MŰEGYETEM 1782

Budapest University of Technology and Economics
Faculty of Electrical Engineering and Informatics
Department of Electron Devices

Smart Systems Integrated Test Environment (SSIT)

User Manual

Rev. 29/10/2014 22:43

Authors:
Ferenc Ender
Péter Horváth
András Timár

Budapest, 2014

SSI
Smart Systems Integration

1 What is SSIT?

Smart Systems Integrated Test Environment is a software framework aiming to simulate integrated systems in behavioral level. Behavioral simulations enable the parameter tuning of the individual components while the system can be investigated as a whole, which results in a system level specification. The framework consists of three main functions:

1. Setting up and manage simulation parameters for a reduced order model (ROM) of a MEMS device, perform a simulation through ANSYS Batch APDL and view the results.
2. Setting up and manage simulation parameters for a circuit of analog components, perform a simulation through Mentor Graphics ELDO and view the results.
3. Setting up and manage simulation parameters for a digital circuit represented in VHDL and initiates a project in Mentor Graphics QuestaSim.

Pressure or acceleration load can be applied to the MEMS device and the transient response is calculated. The resulted data is transferred into the analog simulation engine, whose output is again transferred into the digital simulator. By investigating the system output the desired operation of the system could be validated.

The framework manages the input and output files of the three stages, checks the file dependencies, starts or terminates the simulator engines and provides a feedback of the simulation status. The simulation settings can be saved and later loaded into the framework.

2 Installing and running the software

The latest build of SSIT can be downloaded from the EDU site.

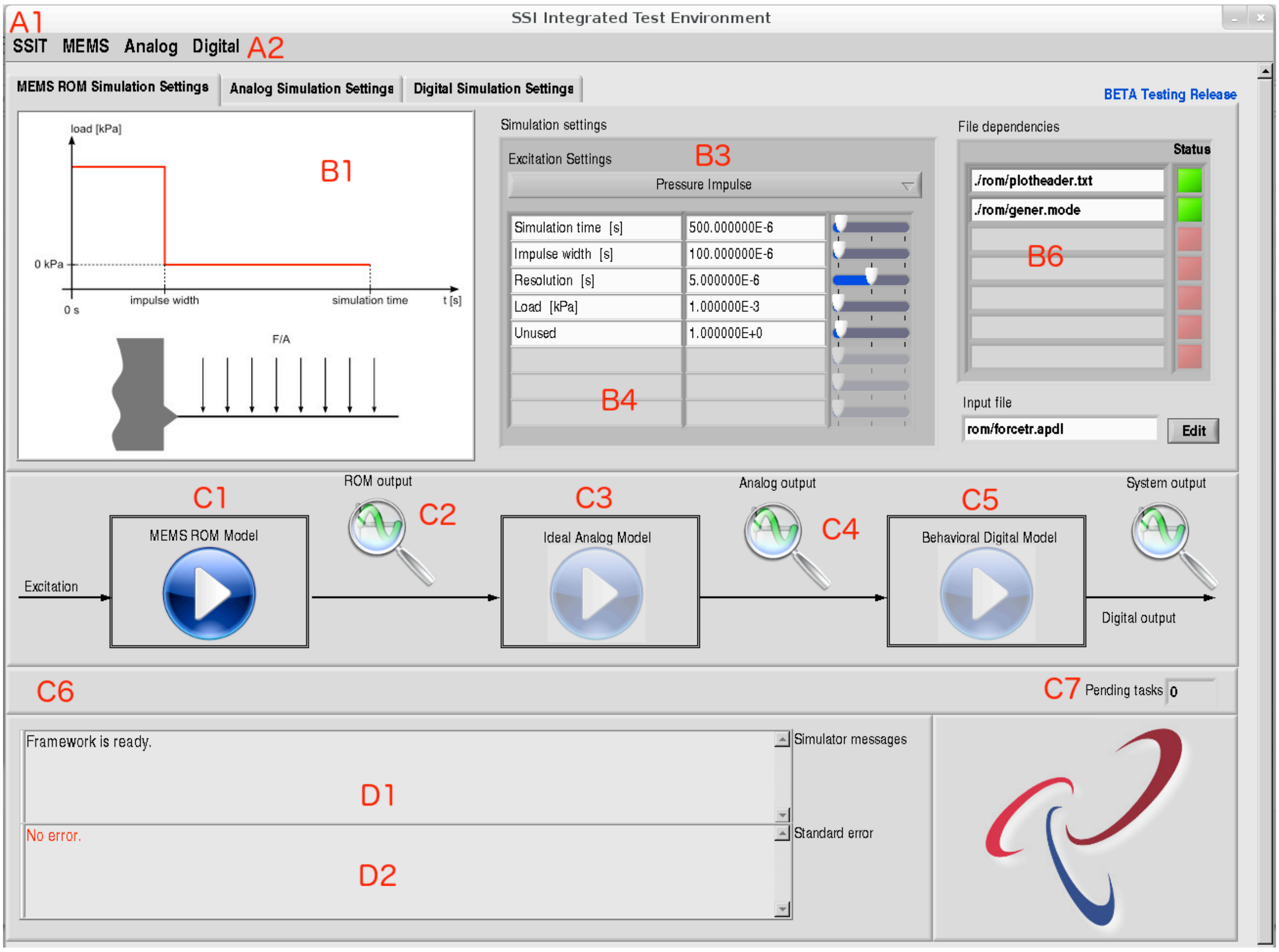
1. Go to the edu.eet.bme.hu site and download the install script.
2. Extract the file in your home directory
3. Start **`ssit_install.sh`**
4. For starting SSIT, click on the SSIT icon on the desktop.

