Design and Verification of USB 3.0 Link Layer (LTSSM)

Rohit Kumar
School of information Sciences
Manipal University
Manipal, India

Hardik Trivedi
School of information Sciences
Manipal University
Manipal, India

Nitish Alok
School of information Sciences
Manipal University
Manipal, India

Abstract-In this proposed design it mainly includes USB 3.0, LTSSM. The Link Training and Status State Machine (LTSSM) have downstream and upstream ports. Transitions of all 12 link states and their subs -states of both downstream and upstream have been designed. The proposed model is implemented using Verilog HDL. Proposed model in this paper has been verified using SystemVerilog.

1. INTRODUCTION

USB is an industry standard developed in the mid-1990s, it defines the cables, connectors and protocols used in a bus for connection and communication between computers and electronic devices. The first USB technology began development in 1994, co-invented by Ajay Bhatt of Intel and the USB-IF (USB Implementers Forum, Inc).

Before USB came into existence, computers used serial and parallel ports to plug devices into computers and transfer data. Expansion cards and custom drivers were often required to connect the devices. Parallel ports transferred data at approximately 100 kilobytes per second, where as serial ports ranged from 115 to more than 450 kilobits per second. The high extent of incompatibilities and the attempt to use multiple interfaces helped in the growth for a technology like USB. Immediate interaction between devices and a host computer without the need to disconnect or restart the computer also enables USB technology to furnish more efficient operation. Consequently, a single USB port can handle up to 127 devices while offering a collective compatibility.

Versions of USB specification:

- Revision 1.0 released on January 15, 1996, introduced a low-speed transfer rate of 1.5 Mbit/s and a full-speed transfer rate of 12 Mbit/s.
- Revision 1.1 released on September 23, 1998, introduced the improved specification and was the first widely used version of USB.
- Revision 2.0 released on April 27, 2000. The major feature of revision 2.0 was the addition of a high-speed transfer rate of 480 Mbit/s.
- Revision 3.0 released on November 17, 2008, brings significant performance enhancements to the USB standard while offering backward compatibility with the peripheral devices currently in use. Delivering data transfer rates up to ten times faster (the raw throughput is up to 5.0 Gbit/s) than Hi-Speed USB (USB 2.0).

2. USB 3.0 LINK LAYER

Link Layer helps in traffic management between the two links connected to each other. It manages the port to port flow of data between the host and the device.

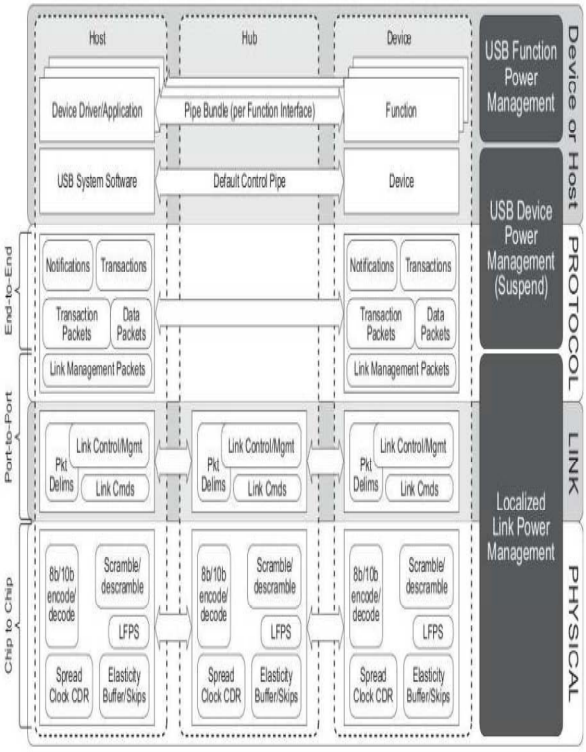

Fig 1: USB Layout Diagram [1]

The Link Layer functions consist of:

- Effective power management for 4 link power state (U0, U1, U2, and U3).
- Packet and link command formation.
- Link training symbol lock and Rx-equalization.
- Packet header formation.
- Different types of error handling.

