UVM based verification of iAPX 186 processor modules

S.Rajthilak and N.Mohankumar

Department of Electronics and Communication Engineering, Amrita School of Engineering, Coimbatore, Amrita Vishwa Vidyapeetham, India

E-mail: rajthilikasathiavrs@gmail.com and n.mohankumar@cb.amrita.edu

Abstract: UVM is used to enable faster development and reuse of System Verilog based Verification Environment. The use of UVM is to develop reusable test bench. The UVM class library contains configuration database, TLM and component hierarchy. Each and every component in the verification environment has a specific functionality. The UVM has a list of base class which are pre-stored and System Verilog environment can be built by deriving the base classes. Two parameters have been used namely Coverage and Assertions. Coverage is used to measure whether all features of the design specification is tested. Assertions are used for checking the behavior of the design. In this work, we design the individual blocks like Interrupt Controller, Timer module, DMA Controller, Execution Unit and Bus Interface Unit of 80186 processor using Verilog and Verify its functionality behavior using the two parameters using UVM.

Keywords: iAPX186 processor, Functional Verification, Universal Verification Methodology (UVM), System Verilog, Functional Coverage, Assertions, Simulation

1. INTRODUCTION

Functional Verification is the important process of VLSI design cycle. It is a process of verifying the RTL design and check whether its specification are met from a functional point of view. Verification is divided into two areas namely functional verification and physical verification. Functional verification is used to check whether the Design Under Test (DUT) executes the operability of the specification correctly. It is one of the most challenging areas in chip design. To implement the specification the designer understands the functionality of the paragraph describing functionality of the RTL code. Since both the process are manual and the specification inevitably leaves room for interpretation, there are large possibility for RTL designer to make mistake. To overcome this System Verilog and then UVM were introduced. It is the derivation of Verilog. The Universal Verification Methodology (UVM) [6] has class libraries that are used for the building System Verilog based verification.

The detailed information of Universal Verification Methodology and 80186 processor functionality will be discussed in the following sections. Chapter 2 containing Methodology which presents the various methods and tools used to validate the Results and Discussions are discussed in Chapter 3. Chapter 4 presents the conclusion.

1.1. Functional Verification

With reference to Figure 1, the test module is in charge for organizing the testbench. It is used to start the testbench blocks by constructing the next level below in the hierarchy environment. It is also used
to start the stimulus by beginning the sequence. The environment module is used for bundling next higher level block namely agent and scoreboard. UVM agent bundles the UVM blocks particular to an interface. The sequence-item defines signals generated and driven to DUT through the driver. UVM Driver is used to drive the signals inside sequence item into DUT. UVM Sequence tells the order in which the stimulus need to be generated and sent to the driver. UVM Sequencer is used to transfer the signals generated in sequence to the driver.

![Proposed UVM architecture for 80186 Microprocessor](image1)

Figure 1: Proposed UVM architecture for 80186 Microprocessor

UVM Monitor observes the signals, samples them and send it to the scoreboards. UVM Scoreboard is used to compare with observed values and expected values that is either got from the golden reference values or got from the reference model.

1.2. 80186 ARCHITECTURE

80186 is a 16 – bit Microprocessor [1]. It is the enhancement of 8086 Micro-processor. It has the upgraded 8086 version-2 CPU, DMA Control, PIC Controller, Clock module, Timer module, Chip selector module, Local Bus Controller which is referenced in Figure 2. Its clock speed is 8 – 10 MHz, Direct Addressing Capability to 1 MB of memory 64 KB I/O and 10 latest instruction types.

![80186 Microprocessor Block](image2)

Figure 2: 80186 Microprocessor Block [1]

1.2.1. INTERRUPT CONTROLLER:

8259 is a programmable interrupt controller [2], it has an capacity to increase interrupt handling capability. It combines the multi-interrupt input source into a single interrupt output. It can be either edge triggered or level triggered interrupt level. It provides 8 interrupt inputs from IR0 to IR7. It can be increased to 64 – interrupt handling capability by cascading using Cascade buffer. Clock cycle is not required. Data Buffer is a mediator between microprocessor and 8259. It takes the control from microprocessor to control logic. Read / Write works when pin CS is low.

