
- The environment builds and connects the Sequence, Agent, Interface, Monitor, and the Scoreboard.
- The agent builds the driver and the Input monitor.

Summary/Conclusions
In this project, a UVM testbench is developed to verify the LMEM module. The test environment developed is reused to verify several functions and specifications. Based on the set bits, registers used in cache controller and SRAM controller are verified. Functional verification of LMEM module will allow exchange of high data rates between processor and memory. Also, verification of the memory module will ensure design works at a desired frequency thereby providing enhanced performance of the processor.

Key References

Acknowledgement
We would like to thank our project advisor Dr. Lili He for guiding us throughout the project and for all the coordination and support provided by her. Also, special thanks to Prof. Morris Jones for his constant advising and mentoring.

We are grateful to our head of the department Dr. Thuy Le for giving us this opportunity to learn and execute our skills.