IDEs SHOULD BE AVAILABLE TO HARDWARE ENGINEERS TOO!

Author: Syed Daniyal Khurram, Horace Chan

Speaker: Syed Daniyal Khurram

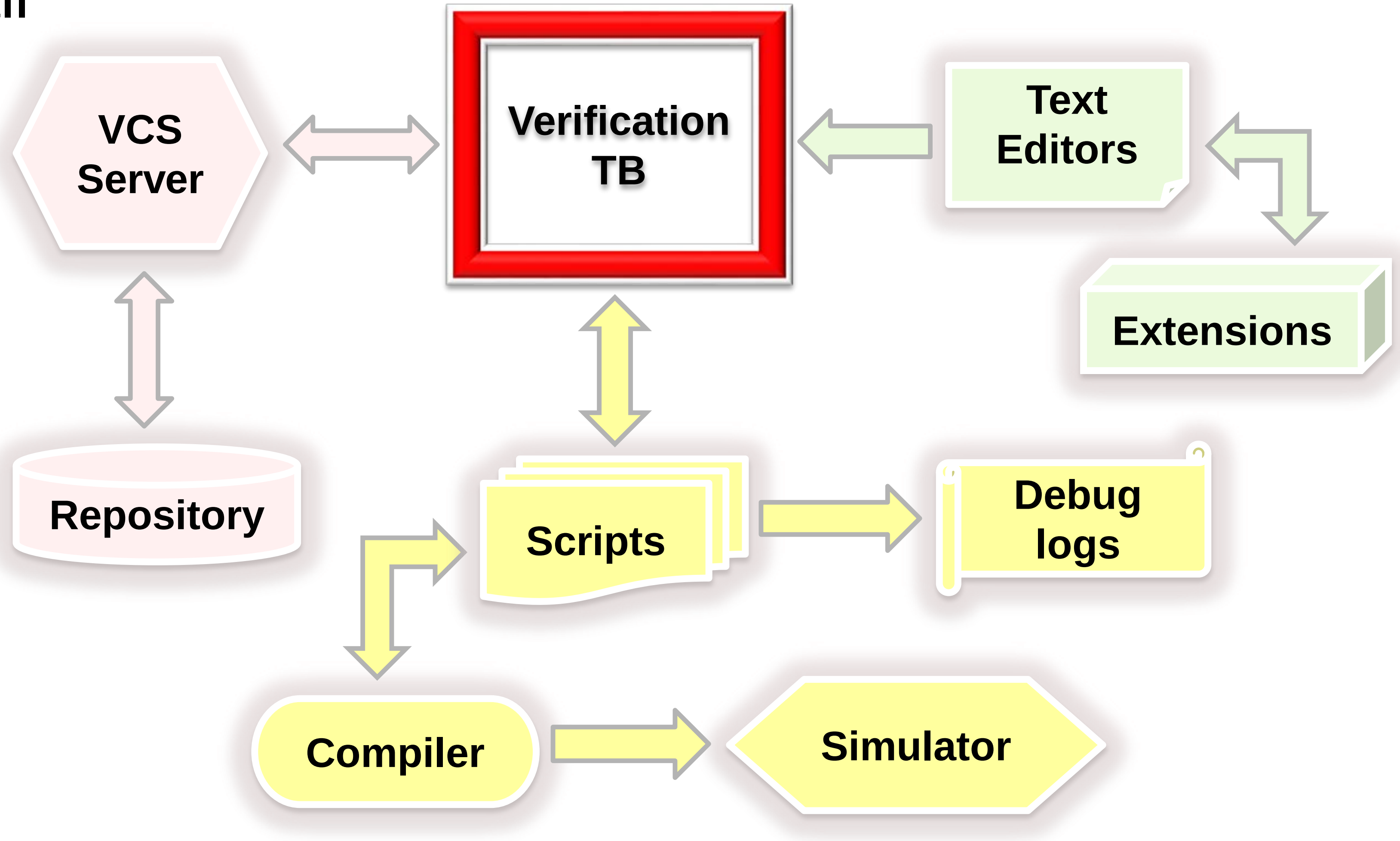
Contribution

- Illustrates productivity benefits of using IDEs in keeping with the demand of a SystemVerilog UVM TB
  - Internal feature evaluation of four popular IDEs: DVT, Sigasi, SlickEdit, SVEditor
- Analyzes application usability and addresses inhibitions towards IDE adoption
  - Helps eliminate/reduce application assessment costs

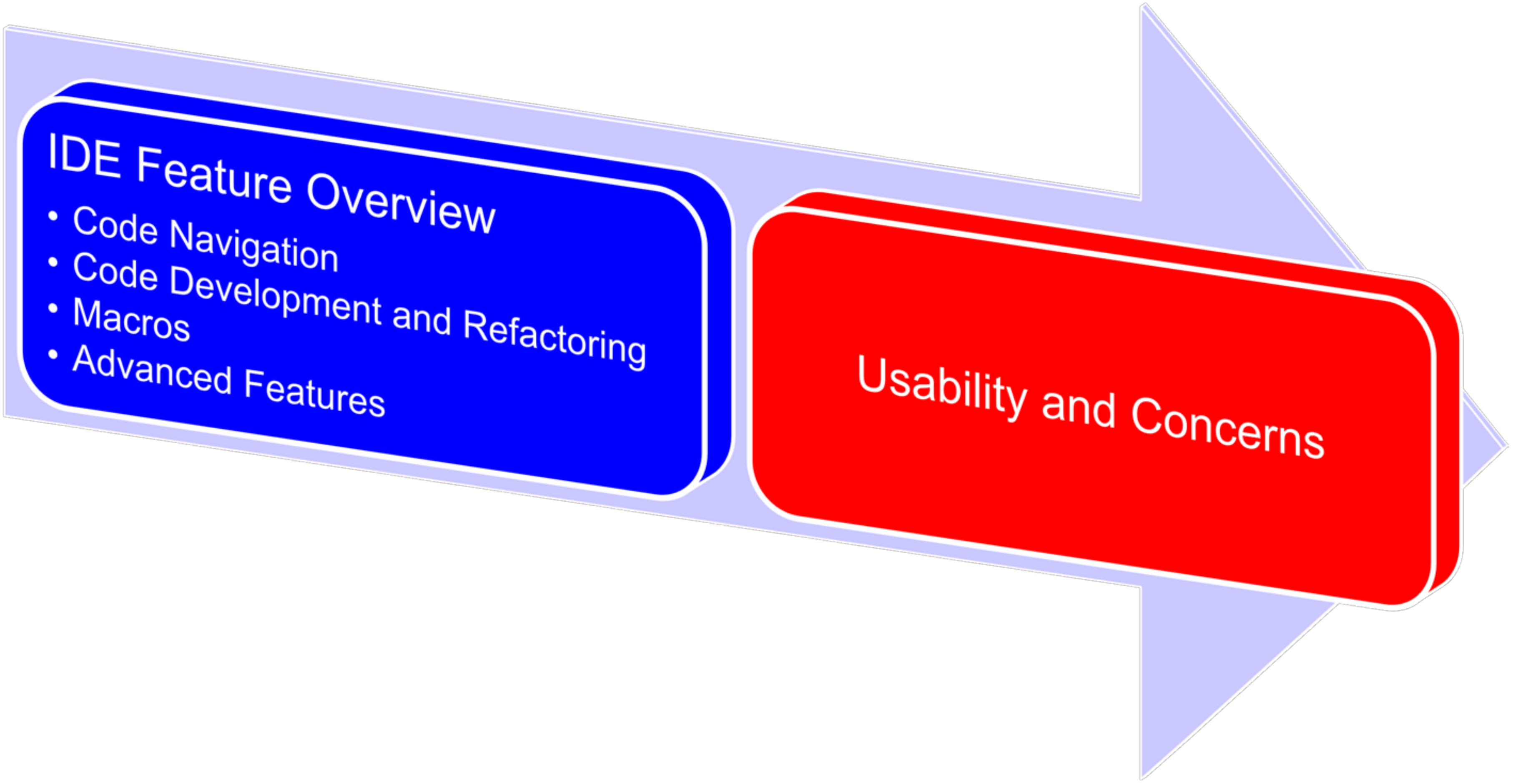
UVM (1.2) “UBus” example verification environment will be used for feature demonstration purposes

Motivation

- “One tool to rule them all”
- Automation of code development and simplification of debug for modern design sizes



Outline


IDE Feature Overview
• Code Navigation
• Code Development and Refactoring
• Macros
• Advanced Features
Usability and Concerns

