Portable NMR Spectroscopy with Direct Sampling

BASc. Thesis

Kevin Kim | 1004915149

Supervisor: Professor Roman Genov

Division of Engineering Science
UNIVERSITY OF TORONTO

Abstract

This thesis presents an analog receiver (Rx) and a deep length Fast Fourier Transform (FFT) hardware implemented in a field programmable gate array (FPGA) for portable nuclear magnetic resonance (NMR) spectroscopy. The system leverages modern advances in high frequency analog-to-digital converters (ADC) to simplify the analog Rx while sampling at a high frequency to reduce distortion and noise introduced to the digital domain. The analog front-end consists of an input buffer, a 3rd order Deliyannis filter, and a voltage matching level shifter as the Rx. To increase resolution of the analysis, a deep-length mixer-based zoom FFT technique was utilized and implemented on the Xilinx RFSoC 2x2 board. With the FFT sample length of 65536 and decimation factor of 2048, an FFT resolution of $\Delta f_{min} = 0.06\,ppm$ was achieved. The sampling frequency was $f_s = 4.096\ GHz$, and target Lamor Frequency $f_L = 500\ MHz$.

Table of Contents

Table of Figures

1. Introduction

Nuclear Magnetic Resonance (NMR) Spectroscopy is a chemical analysis tool that is used ubiquitously in many fields such as chemistry, biomedicine [1], and food safety [2]. Despite its utility, traditional NMR machines are bulky and require high maintenance which limits its use to large facilities [3]. Hence there has been an emerging body of work related to the study of portable NMR spectrometers for use in cases such as in-vivo biomedical studies [4] or on-field autonomous monitoring systems [5].

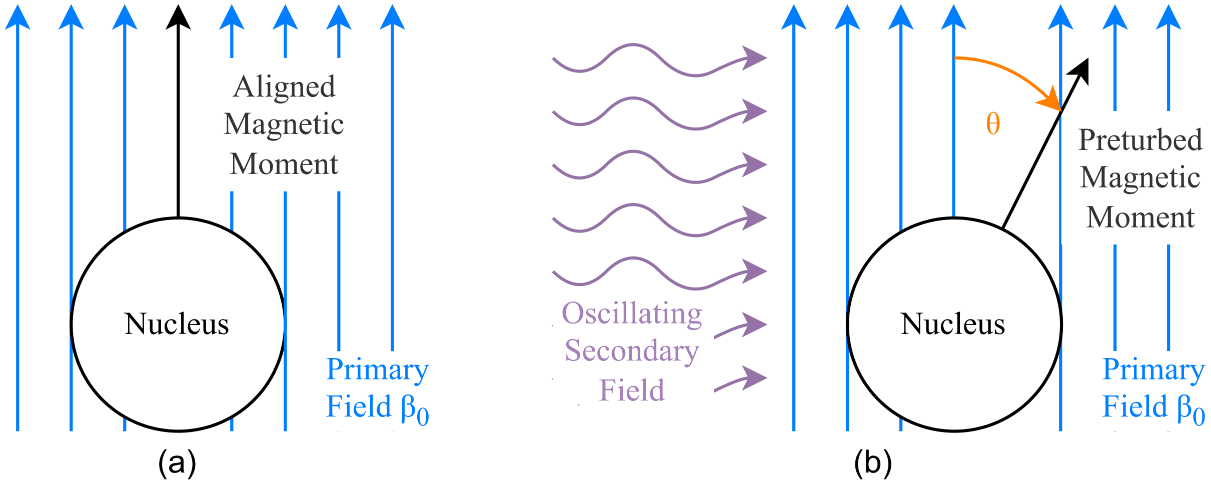


Fig. 1.
(a): Nucleus under a β_0 field. Magnetic moment is aligned with β_0 field.
(b): Nucleus influenced by β_0 and a second field. Magnetic moment is perturbed from the β_0 by θ [6].

NMR spectroscopy is performed by first applying a static magnetic field β_0 to a given sample. This field aligns the spin of nuclei, as shown in Fig. 1 (a), such that their magnetic moments are orientated parallel to the applied field. Then, as shown in Fig. 1 (b), a secondary oscillating field is established perpendicularly to β_0 that perturb the magnetic moment away from β_0, increasing the potential energy of the nuclei. When this secondary field is removed, the nuclei undergo a 'relaxation' in which it realigns with β_0 and releases the potential energy as a signal according to the relation:

$$|E| = hf = \mu\beta_0 Cos(\theta) \tag{1}$$

E: energy released, h: Plank's constant, f: frequency of released signal, μ: magnetic moment, θ: angle of perturbation [6].

This released NMR signal is highly characteristic and can be used to determine the nature of the bonds that exist in the sample. In a frequency spectrogram, they appear as 'peaks' that are 'shifted' from the spectrometer's Larmor frequency f_L, a reference frequency that is a function of the strength of β_0 and the geomagnetic ratio of the nucleus. These shifts are referred to as 'chemical shifts' and are commonly represented in units of parts-per-million (ppm) [6].

Parts-per-million effectively normalizes the NMR spectrum between different NMR spectrometers, accounting for the differences in β_0 and the resulting differences in f_L. It is defined as the ratio of the shift Δf away from f_L over f_L itself, multiplied by a factor of 10^6.

$$ppm = \frac{\Delta f}{f_L} * 10^6 \tag{2}$$

A portable spectrometer with a small β_0 will have a lower f_L. As a result, the shift Δf for a given ppm will also be lower comparatively to spectrometers with a strong β_0 field.

This relationship between f_L, β_0, and ppm constrains existing NMR designs. Using a stronger β_0 magnetic field leads to larger Δf shifts for a given ppm, reducing the requirement on the spectral analysis to detect fine grain Δf changes, while still maintaining ppm accuracy. However, it equally increases f_L, increasing demands on the receiver to process high-frequency signals, in addition to reducing the portability of the system due to the size of the magnet.

Existing portable systems thereby compromise by sacrificing accuracy of analysis. They use a weaker β_0 for portability, capture the signal at a lower f_L and increasing the requirements for accurate fine-grain spectral analysis. Existing systems are further constrained by slow analog-to-digital converters (ADC) that operate below the Nyquist Criterion for the target f_L. They instead rely on complex Receiver (Rx) designs using methods such as I-Q demodulation to allow sampling of the signal at lower frequencies. Increasing Rx complexity will inevitably introduce noise and distortion that would limit the accuracy of the fine grain spectral analysis.

This thesis aims to simplify the Rx by leveraging advances in modern high frequency ADCs. The goal of the system would be to maximize the ability for fine-grain spectral analysis such that high-accuracy NMR spectroscopy can still be performed with low β_0 fields. This will be achieved by maintaining the integrity of the NMR signal through a simple Rx and direct sampling the output, minimizing introduction noise and distortion from complex analog processing techniques. The signal will then be analyzed in the digital domain with a deep, Mixer-Based Zoom Fast Fourier Transform (FFT) for a high-resolution analysis of the sample signal capable of distinguishing peaks in the orders of 5 ppm. For the purposes of this thesis, the system will target a high β_0 and f_L of 500 MHz for testing and verification purposes, as it is easier to achieve a fine-grain ppm analysis at high frequencies. The intent is to reduce the β_0 field in future works.

2. Background and Motivation

Existing portable NMR Systems [3] [5] [7] [8] all broadly share four defined components: Tx – Transmitter of the oscillating field, Rx – Receiver that captures the signal, a timing controller for synchronizing the Rx and Tx, and a form of digital signal processing (DSP) unit responsible for data analysis. For this proposal, the focus is on the Rx and the digital DSP.

D.Ha et.al. [3] is a significant paper in the scalable NMR body of work with over 80 citations. They aimed to create a scalable NMR system on a single Integrated Chip (IC). They utilized a field with β_0 = 0.51 T and the corresponding f_L = 21.8 MHz. The receiver consists of a low noise amplifier (LNA) followed by an I-Q demodulator (IQD), which allows them to use a low frequency 14-bit 20 kHz ADC. As their work's focus was mainly concerned with the transceiver and pulse sequencer, the ADC was not present on the IC but was connected externally.

D.Ariando et.al. [5] is the only existing implementation that did not utilize an I-Q demodulation technique for the analog receiver. They have utilized a common-gate low noise preamplifier followed by a second stage 4th order tunable gain amplifier. A common-gate band pass preamplifier was opted as the first stage over a narrow band pass filter (NBPF) for simplicity, as they found in their previous work that the NBPF was difficult to tune in practice. Both the center frequency and the gain of this receiver can be controlled by the FPGA with the gain having a range of 40-66 dB. After the two-stage pre-amplifier the signal is directly sampled via 14-bit 25 MHz ADC (LTC1746 Analog Devices). This was possible as this system has a relatively weak β_0 of only 0.101 T and hence an f_L of 3 to 5 MHz. As a result, the ADC was able to satisfy the Nyquist criteria. Finally, with respect to digital processing, Ariando et.al. utilizes an Intel Cyclone V FPGA programmed via HDL for control of the analog components such as the receiver and the transmitter. It has the capability to temporarily store 1.28 s of the measured data at 25 MHz but does not process the data on the FPGA. Instead, it then transfers the data to a dual-core ARM Cortex-A9 running Ubuntu which performs digital post-processing via high level languages like python.

H.Burkle et.al. [7] focused on attempting to increase the voltage compliance of the NMR coil such that it can be enlarged to accommodate larger samples with the goal of benefiting from the improved concentration and volume of the sample. The IC design focused mostly on the high-voltage transmitter and thereby does not have an ADC included in the IC. The receiver is composed of an LNA, followed by an IQD. It has a significantly larger β_0 of 1.1 T with f_L of 48 MHz.