Control Logic is the center of the microprocessor and controls every functionality of the block. INTR
takes the interrupt requests and INT pin gives the output. Interrupt Request Register stores all the interrupt interrupt levels. IRR stores the interrupt level which are currently being executed. IMR stores the bits to be masked for the interrupt levels. Priority Solver is used to set priority for the interrupt levels and resets the Interrupt level which is serviced.

1.2.2. TIMER:
Intel 8253 programmable Counter / Timer is a 24-pin package IC [3]. It operates in 4 – Programmable timer modes and it has 3 counters. Timer control byte is a used to control the operation of timer module. Bits 6,7 are the counter selection bits. Bits 5,4 is used to select the read / write mode for 2 – byte counter. Bits 3,2,1 are used to select the count mode. Mode 0 is a Interrupt on terminal count, it starts with initial count and count down to zero. OUT value is made 0 then count value is loaded and starts count down after 1 clock.

The previous count value is stopped by the new value. Mode 1 is Programmable one shot, initially OUT is high. OUT goes to 0 in the clock following the trigger to begin mode 1 count and remain 0 till count value is 0. OUT goes to 1 and will stay till the trigger of next clock. Gate value has null on count. Latest count value doesn’t affect the previous count. Mode 2 is Rate generator, behaves as divided by m-counter and used to get a real time clock interrupt. The previous mode by the OUT is set low when the counter reaches 1 and the following cycle counter is reloaded. Mode 4 makes the timing of 1 and 0 clock pulse of output is 1 for n by 2 counts and 0 for n by 2 counts. Period is n counts, if n=odd additional cycle is spent on the OUT value 1.

1.2.3. DMA CONTROLLER:
Direct Memory Access (8257) [4] is used to transfer data for Intel Microprocessor. Based on the peripheral request, it is used to read / write in to / from the memory and peripheral The DMA Controller Simulation Waveform.

At first the system bus control is acquired by the DMA. Then the requesting peripheral is acknowledged by the DMA. Outputs the Least Significant Bits to A0-A7 and Most Significant Bits to A8-A15. Generates the required memory and Input / Output read or write signal for data transfer. The control signals are sent to CPU that the operation is completed.

DMA channel has two 16-bit register namely DMA address register and Terminal count register. They must be initialized before the terminal count are enabled. The lower 14- bits are TC register is used to specify the number of DMA cycle – 1 and the MSB 2-bits specify the DMA operations. DRQ0 – DRQ3 are individual channel requests input which makes the peripherals acquire DMA cycle. Data Request 0 has the more priority and Data Request 3 has the less priority. The line to be request is held until DMA acknowledgement. DACK0 – DACK3 shows the peripheral connected to the channel. Data bus’s register is a bi-directional three state line and it is programmable by CPU for DMA address and TC register. The Various operating modes are shown in Table 1.

| BIT 15 | BIT 14 | DATA Operation |
|--------|--------|----------------|
| 0      | 0      | Verify         |
| 0      | 1      | Write          |
| 1      | 0      | Read           |
| 1      | 1      | Illegal        |

The Various bits in the MSR activates each channels and allow 4 different options. It is programmed by CPU after DMA address and TC register.

Table 2: Mode Set Register
Bits 0 – 3 is used to enable the DMA channels. Bit-4 is the rotating priority bit which is used to set priority to the channel in circular sequence. Channel0 and Channel3. Bit-5 is the extend write bit is used to extend the duration of Memory Write and Input / Output Write earlier in DMA cycle. Bit-6 is TC stop bit is used to disable the channel. Bit-7 is Auto load is used to block chain operation without software intervention which reference to Table 2. The Status register is used to indicate which channel have achieved a TC state and update flag which is shown in Table 3.

| AUTOLOAD | TC Stop | Extended Write | Rotating Priority |
|----------|---------|----------------|-------------------|
| 7        | 6       | 5              | 4                 |

1.2.4. EXECUTION UNIT AND BUS INTERFACE UNIT: EU and BIU [5] are two important functional units of a microprocessor. The Execution Unit feeds instructions and does operations on the data via Bus Interface Unit. The instruction decoder and the ALU control operations on the data. All units are coordinated via the Control unit. The instruction decoder translates the instruction fetched into a series of operations carried out by Execution unit. Results of the latest arithmetic operations are represented by conditional flags and the operations of the execution unit are controlled by control flag. The function of Bus Interface unit is to handle the data transfer and address transfer on the buses for the Execution unit. Data address and instructions address are stored in Segment register’s memory. The address of the next instruction is held in the Instruction Pointer, which is a 16-bit register. Other general purpose registers and pointers are also available.

