ECE 546: Advanced Signal Integrity
High-Speed Serial Link (HSSL) Design Project
1. Overview

Technology scaling and unprecedented growth in demand for ubiquitous fast, robust computing. With the rise of heavy-duty data centers to handheld mobile devices, the desire for faster, low-power integrated inter-IC communication protocols is at an all-time high and has led the roadmap of the semiconductor industry, making it one of the fastest growing yet fiercely competitive industry. With the growing needs for ultra-low power yet multi-Gbps signaling in both wired as well as wireline applications, integrated systems on chip (SoCs) have become mainstream critical components in modern computing systems. Ability to process and access ‘big-data’ are the fundamental demands in modern society where every second saved in prompt communication as well as computation of information is critical. In order to meet these needs of fast, robust signaling over the same old "lossy" channels, the clock-frequencies need to scale accordingly, and clever I/O links need to be developed. Figure 1 shows a typical Serial Link that consists of a Transmitter (TX) with an **externally provided** TX CLK, channel, Receiver (RX) with a forwarded clock and equalization schemes.

![Figure 1: Basic Link High-level description](image)

In this design-project you (in a team of 2) will explore the signal integrity issues experienced by High-Speed Link designers on both a system level as well as the circuit level. Normally, a Clock and Data Recovery (CDR) circuitry is a part of the receiver to recover the clock signal from the data stream it receives. However, for simplicity, CDR is not in the scope of this study. The TX CLK signal will be provided to you and clk_deskew (RX clock, which is needed for the DFE circuit to work) can be implemented by the ideal delay cell found in **analogLib** library (clk_amplifier can also be omitted). Your task will be to design the channel in HFSS, implement the TX and RX at transistor level (you can put the homework together to have an inverter-based transceiver as the starting point or implement your favorite architecture), and the equalization block which can be implemented using VerilogAMS. We provide a simple 5-tap DFE behavioral model implemented in VerilogA to further simplify your task (see your homework). You can choose to implement single-ended or differential signaling.

The final integration, in principle, cannot take place in Virtuoso because it simply does not support Channel Simulation (remember that, for instance, if you want to simulate your channel to see if it meets the requirement BER of $10^{-16}$ you will have to run at least $10^{16}$ bits, this is prohibitive with transient simulator like Spectre or HSPICE which is the engine behind Virtuoso). For the sake of completeness and the goal to introduce you to how transceivers are implemented in transistors, you are still required to run a long transient simulation in Virtuoso. However, you will also be expected to run actual Channel Simulation in ADS (or PowerSI or Nexxim, whichever you’re familiar with) to obtain some statistical metrics on your high-speed link design.
2. Channel design requirement

A typical channel is shown in Figure 2. In this project, you are provided with a package design (certainly not the best but enough to challenge you), which will be described in detail later; you are required to use it for both TX and RX side.

![Figure 2: A simplified controller (TX) – memory (RX) interface, forming a channel](image)

The PCB trace is required to be at least 6 inches. PCB material is FR4_epoxy, you can design your own stack-up or you can use this as reference. You can either implement a microstrip line or a strip line configuration for the PCB trace. **But you must explicitly have ground plane(s) and power plane(s). You will have to provide power to your driver circuits via the power distribution network (PDN).** The starting point of your PDN could be at the package for simplicity. Figure 3 shows a system level setup for the final channel simulation. In configuration (a), each PDN and the channel are modeled (hence, their S-parameters are collected) separately while in configuration (b), they all modeled together so that all interaction between the channel and the PDN would be captured.

![Figure 3: Power–aware channel simulation](image)
3. Package details

This section is devoted to explaining how the package model was created to help you familiarize yourself with the provided model.

Figure 4 shows 1/8th of the package. Because the package and die are drawn the same way just at different scale, you will find 2 sets of variables starting with “p_...” representing package's variables and “d_...” representing die's variables.
The package (and die) has 3 different groups of vias, pads and solder balls. They are colored differently in Figure 4. Vias in a same column parallel to the x-axis (same color) are created by cloning the via at the bottom of the column, denoted as “1st generation”. The “2nd generation” vias are created, hence same color, along x-axis. Their location is defined by variables $p_{pin1x}, p_{pin1y}, p_{pin2x}, p_{pin2y}, p_{pin3x}, p_{pin3y}$. Moreover, the height of each via group can be controlled by $p_{h1}, p_{h2}, p_{h3}$. In addition, the via has multiple design parameters: pad radius, via radius, solder ball radius, notice that the solder ball radius needs to larger than the via radius. You should use the 3 groups of vias for signal, ground and power conduction.

The die landing pads are also drawn in groups, hence different colors. But they all should be placed on top of the package base. The die follows the same logic and was just a scaled version of the package (it has its own design parameter sets though). For your convenience, the package is drawn in RelativeCS1 coordinate system while the die is drawn in RelativeCS2. You can find it under “Coordinate Systems” tree. Changing the coordinates of the origin of RelativeCS1 and RelativeCS2 will change where the package and die are placed.

