**OBJECTIVE**

*The purpose of this document is to describe in detail* *tasks that mentors need to follow during mentoring trainees during practical part of UVM training.*

**DOCUMENT HISTORY**

| ***Revision*** | ***Date*** | ***Author*** | ***Changes*** |
| --- | --- | --- | --- |
| 01 | 07/04/19 | Milan Bjelobrk | Initial version. Created based on document used during last internship cycle. |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**TABLE OF CONTENTS**

1. project description 3

2. TASK DESCRIPTION 4

2.1 WB2UART 5

2.1.1 Create verification documents for WB2UART project 5

2.1.2 Develop WishBone UVC 5

2.1.3 Develop UART UVC 6

2.1.4 Develop complete verification environment for WB2UART DUT module 7

2.1.5 Develop test-cases for functional verification of WB2UART DUT 7

2.1.6 WB2UART DUT debug on developed TCs 7

# project description

UVM training project is meant to facilitate acquisition of practical skills based on theoretical part of UVM Basic Training. With respect to that idea each trainee will be given task to create two UVCs and build entire verification environment and tests for checking functionalities of DUT.

As DUT, WB2UART IP module will be used. The DUT has been subjected only to some level of testing, so assumption can be made that it contains unknown defects. Defects discovered during verification phase will be reported in Defect Tracking Sheet and will be corrected in RTL only if effort for doing so doesn’t impact largely verification training process or if defect is blocking further verification progress. Assessment for defect correction will be done with mentors/supervisors for each defect.

# TASK DESCRIPTION

Each trainee will be assigned with several tasks.

**NOTE:**

Due to limited number of licenses, running simulations in batch mode is obligatory for all.

In case you want to review the waveform of the latest simulation ran in batch mode, you should do the following:

1. Start **‘SimVision’** tool,
2. Select **‘File->Open Database’**,
3. Select **‘\*.trn’** file from the provided test folder hierarchy,

Click on **‘Open & Dismiss’** button to load the simulation waveform.

**NOTE FOR MENTORS:**

Make sure that no simulation created files or/and folders are checked in to SVN repository.

Make sure that all simulation created files or/and folders are deleted at the end of the training process.

## WB2UART

### Create verification documents for WB2UART project

* Template for Verification plan is available [here](https://alfresco.elsys-eastern.rs:8443/share/page/site/internships/document-details?nodeRef=workspace://SpacesStore/557d2d2c-48b5-4387-a2d0-8d93899a40e8)
* Template for Verification architecture is available [here](https://alfresco.elsys-eastern.rs:8443/share/page/site/internships/document-details?nodeRef=workspace://SpacesStore/8f931cf0-d557-4f1a-842a-b5bd4cee8c3c)
* Use discussions with mentors to understand requirements for writing each chapter of mentioned documents
* Once the draft version of documents are available, mentors will review them and give “green light” for moving to next task
* These doc’s should be updated regularly during the project

### Develop WishBone UVC

WishBone protocol specification is available [here](https://alfresco.elsys-eastern.rs:8443/share/page/site/internships/document-details?nodeRef=workspace://SpacesStore/56d9c8db-078b-4ccc-b516-59d69b630f66).

* Re-arrange existing **‘WB2UART/verif/’** folder structure as following:

**systemverilog/uvc\_lib/wishbone\_uvc**

**|---------> sv** – contains all UVC files needed for reuse.

**|---------> example** – contains UVC2UVC env, tb and example test-case needed to show

user how to instance, configure and use UVC.

