**Interim Report**

September 2022

Alex Barnwell

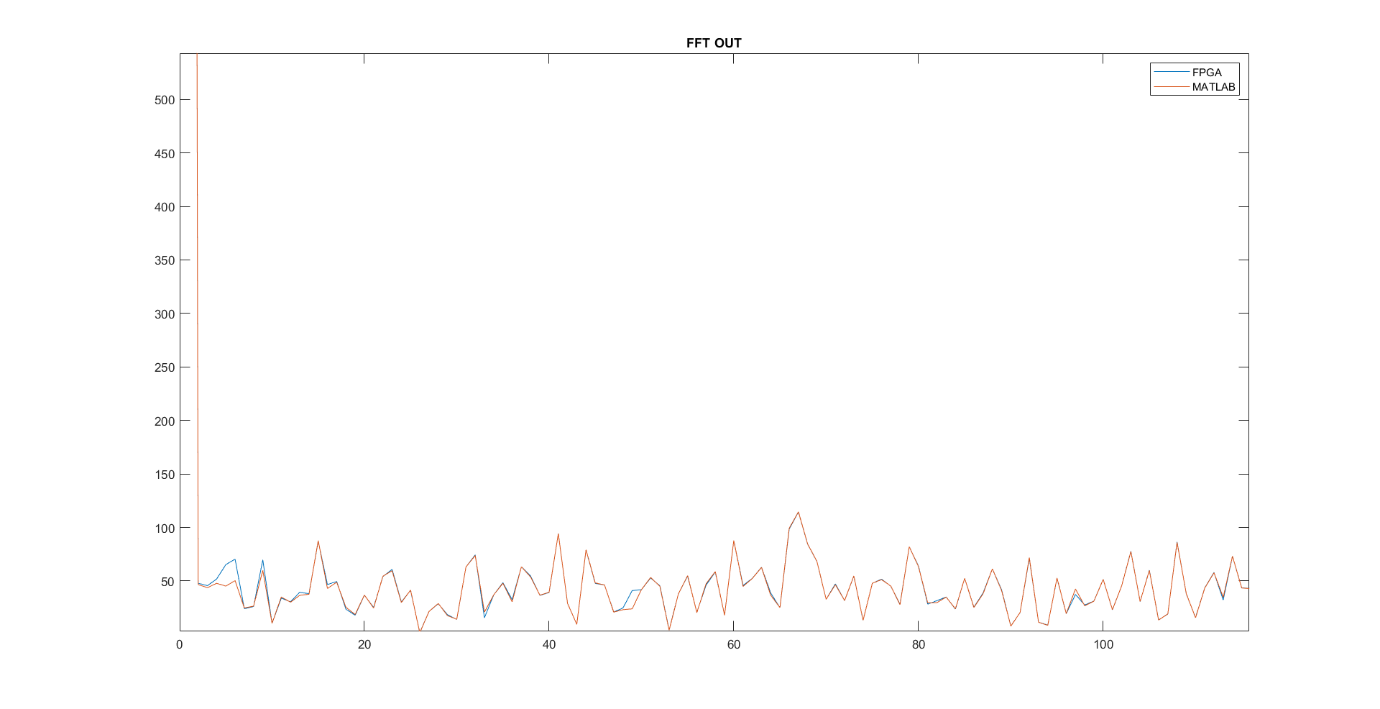
Completed the main design of the algorithm on the FPGA. It has been confirmed in simulation and produces the expected result. The clock rate is able to be increased to 138MHZ which allows the use of just two parallel DSP sections while having the microphone clock running at 4MHz, 4 sections in parallel will be needed if a faster microphone clock speed is desired.

However, it has been identified that as the implementation of the algorithm requires 512 (for an 8192-bit input) multiplicative loops to compute any given FFT bank there is error present. This error is due to the cumulative nature of the innate error from the twiddle factors being stored as 18 bits. This storage arrangement was chosen as the limit for the input size into the DSP block for the second multiplicative channel is 18 bits and thus no more precision can be attainted. The discovery of the culprit of the output error being strongly explained by the twiddle factor precision is that by running the MATLAB model with the twiddle factors at 18bits of precision yield results closer to the output from the FPGA.

To reduce this error the main proposed solutions would be to increase the precision of the twiddle factor This is impossible with the current limitations of the DSP so a redesign would be needed. Secondly to reduce the number of loops required, to do this the radix would have to be increased this however will increase the size of the DFTBD RAMs which would increase Block Ram usage so may not be desirable. To implement an increase in the radix would be possible however at this stage of the project would take to long.

The MATLAB compared to FPGA model shown below.