IDE Feature Overview

Code Navigation

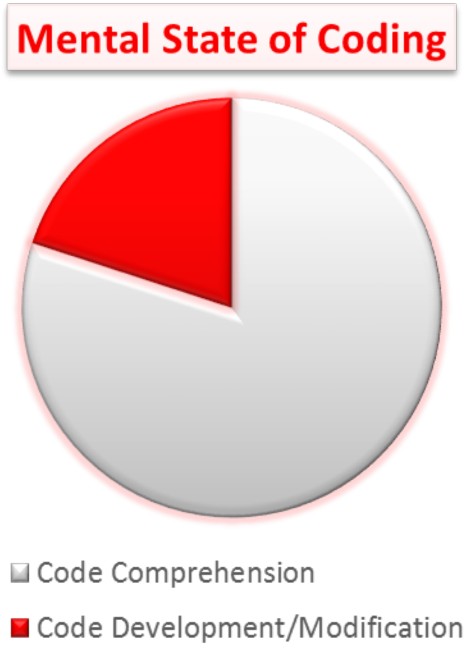
Symbol Lookup

File Browsing

Class Browsing

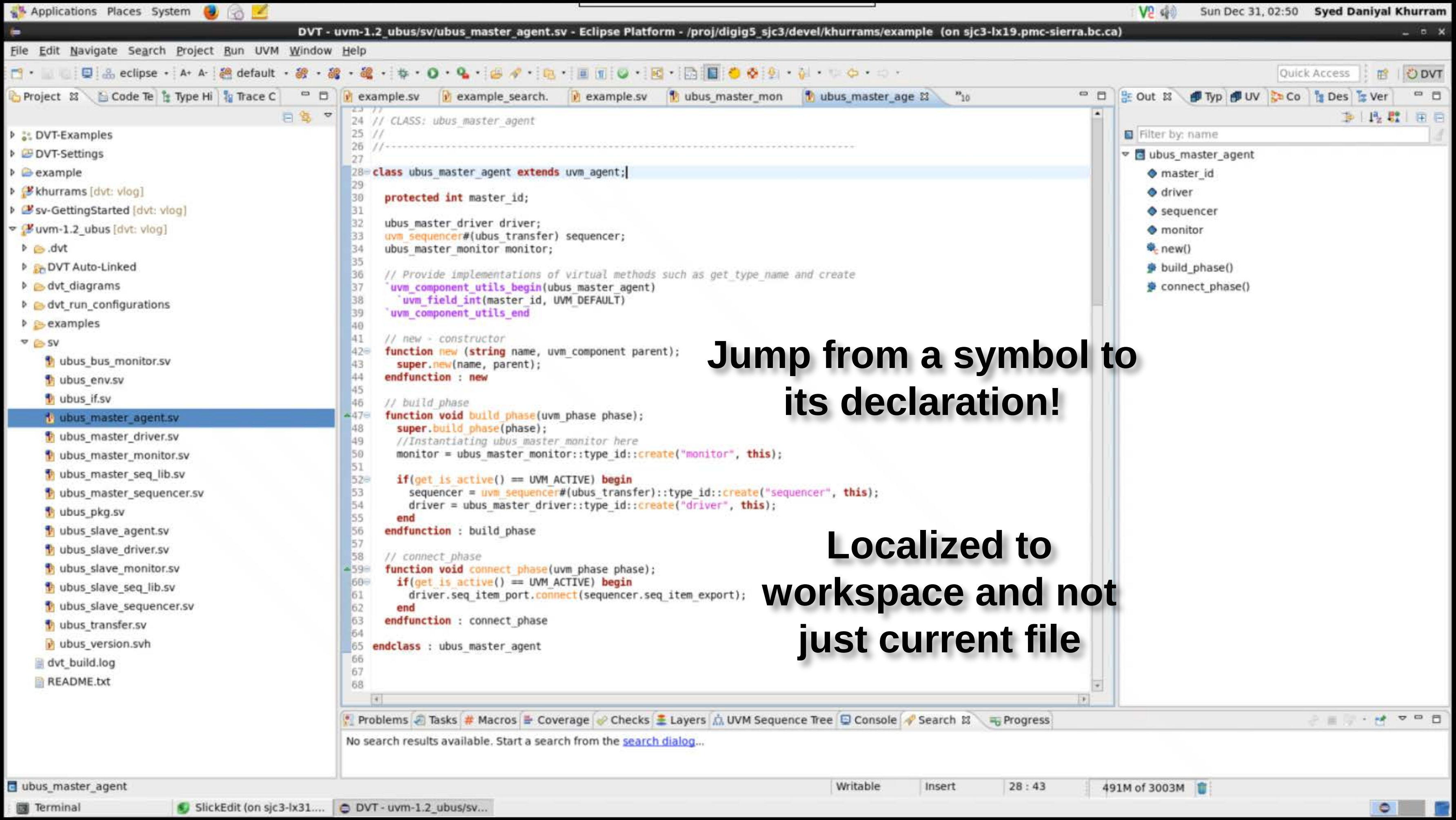
Design Browsing

Reduce time spent on understanding source code and locating design information!



Symbol Lookup :: Search by Definition

IDE : DVT



```
// CLASS: ubus_master_agent
//
//------------------------------------------------------------------------------

class ubus_master_agent extends uvm_agent;

  protected int master_id;

  ubus_master_driver driver;
  uvm_sequencer#(ubus_transfer) sequencer;
  ubus_master_monitor monitor;

  // Provide implementations of virtual methods such as get_type_name and create
  `uvm_component_utils_begin(ubus_master_agent)
    `uvm_field_int(master_id, UVM_DEFAULT)
  `uvm_component_utils_end

  // new - constructor
  function new (string name, uvm_component parent);
    super.new(name, parent);
  endfunction : new

  // build_phase
  function void build_phase(uvm_phase phase);
    super.build_phase(phase);
    //Instantiating ubus_master_monitor here
    monitor = ubus_master_monitor::type_id::create("monitor", this);

    if(get_is_active() == UVM_ACTIVE) begin
      sequencer = uvm_sequencer#(ubus_transfer)::type_id::create("sequencer", this);
      driver = ubus_master_driver::type_id::create("driver", this);
    end
  endfunction : build_phase

  // connect_phase
  function void connect_phase(uvm_phase phase);
    if(get_is_active() == UVM_ACTIVE) begin
      driver.seq_item_port.connect(sequencer.seq_item_export);
    end
  endfunction : connect_phase

endclass : ubus_master_agent
```

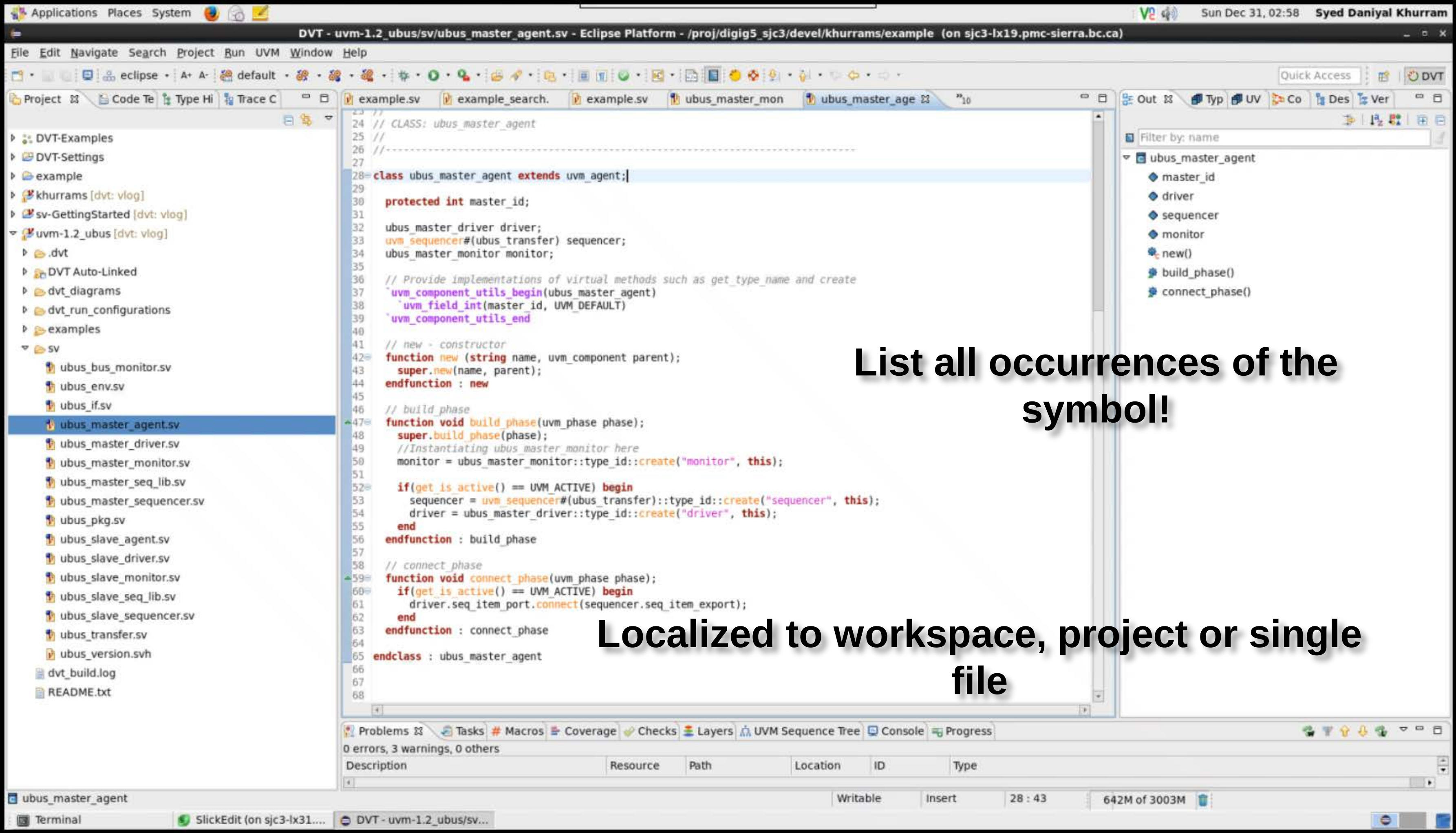
No search results available. Start a search from the search dialog...

Jump from a symbol to its declaration!