S.Hong et.al. [8] in their paper aimed to target the problems with timing issues occurring between Tx and Rx, their synchronization and Rx settling times. Their β_0 and f_L is similar to that of [3] at 0.5 T and 21.07 MHz respectively. The receiver consisted of an LNA, followed by an IQ demodulator, a second order low pass filter (LPF) tunable to 150-650 kHz, then a variable gain amplifier (VGA) tunable to 0-60 dB. The ADC utilized is a 11-bit 2 MHz ADC communicating with an FPGA.

Tab. 1: State-of-Art comparison of NMR receiver systems and the proposed system

Authors	**D.Ha et al. [3]**	**D. Ariando et al. [5]**	**H.Burkle et al. [7]**	**S.Hong et al. [8]**	**This Work**
Journal	PNAS	Journ. Mag. Res.	IEEE	ESSCIRC	IEEE Tran.Circ.Sys.
Year	2014	2018	2021	2021	-
Primary Mag. Field (β_0)	0.5 T	0.101 T	1.1 T	0.5 T	-
Larmor Frequency (f_L)	21.8 MHz	3-5 MHz	48 MHz	21.07 MHz	500 MHz
Instrument Type	Spectroscopy Relaxometry	Relaxometry	Spectroscopy	Relaxometry	Spectroscopy
Receiver Type	I-Q demod.	Direct sampling	I-Q demod.	I-Q demod.	Direct sampling
Receiver Config	LNA→IPQ	LNA→Gain Amp	LNF → IQD	IQD→LPF→VGA	NBPF→ Gain Amp
ADC Specification	• 14-bit 20 kHz • LT1407A	• 14-bit 25 MHz • LTC1746	-	• 11-bit 2 MHz	• 12-bit 4.096 GHz
Digital platform	-	• Intel Cyclone V FPGA • 1.28 s of data	-	Digilent FPGA	Xilinx RFSoC 2x2 FPGA [9]
Post-processing	-	• ARM Cortex A9	-	-	-

Tab. 1 summarizes the state-of-art systems under study. None of the state-of-art portable NMR systems operate at f_L close to this thesis's target of 500 MHz. D.Ariando et.al [5] reports the only system which has utilized direct sampling to simplify Rx, but the target f_L is a very low 3-5 MHz making it only fit for macro level analysis. All other examined systems target f_L=20-50 MHz and use IQ Demodulation to reduce the sampling frequency significantly.

In contrast, this thesis will utilize modern 12-bit 4.096 GHz ADCs on the Xilinx RFSoC 2x2 FPGA [9] which exceeds the Nyquist criteria for the target f_L of 500 MHz by 4x. To avoid intermodulation and distortion that can occur when transmitting high frequency signals, a pre-ADC Filter is proposed as the analog front-end between the NMR Rx Coil and the ADCs of the FPGA as seen in Fig. 2.

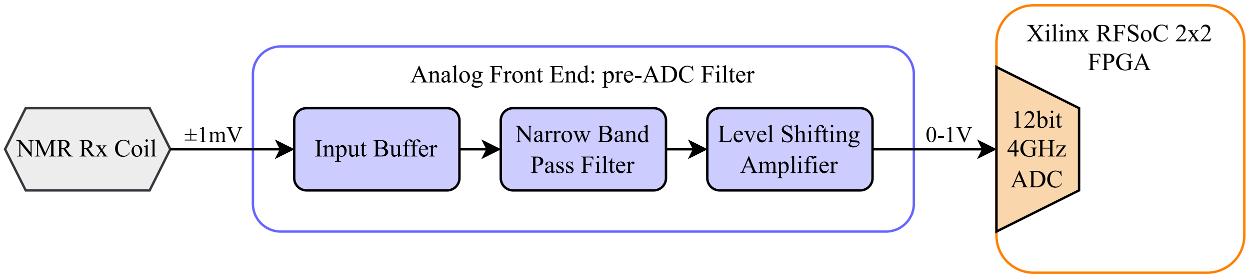


Fig. 2. High level block diagram of proposed analog front-end.
Front end provides band-pass filtering and ADC input voltage matching.

This front-end's goal would be to provide a simple layer of band-pass filtering to attenuate from irrelevant frequencies to reduce intermodulation, in addition to providing gain and adjusting the DC level of the signal to match the input voltages of the Xilinx RFSoC 2x2 Board.

This thesis also proposes a digital back-end implemented on the FPGA to process the sample data at runtime. This back-end will aim for a deep length FFT to achieve fine-grain resolution of the spectrum to be able to distinguish peaks that exist in the order of 5 ppm.

3. Analog Front-End Design

3a. Input Buffer

To protect the integrity of the output of the NMR coil, an input buffer is required as the first stage of the analog front-end. Without this buffer, the coil would effectively become a part of the front-end circuit, and its output behavior may distort significantly due to interactions and loading with other components of the circuit. Hence, a common method of preserving the output of the coil is by a buffer like a unity gain amplifier.

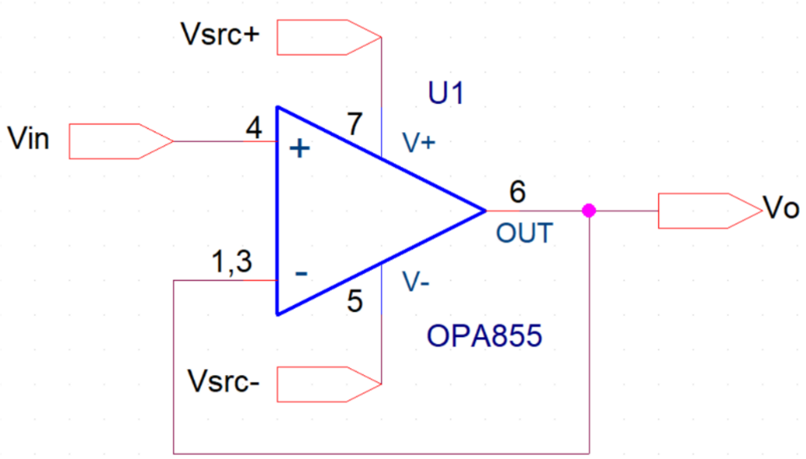


Fig. 3. Circuit diagram of the unity gain amplifier.

The output of this amplifier follows the input while providing a very high impedance to the NMR coils, conveying the signal while isolating the coil from the rest of the circuit. This will ensure that later stages of the front-end do not overload the signal from the NMR coil.

The Texas Instruments OPA855 transimpedance amplifier [10] was chosen for all stages of this front-end. With an 8 GHz gain bandwidth (GBW) product, which is 16x greater than the target f_L=500 MHz, is sufficient to meet the requirements of this research.

3b. NBPF Filter

One of the key considerations in designing a band pass filter is the Q factor that describes how narrow the bandwidth is where higher numbers indicate a narrower width. To completely reduce noise outside of the desired range, a greedy solution to this problem would be to design a very narrow band to completely attenuate any irrelevant frequencies. For the target range of 5 ppm with

f_L of 500MHz, the Q factor of the filter, a metric of how narrow the bandwidth is, would need to be extremely high:

$$5\ ppm = \frac{\Delta f}{f_L} * 10^6 \rightarrow \Delta f = 2.5\ kHz \tag{3}$$

$$Q = \frac{f_c}{BW} = \frac{500\ MHz}{2 * 2.5\ kHz} = 100\ k \tag{4}$$

Where Δf is the shift in frequency from f_L at a given ppm, f_c and BW is the desired center frequency and bandwidth of the desired band pass filter respectively.

This greedy approach is unrealistic in implementation, involving NBPF of extremely high orders which would introduce significant complexity to the circuit. Hence, a heuristic approach was taken. The pre-ADC filter would aim for a much lower and realistic Q factor, not aiming to eliminate all noise outside of relevant frequencies but aiming to reduce intermodulation and distortion from the non-relevant components into the ADC. The required narrower band widths will then be processed in the digital domain, leveraging the direct-sample approach of this thesis.

A Deliyannis Filter (DF) was ultimately chosen as the NBPF of choice due to the ease of its implementation with only resistive and capacitive elements that are less sensitive in high-frequency environments compared to LC circuits. Its downside is the relatively low Q factor, which would have to be compensated for by increasing the order of the filter.

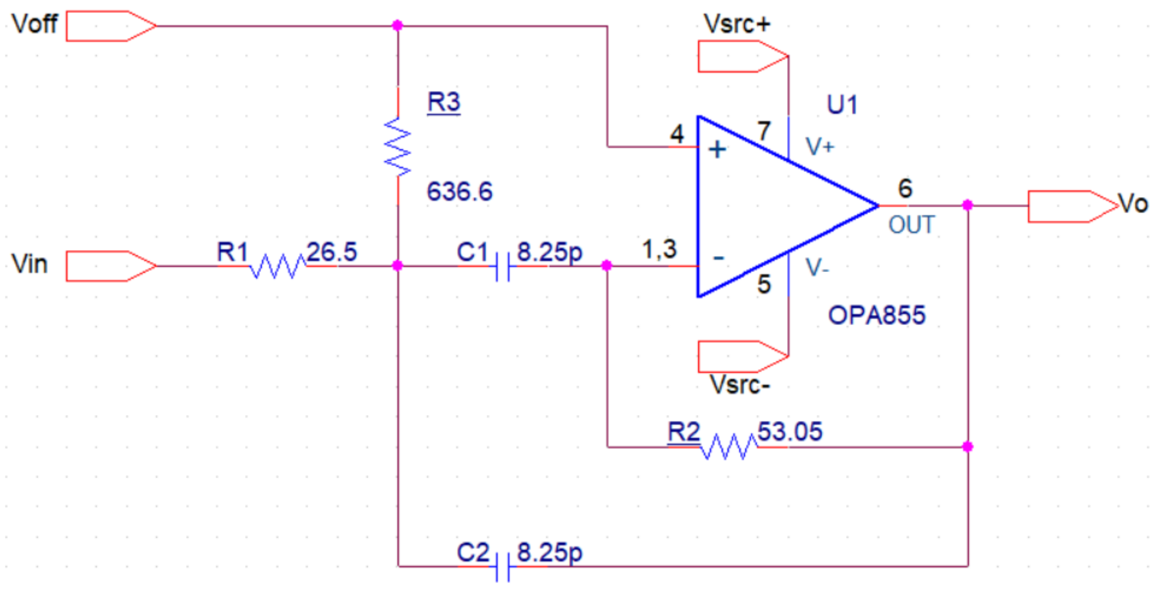


Fig. 4. Circuit diagram of the Deliyannis filter.

