OFDM Tutorial

Introduction

Orthogonal Frequency-Division Multiplexing (OFDM) is a digital multi-carrier modulation scheme in which closely-spaced sub-carriers are summed into a main carrier. The sub-carriers are orthogonal to each other and modulated with conventional modulation schemes at a low symbol rate. The summation is performed using Fast Fourier Transforms.

OFDM is robust against both intersymbol interference and narrow-band co-channel interference. It is spectral-efficient. OFDM is used in applications such as WiMAX, MBWA, Wi-Fi, and UWB.

This tutorial demonstrates how to set up and run a simulation that uses the OFDM module. The exercise begins by demonstrating the setup for MATLAB, then continues by illustrating the setup for Spectre.

At the end of the tutorial, is a summary information about the terminals and parameters of the OFDM module.

Preparing the Database for the Tutorial

1. Copy the file to your directory.

```
`cds_root spectre`/tools/spectre/examples/SpectreRF_workshop/ofdm.tar.Z
```

The `cds_root spectre` command is one way to determine where the Spectre simulator is installed. If you already know the path to the Spectre hierarchy, you can just substitute it into the path given above.

Note: This tutorial database is intended for use with the MMSIM 6.1.2 release or later.

2. Unpack the file.

```
zcat ofdm.tar.Z | tar xvf –
```

Viewing the Setup for the MATLAB Part of the Cosimulation

3. In the working directory of the cosimulation package, type

```
cd ofdm/lab1
```

to move into the `lab1` directory.

4. Start MATLAB.

```
matlab &
```

The OFDM test case opens.

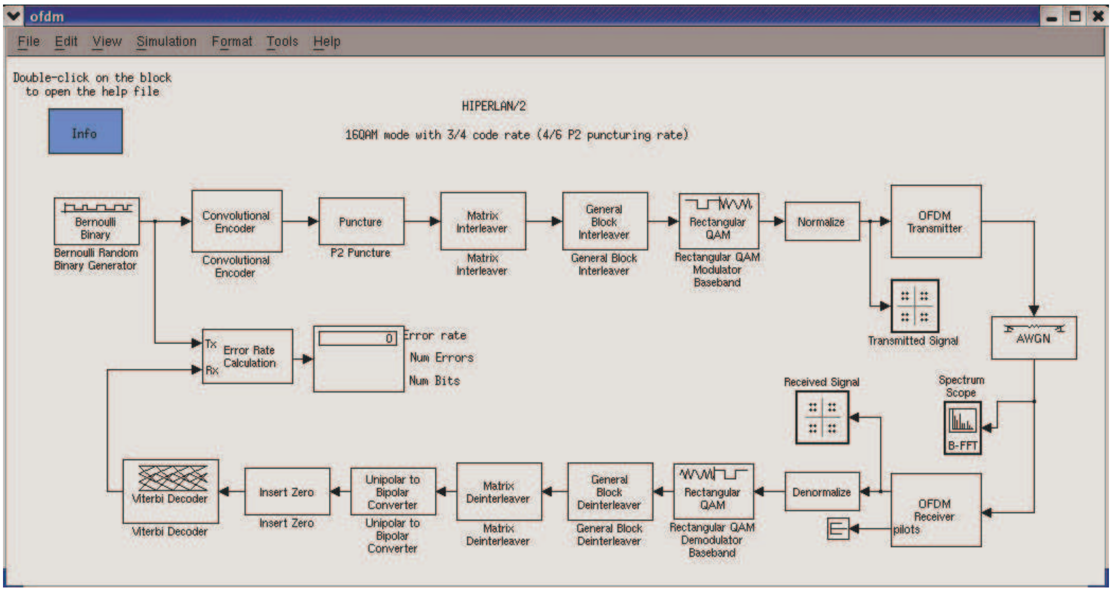

Figure 1: Schematic of test bench in MATLAB

This example uses the `hiperlan2` test bench.

5. Double click the `OFDM transmitter` block.

The OFDM Transmitter schematic appears.