Localized to workspace and not just current file

Symbol Lookup :: Search by Reference

IDE : DVT



Applications Places System
Sun Dec 31, 02:58 Syed Daniyal Khurram
DVT - uvm-1.2_ubus/sv/ubus_master_agent.sv - Eclipse Platform - /proj/digig5_sjc3/devel/khurrams/example (on sjc3-lx19.pmc-sierra.bc.ca)
File Edit Navigate Search Project Run UVM Window Help
Project | Code Te | Type Hi | Trace C
example.sv | example_search. | example.sv | ubus_master_mon | ubus_master_age
Out | Typ | UV | Co | Des | Ver
Filter by: name
ubus_master_agent
master_id
driver
sequencer
monitor
new()
build_phase()
connect_phase()
DVT-Examples
DVT-Settings
example
khurrams [dvt: vlog]
sv-GettingStarted [dvt: vlog]
uvm-1.2_ubus [dvt: vlog]
.dvt
DVT Auto-Linked
dvt_diagrams
dvt_run_configurations
examples
sv
ubus_bus_monitor.sv
ubus_env.sv
ubus_if.sv
ubus_master_agent.sv
ubus_master_driver.sv
ubus_master_monitor.sv
ubus_master_seq_lib.sv
ubus_master_sequencer.sv
ubus_pkg.sv
ubus_slave_agent.sv
ubus_slave_driver.sv
ubus_slave_monitor.sv
ubus_slave_seq_lib.sv
ubus_slave_sequencer.sv
ubus_transfer.sv
ubus_version.svh
dvt_build.log
README.txt
Problems | Tasks | Macros | Coverage | Checks | Layers | UVM Sequence Tree | Console | Progress
0 errors, 3 warnings, 0 others
Description | Resource | Path | Location | ID | Type
ubus_master_agent
Writable | Insert | 28 : 43 | 642M of 3003M
Terminal | SlickEdit (on sjc3-lx31.... | DVT - uvm-1.2_ubus/sv...

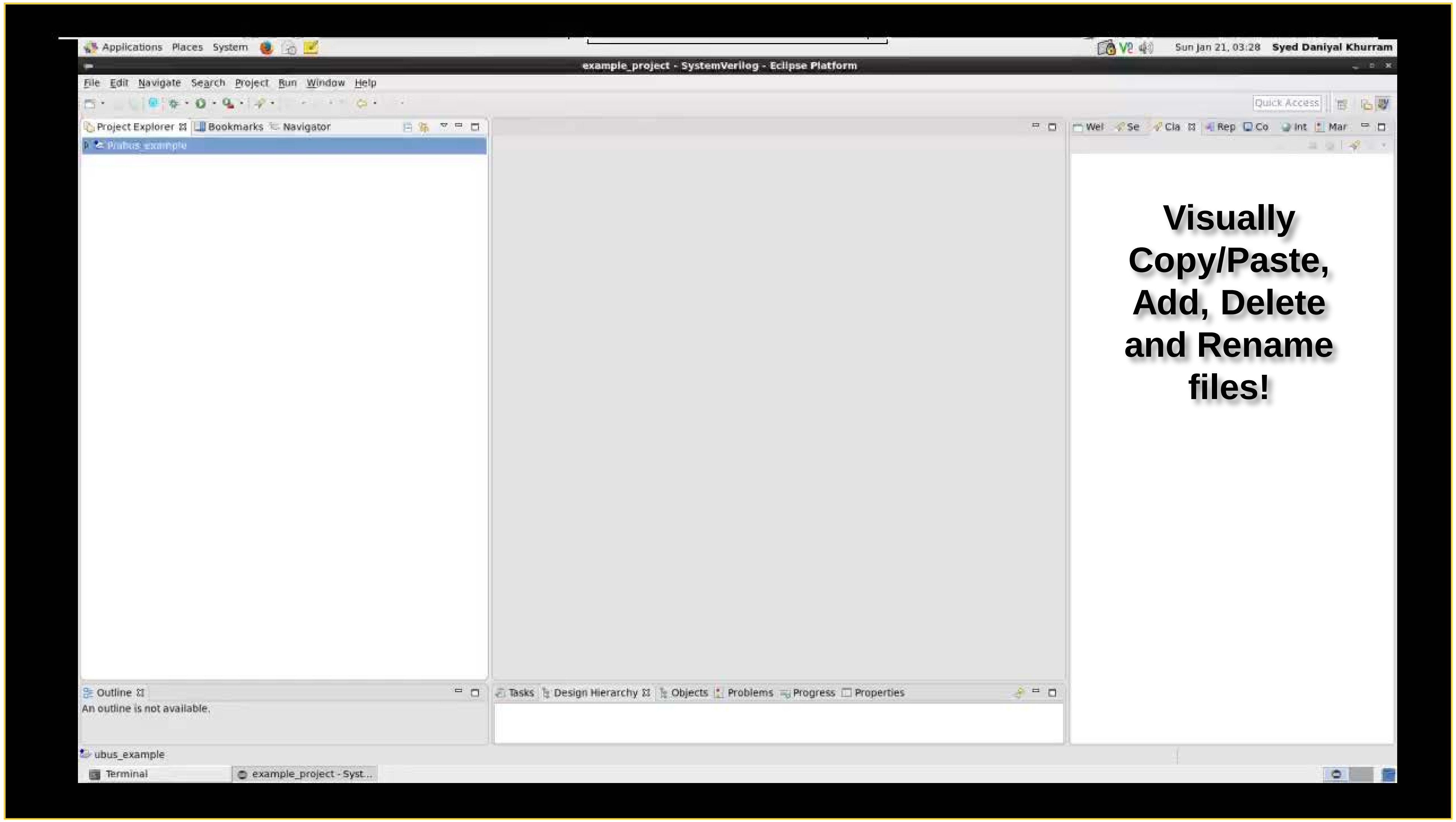
```
// CLASS: ubus_master_agent
//
//-----------------------------------------------------------------------------

class ubus_master_agent extends uvm_agent;

  protected int master_id;

  ubus_master_driver driver;
  uvm_sequencer#(ubus_transfer) sequencer;
  ubus_master_monitor monitor;

  // Provide implementations of virtual methods such as get_type_name and create
  `uvm_component_utils_begin(ubus_master_agent)
    `uvm_field_int(master_id, UVM_DEFAULT)
  `uvm_component_utils_end

  // new - constructor
  function new (string name, uvm_component parent);
    super.new(name, parent);
  endfunction : new

  // build_phase
  function void build_phase(uvm_phase phase);
    super.build_phase(phase);
    //Instantiating ubus_master_monitor here
    monitor = ubus_master_monitor::type_id::create("monitor", this);

    if(get_is_active() == UVM_ACTIVE) begin
      sequencer = uvm_sequencer#(ubus_transfer)::type_id::create("sequencer", this);
      driver = ubus_master_driver::type_id::create("driver", this);
    end
  endfunction : build_phase

  // connect_phase
  function void connect_phase(uvm_phase phase);
    if(get_is_active() == UVM_ACTIVE) begin
      driver.seq_item_port.connect(sequencer.seq_item_export);
    end
  endfunction : connect_phase

endclass : ubus_master_agent
```

List all occurrences of the symbol!

Localized to workspace, project or single file

File Browsing

IDE : SVEditor

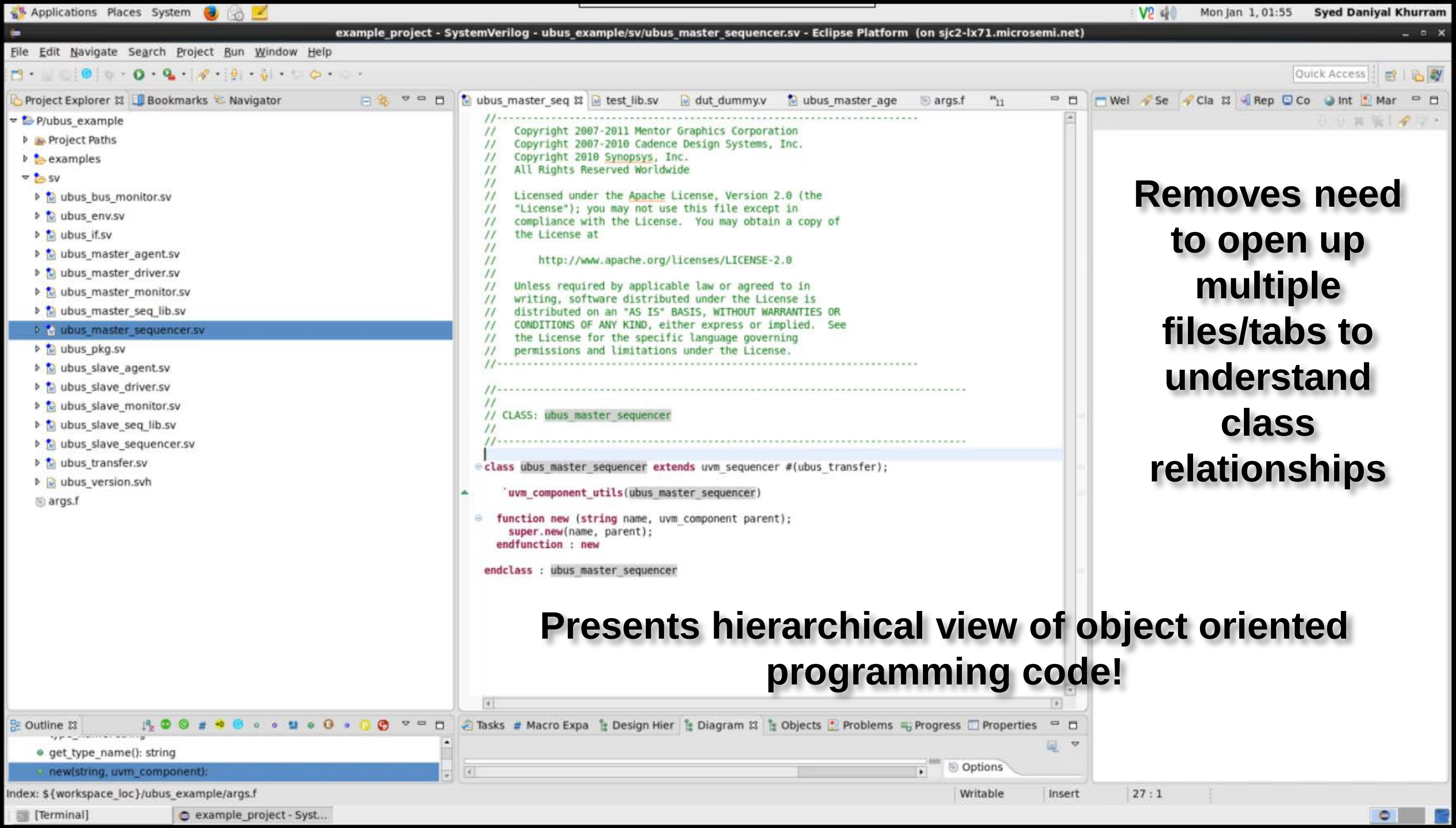

Applications Places System
Sun Jan 21, 03:28 Syed Daniyal Khurram
example_project - SystemVerilog - Eclipse Platform
File Edit Navigate Search Project Run Window Help
Quick Access
Project Explorer
Bookmarks
Navigator
Wel Se Cla Rep Co Int Mar
Outline
Tasks Design Hierarchy Objects Problems Progress Properties
ubus_example
Terminal
example_project - Syst...