The equations for the Deliyannis Filters were adopted from [11]:

$$R_1 = \frac{Q}{G*2\pi f*C}, R_2 = \frac{Q}{(2Q^2-G)*2\pi f*C}, R_3 = \frac{Q}{\pi f*C} \tag{5}$$

The following target parameters were used to obtain initial values for the components: Q = 3, $f_c = 500$ MHz, $C = 3\,pF$, gain (G) = 12. There was a significant iterative process involved in the computation of these values. The equations are empirical and may not accurately model the filters at high frequencies and with the chosen high-frequency OPA855 op-amps. Hence, C had to be chosen through simulations as 8.25 pF to attain the desired f_c.

Due to inaccuracy, the simulated Q and G values were 0.71 and 0.46 dB respectively despite the target Q and G above as shown in Fig. 5. The simulated $f_c = 517.6$ MHz.

To compensate for the low-Q, the filters were the repeated 3 times as to increase Q to 1.53 and G to 2.02 dB, with $f_c = 511.7$ MHz as shown in Fig. 6. Further orders of the filter were not added to reduce complexity in the system.

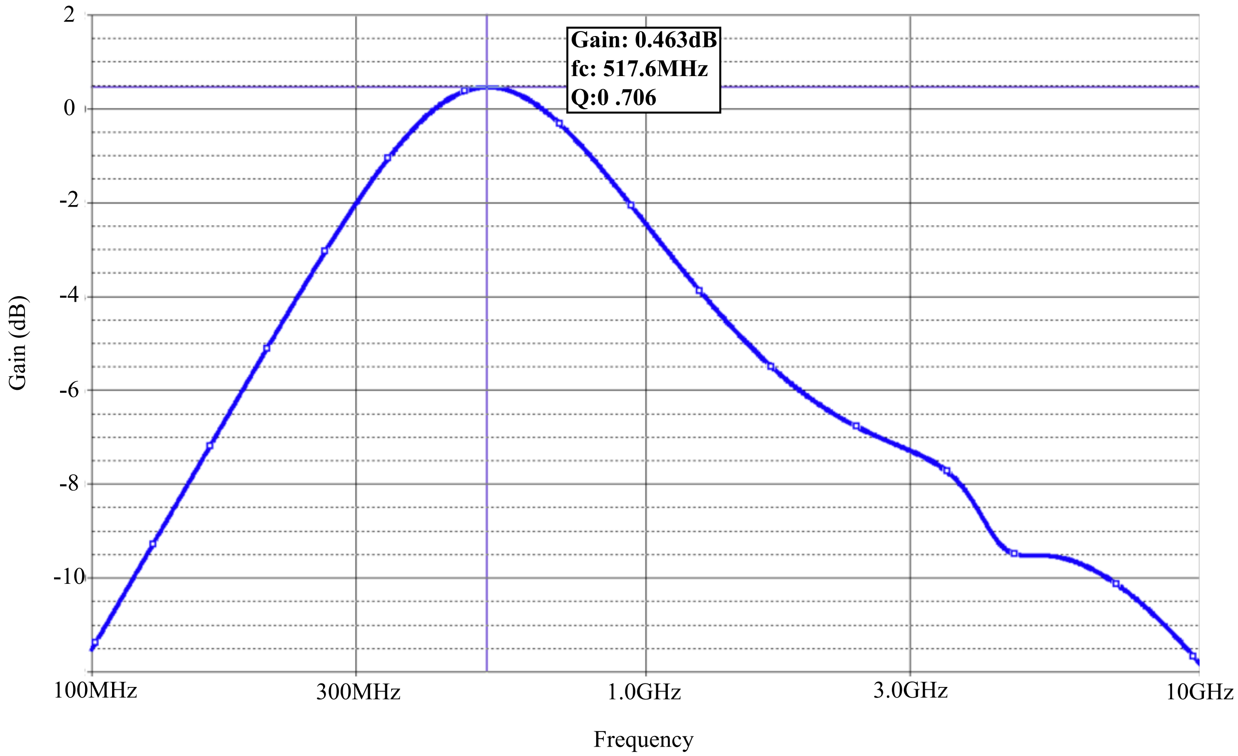


Fig. 5. AC analysis of a single stage Deliyannis filter. G = 0.463dB, f_c = 517 MHz, Q = 0.706.

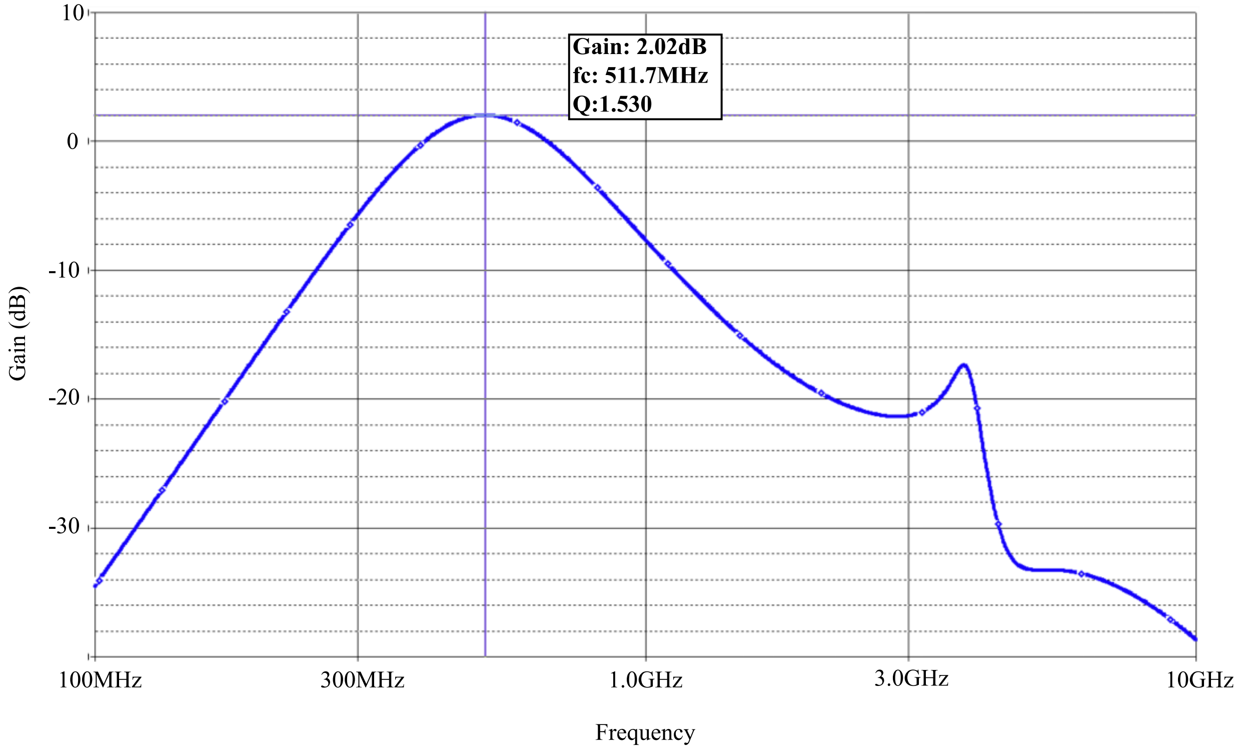


Fig. 6. AC analysis of a three stage Deliyannis filter. G = 2.02 dB, f_c = 511.7 MHz, Q = 1.530.

3c. Level Shifting Amplifier

The NBPFs were followed by a final layer of amplification to shift the DC level of the signal and amplify the signal to match the 0 to 1V input range for compatibility with the ADCs on the Xilinx RFSoC 2x2 board to minimize any quantization noise.

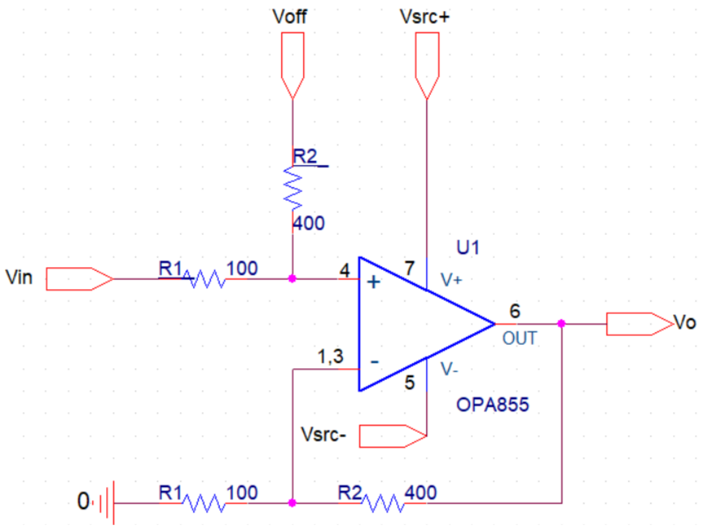


Fig. 7. Circuit diagram of the level shifting amplifier.