Updates – The SSIT framework is ready to use, but currently you are using a Beta Testing Release. The current version may be updated. The above installation script can always download the newest release. To update your current copy

1. Run again **`ssit_install.sh`**
2. Please note, that **`./rom/forcetr.apdl`** will be rewritten by the updated version. If you need your copy, create a backup of your original file and then overwrite the newly installed one with yours.

Patches – in urgent cases patches will be available through the EDU site. Installation instructions will be also provided.

3 Graphical user interface



Excitation Settings: Pressure Impulse

Simulation time [s]	500.000000E-6
Impulse width [s]	100.000000E-6
Resolution [s]	5.000000E-6
Load [kPa]	1.000000E-3
Unused	1.000000E+0

	Status
./rom/plotheader.txt	🟩
./rom/gener.mode	🟩

Input file: rom/forcetr.apdl

Pending tasks: 0

Simulator messages: Framework is ready.

Standard error: No error.

Figure 1. SSIT GUI

The GUI consists of three main parts: the menu bar (A), the simulation settings tab (B), the simulator launcher (C) and the solver message bar (D).

A1. SSIT menu

a. About – Opens the About window of the software

b. Exit – exits the application. Your unsaved data will be lost.

A2. MEMS, Analog and Digital menus

a. Save MEMS/Analog/Digital data – Saves the actual settings of the MEMS/Analog/Digital simulation

b. Load MEMS/Analog/Digital data – Opens an existing MEMS/Analog/Digital simulation data file and loads the values

c. Load initial – Loads the default values of the MEMS/Analog/Digital simulation.

B1. **Figure window** – contains a figure to explain the simulation settings. Click on the figure to open it in an external viewer.

B2. **Simulation type tabs** – the simulation type (MEMS/Analog/Digital) can be set here.

B3. **Simulation mode selector** – every simulation type has more modes. The available parameter settings will change in every single mode. Certain parameters become *Unused* in some modes, changing them does not have any effect in the simulations result.

B4. **Simulation parameter settings** – the parameters of the simulations can be set here even by changing the slider values or typing in the new value in engineering format.

B5. **File dependency window** – the required files of running the actual simulation are listed here. Red box indicates that the file is missing. Simulation can be only started if every required file is present. Files are generated in the previous simulation type (e.g. Analog simulation dependencies are generated in the MEMS simulation).

B6. **Input file editor** – in some simulation types the toplevel input file can be opened here in an external editor.

C1. **Run/Stop MEMS simulation** – Once the simulation is started the button turns to Stop function. By pressing the Stop button the simulation will be interrupted.
C2. **View MEMS simulation results** – Once the MEMS simulation finished the results can be viewed using an external waveform viewer EZWave.
C3. **Run/Stop Analog simulation** – Once the simulation is started the button turns to Stop function. By pressing the Stop button the simulation will be interrupted.
C4. **View Analog simulation results** – Once the MEMS simulation finished the results can be viewed using an external waveform viewer EZWave.
C5. **Run Digital simulation** – Pressing the button launches the digital design framework (QuestaSim). Further simulation steps can be done there.
C6. **Status bar** – The actual progress of the MEMS/Analog simulation is indicated here.
C7. **Pending tasks indicator** – the framework allows to schedule more tasks at once. E.g. you can press the start MEMS simulation button, then change to the Analog tab and press the Digital simulation button and then the view results button. The pending task indicator will increase its value. After the MEMS simulation finished the Analog simulation will start immediately. After the analog simulation the results window will open.