2. METHODOLOGY
2.1. INTERRUPT CONTROLLER
Testbench:
Constraint Interrupt( I dist(1:=25, 2:=25, 4:=25, 8:=25);)
Constraint CS (cs dist(1:=75, 0:=25);)
Constraint d(din == 8'b11111111;)
Design:
READ CS, I, din
state ← defaultvalue
ALWAY S@posedgeclock
IF"CS == 1"then
CASEstateof
3'b000 :
IRR ← InterruptRequest
IF"sizeofIRR == Threshold" state ← PrioritySolverstate
3'b001 :
FORI 2 IRR
HigherPriorityInterruptisselected
ENDFOR
ENDIF
3'b010 :
Interrupt Controller mode ← master
Interruptsignal ← high
Opcode ← systembus processor
IF"IACK"then
InterruptController mode ← slave
Interruptsignal ← low
State ← prioritystate
ENDIF
IF"CS"and"WR"then
IMRvalue ← din[4 : 3]; din[1 : 0]
ENDIF
IF"j == 0"then
State ← defaultvalue
ENDIF
ENDCASE
ENDALWAY
2.2. TIMER MODULE
Testbench:
Constraint data range (data dist ([1:5]:=0,[6:9]:1));
Constraint wr range (wr dist (1:=50, 0:=50));
Constraint rd range (if(wr==1) rd==0; else rd==1);
Design:
READ cs, data, rd, wr
ALWAYS @ posedge clk
FOR i ∈ mode
IF"mode == n"then
IF"out == 1"and"wr == 1"then
IF"rdldmode == 2'b11"then
ctr[15 : 0] ←data
out1 ←withctr
out ←reset
ENDELSE
ELSEIF"rdldmode == 2'b01" then
ctr[7 : 0] ←data
out1 ←withctr
out ←reset
ENDELSE
ELSEIF"rdldmode == 2'b10" then
ctr[15 : 8] ←data
out1 ←ctr
out ← reset
ENDELSE
ELSEIF"out == 0"and"gate == 1"and"data1 == data"
DECREMENTctr
IF"rd == 1"and"rdldmode == 20b01" then
out1 ←ctr
ELSEIF"rd == 1"and"rdldmode == 2'b11" then
out1 ←ctr
ENDELSE
IF"ctr == 8'b0"then
outisset
ELSEIF"data1! = data"and"out == 0"and"wr == 1"
ctr[7 : 0] ←data
out1 ←ctr
out ←reset
ENDFOR
ENDFOR
ELSEIF"a == 2'b11"
state ←initialstate
ctrlreg ←ctrlword
ctrsel ←counter
dldmode ←mode
mode ←countingmode
out ←set

2.3. EXECUTION UNIT
BUS INTERFACE UNIT
Testbench:
Constraint data range (data dist ([1:5]:=0,[6:9]:1));
Constraint opcode range (opcode dist ([1:6]:/5));
Constraint addr range (addrdist ([1:1024]:=50,[1025:4096]:=50));
Design:
INPUT data, address, opcode
ALWAYS @ posedge clock
IF"reset==1" then
State ←2'b10
CASEstateof
2'b00 : IF"ctrl == 1" then
cs ←addressofcode
diptr ←codeinthememory
ds ←startingaddressofdata
siptr ←datainthememory
mem[diptr + +] ←opcode
diptr ←cs
r ←1
instruqueue ←
instructioninthememory
mem[siptr + +] ←data
State ←2'b01
cy ←cleared
ENDIF
2'b01 :
instr ←nextinstruction
ax ←datainthememory
temp ←cleared
cy ←cleared
dx ←mem[address]
Caseinstruof
3'b000 :
CALCULATESUM
CALCULATEPRODUCT
CALCULATEDIFFERENCE
3'b001 :
CALCULATEGUOTIENT
CALCULATERemainder
3'b100 :
ax OR dx
3'b101 :
ax AND dx
3'b110 :
NOT ax
2'b10 :
2.4. DMA Controller

