System Level Modeling using SystemC

Ganavi V S[1], Dr.Rekha K R[2]
[1]Scholar, M. Tech VLSI Design, and Embedded System, SJBIT, India
[2]Professor, Department of Electronics & Communication Engineering, SJBIT, India

***Abstract* - System level design and IP interchange are planned to be made possible via SystemC, a new modelling language based on C++. This paper first explains system level modelling features before quickly reviewing the hardware modelling features included in SystemC. In the SoC design flow, transaction level modelling (TLM) is suggested as a prospective improvement over register transfer level (RTL). In order to address SoC design processes such as early software development, architecture analysis, and functional verification, this article formalises TLM abstractions. The genuine hardware/software co-design is the most satisfying aspect of TLM. SystemC allows for modelling of systems above the RTL level of abstraction, including those that may be implemented in hardware, software, or a combination of the two. The new features make system level design activities easier, by connecting design specifications to hardware and software implementations, as shown by a simple design example.**

***Index Terms* - SystemC, Transaction Level Modelling, Register Implementation, Application of Registers.**

I.INTRODUCTION

Several models have been put out to enhance the level of abstractions and enable hardware-software co-design. In settings like SpecC, specifying at higher levels of abstraction is conceivable. If the hardware modelling descriptions is modeled on the C/C++ languages, which seem to be well-liked in the software community, then a unified and integrated method to hardware-software co-design is feasible.

The above-mentioned problems are addressed by the standard modelling platform SystemC, which is based on C++ and provides designing abstractions at the RTL, behavioural, and system levels. This is because when design complexity rises, very quick executable specifications are needed to evaluate system concepts, and only C/C++ is capable of providing the necessary degrees of abstraction, hardware/software integration, and performance. SystemC is an attempt to standardise a C/C++ design approach and consists of a class library and a simulation kernel.

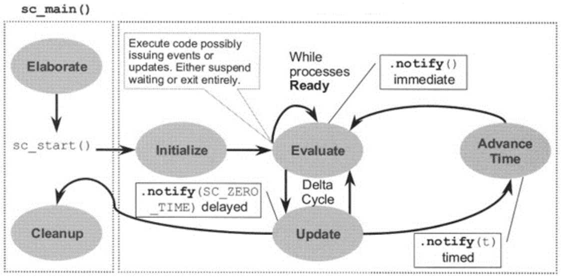


Fig 1: Process Management flow chart of SystemC

The module, which is a container containing one or more processes to represent the parallel behaviour of the design, is the fundamental unit of construction in SystemC. A module may also contain additional modules, illustrating the design's hierarchical structure. The code in each process runs sequentially, and System C has three different types of concurrent processes: SC_METHOD, SC_THREAD, and SC_CTHREAD. The processes are operated on by the simulation kernel. The test environment and the design are both parts of a complete SystemC model. An executable file which simulates the design as in given test environment is produced after compilation.

Modules use channels for communication whereas processes inside of them use signals. Modules have ports which are bound to interface methods, and the channels are abstract and available through their interface methods.

II. METHODOLOGY

1) TLM BUS MODELING

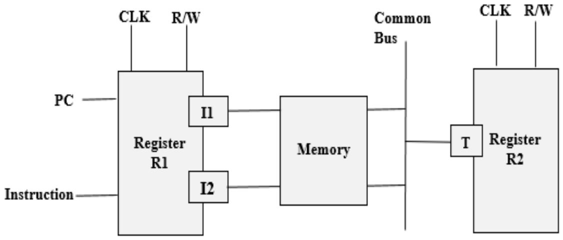


Fig 2: Block Diagram of Bus modeling using two Registers

A) REGISTER (R1)

Registers are memory storage units that are utilised to convey data for the Central Processing Unit (CPU) immediate data processing process. Any type of data, including dates, instruction sets, storage locations, bits, sequences, and characters, can be stored in the register. In register R1, clock, program counter and instructions with read or write signal is given.