D1. **Standard output messages** – Status messages and warning messages of the actual simulation appear here
D2. **Standard error messages** – Simulator engine error messages appear here.

4 Folder structure

The SSIT framework has the following folder structure:

- **`/`** - Root directory – the SSIT.exe executable and other supplementary files can be found here
  - **`/etc`** – runtime setup files, initial files, figures and settings can be found here – do not change
  - input files for the simulations
    - **`/ROM`** – input files of the ROM model: APDL scripts and ROM database items
    - **`/Analog`** – input files of the analog SPICE simulation: netlist files
    - **`/Digital`** – input files of the digital simulation – VHDL sources
  - **`/results`** – the simulation results in *csv* format

5 Simulation settings

5.1 MEMS ROM simulation

In MEMS ROM simulation the ROM model is evaluated which was previously generated in ANSYS. Copy the following resulted ANSYS files into **`./rom`**

- **`*.ph1, *.ph2`**
- **`*.rom`**
- **`use*.*`**

The SSIT framework loads the APDL script **`./rom/forcetr.apdl`** and transfers it to the ANSYS MAPDL in batch mode. ANSYS solver runs in the background, the progress is indicated on the SSIT GUI. Clicking the Edit button in the GUI opens this APDL script.
The file consists of three parts

- Initialize the ROM database **`(<your rom database>.rom`**) . Your ROM database was previously generated by ANSYS and the files were copied to the **`./rom`** folder. The database is loaded by the following command:
  `RMRESU,<your rom database>,rom`
- Creating nodes and assigning them to the ROM 144 element. Basically you do not have to modify anything here.
- The solution part, starting with /SOLU. Three types of loadings are set here according to your settings in SSIT. No need to modify anything.
- The postprocessing part, starting with /POST1. You should process the simulation output. See more at the *Simulation output* paragraph.

Simulation modes

- **Pressure impulse** – the MEMS structure is loaded by a sudden change of pressure using the **`RMLVSCALE`** command. The loading will be the same as it was set as first *TestLoad* during the ROM generation. The value of the loading is scaled up according to the value of the pressure impulse.
- **Pressure ramp** – the MEMS structure is loaded by a linearly increasing value using the **`RMLVSCALE`** command during the time length of the loading. The loading will be the same as it was set as *TestLoad* during the ROM generation. The value of the loading is scaled up according to the value of the pressure impulse.
- **Acceleration impulse** – the MEMS structure is loaded by a sudden change of gravity using the **`RMLVSCALE`** command. The loading will be the same as it was set as second *TestLoad* during the ROM generation. The value of the loading is scaled up according to the value of the pressure impulse.

Simulation output

- The simulation output depends on the MEMS device.
  - In case of piezoresistive devices the displacement of the master nodes should be determined and the overall prolongation of the resistor should be calculated by using any well-known formula. The calculation should be placed in the APDL code.

    Use the **`*GET,<variable>,NODE,<nodenum>,UX`** command embedded in a **`*DO`** loop to obtain the displacements. Refer ROM144 datasheet to find the corresponding nodes.

  - In case of capacitive devices the change of capacitance can be directly obtained from the ANSYS solver.

Use the `*GET,<variable>,ELEM,1,NMISC,<E number>` command embedded in a `*DO` loop to obtain the capacitance values. Refer ROM144 datasheet to find the corresponding element data sequence numbers (E).

- The obtained time dependent values are written in `solu.csv` file. The framework then copy the file to the `/results` folder from where EZWave opens it.

5.2 Analog simulation

In the analog simulation mode, the resulting piezoresistance or capacitance variation of the MEMS ROM output is evaluated. The analog simulation step requires the ROM simulation result as input. By default, the ROM simulation generates a “rom_output.csv” file in the “results” directory. The analog simulator looks for this file when starting simulation.