3. USB 3.0 LTSSM

LTSSM (Link Training and Status State Machine) is the state machine used for link connectivity and link power management. LTSSM consist of 12 main link states and their sub-states. These states and sub-states are responsible for link training, power management and error testing.

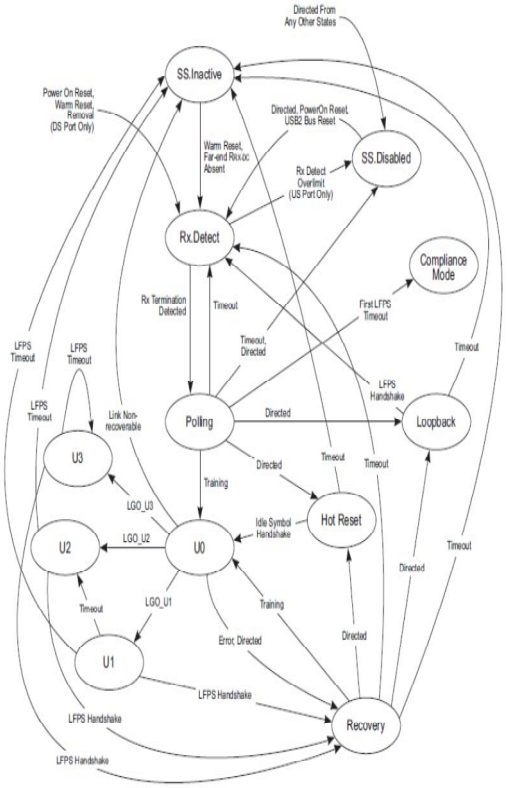

Fig 2: State diagram of LTSSM [1]

4. USB 3.0 LTSSM STATE DESIGN

The proposed USB3.0 LTSSM architecture is consisting of following states and their sub-states:

4.1 SS.Disabled

It is a state where a port's SuperSpeed connectivity is disabled. It does not contain any sub-states in case of downstream port and hub upstream port but contain two sub-states for peripheral upstream port.

- SS.Disable.Default
- SS.Disable.Error

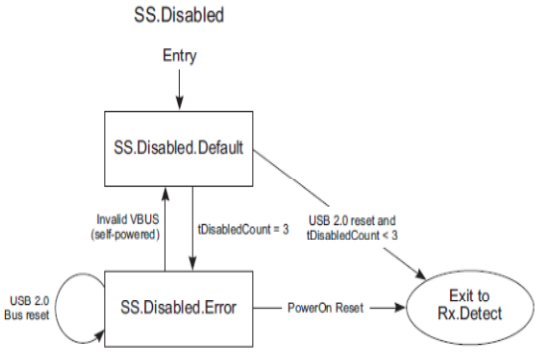

Fig 3: SS.Disabled Sub-state Machine [1]

4.2 SS.Inactive

It is a state where a link has failed SuperSpeed operation and USB is non-operable. It contains two sub-states:

- SS.Inactive.Disconnect.Detect
- SS.Inactive.Quiet

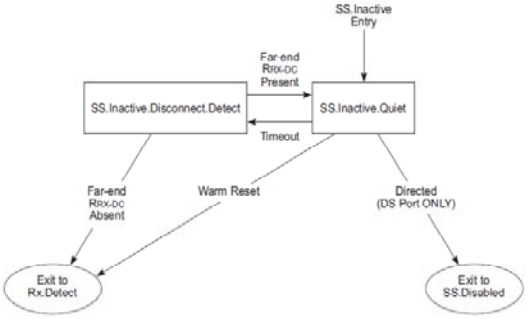

Fig 4: SS.Inactive Sub-state Machine [1]

4.3 Rx.Detect

It is the initial state after reset, called as power on state of LTSSM for both downstream port and upstream port. It detects the presence or absence of a device connected at far end of the link. It contains three sub-states:

