# Elektrischer Aufbau der Schnittstellen

Julian Hatzky

August 10, 2019

# Contents

# Part I

# Introduction

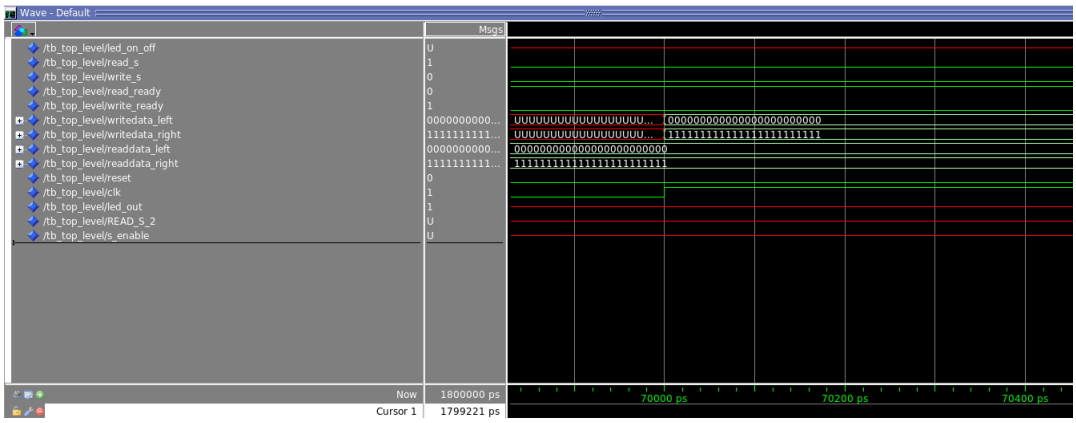
In this document the specific routing of the important signals is described in detail. The reader should get a feeling of timings, states and of the overall behaviour of the FPGA architecture. However the author highly recommends to use the Altera Modelsim simulation software to do the exact timing analysis in order to get the fully understanding of the ongoing processes. Note: Because the AudioCodec is well documented IP of Altera, only the copy-machine, which was developed in this project, is described in detail.

# Part II

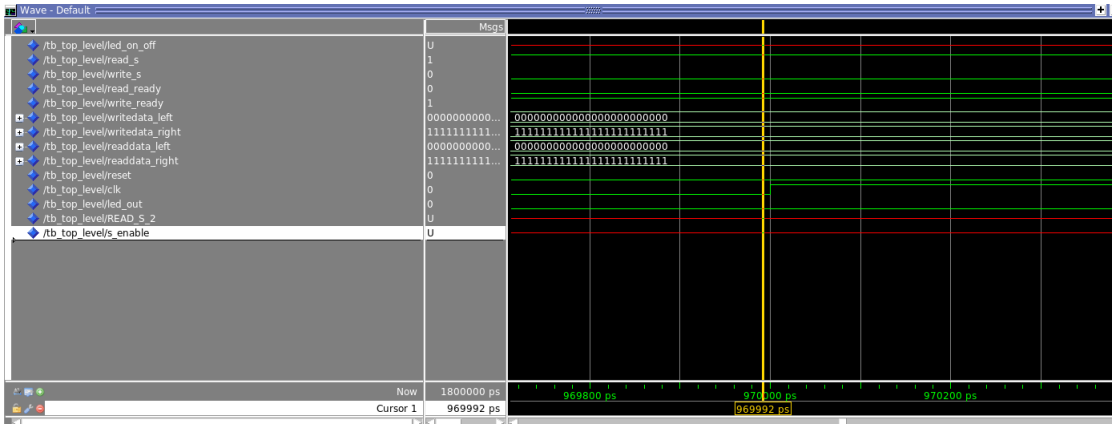
# Top-Level signal routing

The following snippets made from a Model-Sim simulation should point out the most important and interesting signals and timings when looking at the top-level architecture. As already mentioned the focus lies on the copy-machines behaviour. In this section we look at the copy-machine as a black box and point all its relevant signals and time stamps. The copy-machine's most relevant signals are the output of audio data, in other words how much time is needed to process the audio data from input to output? And also LED-out signal that answers the question on how much time it needs to detect the signal. The following two grafics are giving explanations.

Note: Unfortunitely it is necessary to zoom in on most of the pictures.