An outline is not available.

Visually Copy/Paste, Add, Delete and Rename files!

Class Browsing

IDE : SVEditor


Applications Places System
Mon Jan 1, 01:55 Syed Daniyal Khurram
example_project - SystemVerilog - ubus_example/sv/ubus_master_sequencer.sv - Eclipse Platform (on sjc2-lx71.microsemi.net)
File Edit Navigate Search Project Run Window Help
Quick Access
Project Explorer Bookmarks Navigator
ubus_master_seq test_lib.sv dut_dummy.v ubus_master_age args.f
Wel Se Cla Rep Co Int Mar
Outline
Tasks Macro Expa Design Hier Diagram Objects Problems Progress Properties
Options
Writable Insert 27 : 1
[Terminal] example_project - Syst...

```
P/ubus_example
  Project Paths
  examples
  sv
    ubus_bus_monitor.sv
    ubus_env.sv
    ubus_if.sv
    ubus_master_agent.sv
    ubus_master_driver.sv
    ubus_master_monitor.sv
    ubus_master_seq_lib.sv
    ubus_master_sequencer.sv
    ubus_pkg.sv
    ubus_slave_agent.sv
    ubus_slave_driver.sv
    ubus_slave_monitor.sv
    ubus_slave_seq_lib.sv
    ubus_slave_sequencer.sv
    ubus_transfer.sv
    ubus_version.svh
  args.f
```

```
//----------------------------------------------------------------------
//   Copyright 2007-2011 Mentor Graphics Corporation
//   Copyright 2007-2010 Cadence Design Systems, Inc.
//   Copyright 2010 Synopsys, Inc.
//   All Rights Reserved Worldwide
//
//   Licensed under the Apache License, Version 2.0 (the
//   "License"); you may not use this file except in
//   compliance with the License.  You may obtain a copy of
//   the License at
//
//       http://www.apache.org/licenses/LICENSE-2.0
//
//   Unless required by applicable law or agreed to in
//   writing, software distributed under the License is
//   distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR
//   CONDITIONS OF ANY KIND, either express or implied.  See
//   the License for the specific language governing
//   permissions and limitations under the License.
//----------------------------------------------------------------------

//------------------------------------------------------------------------------
//
// CLASS: ubus_master_sequencer
//
//------------------------------------------------------------------------------

class ubus_master_sequencer extends uvm_sequencer #(ubus_transfer);

    `uvm_component_utils(ubus_master_sequencer)

  function new (string name, uvm_component parent);
    super.new(name, parent);
  endfunction : new

endclass : ubus_master_sequencer
```

```
get_type_name(): string
new(string, uvm_component):
```

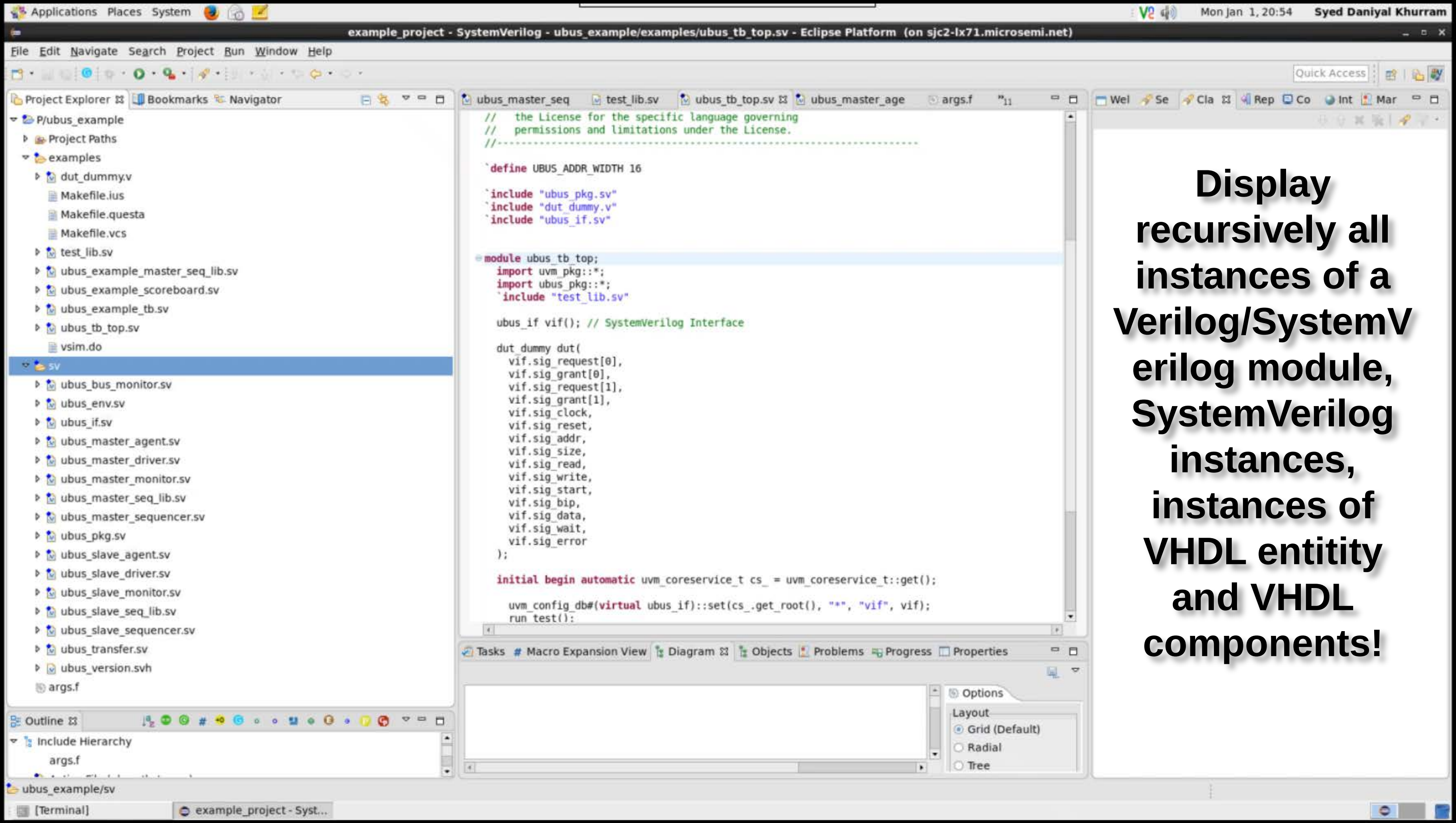
```
Index: ${workspace_loc}/ubus_example/args.f
```

Removes need to open up multiple files/tabs to understand class relationships

Presents hierarchical view of object oriented programming code!

Design Browsing

IDE : SVEditor



example_project - SystemVerilog - ubus_example/examples/ubus_tb_top.sv - Eclipse Platform (on sjc2-lx71.microsemi.net)

```
P/ubus_example
  Project Paths
  examples
    dut_dummy.v
    Makefile.ius
    Makefile.questa
    Makefile.vcs
    test_lib.sv
    ubus_example_master_seq_lib.sv
    ubus_example_scoreboard.sv
    ubus_example_tb.sv
    ubus_tb_top.sv
    vsim.do
  sv
    ubus_bus_monitor.sv
    ubus_env.sv
    ubus_if.sv
    ubus_master_agent.sv
    ubus_master_driver.sv
    ubus_master_monitor.sv
    ubus_master_seq_lib.sv
    ubus_master_sequencer.sv
    ubus_pkg.sv
    ubus_slave_agent.sv
    ubus_slave_driver.sv
    ubus_slave_monitor.sv
    ubus_slave_seq_lib.sv
    ubus_slave_sequencer.sv
    ubus_transfer.sv
    ubus_version.svh
  args.f
```

```
//   the License for the specific language governing
//   permissions and limitations under the License.
//----------------------------------------------------------------------

`define UBUS_ADDR_WIDTH 16

`include "ubus_pkg.sv"
`include "dut_dummy.v"
`include "ubus_if.sv"


module ubus_tb_top;
  import uvm_pkg::*;
  import ubus_pkg::*;
  `include "test_lib.sv"

  ubus_if vif(); // SystemVerilog Interface

  dut_dummy dut(
    vif.sig_request[0],
    vif.sig_grant[0],
    vif.sig_request[1],
    vif.sig_grant[1],
    vif.sig_clock,
    vif.sig_reset,
    vif.sig_addr,
    vif.sig_size,
    vif.sig_read,
    vif.sig_write,
    vif.sig_start,
    vif.sig_bip,
    vif.sig_data,
    vif.sig_wait,
    vif.sig_error
  );

  initial begin automatic uvm_coreservice_t cs_ = uvm_coreservice_t::get();

    uvm_config_db#(virtual ubus_if)::set(cs_.get_root(), "*", "vif", vif);
    run_test();
```

Outline: Include Hierarchy, args.f

Layout: [✓ Grid (Default)] Radial Tree

Display recursively all instances of a Verilog/SystemVerilog module, SystemVerilog instances, instances of VHDL entitity and VHDL components!