- Rx.Detect.Reset
- Rx.Detect.Active
- Rx.Detect.Quiet

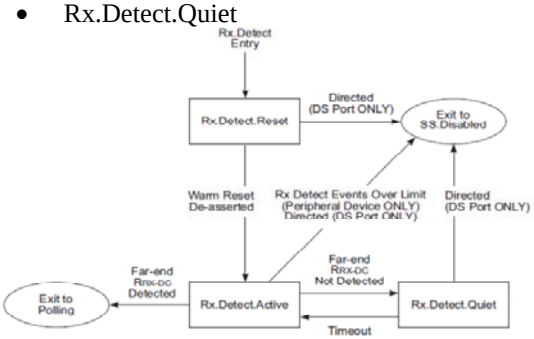

Fig 5: Rx.Detect Sub-state Machine [1]

4.4 Polling

It is a state where link training starts. It contains five sub-states:

- Polling.LFPS
- Polling.RxEQ
- Polling.Active
- Polling.Configuration
- Polling.Idle

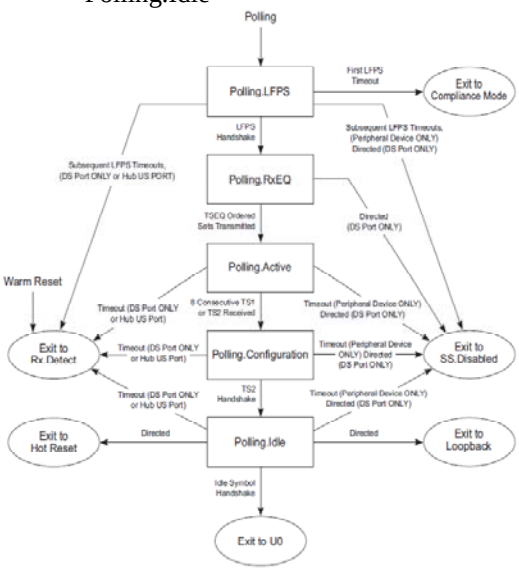

Fig 6: Polling Sub-state Machine [1]

4.5 Compliance Mode

It is used to test the transmitter as per given voltage and timing specification. It does not contain any sub-state.

4.6 U0

It is normal power state and does not conation any sub-state.

4.7 U1

It is low power state where no transmission of packets is done.

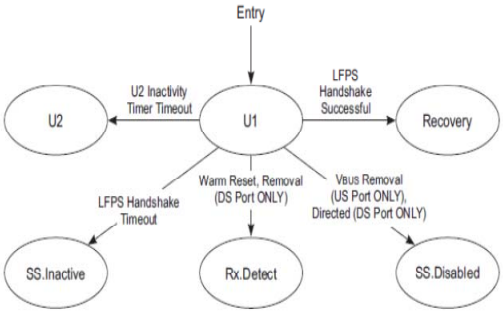

Fig 7: Transition to other state from U1 [1]

4.8 U2

It is even low power state than U1 but exit latency increased in this state.

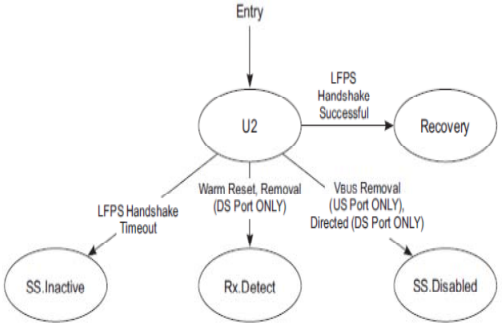

Fig 8: Transition to other state from U2 [1]

4.9 U3

It is lowest power state, where device is put into a suspend state.