When zooming in it can be seen that after 70 nanocesonds (7000 ps) the input audio data is forwarded to the output. If you would run the architecture on the FPGA itself, that would be roughly the time spot when you would start hearing sounds out of a speaker adapted to the FPGA.

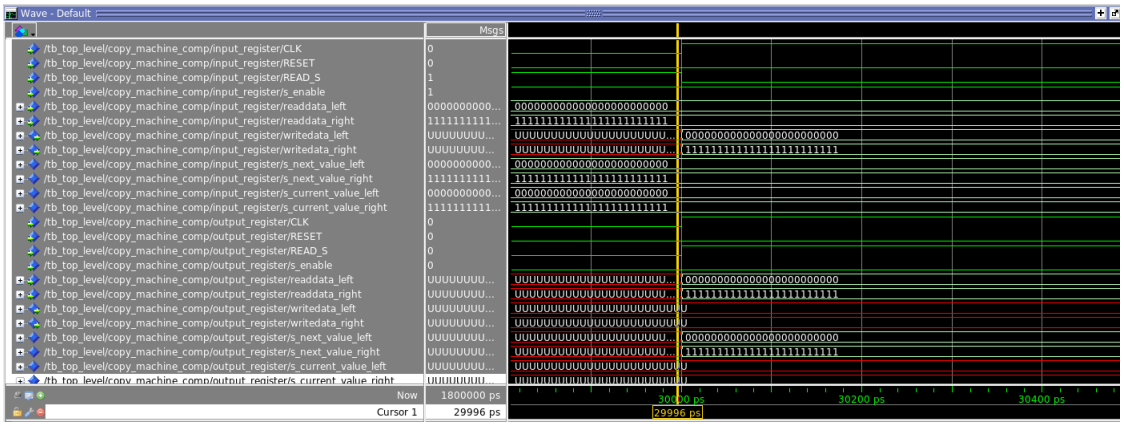



The other important change is when the input signal is fully processed and the LED gets switched on the first time. The procedure needs 970 ns (970000ps) that can be seen as the detection time the architecture needs to classify input signals.

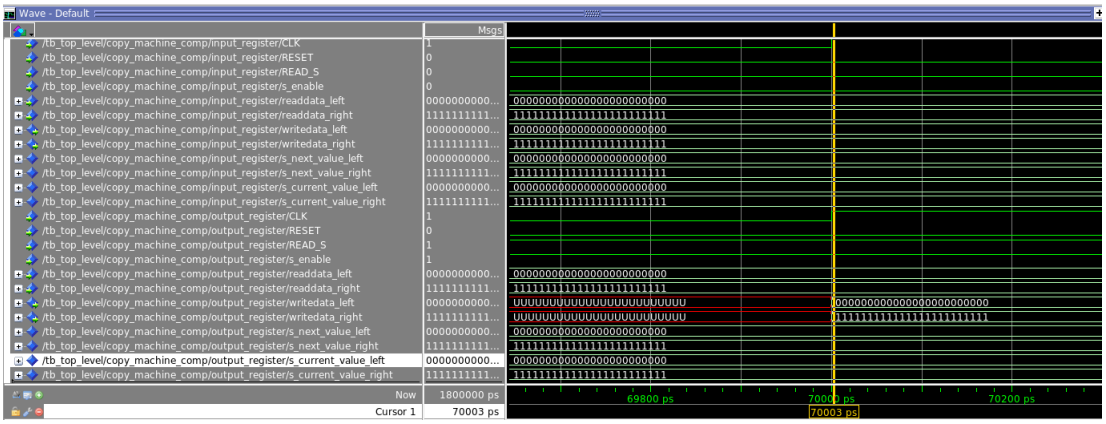
# Part III

# Buffer-in and -out components signal routing

This section shows up the relevant signals of the buffer-in and buffer-out components. Basically the buffer-in component is the first layer buffer to store the incoming Audio Data. Like buffer-out it works as a simple FIFO buffer. From buffer-in the data goes on the one hand directly to buffer-out, so that the audio-signal is forwarded to the AudioCodecs output and also to the ring-buffer that stores more samples for a proper detection.