Chart

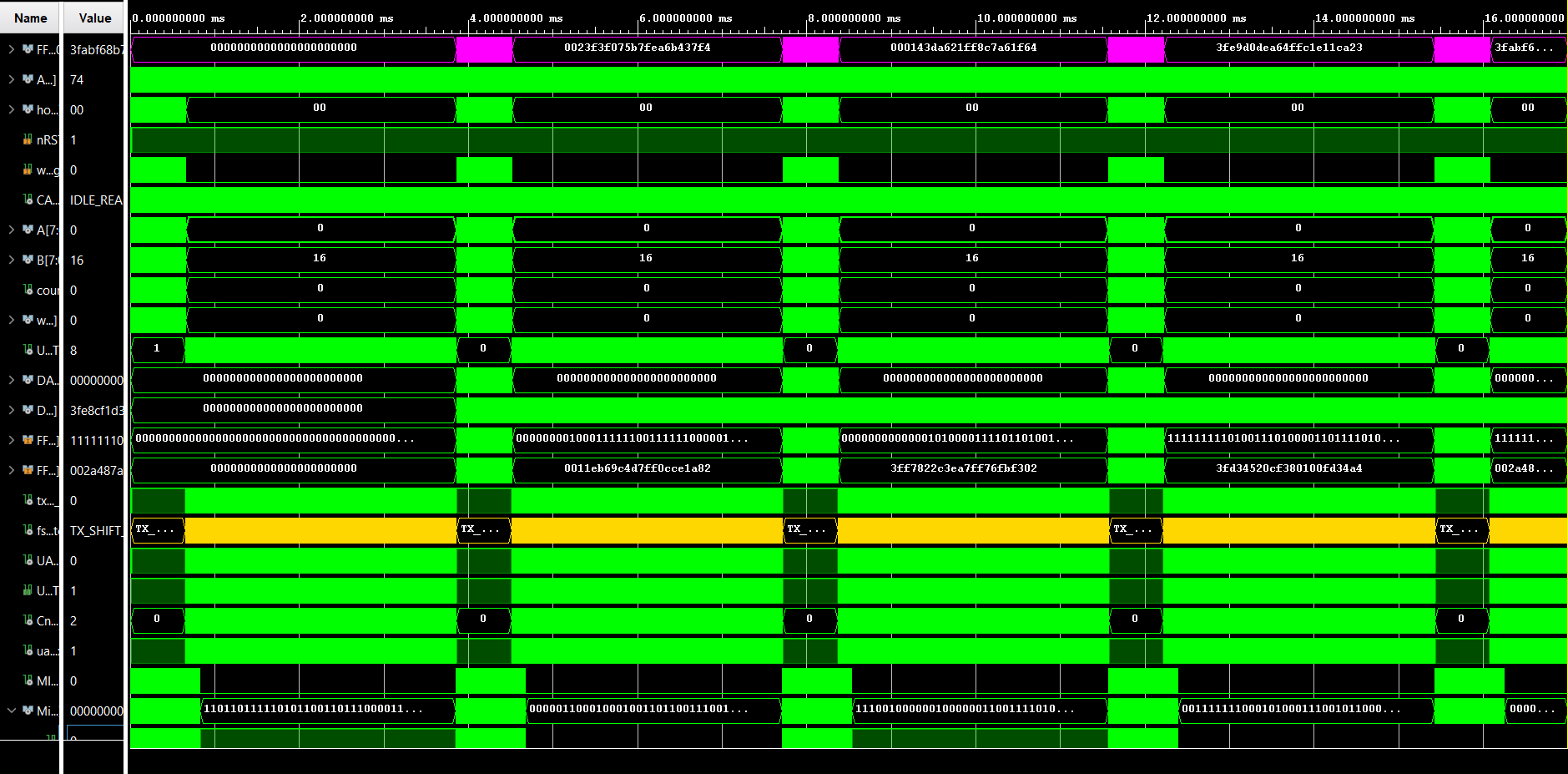
Description automatically generated

Wher the upper one plots the FPGA output against the MATLAB model when the Twiddle precision is the same, the lower one is when the MATLAB s full precision.