Simulation modes

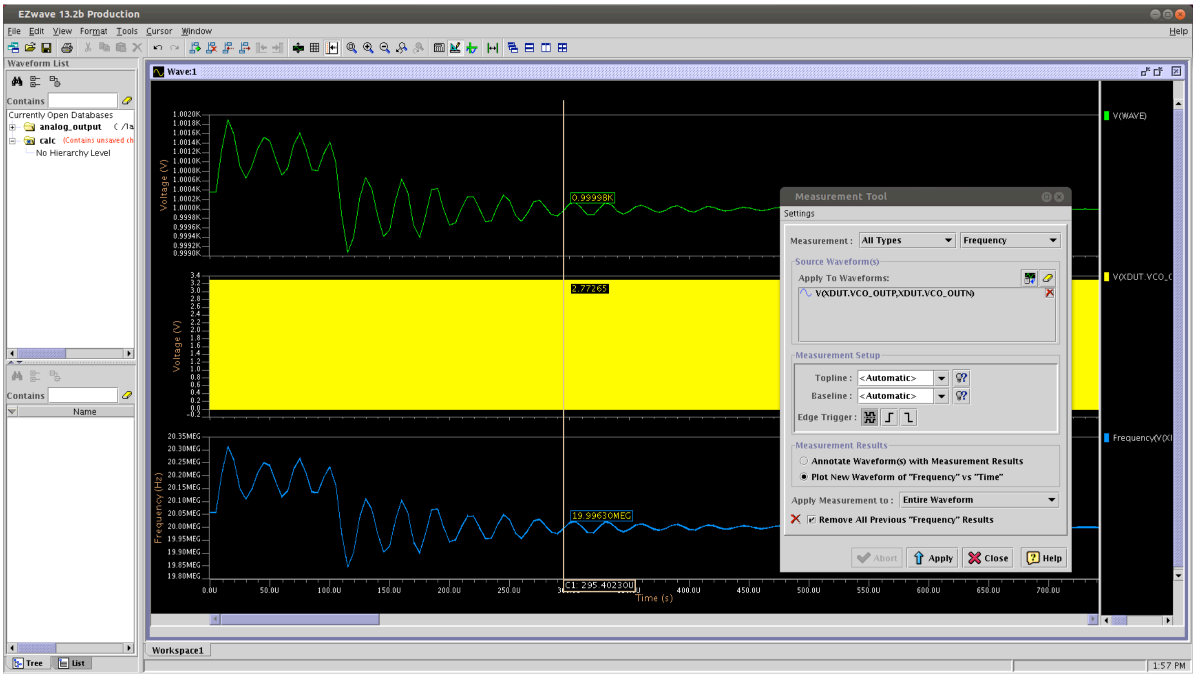
Depending on the type of the ROM and the readout circuit two types of readout circuits can be simulated.

- **Piezoresistive readout circuit**– the piezoresistive readout circuit consists of a voltage divider, an OPAMP and a VCO. In the test circuit the OPAMP and VCO are ideal macro components. They serve as a guide for designing the actual readout circuit. Simulation of this ideal circuit should be done by tuning the component parameters (OPAMP gain, VCO sensitivity, center frequency, etc.) to achieve good simulation results. If that is accomplished, the preparation of the actual design can be started with the tuned component parameters as specification.
- **Capacitive readout circuit** – the capacitive readout circuit consists of a sole VCO. The parameters should be tuned as in the piezoresistive case to achieve good simulation results. The tuned parameters can be used as a specification of the to-be-designed VCO circuit. In detail, the real VCO circuit will be a ring oscillator where a variable capacitance is inserted between two of the ring oscillator inverters. The variable capacitance will tune the output frequency of the VCO in concert with the ROM output capacitance variation.

Simulation output

- In both cases the simulation output will be the VCO’s output frequency depending on the variable piezoresistance or capacitance. The output frequency-variation should mimic the piezoresistance or capacitance variation. To verify that the output frequency changes the same way as the input variable, additional measurement can be executed in the EZWave waveform viewer.

  By displaying the frequency-time diagram of the VCO’s output signal, the relation of the variable input and the output frequency can be investigated. To achieve this, the **Measurement tool** should be used in EZWave.



Measurement Tool

Settings

Measurement : All Types ▾ Frequency ▾

Source Waveform(s)

Apply To Waveforms:

V(XDUT.VCO_OUTP,XDUT.VCO_OUTN)

Measurement Setup

Topline : <Automatic> ▾

Baseline : <Automatic> ▾

Edge Trigger :

Measurement Results

○ Annotate Waveform(s) with Measurement Results

● Plot New Waveform of "Frequency" vs "Time"

Apply Measurement to : Entire Waveform ▾

☑ Remove All Previous "Frequency" Results

Abort Apply Close Help

Figure 2 - Obtaining frequency data

5.3 Digital simulation

The digital simulation includes a whole test environment instantiating the behavioral model of the frequency measurement circuit and the serial communication interface (*Design Under Verification*, DUV). The behavioral model of the DUV is not intended to be used for synthesis but it is able to generate the appropriate waveforms according to the configuration of its parameters (measurement window, baud rate etc.).