3d. Front-End System Overview

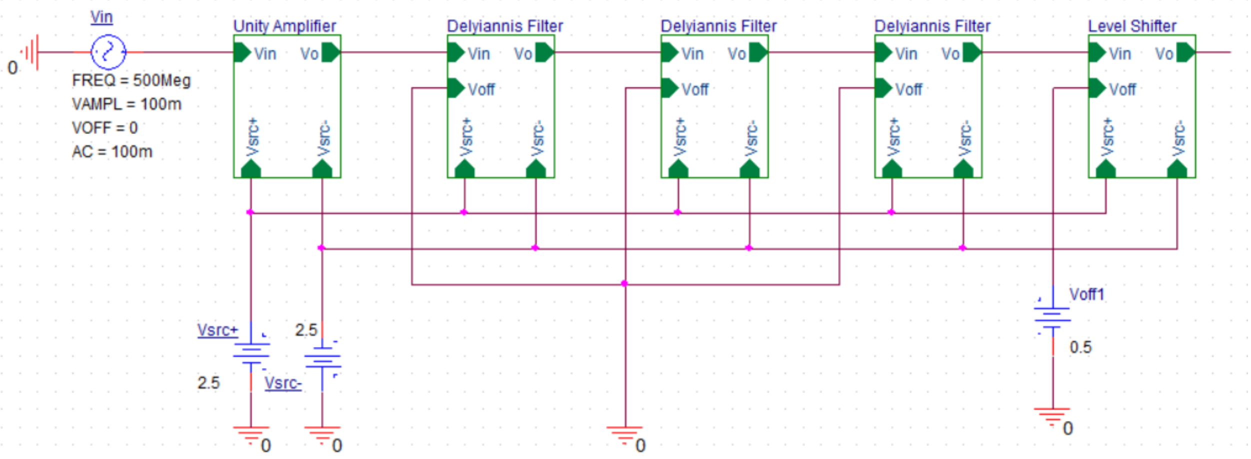


Fig. 8. Final block level diagram of the pre-ADC filter driver circuit.

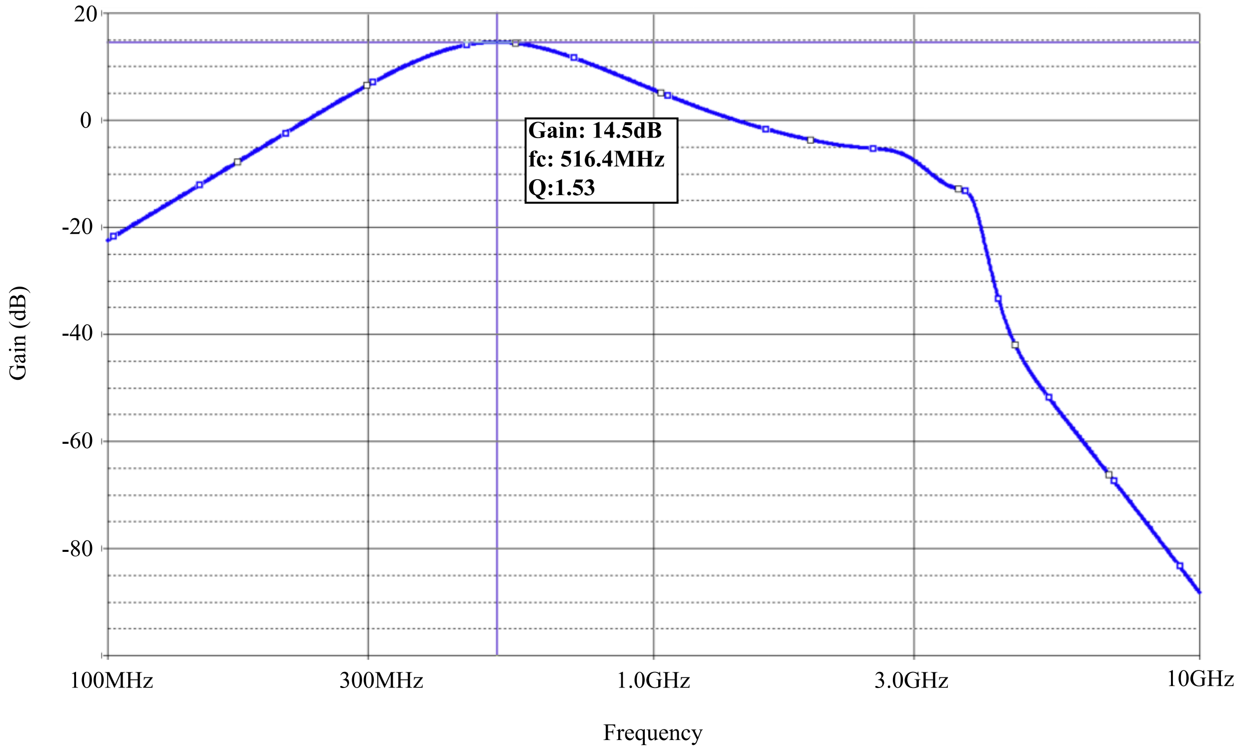


Fig. 9. AC analysis simulation of output of the pre-ADC driver circuit. G = 14.5 dB, f_c = 516.4 MHz, Q = 1.53.

The above figures (Fig. 8, Fig. 9) show the final block diagrams and AC analysis simulation of the pre-ADC circuit. The final parameters achieved were $f_c = 516.4\text{MHz}$, $G = 14.5dB$, $Q = 1.53$.

4. Analog Front End Assembly and Verification

4a. Front-End Layout

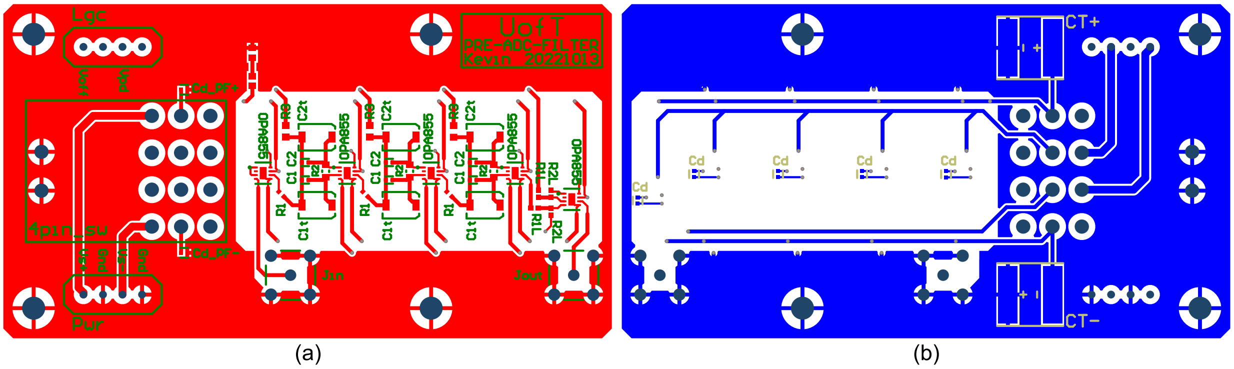


Fig. 10.
(a): Top layer PCB layout diagram of the pre-ADC filter circuit.
(b): Bottom layer PCB layout diagram of the pre-ADC filter circuit.

The pre-ADC front-end was implemented into a PCB layout. Special care was taken in the layout to minimize the introduction of noise. Components of size standard 0603 or smaller were used exclusively in any high-frequency paths of the circuit to reduce parasitic capacitances. Decoupling capacitors C_d were used at the power inputs of all opamp chips. Special care was taken to ensure symmetry in trace lengths in the signal inputs to the op-amps. Finally, a ground pour was applied in all non-high-frequency regions of the board to further reduce parasitic capacitances.

4b. Front End Assembly

The assembly of the pre-ADC filter was done stage-by-stage, fully assembling one stage of the circuit, validating the functionality, then moving on after validation.

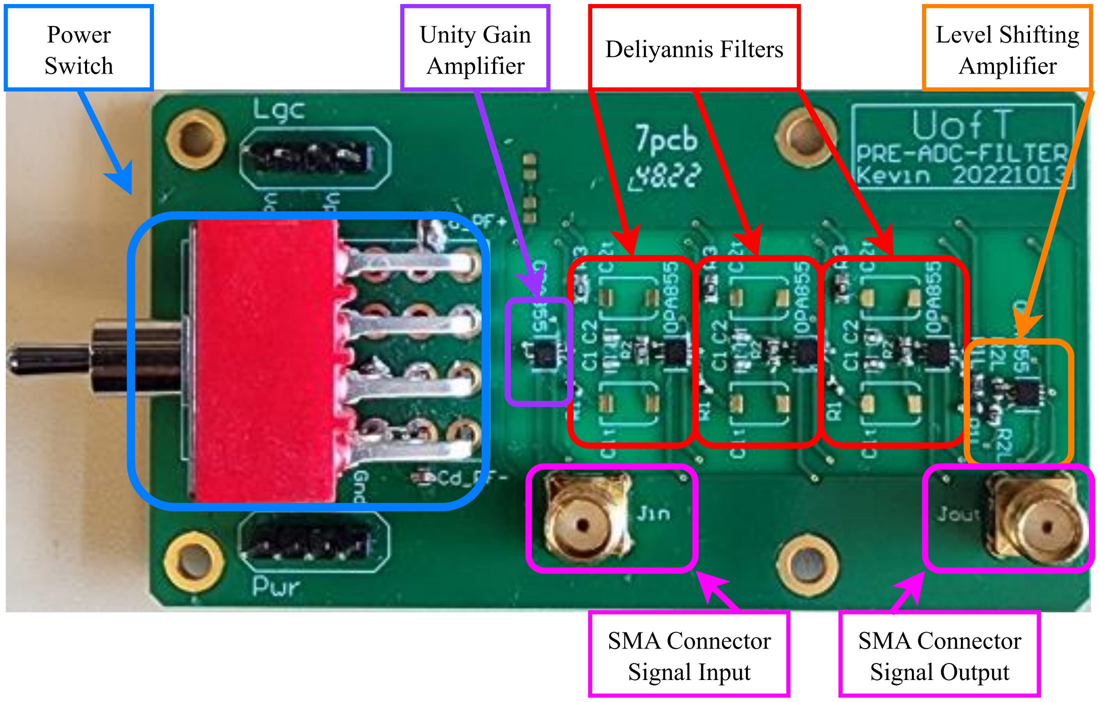


Fig. 11. Image of the assembled pre-ADC filter.