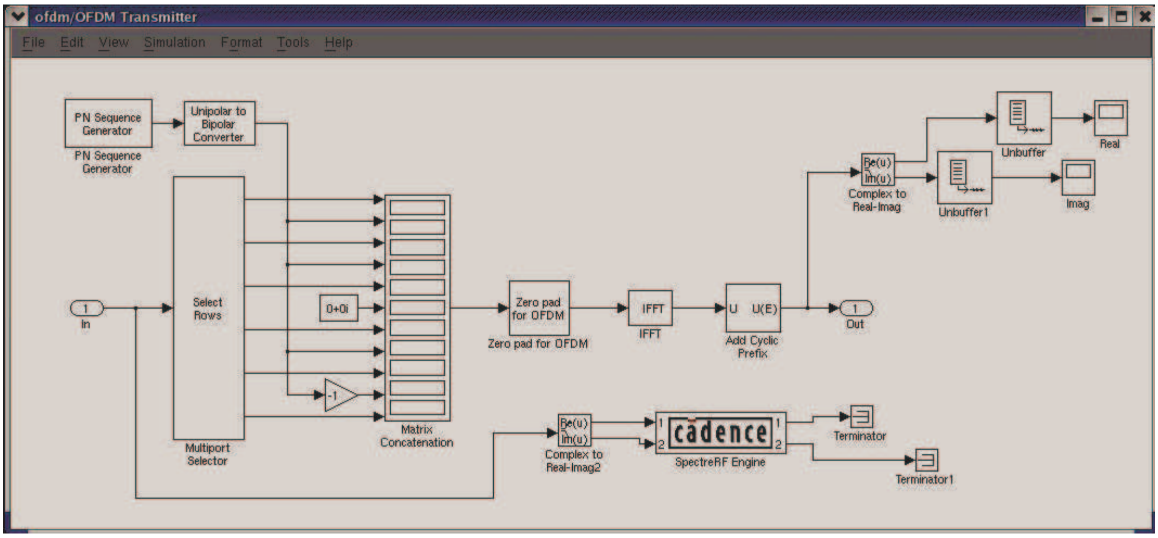

Figure 2: Schematic of SpectreRF Engine in the test bench

Notice that the `SpectreRF Engine` is attached, via `Complex to Real Imag2`, to the `In` port. The outputs of the `SpectreRF Engine` go first to unbuffer blocks, then to scopes. This path uses the `Spectre RF Engine` to generate the data of interest.

There is another path that does not use the `SpectreRF Engine`, generating data that can be compared to the data generated by the `SpectreRF Engine`. This second path is connected at the `Out` port and takes the data through `Complex to Real-Imag3`, through unbuffer blocks, and then to scope blocks.

Viewing the Setup for the Spectre RF Part of the Simulation

6. In the ofdm/OFDM Transmitter schematic, double-click the `SpectreRF Engine` block.

The Function Block Parameters: SpectreRF Engine form appears.