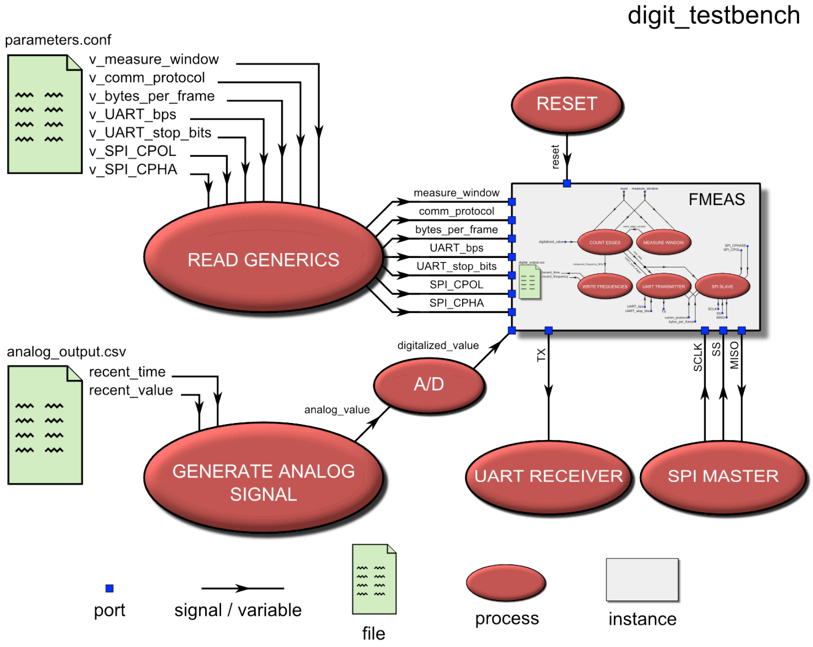
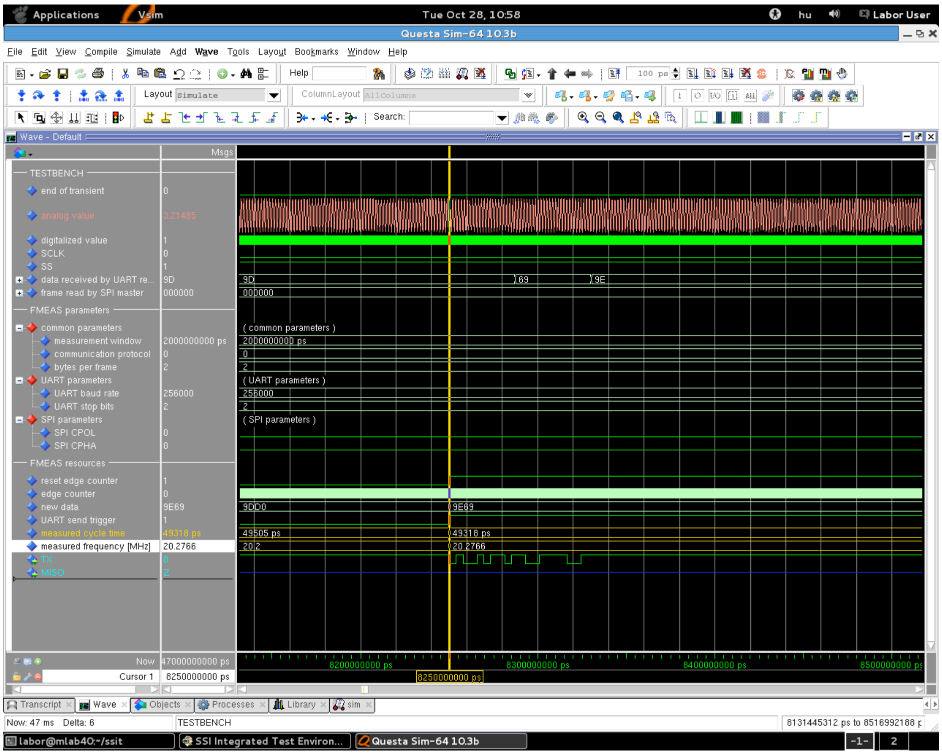


Figure 3 - Structure of the test environment

Components of the test environment (see Fig. 3):