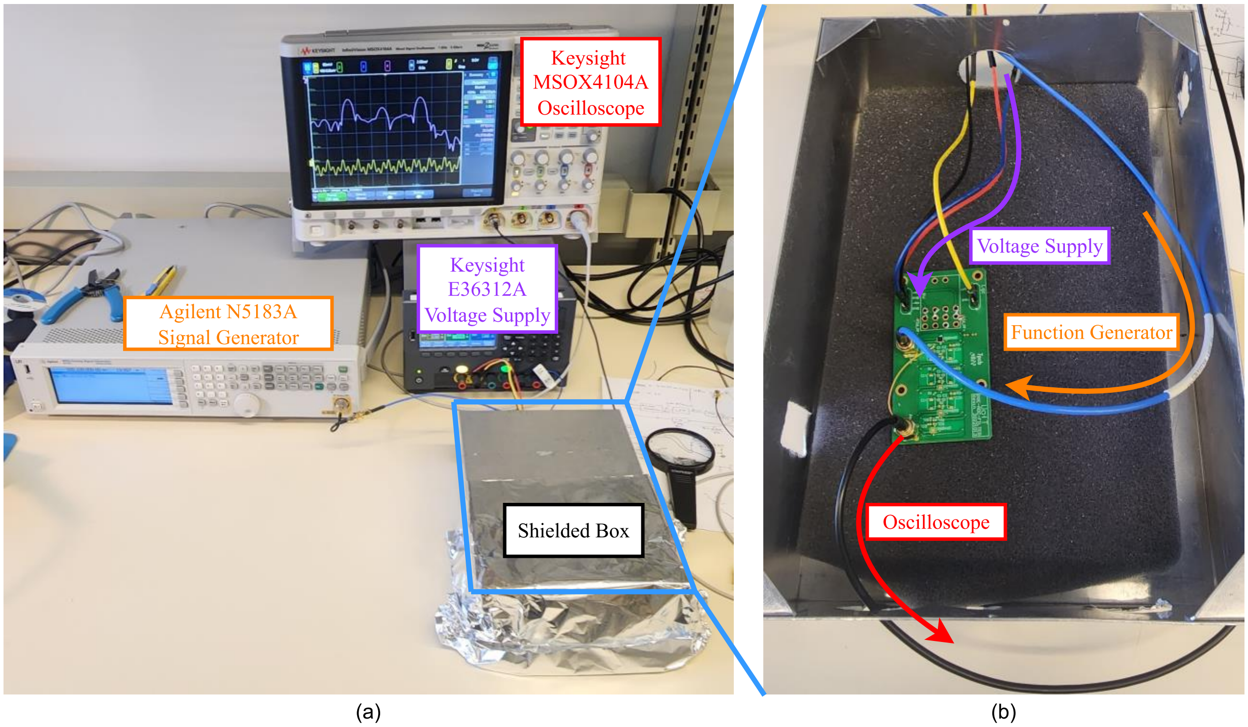
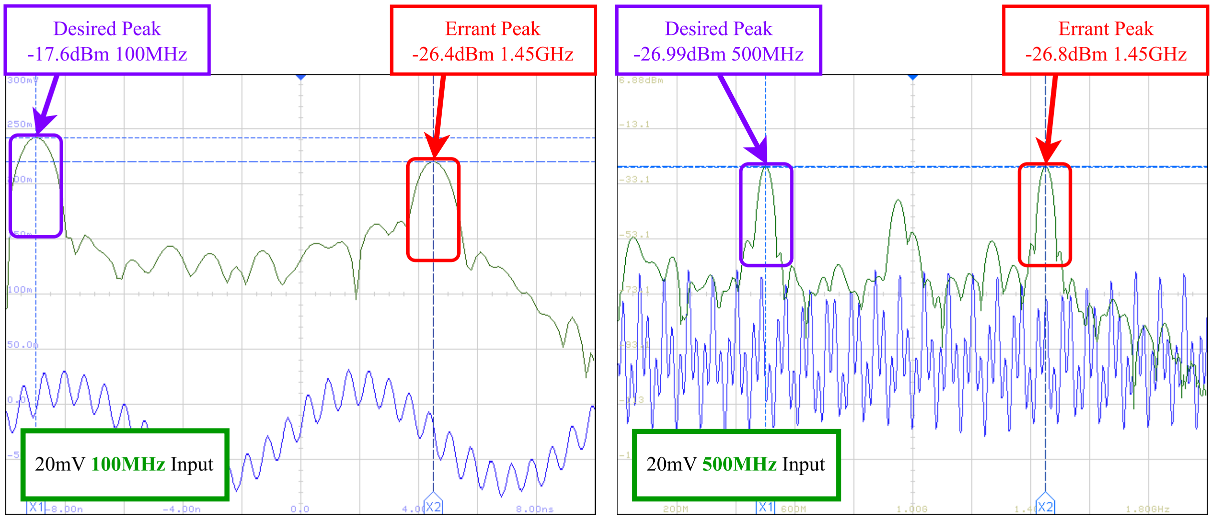

Keysight MSOX4104A Oscilloscope
Agilent N5183A Signal Generator
Keysight E36312A Voltage Supply
Shielded Box
(a)
Voltage Supply
Function Generator
Oscilloscope
(b)

Fig. 12.
(a): Overview image of front-end validation setup.
(b): Image of the circuit-under-test inside the metal shielded box.

The validation of a block was performed with a high-frequency function generator and oscilloscope as shown on Fig. 12. The function generator would be used to create a sinusoidal input at a target frequency. The output of a block was then analyzed via an oscilloscope with the desired output being a clean sinusoidal of the same input frequency albeit with the same gain as in simulation. A metal box was utilized as a shield to minimize noise from external sources such as radio stations.

4c. Input Buffer Validation

The unity gain amplifier exhibited an unexpected behavior during validation. While the input of the system was a sinusoidal, the time domain output was clearly a modulation of multiple frequencies as can be seen in Fig. 13. Under frequency domain, there was a peak present with an amplitude of -26dBm at 1.45GHz. This peak was present with or without the presence of the metal shield and was static in magnitude and frequency when the input frequency was modified from 500MHz to100MHz.



(a)

(b)

Fig. 13.
(a): Output of unity gain amplifier at 100MHz input. (b): Output of unity gain amplifier at 500MHz input.

4d. Deliyannis Filter Validation

The errant peak observed in the input buffer was also observed in the Deliyannis stages. As seen in Fig. 14, at the first Deliyannis stage it was present with an amplitude of -17.2dBm at 1.18GHz and at the second Deliyannis stage it was present with an amplitude of -13.8dBm at 1.14GHz. This suggests that the noise is increasing with subsequent stages and getting closer to the target f_L. Furthermore, with an input amplitude of 50mV, even at the first Deliyannis stage its amplitude was greater than the desired peak at -22.5dBm.

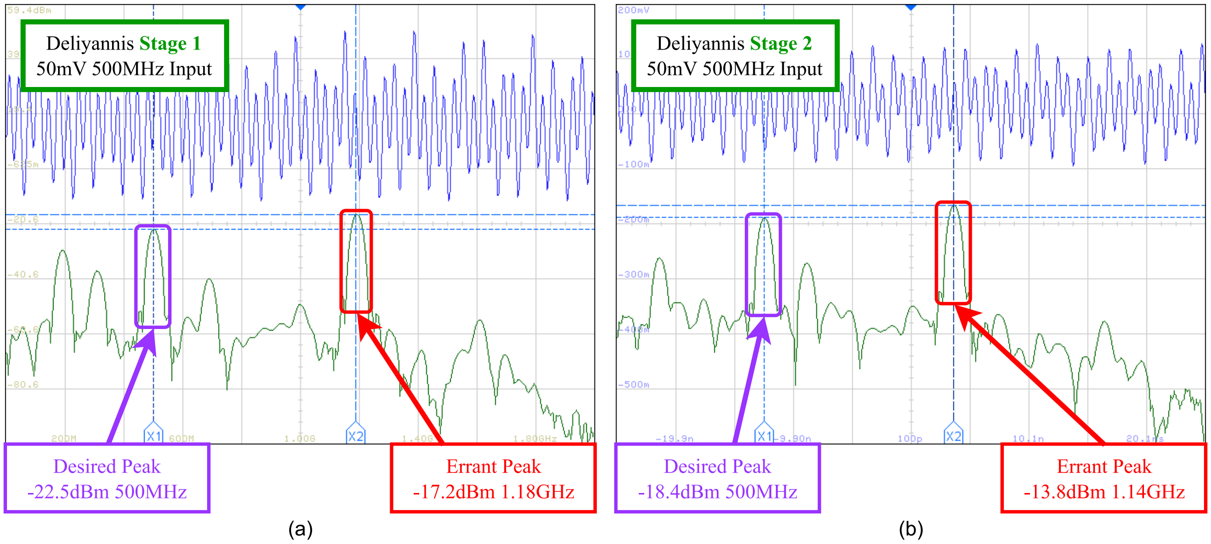


Fig. 14. (a): Output of Delyiannis Stage 1 at 500MHz. (b): Output of Delyiannis Stage 2 at 500MHz.

There are also significant intermodulation based distortions observed in addition to the desired signal, as shown in Fig. 15. As the input frequency changed from 500MHz to 540MHz, intermodulation could be seen in the frequency domain as intermodulation peaks appeared and disappeared around the two peaks.