Code Navigation :: Takeaways

- Symbol Lookup
  - *Push-button alternative to external/built-in search plugins such as 'Grep'*
- File Browsing
  - *File management through an easy to use interface*
- Class Browsing
  - *Visualization of the hierarchy makes it easier to understand class-based TB organization and relationships*
- Design Browsing
  - *Design engineers benefit when analyzing external IP or during design audits.*

Next to none prior knowledge of the workspace/project hierarchy required!

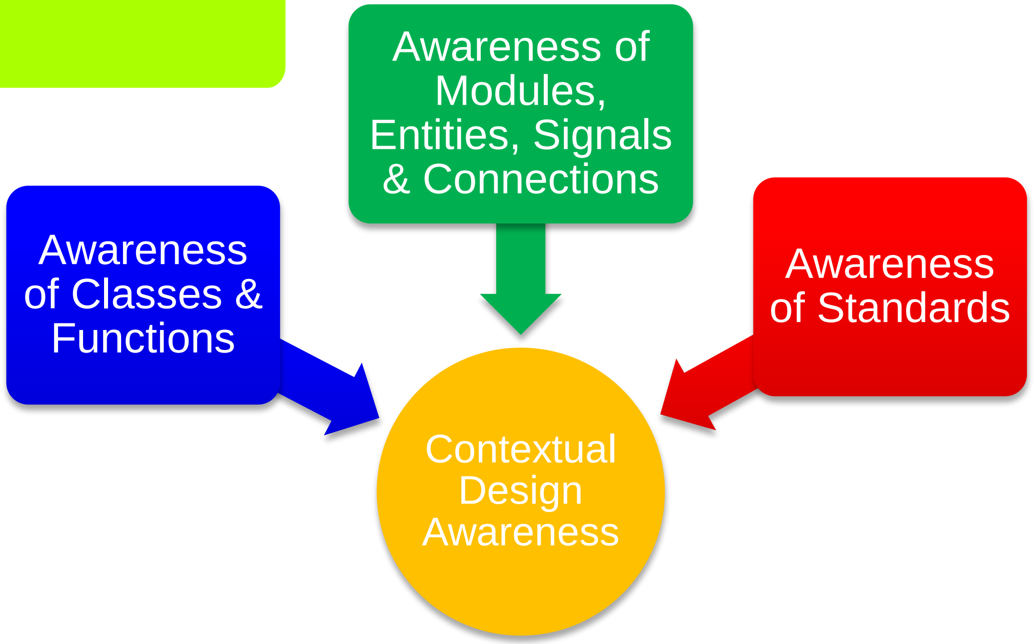
Code Development & Refactoring

Auto-Editing

Intelligent Refactoring

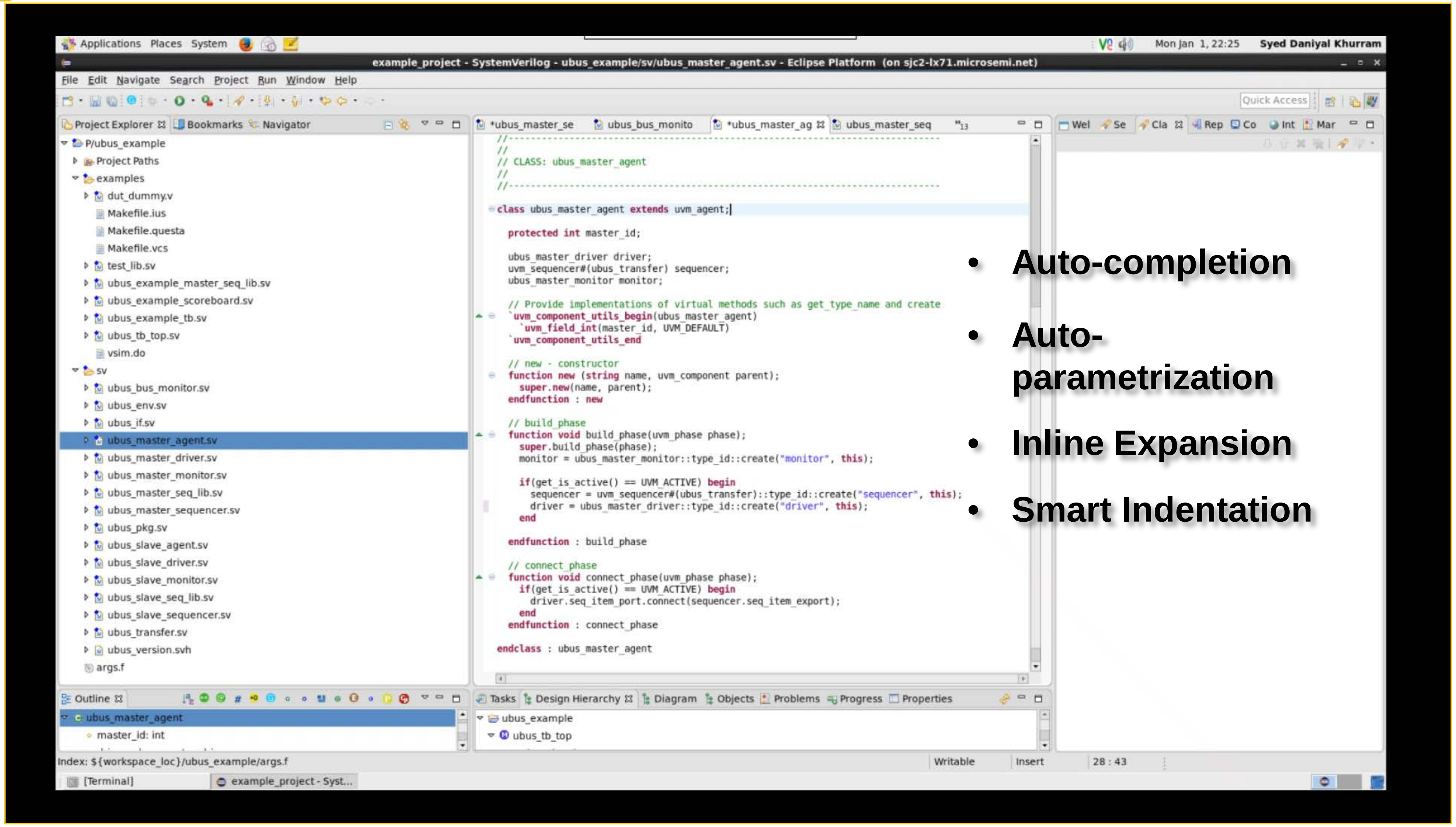
Code Collapse

Automate common programming tasks through intelligent code formatting and code prediction!


Awareness of Modules, Entities, Signals & Connections
Awareness of Classes & Functions
Awareness of Standards
Contextual Design Awareness

Auto-Editing

IDE : SVEditor

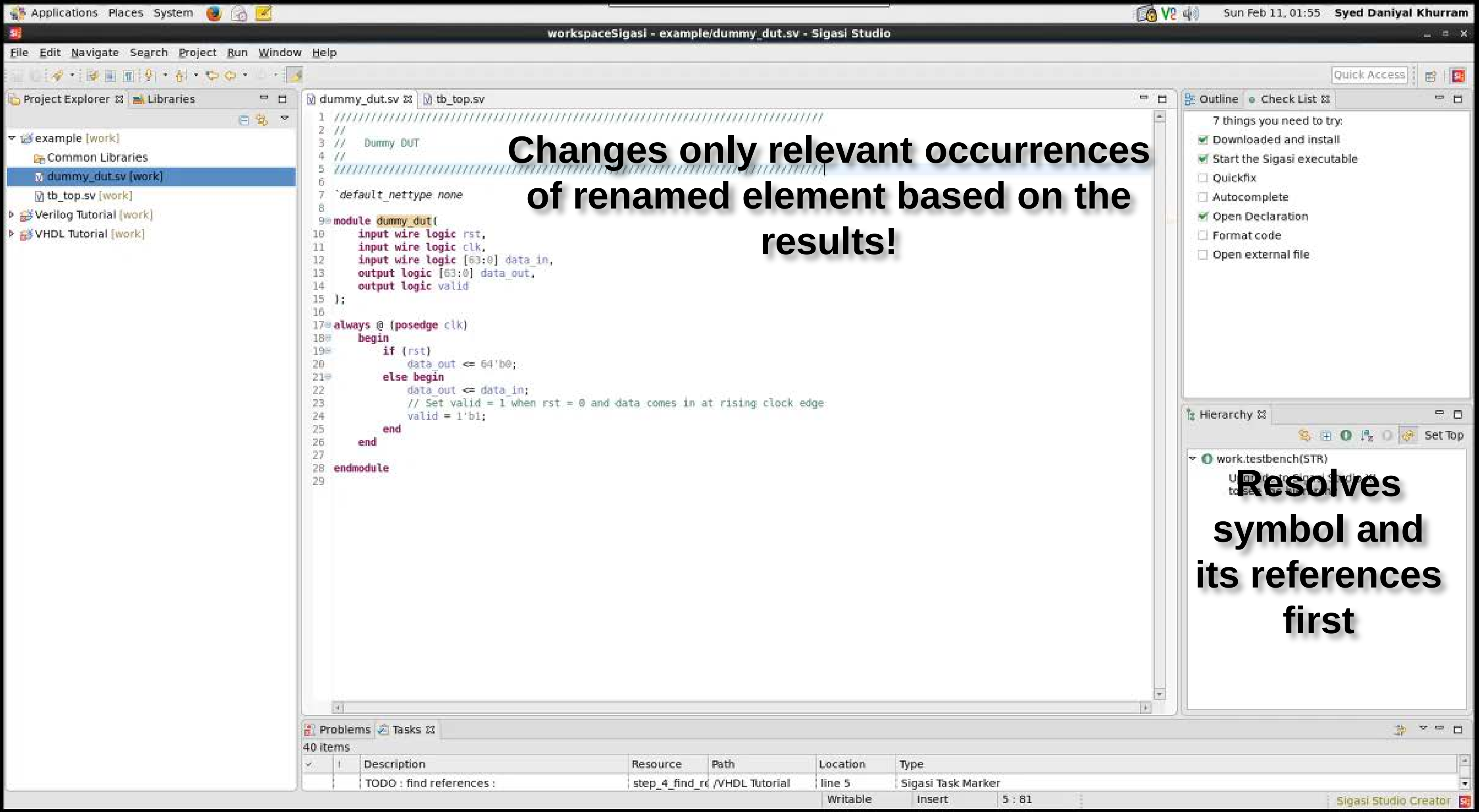


```
//------------------------------------------------------------------------------
//
// CLASS: ubus_master_agent
//
//------------------------------------------------------------------------------

class ubus_master_agent extends uvm_agent;

  protected int master_id;

  ubus_master_driver driver;
  uvm_sequencer#(ubus_transfer) sequencer;
  ubus_master_monitor monitor;

  // Provide implementations of virtual methods such as get_type_name and create
  `uvm_component_utils_begin(ubus_master_agent)
    `uvm_field_int(master_id, UVM_DEFAULT)
  `uvm_component_utils_end

  // new - constructor
  function new (string name, uvm_component parent);
    super.new(name, parent);
  endfunction : new

  // build_phase
  function void build_phase(uvm_phase phase);
    super.build_phase(phase);
    monitor = ubus_master_monitor::type_id::create("monitor", this);

    if(get_is_active() == UVM_ACTIVE) begin
      sequencer = uvm_sequencer#(ubus_transfer)::type_id::create("sequencer", this);
      driver = ubus_master_driver::type_id::create("driver", this);
    end

  endfunction : build_phase

  // connect_phase
  function void connect_phase(uvm_phase phase);
    if(get_is_active() == UVM_ACTIVE) begin
      driver.seq_item_port.connect(sequencer.seq_item_export);
    end
  endfunction : connect_phase

endclass : ubus_master_agent
```