![Relative Coordinate System 1 to draw the die](image)

You will have to draw traces yourself. It is recommended that you use the Line drawing tool.

![Line drawing tool](image)
You can draw a polyline, add many segments as you want with multiple clicks and end the sequence by a double click, then extend the line into a 3D objects with your choice of width and thickness, the process is shown in Figure 7.
3. Design Specifications

The following design specifications should be met by your design:

1. Package size is not larger than 15x15 mm².
2. Die size is not larger than 1x1mm².
3. Supply \( V_{\text{Supply}} \leq 1.8V \)
4. Receiver eye height \( \geq 600mV \), receiver eye width \( \geq 0.8UI \)
5. PCB trace(s) length \( \geq 6 \) inches

4. Analysis

To validate your design, you will need to perform the following analyses:

1. **Timing Budget**: Calculate all contributions to timing uncertainty and compute the net timing margin.
2. **Noise Budget**: Calculate all contributions to noise and interference and compute the net noise-margin, SNR induced on the channel.
3. **Channel Design and Modeling**: The channel connects chip through package to board as shown in Figure 2. Define appropriate ports and generate S-parameters using HFSS. Describe the process of optimizing your channel design. An external optimization method is strongly recommended and will be rewarded with extra credits. Include appropriate type (single-ended or mixed-mode) of return loss and insertion loss of your channel and characterize it using a simulated TDR response. Run a transient simulation in ADS to find the bit response of the channel, calculate cursor’s values. Additionally, simulate your timing and signaling conventions on this channel using a PRBS excitation to generate an eye-diagram. This step should be done in ADS using ChannelSim and generic TX, RX, like the setup in Homework 7. Run the simulation with at least 1 million bits. Both TX and RX eye diagram, timing and voltage bathtub curve needs to be included in your report.
4. **Circuit Simulation**: Perform transistor-level simulation of your design using Cadence Virtuoso using the NCSU 180nm PDK provided in the class server or any other PDK you have access to through your research group. Presenting test benches for each part of your design to earn partial credit if your final design does not work as expected. Simulate the whole design in Virtuoso (transient simulation) with as many bits as you can (pay attention to time constraint, ChannelSim type of simulation can be fast for a large number of bits, but transient simulation, especially at transistor-level, will take a very long time). Show the effect of equalization on improving the eye opening. Verify the result with ADS ChannelSim.
5. **Power-aware channel simulation**: Compare the eye diagram in 2 cases: with and without PDN. In the case PDN is ignore, your transceiver should be powered directly with an ideal DC voltage, same as your homework, while in the case PDN is considered for, the power supply is fed through a PDN. You can perform this analysis either with your Virtuoso flow from above for you can just simply use ChannelSim in ADS.

Extra credits are given for the one or more of following additional implementations and analyses:

1. **Channel operating margin (COM)** analysis of your channel.
2. **Peak-distortion analysis** of your channel, and **worst-case eye diagram generation**.
3. Implementing the given behavioral DFE in NCSU PDK at transistors level.
4. **Implementing a behavioral CDR** to generate RX clock instead of manually delaying TX clock as suggested at the beginning.
5. **Jitter** extraction at RX, identify the source, discuss possible fix.
5. Conclusion

1. Why is a specialized simulator needed for Channel Simulation? Hint: assuming that the time requires
to simulate the response of 1 bit is 1ms (you must have seen how long it actually took to simulate 1
bit when you found the bit response of your channel in homework 10), for modern communication
system, a BER of $10^{-16}$ or lower is often required. That would mean at least $10^{16}$ bits needed
simulating. How long it would take for such large number of bits to be simulated?
2. Show the bit response of your channel and the process to identify the tap values for DFE
equalization? Show data on plot to support your answer.
3. Comment on the effects of power integrity on signal integrity using the results obtained when
simulating the channel with and without the PDN.

6. Report Requirement

Your report should show your design and optimization process. List out your design parameters, show
relevant schematics, plots, data to support your statements and to validate your design. It should reflect all
the required analyses mentioned in part 4 and the answers/comments to questions in part 5.

You are required to use IEEE format for your report. You can find more information about IEEE template in
this Quora answer and here if you use Overleaf

There is no page requirement and limit to your report. Submit your report in .pdf format along with
your design files, including S-parameter of your channel to Compass2g.

7. Recommended readings

1. Introduction to BGA package
2. Intel's Packaging Databook – Chapter 14
3. Intel's PDN design methodology
4. Qualcomm's tutorial on PDN
5. Eye diagram and Signal Integrity 1 & 2
6. Channel Simulation Methodology 1 & 2 Video1 & 2
7. jitter1, jitter2 and jitter decomposition