B) MEMORY

Binary data is stored in memory units as words, which are collections of bits. Data output lines convey the information out of the memory after it has been stored there by data input lines. The direction lines The direction of data transfer is specified by read and write operations.

C) REGISTER (R2)

It has clock, read or write data and a target. The target consists of write data and address.

2) IMPLEMENTATION OF REGISTER APPLICATIONS

a) 4-bit UP COUNTER

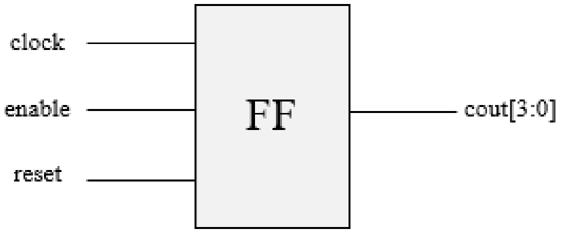


Fig 3: 4-bit Up Counter Implementation

Diagram depicts a 4 bit asynchronous UP counter. It can count between zero and fifteen. As a result, the flip flops will switch on and off at each positive or active edge of clock signal. The first flip flop is connected to the clock input. The 4-bit counter starts at 4'b0000 and increases to 4'b1111 before rolling over to 4'b0000. As long as a running clock is provided and reset is held high, it will continue to count.

b) 1:2 DECODER

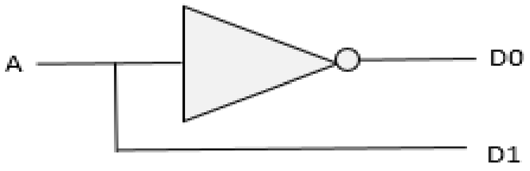


Fig 4: 1:2 Decoder

A digital decoder converts a number of digital input into an equal decimal code at its output since the word "decoder" means to convert or decode coded information from a format into another. A combinational circuit known as a decoder transforms binary data from n input signals to a maximum of 2n distinct output lines.

c) Arithmetic Logical Unit (ALU)

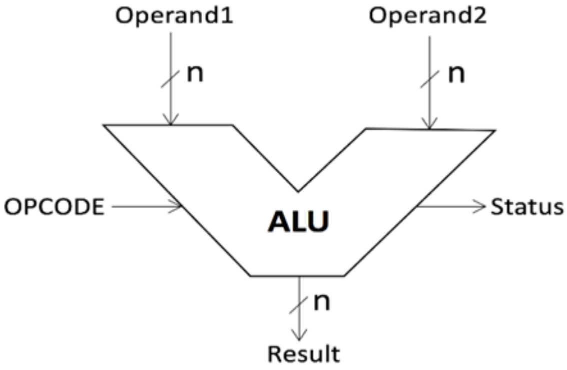


Fig 5: ALU Diagram

An ALU's inputs are the data to be processed, or operands, and a code designating the operations to be carried out. The ALU's output would be the outcome of the processed data. In many architectures, the ALU also features status inputs, outputs, or both, that communicate data about an operation's history or current status to external status registers.

III.RESULTS AND DISCUSSIONS

The Results obtained using EDA playground is depicted in below Figure 6:

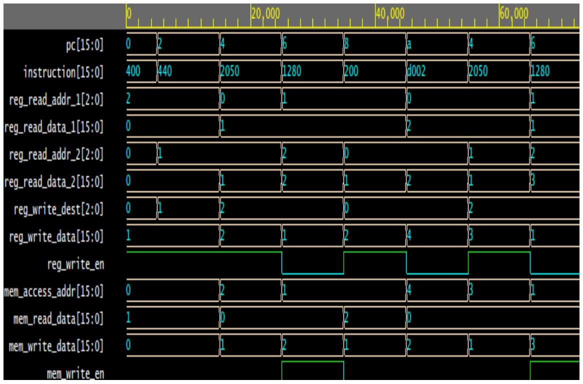
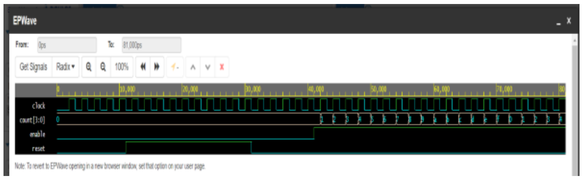