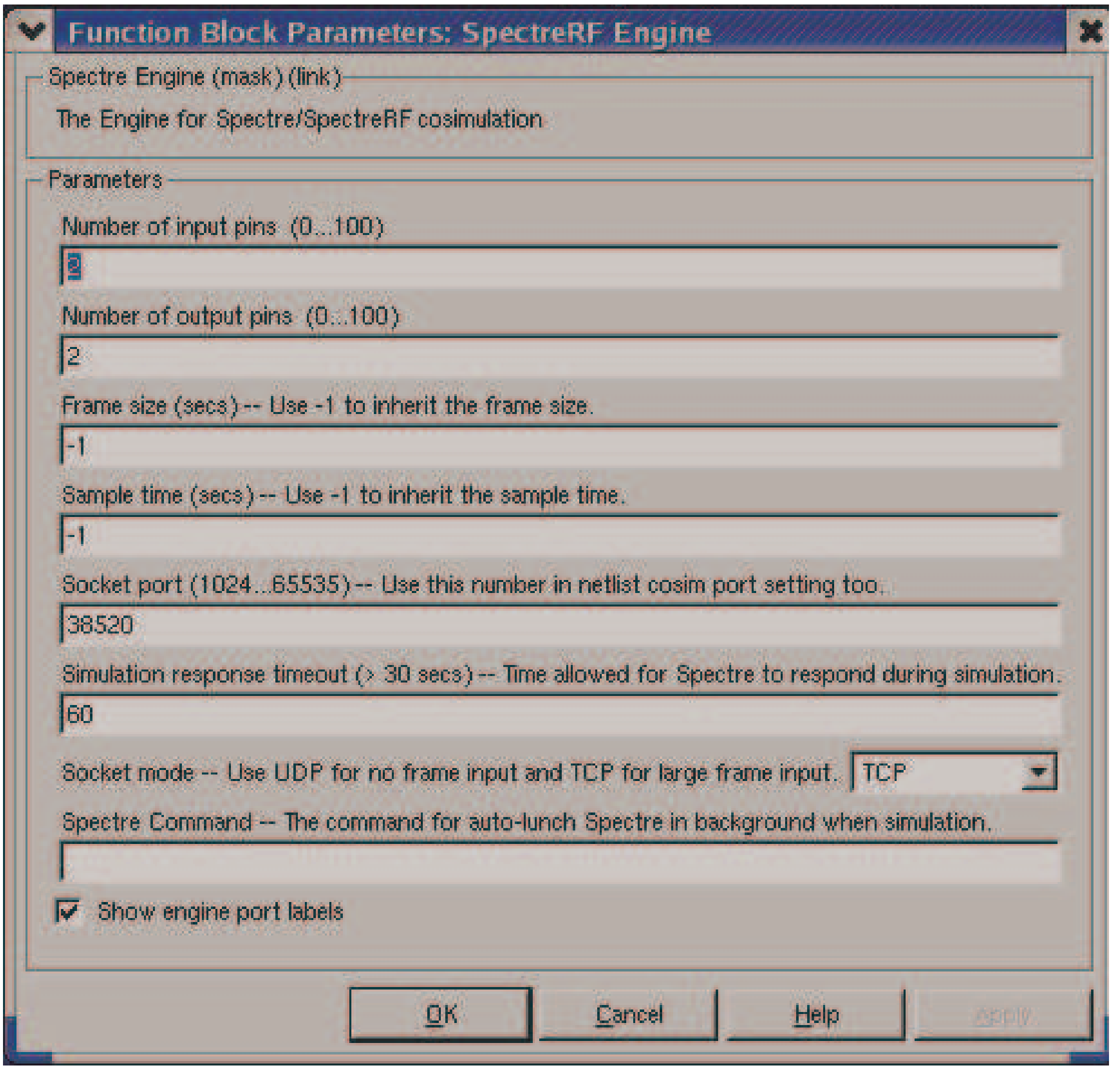

Figure 3: Setup for a Discrete-Time Eye Diagram Scope

7. Ensure that the values match those shown in Figure 3, then click *OK*.

   The form closes.

8. Examine Figure 4, which is the schematic for the Spectre RF circuit.

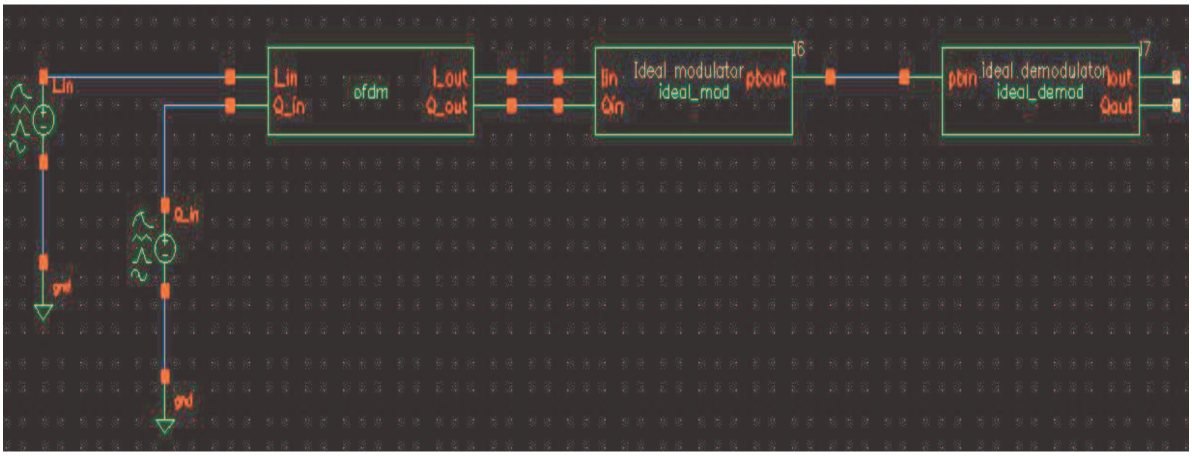

Figure 4: The schematic for Spectre RF in passband

This simple circuit is the source of the netlist used in this exercise, `ofdm_env.scs`. The circuit includes an instance of the Verilog-A OFDM module, which is described in more detail at the end of this tutorial.