PSEUDOCODE

Design:
ALWAYS @ posedge clk
CASE state of
3'b000:
IF 'reset == 1" then
CASE state0 of
2'b00:
mode set reg ← data
state0 ← nextstate
2'b01:
statusreg ← data
state0 ← initialstate
state ← nextstate
memory[x][y] ← cleared
ENDCASE
3'b001:
CASE state of
2'b00:
dmaaddreg ← data
data1 ← dmaaddreg
state1 ← nextstate
2'b01:
tcreg ← data
DMAcycle ← tcreg
state1 ← initialstate
state ← 3'b101
data1 ← tcreg
DMAcycle ← tcreg[13 : 0]
Operation ← tcreg[15 : 14]
state1 ← initialstate
state ← 3'b101
state4 ← initialstate
3'b101:

3. RESULTS AND DISCUSSION

This section presents the simulation results, Coverage report of each modules and its inference. The Figure 3 is the simulation waveform of Interrupt Controller. The interrupts are given in the order 1, 2, 4, 8 and they are executed in the order specified by Priority solver as 8, 2, 4, 1. Once the Interrupt is sent an opcode is generated indicating the execution of a particular interrupt and it is sent to the CPU. The value of the IMR is taken from datain value and only when all bits are 1 sent as input, the correct interrupt value is sent as output on read signal. In order to generate the output on read signal IMR value is with Interrupt value.

The Table 4 shows the coverage report of Interrupt Controller. The Coverage is 100 percentage all the features of the specification are checked.
3.2. 8253-Timer Module
The Figure 4 shows the Time Module Simulation Waveform of Mode 0. It starts with initial count value load and count down to 0. Out value is set 0 after count value is loaded and starts counting down after one clock cycle. Out stays 0 till the counter reaches zero. Out value is 1 till reloading the counter value. For normal count process Gate value is set 1 and counting stops when 0 and starts when 1. New count value stops the previous count. The Figure 5 shows the Time Module Simulation Waveform of Mode 1. Initially out is high. Out value goes to 0 the trigger after clock to begin mode 1 and will stay 0 until count value changes to 0. The out value is 1 and stay till next clock. Gate value has no effect on count.

![Figure 3: 8259-Interrupt Controller UVM Simulation Waveform](image)

Table 4 – 8259 Interrupt Controller Coverage Report

| Covergroup                  | Hits | Goal | Status  |
|-----------------------------|------|------|---------|
| Package interrupt_controller.bg | 100.00 | 100.00 | Covered |
| Instance                    | 100.00 | 100.00 | Covered |
| Coverpoint trans.allocated₁ | 100.00 | 100.00 | Covered |
| bin cs₀                     | 12    | 1    | Covered |
| bin cs₁                     | 5     | 1    | Covered |
| Coverpoint trans.allocated₂ | 100.00 | 100.00 | Covered |
| bin WR₀                     | 9     | 1    | Covered |
| bin WR₁                     | 8     | 1    | Covered |
| Coverpoint trans.allocated₃ | 100.00 | 100.00 | Covered |
| bin RD₀                     | 14    | 1    | Covered |
| bin RD₁                     | 3     | 1    | Covered |
| Coverpoint trans.allocated₄ | 100.00 | 100.00 | Covered |
| bin I₀                      | 4     | 1    | Covered |
| Coverpoint trans.allocated₅ | 100.00 | 100.00 | Covered |
| bin I₁                      | 3     | 1    | Covered |
| Coverpoint trans.allocated₆ | 100.00 | 100.00 | Covered |
| bin I₂                      | 2     | 1    | Covered |
| Coverpoint trans.allocated₇ | 100.00 | 100.00 | Covered |
| bin I₃                      | 1     | 1    | Covered |


###Figure 4: 8253-Timer UVM Simulation Waveform Mode 0

###Figure 5: 8253-Timer UVM Simulation Waveform Mode 1

New count value doesn’t effect the previous count.