**|---------> makefile** –script for running UVC2UVC test case.

* Move your **‘training\_uvc’** code under **‘wishbone\_uvc’** folder – complying with folder structure above
* Rename the files to have them represent **‘wishbone\_uvc’** code
* Continue **‘wishbone\_uvc’** development from existing code now placed under **‘wishbone\_uvc’** folder
* Extend current agent (master) to comply with **‘WishBone’** protocol (supported features)
* Develop slave agent for it:
  + Implement driver & IF classes --> [**1** day, upon completion review with assigned mentor]
    - IF should be common for both agents
  + Implement monitor class --> [**1** day, upon completion review with mentor]
  + Implement sequence lib class (basic sequences only) --> [**1** day, upon completion review with mentor]
* Implement test class --> [**1** day, upon completion review with mentor]
* Update interface class with protocol checkers (a.k.a. SVAs)
* Test it with UVC\_2\_UVC test case --> [**2** days, upon completion review with mentor]
* **Features that should be supported**:
  + Tag types (supported for all transaction types):
    - Address Tag
    - Data Tag
    - Cycle Tag
  + Transaction types:
    - Single Read/Write
    - Single Pipelined Read/Write
    - Block Read/Write
    - Read Modified Write

### Develop UART UVC

UART protocol specification is available [here](https://alfresco.elsys-eastern.rs:8443/share/page/site/internships/document-details?nodeRef=workspace://SpacesStore/1ce8555b-dd21-4c21-9301-384646e75372).

* Create UVC skeleton using **SV UVC\_GEN** tool
  + Manual for using UVC generator: **SV UVC\_GEN User Guide**

Location on [Intranet](https://intranet.elsys-eastern.rs): **‘Tech Center -> Tools -> SV GEN &LINT tool’**

* + All UVC related files should be placed under **‘systemverilog/uvc\_lib/’**, and hence:
    - Provide the destination directory path in the named text-box element,
    - Select the **‘Two agents’** radio button,
    - Check the **‘Create testbench’** check-box
  + Re-arrange generated folder structure as following:

**systemverilog/uvc\_lib/uart\_uvc**

**|---------> sv** - contains all UVC files needed for reuse.

**|---------> example** – contains UVC2UVC env, tb and example test-case needed to show

user how to instance, configure and use UVC.

**|---------> makefile** –script for running UVC2UVC test case.

* Develop both agents (master and slave) for UART protocol, and for it:
  + Implement item class --> [**2** days, upon completion review with mentor]
  + Implement driver & IF classes --> [**4** days, upon completion review with mentor]
    - IF should be common for both agents
  + Implement monitor class --> [**4** days, upon completion review with mentor]
  + Implement sequence lib class --> [**2** days, only basic sequences, upon completion review with mentor]
* Implement test class --> [**1** day, upon completion review with mentor]
* Update interface class with protocol checkers (a.k.a. SVAs)
* Test it with UVC\_2\_UVC test case --> [**2** days, upon completion review with mentor]
* **Features that should be supported:**
  + Data length:
    - 5bits – 8bits
  + Stop bits:
    - 1bit and 2bits
    - 1.5bits if data length is 5bits
  + Parity:
    - Enable/Disable parity bit
    - Odd/Even number of “1” in transfer

### Develop complete verification environment for WB2UART DUT module

WB2UART IP functional specification is available [here](https://alfresco.elsys-eastern.rs:8443/share/page/site/internships/document-details?nodeRef=workspace://SpacesStore/acc6e5c5-3df0-4664-b36c-f6046f1b78a2).

* Create <project\_name>\_top\_tb.v module containing --> [**2** days, upon completion review with mentor]:
  + DUT instantiation
  + Virtual IFs
  + Clock & reset generator
* Create top\_env containing --> [**2** days, upon completion review with mentor]:
  + Instances of UVCs
  + Configuration of virtual IFs
  + Env config
  + Virtual sequencer
  + Global monitor, if needed
  + Scoreboard
* Test (only base test class) --> [**1** day, upon completion review with mentor]
* Develop virtual sequences --> [**2** day, upon completion review with mentor]
* Further develop Scoreboard according to verification plan (checkers/coverage) --> [**7** days, upon completion review with mentor; debug time included]
* **Features that should be supported:**
  + Everything described in functional spec except:
    - Modem related features (signals, registers, …)
    - Register “Debug 2”

### Develop test-cases for functional verification of WB2UART DUT

* This task should be done accordingly to verification plan document. ) --> [**6** days, upon completion review with mentor; debug time included]

### WB2UART DUT debug on developed TCs

* At least xxx RTL bugs are present in the design.