- Auto-completion
- Auto-parametrization
- Inline Expansion
- Smart Indentation

Intelligent Refactoring

IDE : Sigasi (Studio Creator)



Changes only relevant occurrences of renamed element based on the results!

Resolves symbol and its references first

```
////////////////////////////////////////////////////////////////////////////////
//
//   Dummy DUT
//
////////////////////////////////////////////////////////////////////////////////

`default_nettype none

module dummy_dut(
    input wire logic rst,
    input wire logic clk,
    input wire logic [63:0] data_in,
    output logic [63:0] data_out,
    output logic valid
);

always @ (posedge clk)
    begin
        if (rst)
            data_out <= 64'b0;
        else begin
            data_out <= data_in;
            // Set valid = 1 when rst = 0 and data comes in at rising clock edge
            valid = 1'b1;
        end
    end

endmodule
```

7 things you need to try:
- [x] Downloaded and install
- [x] Start the Sigasi executable
- [ ] Quickfix
- [ ] Autocomplete
- [x] Open Declaration
- [ ] Format code
- [ ] Open external file

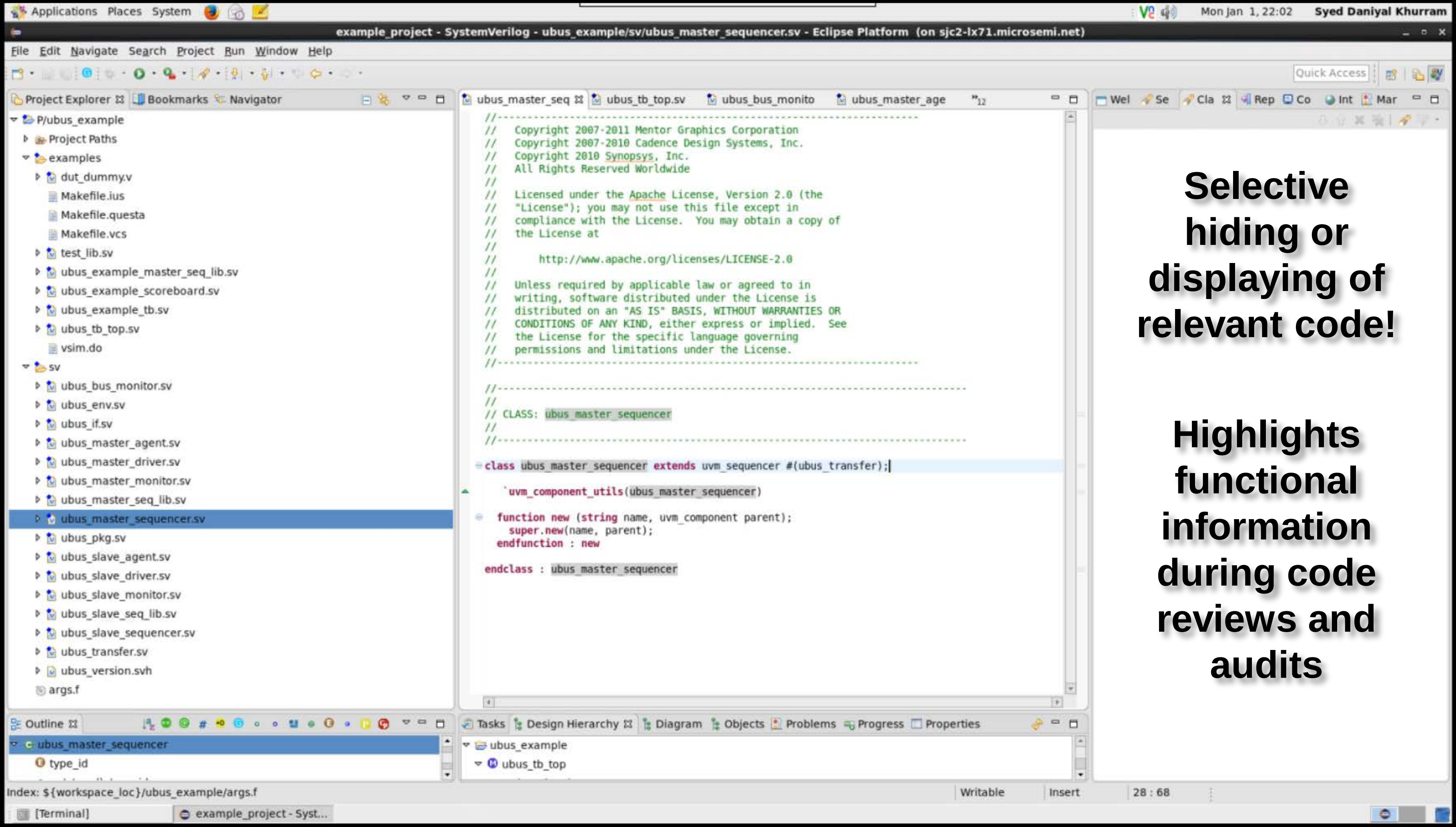
Problems Tasks

40 items

	!	Description	Resource	Path	Location	Type
		TODO : find references :	step_4_find_r	/VHDL Tutorial	line 5	Sigasi Task Marker

Code Collapse

IDE : SVEditor



Applications Places System
Mon Jan 1, 22:02 Syed Daniyal Khurram
example_project - SystemVerilog - ubus_example/sv/ubus_master_sequencer.sv - Eclipse Platform (on sjc2-lx71.microsemi.net)
File Edit Navigate Search Project Run Window Help
Quick Access
Project Explorer Bookmarks Navigator
ubus_master_seq ubus_tb_top.sv ubus_bus_monito ubus_master_age
Outline
Tasks Design Hierarchy Diagram Objects Problems Progress Properties
Writable Insert 28 : 68
[Terminal] example_project - Syst...

```
P/ubus_example
  Project Paths
  examples
    dut_dummy.v
    Makefile.ius
    Makefile.questa
    Makefile.vcs
    test_lib.sv
    ubus_example_master_seq_lib.sv
    ubus_example_scoreboard.sv
    ubus_example_tb.sv
    ubus_tb_top.sv
    vsim.do
  sv
    ubus_bus_monitor.sv
    ubus_env.sv
    ubus_if.sv
    ubus_master_agent.sv
    ubus_master_driver.sv
    ubus_master_monitor.sv
    ubus_master_seq_lib.sv
    ubus_master_sequencer.sv
    ubus_pkg.sv
    ubus_slave_agent.sv
    ubus_slave_driver.sv
    ubus_slave_monitor.sv
    ubus_slave_seq_lib.sv
    ubus_slave_sequencer.sv
    ubus_transfer.sv
    ubus_version.svh
  args.f
```

```
//----------------------------------------------------------------------
//   Copyright 2007-2011 Mentor Graphics Corporation
//   Copyright 2007-2010 Cadence Design Systems, Inc.
//   Copyright 2010 Synopsys, Inc.
//   All Rights Reserved Worldwide
//
//   Licensed under the Apache License, Version 2.0 (the
//   "License"); you may not use this file except in
//   compliance with the License.  You may obtain a copy of
//   the License at
//
//       http://www.apache.org/licenses/LICENSE-2.0
//
//   Unless required by applicable law or agreed to in
//   writing, software distributed under the License is
//   distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR
//   CONDITIONS OF ANY KIND, either express or implied.  See
//   the License for the specific language governing
//   permissions and limitations under the License.
//----------------------------------------------------------------------

//------------------------------------------------------------------------------
//
// CLASS: ubus_master_sequencer
//
//------------------------------------------------------------------------------

class ubus_master_sequencer extends uvm_sequencer #(ubus_transfer);

   `uvm_component_utils(ubus_master_sequencer)

  function new (string name, uvm_component parent);
    super.new(name, parent);
  endfunction : new

endclass : ubus_master_sequencer
```

```
ubus_master_sequencer
  type_id
```

```
ubus_example
  ubus_tb_top
```

Index: ${workspace_loc}/ubus_example/args.f

Selective hiding or displaying of relevant code!