###Figure 6: 8253-Timer UVM Simulation Waveform Mode 2

The Figure 6 shows the Time Module Simulation Waveform of Mode 2. Out is set high initially and when count value is loaded it is set low. For normal count process Gate value is set 1 and count suspends when 0 then resumes when high. The counter is loaded with the initial count after the

###Table 5 — 8253 Timer Module Coverage Report

| Covergroup                  | Hits    | Goal  | Status  |
|-----------------------------|---------|-------|---------|
| Package timer_pkg           | 83.33   | 100.00| Uncovered|
| Instance                    | 83.33   | 100.00| Uncovered|
| Coverpoint trans_collected_s| 100.00  | 100.00| Covered |
| bin c00                     | 15.00   | 1.00  | Covered |
| bin c01                     | 1.00    | 1.00  | Covered |
| Coverpoint trans_collected1D| 100.00  | 100.00| Covered |
| bin RD0                     | 10.00   | 1.00  | Covered |
| bin RD1                     | 6.00    | 1.00  | Covered |
| Coverpoint trans_collectedwR| 100.00  | 100.00| Covered |
| bin WR0                     | 10.00   | 1.00  | Covered |
| bin WR1                     | 6.00    | 1.00  | Covered |
| Coverpoint trans_collecteds | 50.00   | 100.00| Uncovered|
| bin a0                      | 4.00    | 1.00  | Covered |
| bin a1                      | 0.00    | 1.00  | Covered |
| bin a2                      | 0.00    | 1.00  | Covered |
| bin a3                      | 12.00   | 1.00  | Covered |
| Coverpoint trans_collectede| 100.00  | 100.00| Covered |
| bin gate0                   | 14.00   | 1.00  | Covered |
| bin gate1                   | 2.00    | 1.00  | Covered |
| Coverpoint trans_collecteddata| 50.00  | 100.00| Uncovered|
| bin data0                   | 2.00    | 1.00  | Covered |
| bin data1                   | 0.00    | 1.00  | Uncovered|
completion of the current count. It is repeated.

![Figure 7: 8253-Timer UVM Simulation Waveform Mode 3](image)

The Figure 7 shows the Time Module Simulation Waveform of Mode 3. Out is set high initially and when the counter is loaded it is set low and for normal count process Gate value is set 1 and count suspends when 0 then resumes when high. If the count value is even number, it is decremented by 2.

If the count value is odd number, it is decremented by 1 on first clock and remaining clock by 2. If new count value is given as input in the event of present counting, its value is stored and it is used as counter value after the completion of the present counter.

The Table 5 shows the Coverage report of the Timer module. The Coverage is 83.333 not all features are checked as Timer 2 and 3 features are not checked.

### 3.3. 8257-DMA Controller

The Figure 8 shows the DMA Controller Simulation Waveform. At first the system bus control is acquired by the DMA. Then the requesting peripheral is acknowledged by the DMA. Outputs the Least Significant Bits onto A0-A7 and Most Significant Bits onto A8-A15. Gets the required memory and Input / Output read or write signal for data transfer. It retains the control until the DMA request exists. The transfer of data is by single burst method. The control signals are sent to CPU that the operation is completed.

The Table 6 shows the Coverage report of the DMA Controller. The Coverage is 91.666 not all features are checked.

### 3.4. Execution Unit and Bus Interface Unit

The Figure 9 is the simulation waveform of the EU and the BIU. At first the data is loaded into the memory. Then the instruction is loaded into the instruction queue of size 6 Bytes. The instruction is fetched one by one and the corresponding data is loaded into the register from the memory. The operation is performed and the answer is kept in the Accumulator. Later data’s are moved into memory.
Table 6 – DMA Controller Coverage Report