- **L_READ_GENERICS**: A process reading the generic parameters passed through the SSIT graphical user interface.
- **L_GENERATE_ANALOG_SIGNAL**: A process reading the .csv file containing the transient response of the analog circuit.
- **L_AD**: A process converting the analog (sinusoidal) signal into a digital square signal. This process basically describes a comparator with a threshold voltage of 1.65 V.
- **L_FMEAS**: The instantiation of the DUV.
- **L_SPI_MASTER**: The DUV may include an SPI slave interface. The test environment includes an SPI slave module in order to make sure that the DUV generates valid SPI bit streams.
- **L_UART_RECEIVER**: The DUV may include a UART transmitter. The test environment includes a UART receiver module in order to make sure that the DUV generates valid UART bit streams.
- **L_RESET**: A process responsible for generating the global reset signal.

Once the generic parameters have been configured the simulation may be started. The simulation itself is performed by QuestaSim whose waveform window will appear after the simulation. A pre-defined .do file is loaded to ensure that the relevant signals can be observed (see Fig. 4).



	Msgs
TESTBENCH	
end of transient	0
analog value	3.21485
digitalized value	1
SCLK	0
SS	1
data received by UART re...	9D
frame read by SPI master	000000
FMEAS parameters	
common parameters	
measurement window	2000000000 ps
communication protocol	0
bytes per frame	2
UART parameters	
UART baud rate	256000
UART stop bits	2
SPI parameters	
SPI CPOL	0
SPI CPHA	0
FMEAS resources	
reset edge counter	1
edge counter	0
new data	9E69
UART send trigger	1
measured cycle time	49318 ps
measured frequency [MHz]	20.2766
TX	0
MISO	Z
Now	47000000000 ps
Cursor 1	8250000000 ps

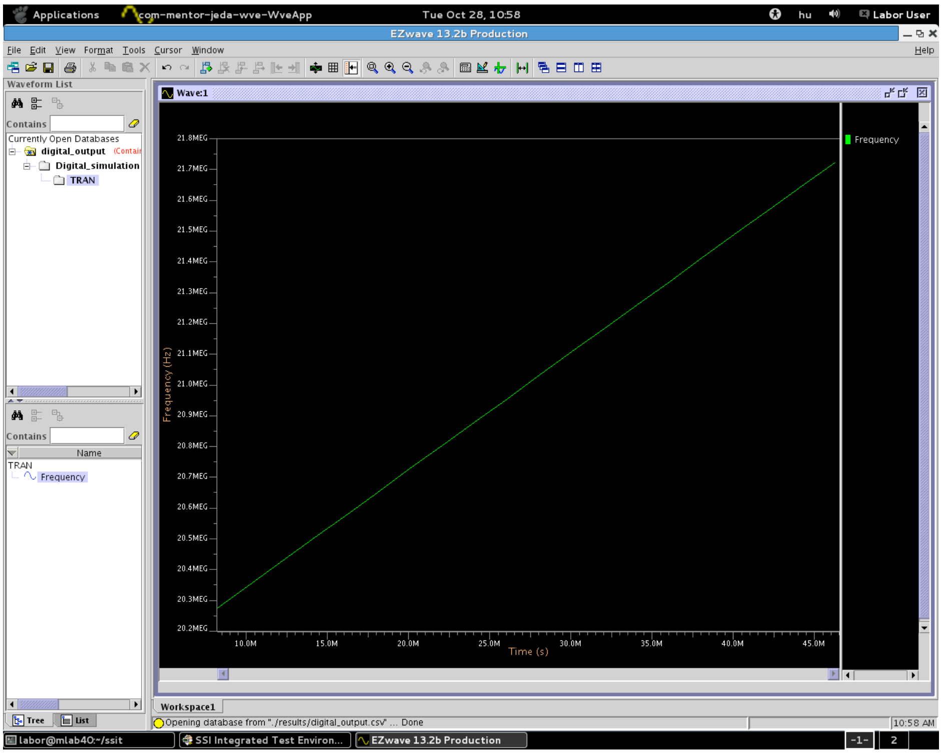
Now: 47 ms Delta: 6

TESTBENCH

8131445312 ps to 8516992188 ps

Figure 4 The wafeform window of QuestaSim

Although the DUV does not have to determine the frequency values themselves, the test environment produces them during the transient simulation. The measured frequencies are dumped into a .csv file which can be observed with EZWave (see Fig. 5).



Opening database from "./results/digital_output.csv" ... Done

Figure 5 - .csv file describing the measured frequencies plotted by EZWave