Highlights functional information during code reviews and audits

Code Development & Refactoring :: Takeaways

- Auto-Editing
  - *Solution to typical questions that arise during code development:*
    - *What is the name of the method that you wish to use?*
    - *What is a methods order of arguments?*
    - *What are the possible values of an enumerated type?*
- Intelligent Refactoring
  - *Code transformations that maintain the behavior of the design*
    - *Especially handy when dealing with port related changes in design modules!*
- Code Collapse
  - *Useful for masking irrelevant information*

Enter the Macro

Macro Recording/Keyboard Macros

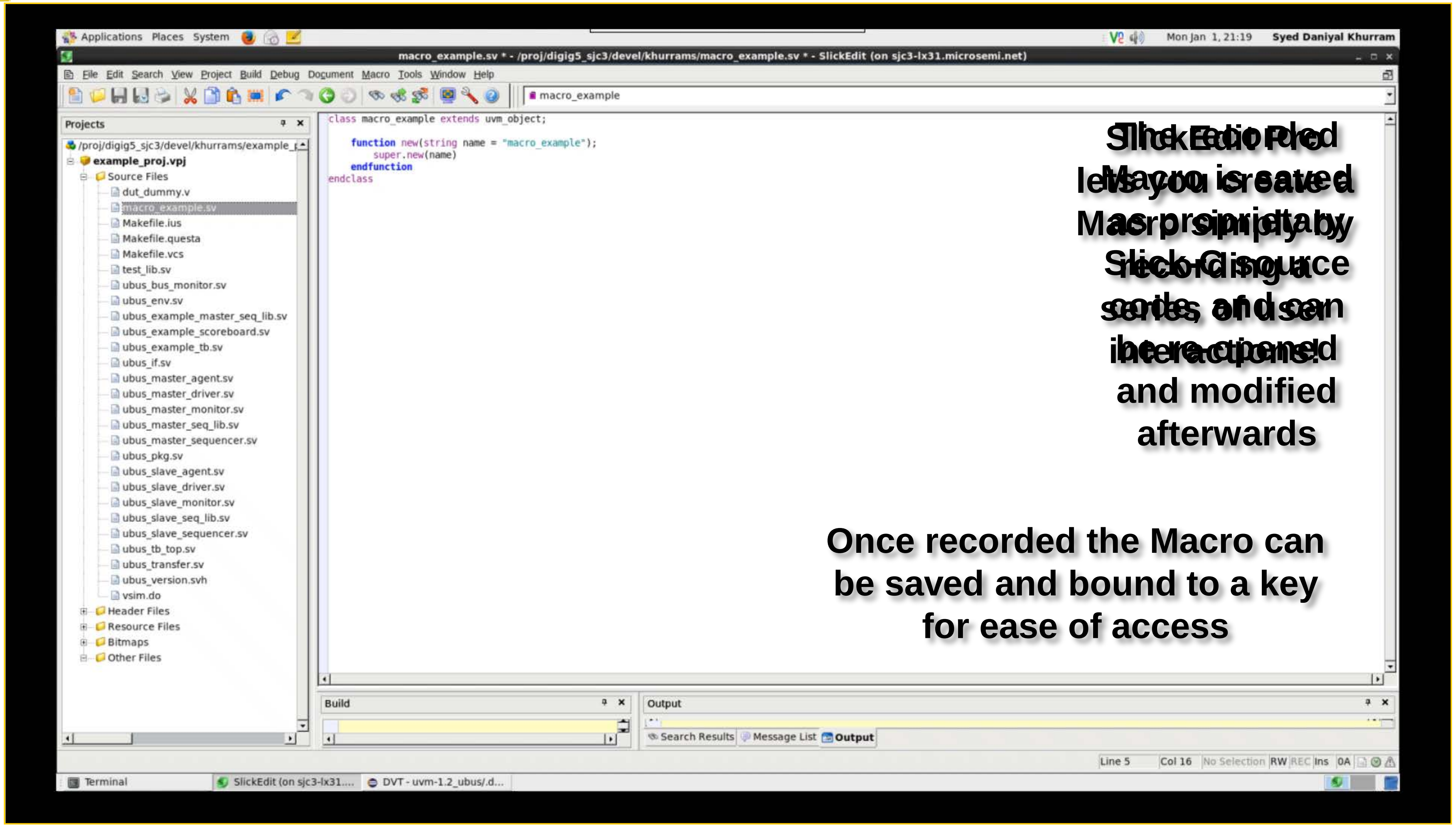
Macro Expansion

Automate repetitive actions performed frequently while writing code

Open pre-existing Macros in an interactive window and trace line by line for debug and analysis

Macro recording/Keyboard Macros

IDE : SlickEdit Pro



```
class macro_example extends uvm_object;

    function new(string name = "macro_example");
        super.new(name)
    endfunction
endclass
```

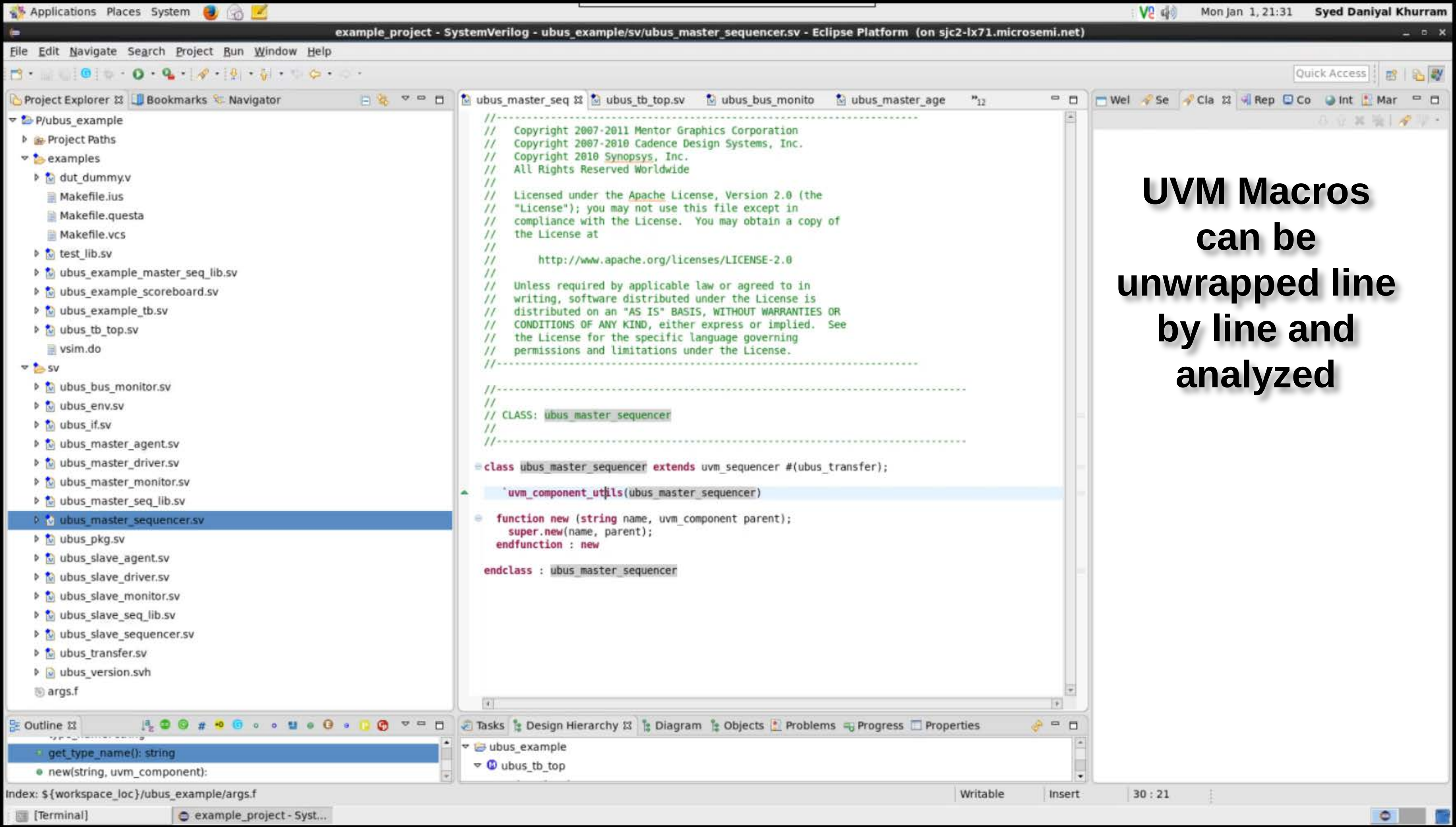
SlickEdit Pro lets you create a Macro simply by recording a series of user interactions!