| Covergroup                  | Hits | Goal | Status  |
|----------------------------|------|------|---------|
| Package DMA pkg            | 91.666 | 100.00 | Uncovered |
| Instance                   | 91.666 | 100.00 | Uncovered |
| Coverpoint transcollected a1| 25.00 | 100.00 | Covered  |
| bin a1 0                   | 19   | 1    | Covered  |
| bin a1 1                   | 0    | 1    | Covered  |
| bin a1 2                   | 0    | 1    | Covered  |
| bin a1 3                   | 0    | 1    | Covered  |
| Coverpoint transcollected data | 100.00 | 100.00 | Uncovered |
| bin data 0                 | 12   | 1    | Covered  |
| bin data 1                 | 4    | 1    | Covered  |
| bin data 2                 | 2    | 1    | Covered  |
| bin data 3                 | 1    | 1    | Uncovered|
| Coverpoint transcollected cs | 100.00 | 100.00 | Covered  |
| bin cs 0                   | 17   | 1    | Covered  |
| Coverpoint transcollected ior | 100.00 | 100.00 | Covered  |
| bin ior 0                  | 17   | 1    | Covered  |
| Coverpoint transcollected isw | 100.00 | 100.00 | Covere  |
| bin isw 0                  | 4    | 1    | Covered  |
| Coverpoint transcollected ready | 100.00 | 100.00 | Uncovered|
| bin ready 0                | 9    | 1    | Covered  |
| bin ready 1                | 189  | 1    | Uncovered|
| Coverpoint transcollected hlda | 100.00 | 100.00 | Covered  |
| bin hlda 0                 | 16   | 1    | Covered  |
| Coverpoint transcollected memrw | 100.00 | 100.00 | Uncovered|
| bin memrw 0                | 17   | 1    | Covered  |
| Coverpoint transcollected memuw | 100.00 | 100.00 | Uncovered|
| bin memuw 0                | 9    | 1    | Uncovered|
| bin memuw 1                | 109  | 1    | Uncovered|

Table 7 – Execution and Bus Interface Coverage Report

| Covergroup                  | Hits | Goal | Status  |
|----------------------------|------|------|---------|
| Package EU BUI pkg         | 66.666 | 100.00 | Uncovered |
| Instance                   | 66.666 | 100.00 | Uncovered |
| Coverpoint transcollected opcode | 100.00 | 100.00 | Covered  |
| bin opcode 1               | 3    | 1    | Covered  |
| bin opcode 2               | 1    | 1    | Covered  |
| bin opcode 3               | 1    | 1    | Covered  |
| bin opcode 4               | 1    | 1    | Covered  |
| Coverpoint transcollected data | 50.00 | 100.00 | Covered  |
| bin data 0                 | 2    | 1    | Covered  |
| bin data 1                 | 23   | 1    | Covered  |
| bin data 2                 | 0    | 1    | Uncovered|
| bin data 3                 | 0    | 1    | Covered  |
| Coverpoint transcollected address | 50.00 | 100.00 | Covered  |
| bin address 0              | 1    | 1    | Covered  |
| bin address 1              | 0    | 1    | Covered  |
| bin address 2              | 0    | 1    | Covered  |
| bin address 3              | 24   | 1    | Covered  |

Figure 8: 8257-DMA Controller UVM Simulation Waveform
The Table 7 shows the Coverage report of the Execution and Bus Interface Unit. The Coverage is 66.666% not all features are checked.

**Design and Verification.** The modules of 80186 Processor is implemented using EDAplayground. For each module design, Verilog was used for coding and Edaplayground is used for simulation purposes. For module verification, UVM testbenches are built to generate Constraint Randomized Input Vectors. The Simulation waveform and Coverage reports were used to verify the functionality of the design.

The maximum operating frequency of the implemented design is 16 MHz.

**Utilization**

Xilinx Vivado 14.0 is used for synthesis and Basys 3 board is used. The Utilization Report consists of Slice LUTs, Slice Registers and Bonded IOBs. FPGA resources are grouped in Slices to create configurable logic blocks. Each slice consists of a set number of LUTs, flipflops and multiplexers. From Table 8, Each LUT has a collection of logic gates wired into the FPGA. Flipflops are used to store a single bit value and can be placed anywhere between the LUTS. Multiplexers are used as signal selection blocks.

| Resource | Utilization | Available |
|----------|-------------|-----------|
| LUT      | 569         | 20800     |
| FF       | 359         | 41600     |
| I/O      | 94          | 106       |

In Basys3 FPGA there are 5200 Slices and 33280 logic cells. Each slice has 4 6-input LUTs and 8 flipflops. Input Output Buffer calculated with respect to the number of input and output pins of the design. The number of CLBs for Timer Module is 49. The number of CLBs for DMA Controller is 41. The number of CLBs for Interrupt Controller is 54.