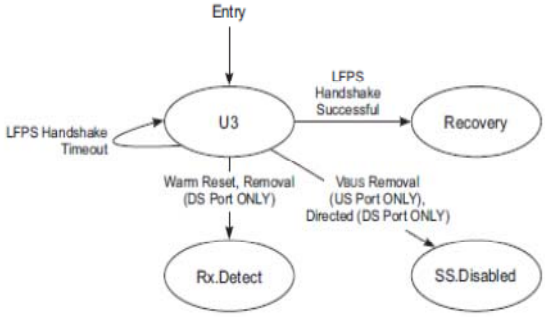

Fig 9: Transition to other state from U3 [1]

4.10 Recovery

Retraining of the link is done in this state. It contains three sub-states:

- Recovery.Active
- Recovery.Configuration
- Recovery.Idle

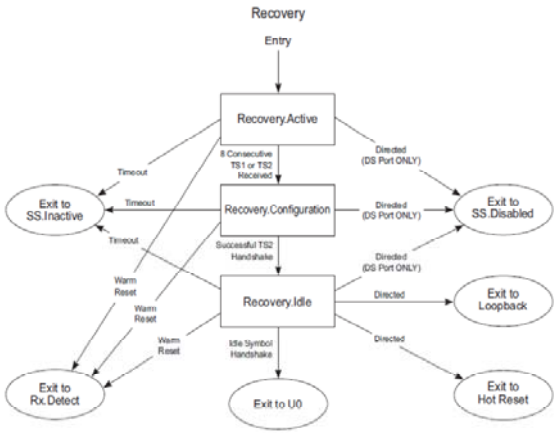

Fig 10: Recovery Sub-state machine [1]

4.11 Loopback

It is used as test and fault isolation state. It contains two sub-states:

- Loopback.Active
- Loopback.Exit

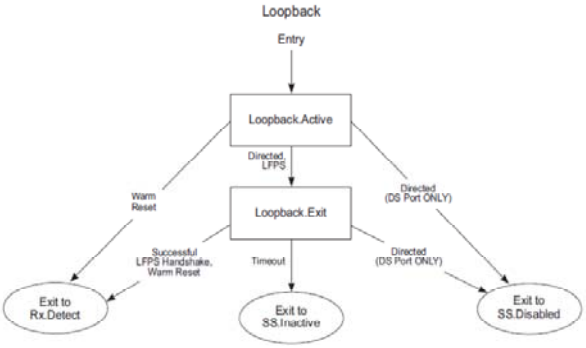

Fig 11: Loopback Sub-state machine [1]

4.12 Hot Reset

This state is entered when it is directed to do so by a device's higher layer. It contains two sub-states:

- Hot Reset.Active
- Hot Reset.Exit

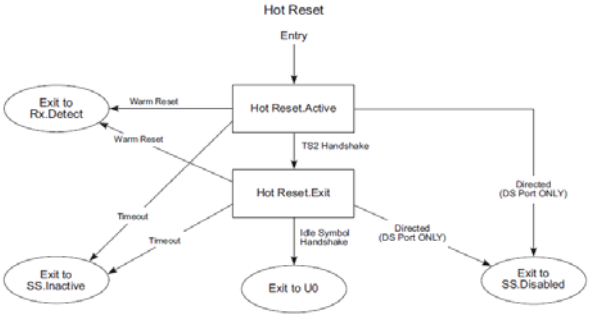

Fig 12: Hot Reset Sub-state machine [1]

5. VERIFICATION

The verification of above LTSSM design for Upstream port is carried out to check that if all the states and sub-states are covered or not. There are 18 different signal which drives LTSSM. All 18 different signals have been given as input to design and respected state has been monitored. These states have been assigned as output. Verification is done using SystemVerilog.