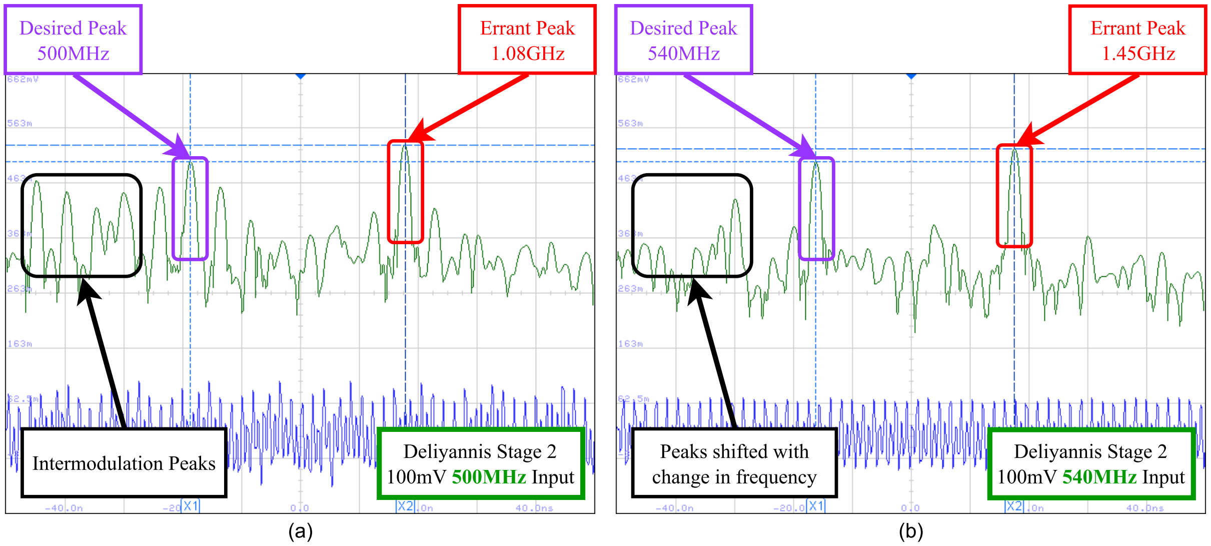


Fig. 15. (a): Output of Delyiannis Stage 2 at 500MHz. (b): Output of Delyiannis Stage 2 at 540MHz.

4e. Pre-ADC Filter Result Discussion

The source and nature of this errant peak is yet undetermined. It is unlikely to be external noise as the noise is not attenuated by the presence of the metal shielding. Its static nature with respect to the input frequency suggests that it is also not a harmonic of the input, as a harmonic, being a multiple of the fundamental frequency, would surely change with the change in the input. It also cannot be an intermodulation with another higher-order frequency as intermodulation would imply that it be a function of the input frequency and hence would be subject to change with the input frequency. It could be some sort of distortion in the system but confirming this would be difficult.

5. Digital Back-End Design

The design of the digital back end is a modification of the spectrum analyser implemented on the RFSoC 2x2 PYNQ platform by ‘StrathSDR’: University of Strathclyde’s Software Defined Radio Research Laboratory [12].

The ‘StrathSDR’ analyser utilizes a Mixer-Based Zoom FFT implemented in the RFSoC 2x2 PYNQ Platform [13]. This system involves the utilization of a mixer to shift the target frequency to DC, then applying a series of decimations to control the bandwidth and resolution of the FFT window.

The mixer, decimator and FFT blocks are all implemented and connected in hardware on the FPGA for high-frequency processing. A ZYNQ Ultrascale+ MPSoC v3.3 processor is also included in implementation to control and configure each stage. The open-sourced PYNQ software environment is installed atop this processor to provide a Linux operating system, hardware drivers, and Python-Based application programming interfaces (APIs). The PYNQ environment is then used to interface with the implemented hardware at runtime.

All connections between blocks utilize some form of the Advanced Extensible Interface (AXI) protocol [14]. Fig. 16, Fig. 17, Fig. 18 are simplified versions of the implemented system to highlight key specifications. The system can broadly be broken into three stages with respect to the function of the spectral analyzer. See Appendix I: Vivado – High Level Block Design - Appendix IV: Vivado – Channel Block Design for detailed block diagrams and connections.

5a. ZYNQ Ultrascale+ RF Data Converter

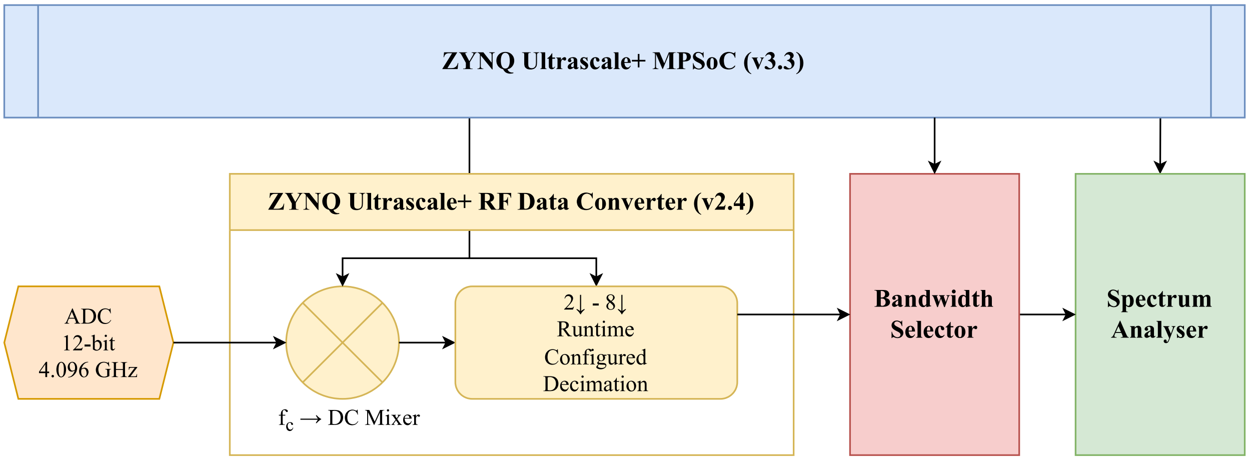


Fig. 16. Simplified block diagram of the ZYNQ Ultrascale+ RF Data Converter (v2.4).

The first stage of the system as shown in Fig. 16 is the ZYNQ Ultrascale+ RF Data Converter v2.4 (RFDC) [15] included in the Vivado Design Suite. This stage includes the frequency shift mixer as well as a runtime configurable decimator capable of decimations factors from 2 to 8. Both features are controllable through the PYNQ environment via the MPSoC processor.

5b. Bandwidth Selector

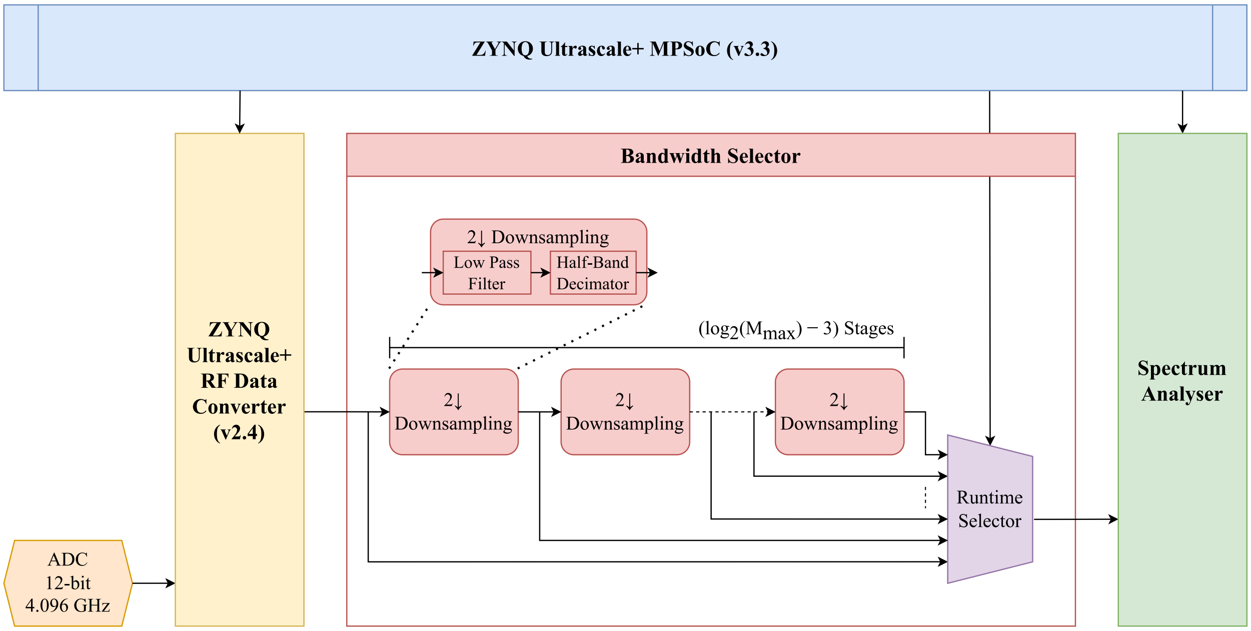


Fig. 17. Simplified block diagram of the bandwidth selector.

The second stage of the system shown in Fig. 17 is the custom bandwidth selector created by ‘StrathSDR’. This selector is used if more decimation is required past the factor of 8 provided by the RFDC. It consists of a series of half-band down samplers to decimate up to a factor of M_{max}, with each down sampler including a low-pass filter to avoid aliasing caused by the decimation. A selector block is used at the output provide the MPSoC control of the level of down sampling.