**4. CONCLUSION**

The Universal Verification Methodology is a standardized methodology for verifying integrated circuit designs. Simulation waveform verifies the functionality of the design. The Coverage report checks whether all the features of the specification are checked. The use of Bloom filter helps to validate correctness of the instruction executed by the processor and prevents the system to crash.

**REFERENCES**

[1] Bahadure, Nilesh B. Microprocessors: The 8086/8088, 80186/80286, 80386/80486 and the Pentium Family. PHI Learning Pvt. Ltd., 2010.

[2] Culley, Paul R., et al. “Programmable interrupt controller.” U.S. Patent No. 5,101,497. 31 Mar. 1992.

[3] Berney, Rex, and John Nowakowski. “Computer-Controlled Precise Square Wave Generation...
Using the 8254 Programmable Interval Timer.” *APS Meeting Abstracts.* 1996.

[4] Koti, Jhansirani, and G. Mani Kumar. “Cost Effective and High Speed Design of DMA Memory Copy Accelerator.”

[5] Brey, Barry B. *The Intel microprocessors: architecture, programming, and interfacing.* Prentice-Hall, 1997.

[6] Fiergolski, Adrian. ”Simulation environment based on the Universal Verification Methodology.” *Journal of Instrumentation* 12.01 (2017): C01001.

[7] D. Malandain, P. Palmen, M. Taylor, M. Aharoni, Y. Arbetman and Y. Arbetman, “An effective and flexible approach to functional verification of processor families,” Seventh IEEE International High-Level Design Validation and Test Workshop, 2002., Cannes, France, 2002, pp. 93-98, doi: 10.1109/HLDVT.2002.1224435.

[8] S. El-Ashry, M. Khamis, H. Ibrahim, A. Shalaby, M. Abdel-salam and M. W. El-Kharsi, “On Error Injection for NoC Platforms: A UVM-Based Generic Verification Environment,” in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 39, no. 5, pp. 1137-1150, May 2020, doi: 10.1109/TCAD.2019.2908921.

[9] A. Bolat, L. Cassano, P. Reviriego, O. Ergin and M. Ottavi, “A Microprocessor Protection Architecture against Hardware Trojans in Memories,” 2020 15th Design Technology of Integrated Systems in Nanoscale Era (DTIS), Marrakech, Morocco, 2020, pp. 1-6, doi: 10.1109/DTIS48698.2020.9080961.

[10] M. Atamaner, O. Ergin, M. Ottavi and P. Reviriego, “Detecting errors in instructions with bloom filters,” 2017 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT), Cambridge, 2017, pp. 1-4, doi: 10.1109/DFT.2017.8244458.

[11] D. M. Reddy, K. P. Akshay, R. Girdar, S. D. Karan and N. Mohankumar, “BHARKS: Built-in hardware authentication using random key sequence,” 2017 4th International Conference on Signal Processing, Computing and Control (ISPCC), Solan, India, 2017, pp. 200-204, doi: 10.1109/ISPCC.2017.8269675.

[12] Shankar, Anuradha, Srinivas Manchala, and N. Mohankumar. “Low power implementation of DTMF chip by using power gating technique with merge flops.” *Journal of Engineering and Applied Sciences* 13 (2018): 6335-6340.

[13] Kotha, Srivasthav, Rajin Ravimony, and N. Mohankumar. “Automated UVM Based Verification of Device Life Cycle Management IP.” International Conference on Intelligent Computing, Information and Control Systems. Springer, Cham, 2019.

[14] S. Karlapalem and S. Venugopala, “Scalable, Constrained Random Software Driven Verification,” 2016 17th International Workshop on Microprocessor and SOC Test and Verification (MTV), Austin, TX, 2016, pp. 71-76, doi: 10.1109/MITV.2016.19.

[15] Khalifa, Khaled, and Khaled Salah. “An RTL power optimization technique based on System Verilog assertions.” 2016 IEEE 7th Annual Ubiquitous Computing, Electronics Mobile Communication Conference (UEMCON). IEEE, 2016.

[16] Balamurugan, Karthigha, M. Nirmala Devi, and M. Jayakumar. “Design of V-band low noise amplifier using current reuse topologies.” *International Journal of Applied Engineering Research* 9.24 (2014): 27319-27330.