In this schematic, the OFDM signal is up-converted by the carrier frequency and then down-converted into the baseband. As such, the outputs of the circuit are baseband signals. The circuit is working on the PASS band and the carrier frequency is 2.4G.

Simulating the Design

9. In the ofdm schematic window, choose *Simulation -- Start*.

This starts the MATLAB part of the simulation.

10. Then, quickly, in your terminal window, type

```
spectre ofdm_env.scs
```

This starts the Spectre part of the simulation. The MATLAB and Spectre tools work together to run the simulation.

Viewing the Results

11. After the simulation finishes, in the upper right-hand corner of the ofdm/OFDM Transmitter window, double-click the scope named `Real`.

12. When the plot appears, click the *Autoscale* button, which looks like a pair of binoculars.

The plot looks like this.

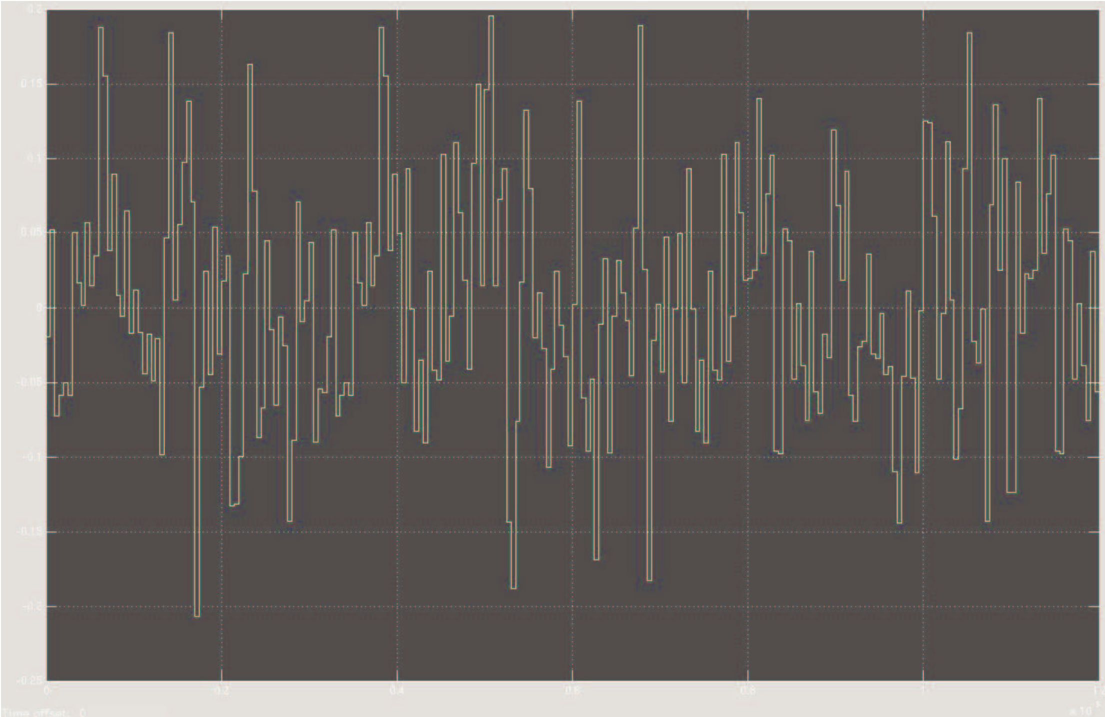

Figure 5: The real part of the baseband data from MATLAB OFDM

13. In your terminal window, type `wavescan &`.

The wavescan window appears.

14. Choose *Tools -- Results Browser*.

The Results Browser window appears.