The recorded Macro is saved as proprietary Slick-C source code, and can be re-opened and modified afterwards

Once recorded the Macro can be saved and bound to a key for ease of access

Macro Expansion

IDE : SVEditor



```
//------------------------------------------------------------------------
//   Copyright 2007-2011 Mentor Graphics Corporation
//   Copyright 2007-2010 Cadence Design Systems, Inc.
//   Copyright 2010 Synopsys, Inc.
//   All Rights Reserved Worldwide
//
//   Licensed under the Apache License, Version 2.0 (the
//   "License"); you may not use this file except in
//   compliance with the License.  You may obtain a copy of
//   the License at
//
//       http://www.apache.org/licenses/LICENSE-2.0
//
//   Unless required by applicable law or agreed to in
//   writing, software distributed under the License is
//   distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR
//   CONDITIONS OF ANY KIND, either express or implied.  See
//   the License for the specific language governing
//   permissions and limitations under the License.
//------------------------------------------------------------------------

//----------------------------------------------------------------------------
//
// CLASS: ubus_master_sequencer
//
//----------------------------------------------------------------------------

class ubus_master_sequencer extends uvm_sequencer #(ubus_transfer);

  `uvm_component_utils(ubus_master_sequencer)

  function new (string name, uvm_component parent);
    super.new(name, parent);
  endfunction : new

endclass : ubus_master_sequencer
```

UVM Macros can be unwrapped line by line and analyzed

Macros:: Takeaways

- Macro Recording/Keyboard Macros
  - *Save time spent on typing re-use code*
  - *Extend existing command functionality or add new commands e.g Macros to display duplicate lines of text, file attributes etc*

Emacs still reigns supreme in this category! As macro programming in Emacs Lisp can be remarkably powerful in the hands of an experienced user

- *Macro Expansion*
  - *Expand and analyze :*
    - *Proprietary simulator pre-processing code*
    - *Macros included as part of a verification methodology standard such as UVM*
    - *Macros created in a tool specific programming language such as Slick-C*

Advanced Macro debug features such as breakpoint insertion can be utilized in IDEs that support compiler/simulator integration capabilities

Advanced Features

Integration with UVM

Integration with Simulation Tools

Revision Control Integration

- Each IDE integrates with UVM
- Integration with popular compilers and simulators through the use of add-ons/licenses/tools & build configuration

- Revision Control from within the IDE
- Eclipse based IDEs require plugins e.g Subclipse
- SlickEdit Pro has inbuilt VCS support

IDE Integration with Tools and Standards

- *Additional UVM debug features available in DVT:*
  - *UVM Factory queries*
  - *UVM Templates*
  - *UVM Browser & Sequence Tree*

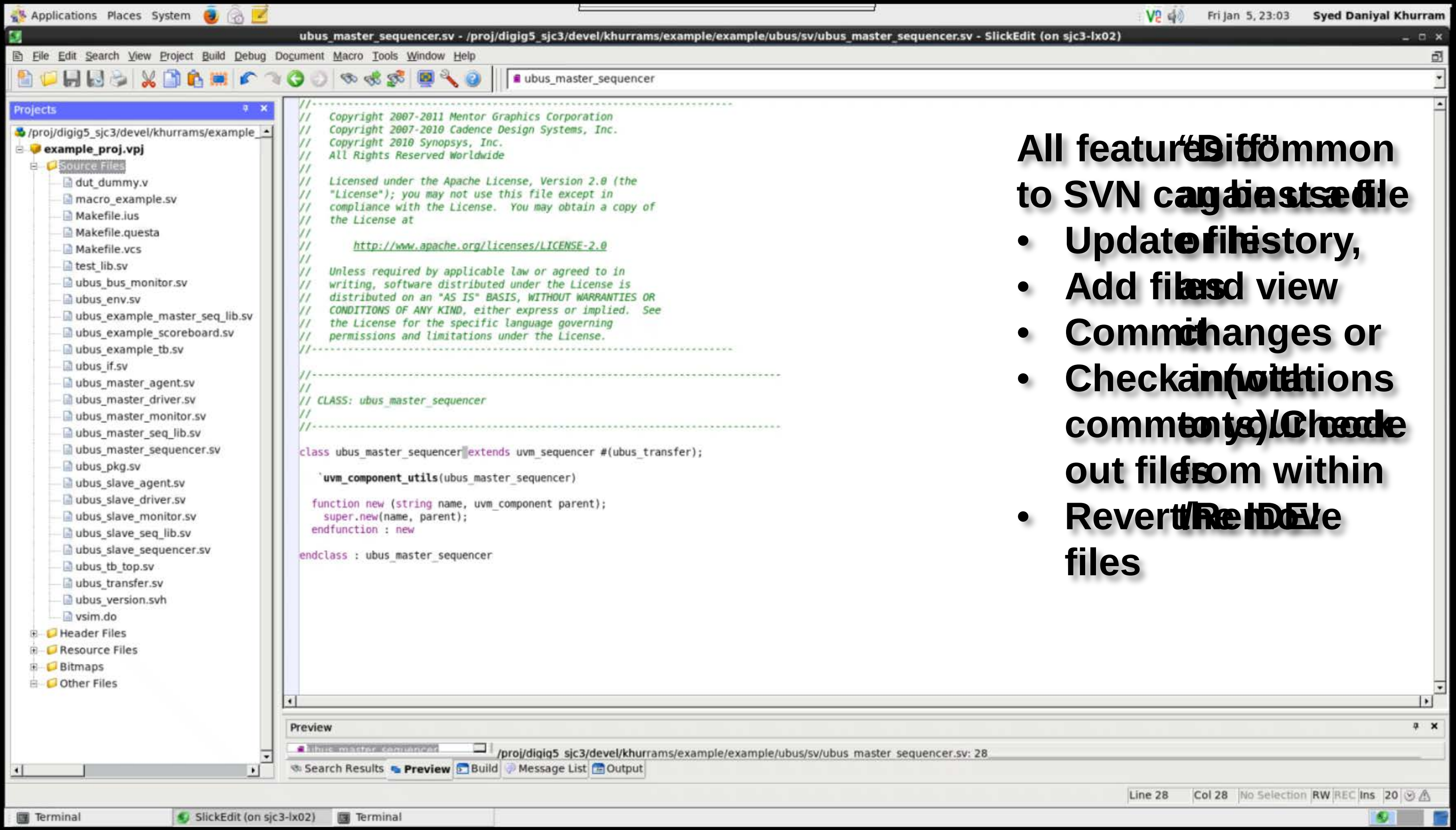
While UVM debug features are supported in most advanced simulators available in the market, they can only be used post-compilation!

- *Integration with simulation tools:*
  - *External tools and build configuration in Eclipse based IDEs. However this requires strenuous effort and may not work for all tools*
  - *DVT : Add-on tool (DVTDebugger)*
  - *Sigasi : Sigasi Studio Creator and higher*

Refer to the product website or the full paper for a list of supported simulators!

Revision Control Integration

IDE : SlickEdit Pro



Applications Places System
Fri Jan 5, 23:03 Syed Daniyal Khurram
ubus_master_sequencer.sv - /proj/digig5_sjc3/devel/khurrams/example/example/ubus/sv/ubus_master_sequencer.sv - SlickEdit (on sjc3-lx02)
File Edit Search View Project Build Debug Document Macro Tools Window Help
ubus_master_sequencer
Projects
/proj/digig5_sjc3/devel/khurrams/example
example_proj.vpj
Source Files
dut_dummy.v
macro_example.sv
Makefile.ius
Makefile.questa
Makefile.vcs
test_lib.sv
ubus_bus_monitor.sv
ubus_env.sv
ubus_example_master_seq_lib.sv
ubus_example_scoreboard.sv
ubus_example_tb.sv
ubus_if.sv
ubus_master_agent.sv
ubus_master_driver.sv
ubus_master_monitor.sv
ubus_master_seq_lib.sv
ubus_master_sequencer.sv
ubus_pkg.sv
ubus_slave_agent.sv
ubus_slave_driver.sv
ubus_slave_monitor.sv
ubus_slave_seq_lib.sv
ubus_slave_sequencer.sv
ubus_tb_top.sv
ubus_transfer.sv
ubus_version.svh
vsim.do
Header Files
Resource Files
Bitmaps
Other Files
Preview
/proj/digig5_sjc3/devel/khurrams/example/example/ubus/sv/ubus_master_sequencer.sv: 28
Search Results Preview Build Message List Output
Line 28 Col 28 No Selection RW REC Ins 20
Terminal SlickEdit (on sjc3-lx02) Terminal

```
//----------------------------------------------------------------------
//   Copyright 2007-2011 Mentor Graphics Corporation
//   Copyright 2007-2010 Cadence Design Systems, Inc.
//   Copyright 2010 Synopsys, Inc.
//   All Rights Reserved Worldwide
//
//   Licensed under the Apache License, Version 2.0 (the
//   "License"); you may not use this file except in
//   compliance with the License.  You may obtain a copy of
//   the License at
//
//       http://www.apache.org/licenses/LICENSE-2.0
//
//   Unless required by applicable law or agreed to in
//   writing, software distributed under the License is
//   distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR
//   CONDITIONS OF ANY KIND, either express or implied.  See
//   the License for the specific language governing
//   permissions and limitations under the License.
//----------------------------------------------------------------------

//------------------------------------------------------------------------------
//
// CLASS: ubus_master_sequencer
//
//------------------------------------------------------------------------------

class ubus_master_sequencer extends uvm_sequencer #(ubus_transfer);

  `uvm_component_utils(ubus_master_sequencer)

  function new (string name, uvm_component parent);
    super.new(name, parent);
  endfunction : new

endclass : ubus_master_sequencer
```

All features common to SVN can be used:

- Update files
- Add files
- Commit
- Checkin (with comments)/Check out files
- Revert/Remove files

"Diff" against a file of history, and view changes or annotations to your code from within the IDE

Advanced Features:: Takeaways

- Integration with UVM
  - *Significant in modern ASIC verification*
- Integration with simulation tools
  - *On the fly debug!*
  - *Invoke compiler/simulator from within tool GUI and trace warnings/errors to problematic source code*
- Revision Control Integration
  - *Removes time spent switching between command line and text editor*
  - *Visual ‘diff’ is powerful and interactive*

Usability and Concerns

Usability and Concerns

- *Learning Curve*
  - *User-friendly and easy to pick-up by junior engineers*
  - *Prior experience in using established IDEs(Eclipse, Visual Studio) reduces training time*
- *Reduction of Tools*
  - *Vast array of features in a centralized environment*
  - *External plugins are supported*
- *Support*
  - *Customer specific support available for all commercial IDEs*
- *Setup Flow*
  - *Possibly the biggest adherence towards IDE adoption*
  - *Quite simple actually with clear instructions*

Conclusion

- There are tools!
  - Established tools exist in the marketspace
  - Choose what best fits your needs
- Worth your time!
  - Invest to save time
- Less is more!
  - Centralize your environment
  - Reduce resource consumption

A comprehensive summary of features available per tool is tabulated in the full paper as a reference!

Questions?