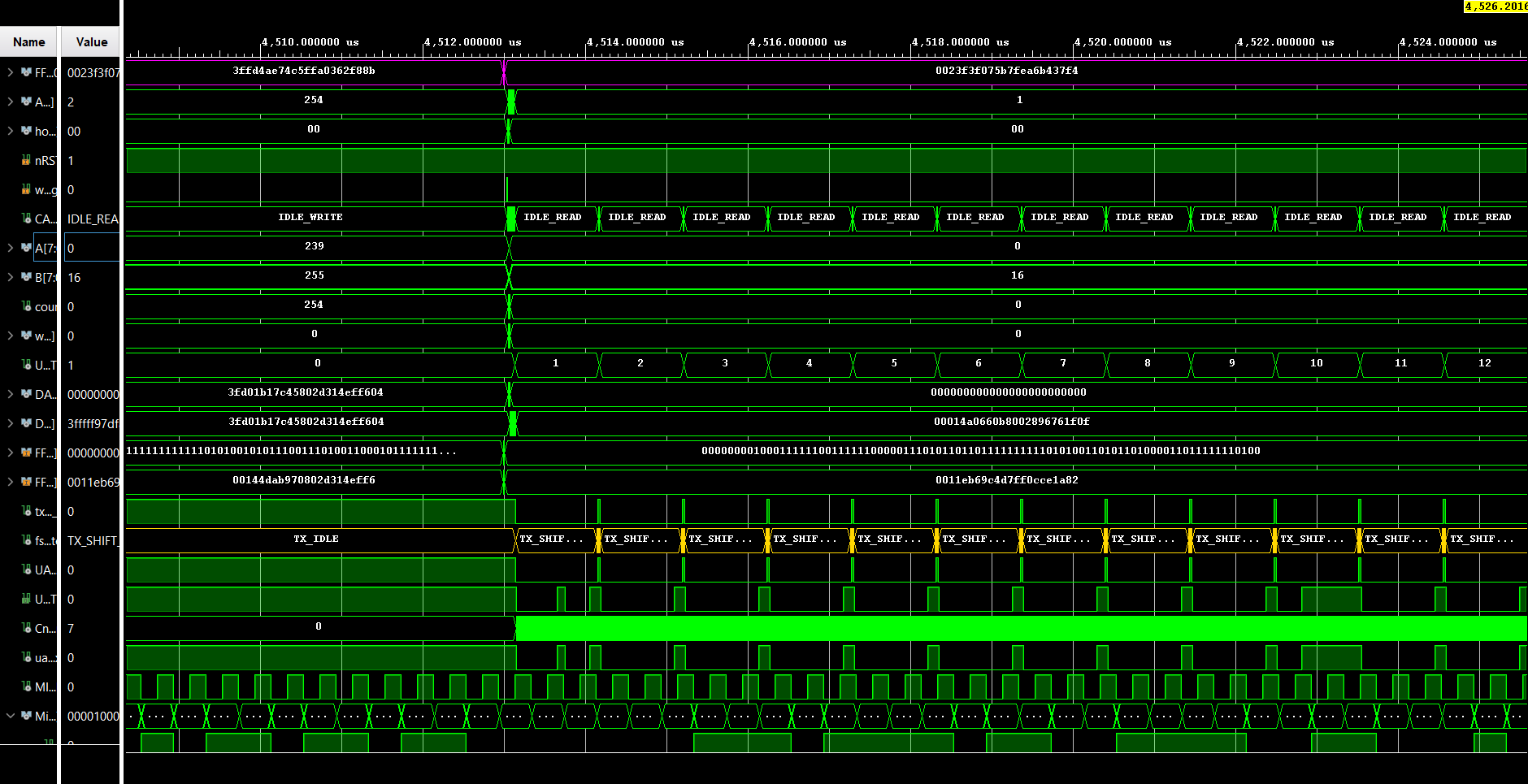
To Further verify the model a physical implementation is being conducted. This implementation involves using a microcontroller to mimic the microphone through its SPI interface and transmit know ‘microphone’ inputs. The FFT is then computed on these inputs and the result transmitted through UART to a computer. Once at computer the resulting serial stream will be processed to pull out the given FFT which will be analysed against the predicted MATLAB result.

So far, the UART system has been implanted in the FPGA so only the SPI interface n the microcontroller remains.

The simulation output of 5 cycles of the FFT with the UART transmission where the yellow is the UART start and the purple the aggregate form of the two FFT banks most recently computed



A close up of the UART states



Also, as the data need to be reorganised before the UART can begin transmitting the FFT data was rearranged and aggregated such that the real and imaginary inputs where concatenated with the order of magnitude such that each entry into the RAM was the entirety of a given FFT bank. The storage arrangement of this RAM also rearranged the order of the FFT banks such that by incrementing the address a new FFT bank will be read out in order of the right to left plotting of the FFT (from 0 to Fs to -Fs to 0). The simulation showing the state machine to operate this is shown below.

Where the purple is the most recent FFT bank aggregate output the yellow is the current sate of the UART transmitter and the blue is the state of the RAM handler.

