Advanced Digital Design [LU] – WS2022

Design I

Florian Huemer & Andreas Steininger

Department of Computer Engineering, TU Wien

Vienna, November 10, 2022

1 Metastability Measurement

For the first part create a simple VHDL design containing just a single D flip flop. Connect its clock input to the main clock source of the FPGA board (50 MHz on input pin `Y2`) and its data input to an independent (uncorrelated) clock source. For the latter use a function generator and connect it via the SMA connector J26 (see Figure 2.1).

Your task is to measure the metastability behavior of the flip flop using the setup shown in Figure 1.1. Notice that for meaningful results the flip flop's clock input signal as well as its data output are required. Thus both have to be connected to some output pin of the FPGA (use the SMA connectors J25 and J27).

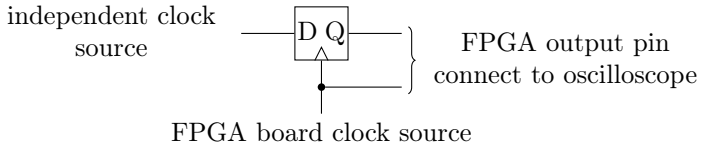

Figure 1.1: Metastability measurement setup

Connect these outputs to the 8 GHz Infiniium High Performance Oscilloscope in the rack and configure it to trigger on either the rising or the falling edge of the data output (Q) of the flip flop. Make sure to activate infinite signal persistency: In this operation mode the oscilloscope records subsequent signals traces and draws them on top of each other. Show the plot in your slides and determine the maximum clock to output delay!

Activate the histogram recording feature of the oscilloscope to extract the occurence of the different clock to output delays. Again show the data in your presentation and use it to estimate the metastability parameter τ of the flip flop.

2 Clock Domain Crossing

For the second part open the provided quartus project (`make quartus_gui` in folder quartus; pinout in figure 2.1), synthesize the design and download it to the DE2-115 FPGA board. Do not forget to adjust the timing assumptions (.sdc file).

The design contains a data source and a data sink that can be clocked independently via the top-level inputs `clk_src` and `clk_dst`. The source produces a new data word (12 bit) every clock cycle (`data_src`); the 8 most significant bits alternate between the values "0x55" and "0xAA" and the 4 least significant bits implement a counter that increments by one with each step. The receiver's task is to capture the data word using its own clock and make it available at port `data_dst`. A two-phase encoded request line is used to indicate valid new data (signal `req`).

Connect a clock source with a fixed frequency of 60 MHz to the receiver's clock input `clk_dst`. For the sender' clock `clk_src` use a 60 MHz clock source as well, but modulate it by ± 1 MHz with a modulation frequency of 100 Hz using a triangular modulation waveform.

The required pulse generators are available in the lab (rack). If you are not aware how they have to be set up for the desired waveform take a look at the user manual available on the web. For the modulated clock use the Arbitrary Waveform Generator from Agilent; for the fixed clock you may also use the clock source on the FPGA board or the other function generator. To control the waveform/function generators over USB we provide you with two Python scripts in the folder `tools/`. Their respective names indicate which device they can be used for (Agilent33250A.py, Agilent81130A.py). The file `tools/README.md` contains a brief description of the command line interface.

Be careful with the clock generator impedance settings: check whether the FPGA has a 50 Ω input (usually not) and set the generator impedance / output voltage accordingly. If you are not sure what you are doing, ask a supervisor or tutor **before** starting the signal generator.

Use the logic analyzer to observe the behavior of the circuit. You will notice two error sources: missing flow control (skipped and doubly received datawords) and missing synchronization (wrong/inconsistent data). Show an example for each in your presentation.

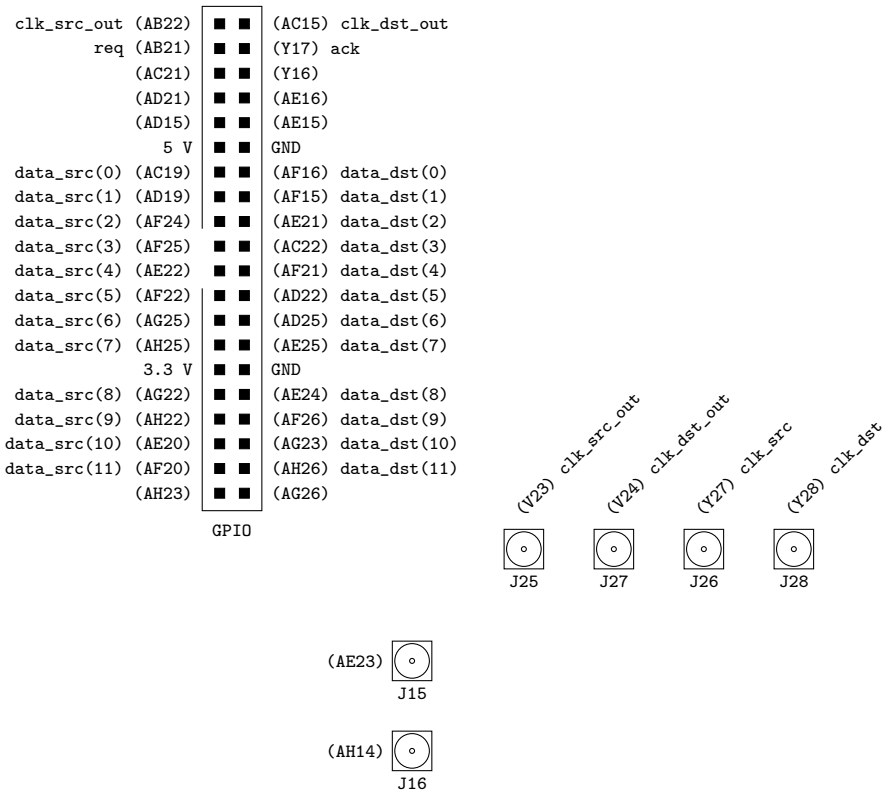

Figure 2.1: FPGA board pinout (The GPIO connector and the SMA sockets J15 and J16 are located on the FPGA board itself while the SMA sockets J25-J28 are located on the daughter board) . The identifier in parenthesis denotes the name of the I/O pin of the FPGA as used in the Pin Planner in Quartus.

2.1 Flow Control

Your task is to remove the first error source due to missing flow control. More specifically you are supposed to alter the data source such that new data values are sent in the correct order and only after the data has been consumed by the sink. Document what problems you encountered and how you fixed them in your presentation. Find and display an event where an erroneous data word is received (screenshot ⇒ slides). Furthermore, observe the rate of erroneous data words at the receiver and compute the MTBU (⇒ measurement concept & result ⇒ slides) as well as the error probability (⇒ slides) for a transmission.

2.2 Synchronization

Change the design to also tackle the second error source and ultimately improve the MTBU to a value of at least 1×10^6 years. For the presentation, prepare a block diagram and sketch the key concept behind your solution. Can you sustain the data rate? If not determine the new maximum data rate (both throughput and min./max./avg. latency ⇒ slides).