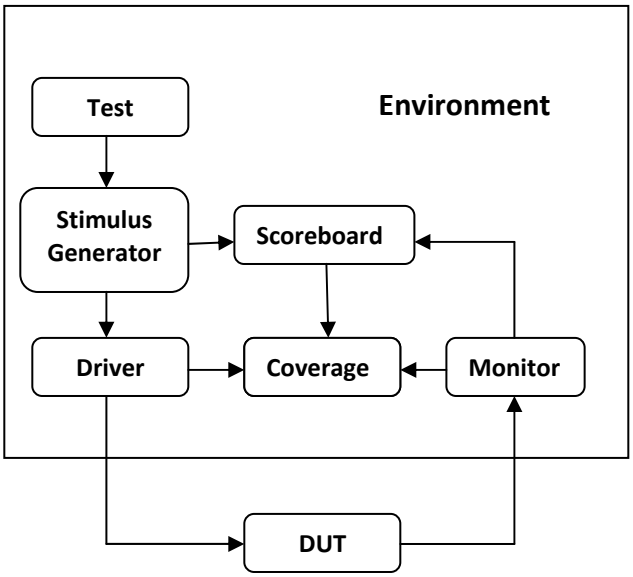

Fig 13: Test Bench Architecture

5.1 Interface

It consists of bundle of signals which can be referenced throughout a design to simplify hierarchical connections and module instantiation i.e. used to connect the testbench to the DUT.

5.2 Stimulus

Using SystemVerilog randomization, stimulus is generated automatically. SystemVerilog high-level data structures helps in storing and processing of stimulus in an efficient way.

5.3 Stimulus Generator

It generates stimulus which are sent to DUT by driver. Randomization of stimulus has been done in this block. Automatic randomization didn't cover all the states of LTSSM, so some directed random testcases is written here. Mailbox sg2dr is created to send the generated stimulus to the driver block.

5.4 Driver

It repeatedly receives a data items from mailbox sg2dr and drives it to DUT by sampling and driving the DUT signals. The driver also sends the stimulus to scoreboard using drv2sb mailbox.

5.5 Monitor

This block receives the output of the DUT i.e. output of LTSSM design. The output is sent to the scoreboard using mailbox rcv2sb.

5.6 Scoreboard

This block receives the stimulus from the driver through the mailbox drv2sb and also receives the DUT output from monitor through the mailbox rcv2sb. In LTSSM design, for different stimulus there will be different states, so in scoreboard expected state for all the combination of stimulus has been written and compared with DUT output receives from mailbox rcv2sb. If it matches then LTSSM design is working correctly.

5.7 Environment

It contains the instances of the entire verification component

Methods defined in Environment class:-

- build(): all the objects like driver, monitor etc are constructed.
- reset(): reset the DUT.
- start(): To call the methods which are declared in the other components like driver and monitor.
- run(): This method calls all the above declared methods in a sequence order.
- Report(): Its main function is to detect the errors in the design and report the errors.

.

6. SIMULATION RESULT

The output Link_state changes with change in the input signal.

- LTSSM design for Downstream port

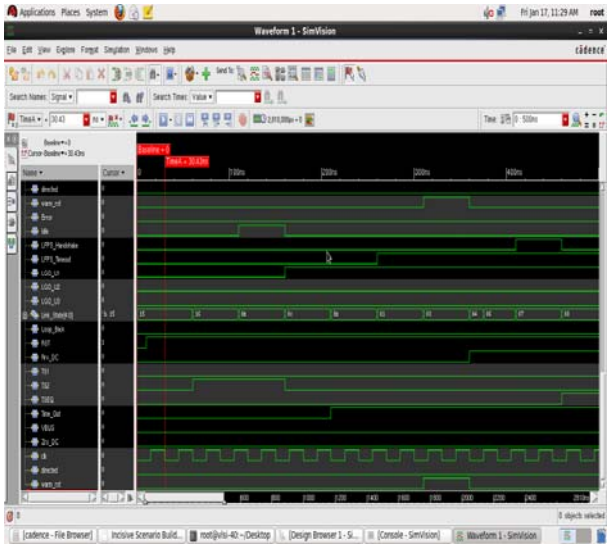

Fig 14: Downstream port simulation

- LTSSM design for Upstream port

LTSSM design for upstream port consists of SS.Disabled sub-states.