5c. Spectrum Analyser

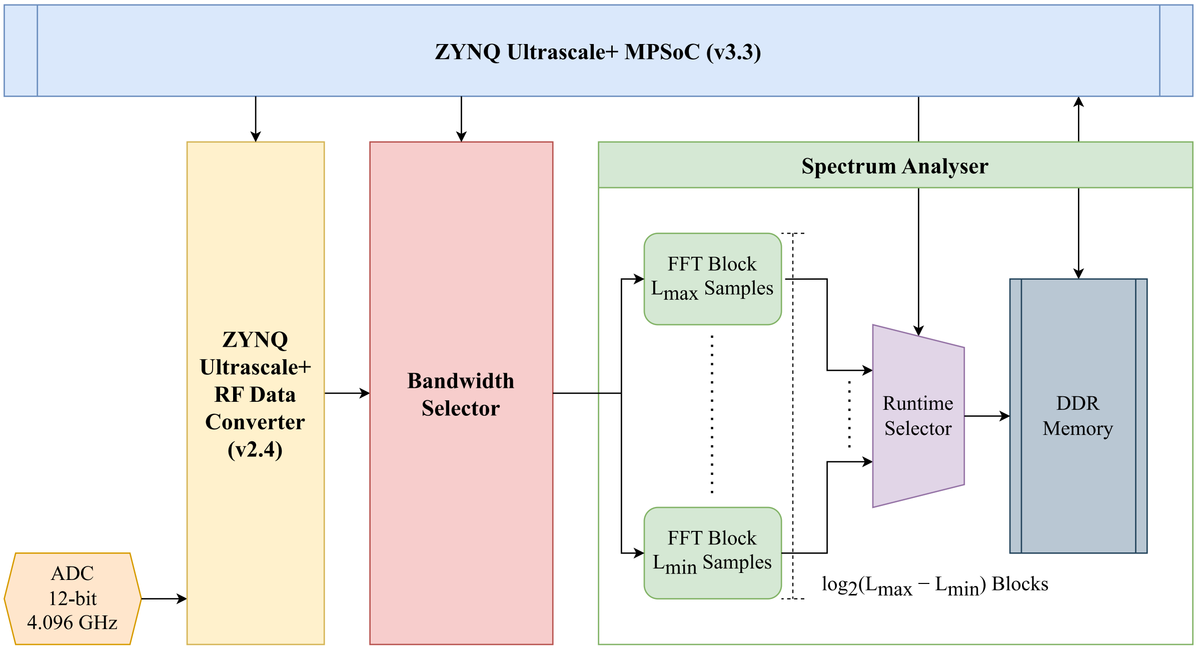

ZYNQ Ultrascale+ MPSoC (v3.3)
ZYNQ Ultrascale+ RF Data Converter (v2.4)
Bandwidth Selector
Spectrum Analyser
FFT Block L$_{max}$ Samples
FFT Block L$_{min}$ Samples
Runtime Selector
DDR Memory
log$_2$(L$_{max}$ − L$_{min}$) Blocks
ADC 12-bit 4.096 GHz

Fig. 18. Simplified block diagram of the spectral analyser.

The final stage is the custom spectrum analyser block created by 'StrathSDR'. This block performs the FFT analysis, then writes the transform to DDR RAM, where it can then be read by a Python program operating in the PYNQ environment. It contains several FFT blocks, each capable of performing FFT on a different number of samples. A selector is used to give control of the length of the FFT to the PYNQ environment at runtime.

The resolution of an FFT window under this architecture is governed by:

$$\Delta \mathrm{f}_{min} = \frac{f_s}{M * L} \tag{6}$$

Where $\Delta\mathrm{f}_{min}$ is the minimum frequency granularity of the FFT, f_s is the sampling frequency, M is the selected decimation, and L is selected length of the FFT window.

Applying Eq. (2) to Eq. (6), the following then provides the minimum ppm granularity of the FFT system.

$$\Delta ppm_{min} = \frac{f_s}{f_L} * \frac{10^6}{M * L} \tag{7}$$

The implementation provided by ‘StrathSDR’ for the RFSoC 2x2 Board allows FFT lengths up to 8192 samples and decimation factors of up to 2048 [16]. The RFSoC 2x2 platform, as aforementioned, has ADC with $f_s = 4.096\,Gsps$. This results in $\Delta f_{min} = 244.14\,Hz$ or $\Delta ppm_{min} = 0.49$.

To increase the FFT resolution even further, the spectrum analyser block was modified to support FFT lengths to $L_{max} = 65536$, leading to $\Delta f_{min} = 30.52\,Hz$ or $\Delta ppm_{min} = 0.06$ when decimation of 2048 is applied. This was the maximum number that could be synthesized and implemented onto the RFSoC 2x2 Board without being constrained by the resources available. The modified Simulink diagram for the block is shown in Appendix V: Simulink – Block Diagram of the Modified FFT Block – Appendix VIII: Simulink – Modifications to Pre-Processing Window.

6. Digital Back-End Implementation and Verification.

6a. Synthesis, Implementation, and PYNQ environments.

The custom bandwidth selector and spectrum analyser blocks are designed and implemented by a few tools. MATLAB 2020a, Simulink v10.1 was used with Vivado 2020.2's System Generator to design and verify the functionality of the blocks at a high pre-synthesis level. The Simulink HDL Coder v3.16 Addon is then used to create VHDL code for the verified block. Finally, Vivado 2020.2 is again used to create a block design to connect and integrate with MPSoC and the RF Data Converter block. Synthesis and integration of the whole system is then performed by Vivado, and a bitstream file for the RFSoC 2x2 board written. The relevant Vivado block diagrams are available in Appendix I: Vivado – High Level Block Design – Appendix IV: Vivado – Channel Block Design.

To allow interfacing with the implemented blocks in the PYNQ environment, modifications had to be made on the Python APIs and drivers that came with the original 'StrathSDR' spectrum analyser. This involved modifying a few values to account for the different lengths of FFTs implemented and changing a few names to point to the new versions of the IP Blocks.

All modified drivers and APIs were packaged with the bitstream as a Python library and installed into the PYNQ 2.7 environment.

The RFSoC 2x2 Board is configured on boot-up to automatically starts a Jupyter Notebook Server through which the command line PYNQ environment is exposed. This server can be accessed over a USB interface. A few lines of a Python script can then be used to tell the board which bitstream it should flash onto PL core, and what sets of drivers it should load to interface with the hardware.

6b. Digital Back-End Verification

The implemented back-end design was verified empirically by comparing the FFT performed by the system to that of a known valid FFT transform. For the Tx signal, the 12-bit 6.554 GHz Digital to Analog Converter (DAC) on the same RFSoC 2x2 Board was used to generate a frequency sweep of centered at $f_c = f_L = 500$ MHz, with a total sweep width of 50 kHz at increments Δf of 2.5 kHz (5 ppm). This DAC was controlled by the same Jupyter Notebook environment as described above.

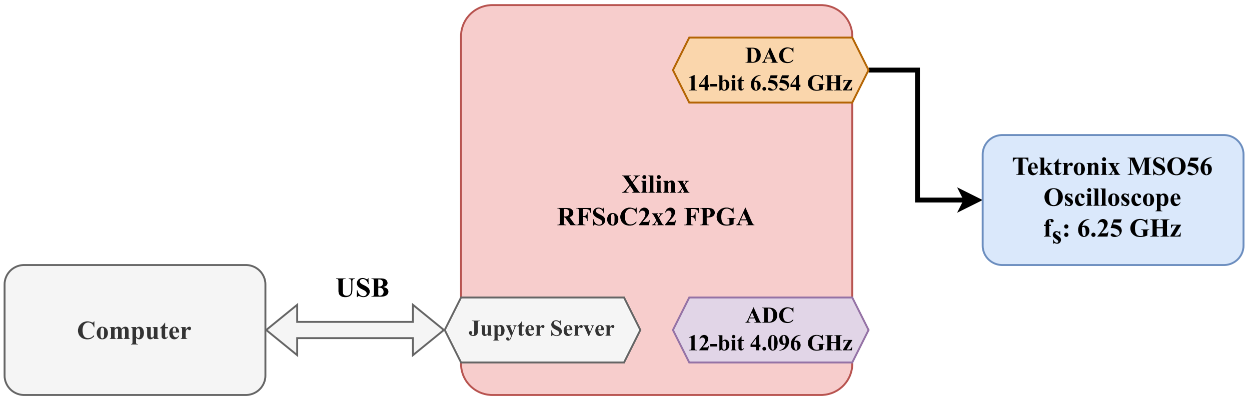


Fig. 19. Block diagram of back-end validation setup. Direct coupled with the Tektronix MSO56 oscilloscope.