15. Use the Results Browser to load `ofdm_env.raw`, then choose `envlp_td.envlp`, and finally double click the `neti` node.

The Spectre result appears.

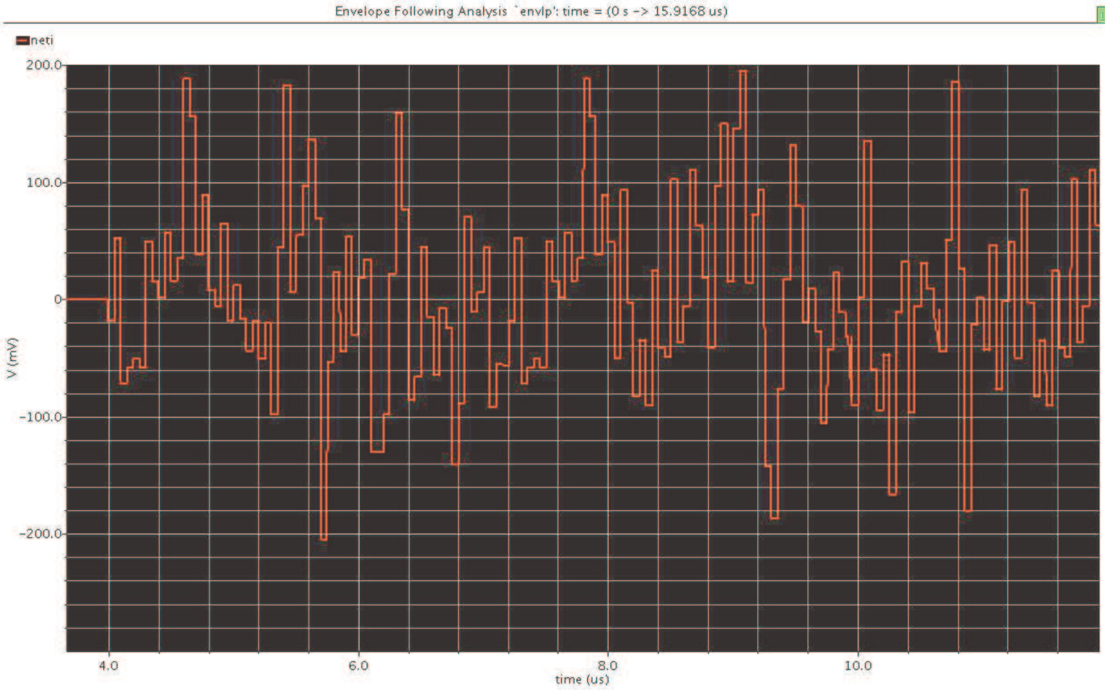

Figure 6: **The plot for neti**

The plots in Figure 6 and Figure 7 match, verifying that the OFDM module used by the `SpectreRF Engine` produces the same results as MATLAB produces.

16. (Optional) You can also verify that the imaginary part of the baseband matches also. Be aware, though, that the waveform in Spectre is delayed by one frame.

This ends the steps in the tutorial. The rest of this document describes the terminals and parameters of the OFDM module.

OFDM Module

Figure 7: **Symbol of the OFDM module**

Terminals:

`I_in, Q_in` [v] input signals, activated only when `input_enable` is set to 1.

`I_out, Q_out` [v] output signals, such as the OFDM baseband signals.

Instance parameters:

Name	Meaning	Type	Default Value	Range
frame_time	Frame time	real	4.0e-6	> 0
samples	Samples in one frame	integer	80	>= 1
Poly_length	The length of vector for shift register's feedback connections.	integer	8	>=1
Shift_length	The length of vector for the delay of PN sequence	Integer	1	>= 1
Poly_order	The order of polynomial function	Integer	7	>=1
Init_state_size	The length of vector of initial state	integer	7	>=1
poly	The array of polynomial function	Vector of integer	{1, 0, 0, 1, 0, 0, 0, 1}	[0, 1]
state	The array of initial state	Vector of integer	{1, 1, 1, 1, 1, 1, 1}	[0, 1]
shift	The array for delay of PN	Vector of integer	{0}	[0, 1]
Dump_frames	How many frames are skipped initially	integer	1	>0