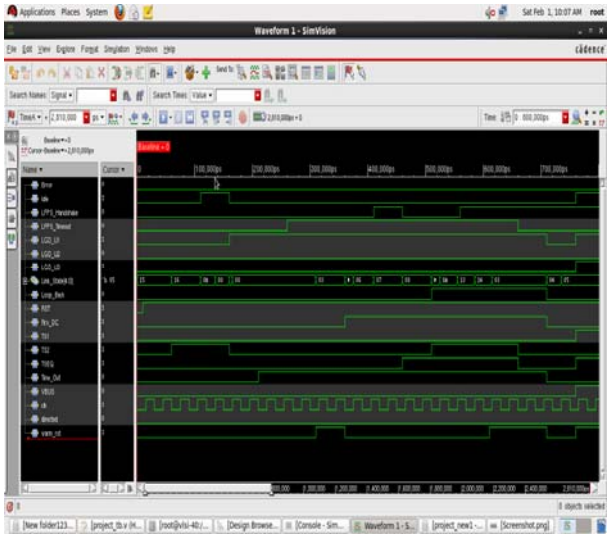

Fig 15: Upstream port simulation

7. VERIFICATION RESULT

7.1 Scoreboard

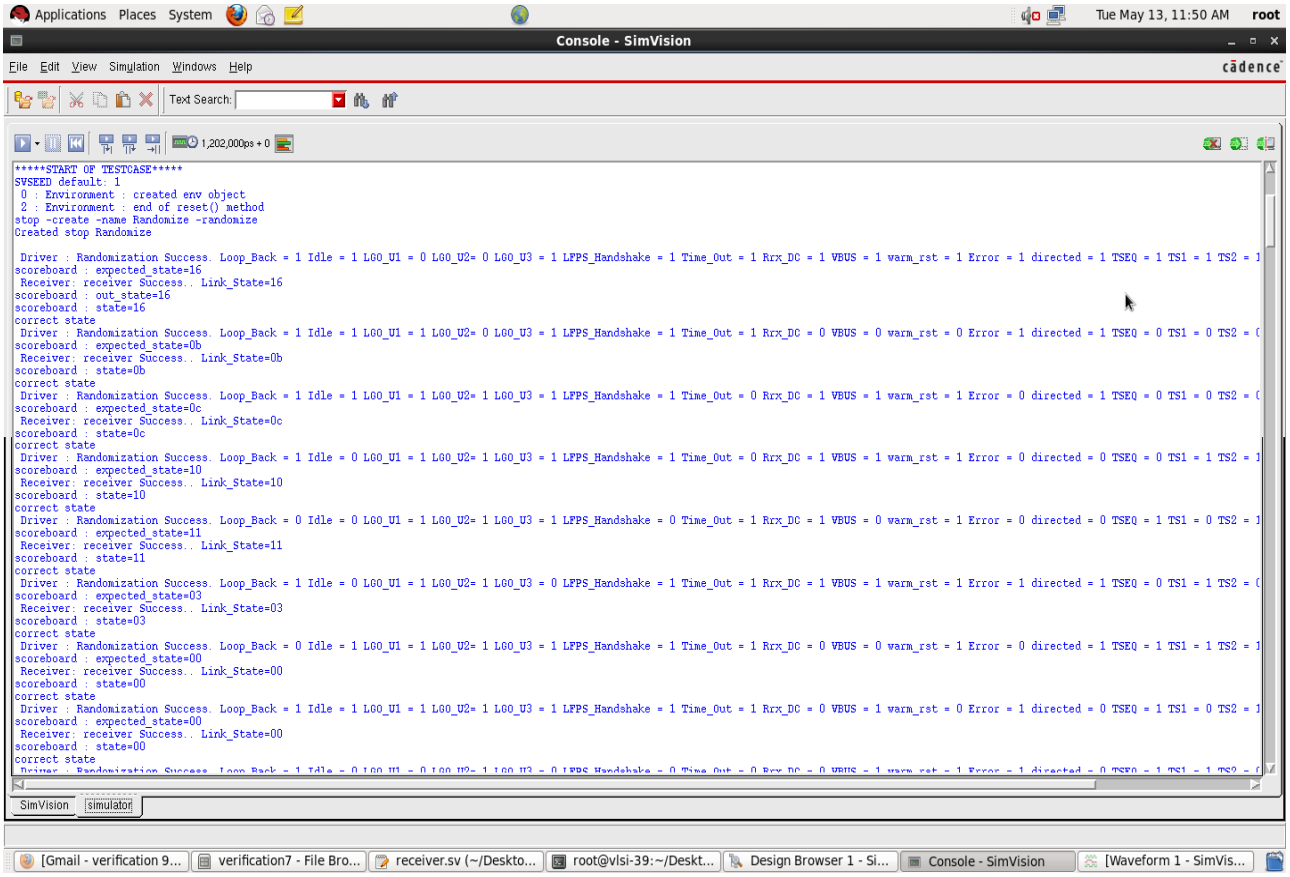

Fig 16: Scoreboard

7.2 Coverage Report

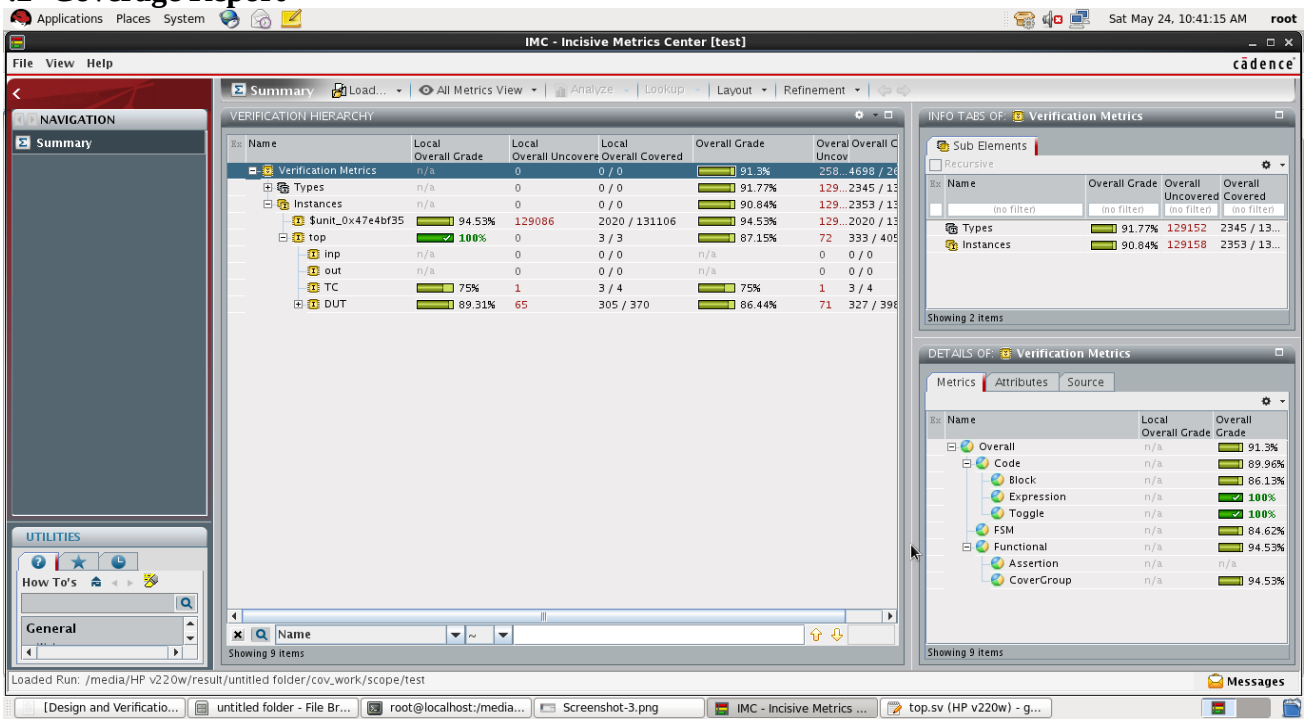

Fig 17: Coverage result

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS
Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS
Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS
Sync your system to PACER to automate legal marketing.