Fig 6: The data transaction output between two registers

The results of applications of registers is shown in Figure 7, 8 and 9:

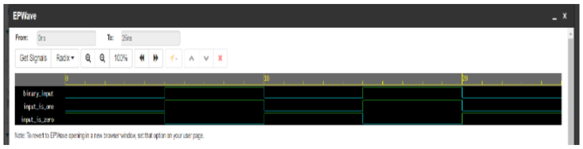


EPWave

From: 0ps To: 81,000ps

Note: To revert to EPWave opening in a new browser window, set that option on your user page.

Fig 7: The 4-bit counter Output

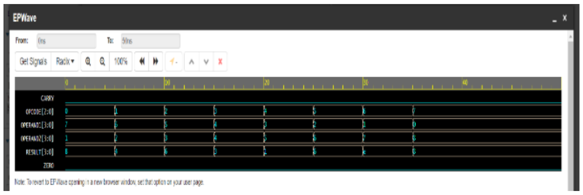


EPWave

From: 0ns To: 25ns

Note: To revert to EPWave opening in a new browser window, set that option on your user page.

Fig 8: The 1:2 Decoder Output



EPWave

From: 0ns To: 50ns

Note: To revert to EPWave opening in a new browser window, set that option on your user page.

Fig 9: The ALU Output

IV.ADVANTAGES

1. Cycle-Accurate model for Software Algorithm.
2. Interface of SoC (System-on-Chip).
3. Modeling at higher level of abstraction increases performance.
4. Executable Specification

V.APPLICATION

1. The complexity level of SoC sub-modules can be reduced.
2. Modeling contributes to a successful software and hardware organization.

VI.CONCLUSION

The SystemC modeling plays a very imporatant role in SoC design cycle. It incorporates both software and hardware models. So, it reduces complexity of using SoC submodules. Here, we had implemented how the transaction takes place between two registers using common bus. By using this register transfer process, we had applied to the 4-bit up_counter, 1:2 decoder, arithmetic logical unit (ALU) and the waveform is obtained, and the result is analyzed.

VII. FUTURE SCOPE

The approach applies to a wide range of SoC design applications. It can still be implemented for ADC or DAC operations for reduction of SoC design and its sub-modules.

REFERENCE

[1] Automated Design Understanding of SystemC-based Virtual Prototypes: Data Extraction, Analysis and Visualization, Mehran Goli and Rolf Drechsler Cyber-Physical Systems, DFKI GmbH, 28359 Bremen, Germany Institute of Computer Science, University of Bremen, 28359 Bremen, Germany 2020.

[2] Scalable Simulation-based Verification of SystemC-based Virtual Prototypes, Mehran Goli and Rolf Drechsler Cyber-Physical Systems, DFKI GmbH, 28359 Bremen, Germany Institute of Computer Science, University of Bremen, 28359 Bremen, Germany 2019.

[3] SystemC - A modeling platform supporting multiple design abstractions, Preeti Ranjan Panda Synopsys Inc. 700 E. Middlefield Rd. Mountain View, CA 94043, USA 2001.

[4] A Tractable and Fast Method for Monitoring SystemC TLM Specifications Laurence Pierre, Member, IEEE, and Luca Ferro, 2008.

[5] On the Mutation Analysis of SystemC TLM-2.0 Standard Nicola Bombieri, Franco Fummi and Graziano Pravadelli Department of Computer Science University of Verona Verona, Italy, 2009.

[6] Fast and Accurate Protocol Specific Bus Modeling using TLM 2.0, H.W.M. van Moll, H. Corporaal Technical University, V. Reyes, M. Boonen, NXP Semiconductors, Eindhoven, The Netherlands, 2009.