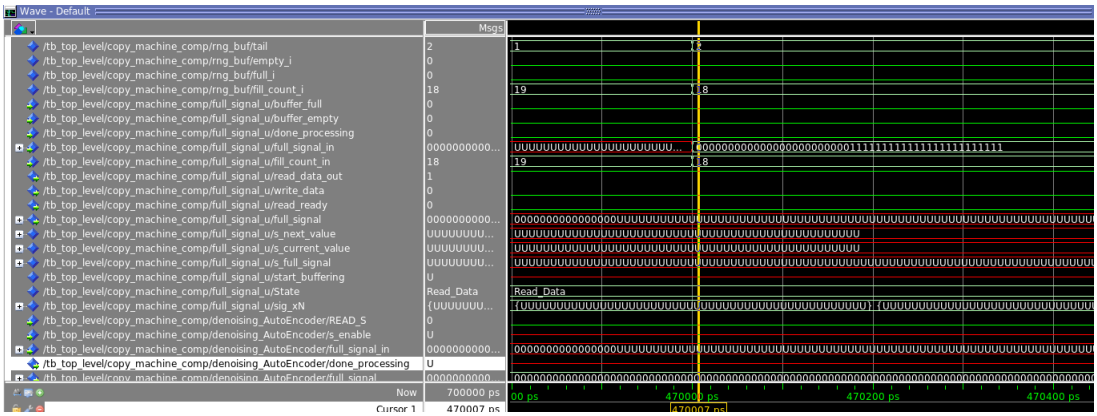

The first relevant timestamp is the duration it needs for the output-buffer to receive the information from the input buffer. The diagram shows 30ns (30000ps). It then needs roughly 40ns more until the signal reaches the output port of the out-buffer (picture below).



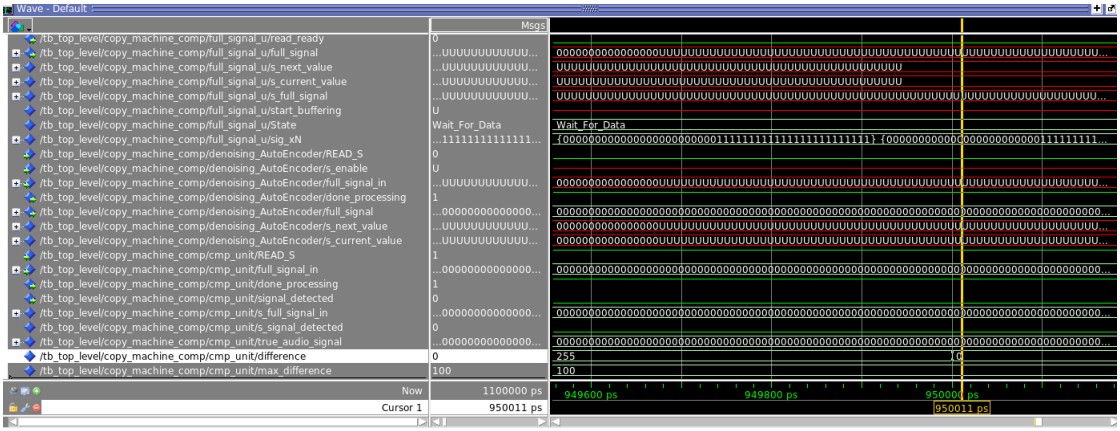
# Part IV

# Detection cycle signal routing

In this section the relevant signals and timings of the detection cycle are pointed out.




During the first clock cycles the the ring buffer gets filled with data. The maximum RAM storage is 20. That means 20 vectors of 48 bits each. When the storage reaches it's limit the full-signal-in component starts to read in the data. Because of the RAM architecture the data gets fetched field by field. The fill-count-in signal gets counted down to show that a reading process is happening. When there is a writing process it gets counted upwards. However the data in the RAM does not get cleared during the reading process. You can also see the State signal that is in the Read-Data state after the ring-buffer contains 20 elements. The signal in unit receives the whole signal.




Roughly 950ns are needed until the compare-unit calculates a difference of zero, hence the simulated input signal is exactly the same as the ideal signal that is hard-coded.(picture above)

The picture below shows the moment the LED gets the high signal, after 970 ns. That is the moment when a whole detection cycle is over, however the state already shifted back to wait-for-data so another fetching cycle is already in progress.