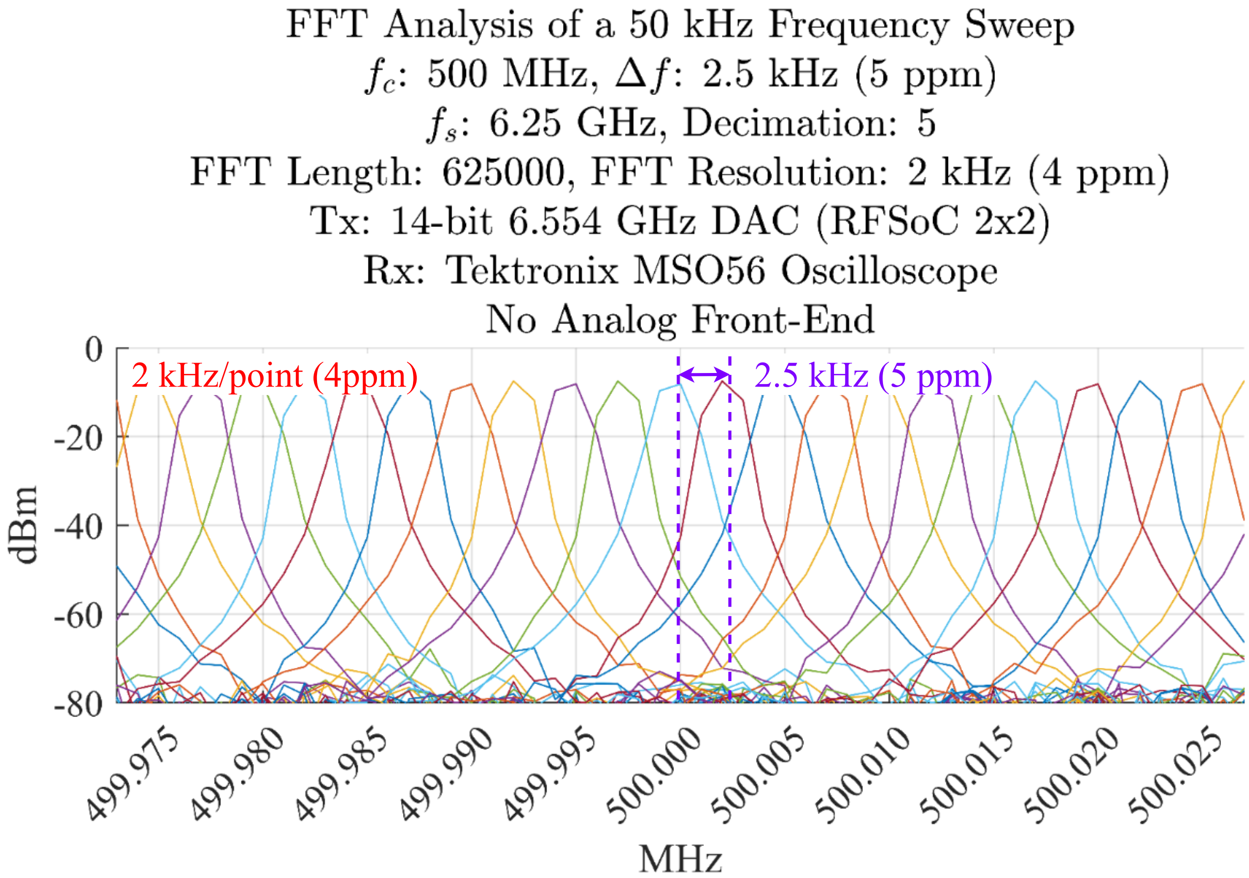


Fig. 20. Superimposed FFTs from a direct coupled Tektronix MSO56 Oscilloscope.

Fig. 20 shows the FFT performed by a 6.25 GHz Tektronix MSO56 oscilloscope with the TX directly coupled to the oscilloscope as shown in Fig. 19. The FFT length is 625 k, with a decimation factor of 5 resulting in an FFT Resolution of 2 kHz or 4 ppm. The oscilloscope does not support mixing to decrease in bandwidth with each decimation. Hence, the decimation must be kept above a certain limit to ensure 500 MHz is within the bandwidth.

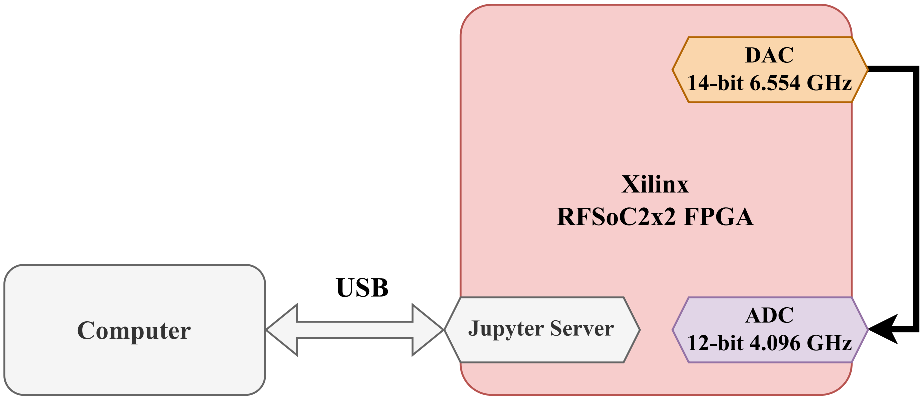


Fig. 21. Block diagram of back-end validation setup. Direct coupled with ADC on RFSoC 2x2.

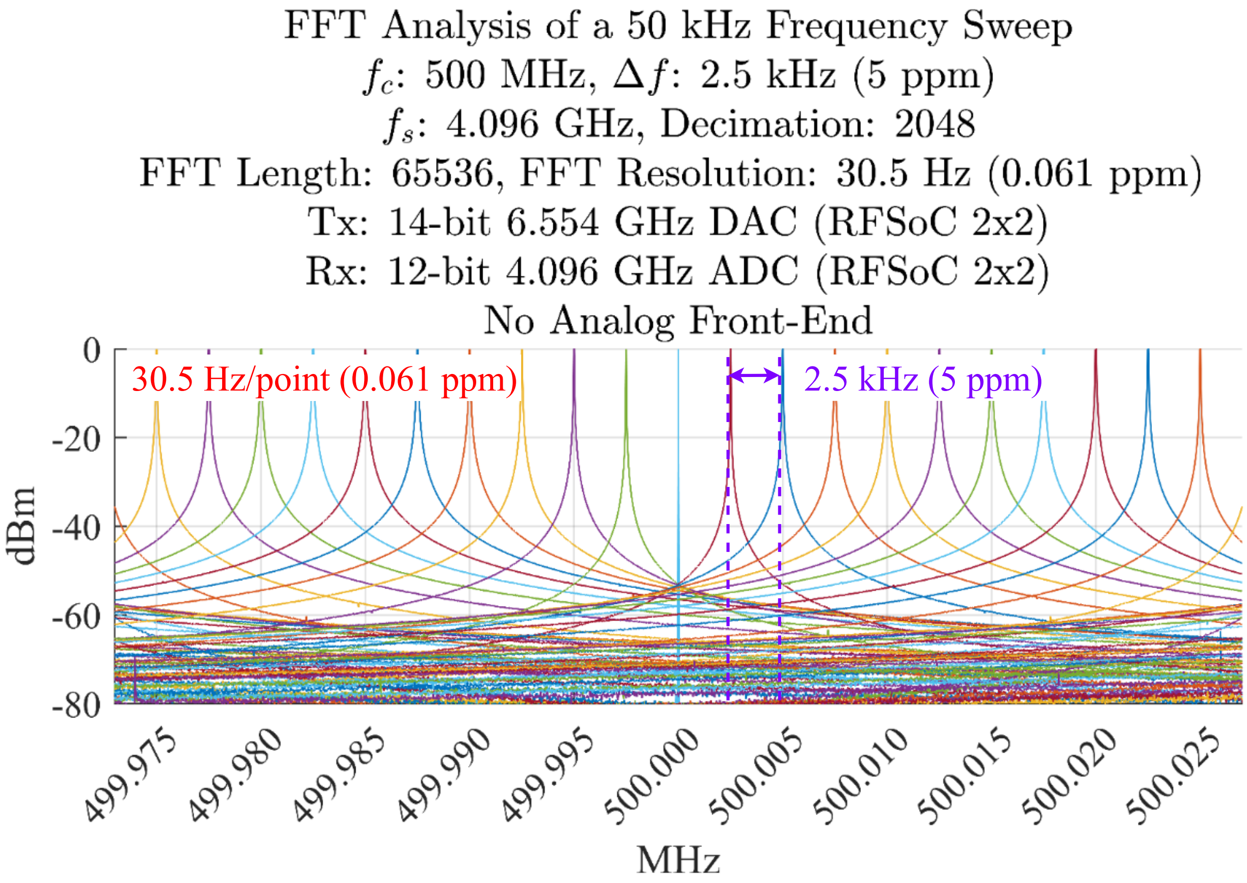


Fig. 22. Superimposed FFTs from a direct coupled digital back-end.

The same sweep was then performed as shown in Fig. 22 by the digital back-end, directly coupled with the Tx signal as shown in Fig. 21. It is evident that the back-end is capable of performing at the required level of resolution. It can display peaks in 2.5 kHz intervals similarly to the performance of the Tektronix MSO56 oscilloscope. However, in contrast to the MSO56, each peak is much more well defined, due to the orders of higher FFT resolution. While a 4-ppm granularity achieved by the Tektronix MSO56 oscilloscope is certainly enough to identify each peak, the peaks are less distinctly defined. The digital back-end with significantly finer ppm granularity more than meets the requirements for this thesis.

7. System Integration Results

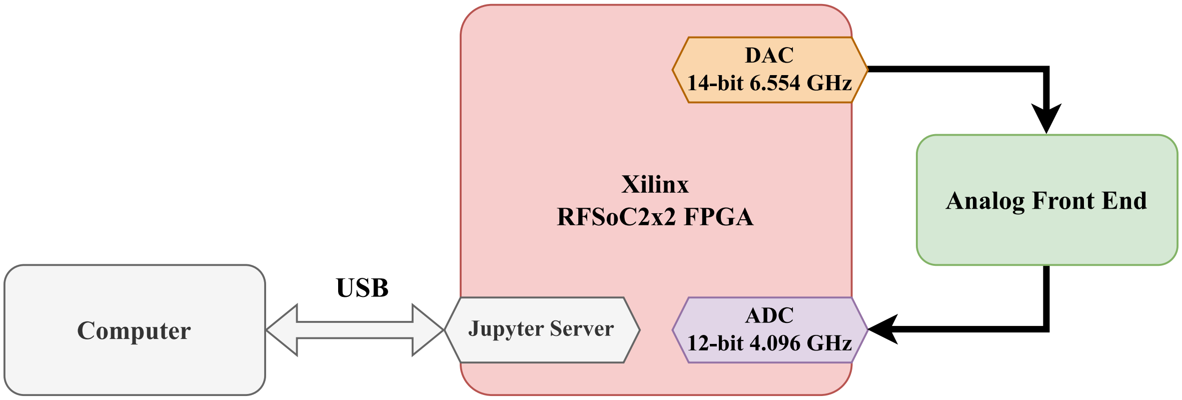


Fig. 23. Block diagram of the fully integrated system.

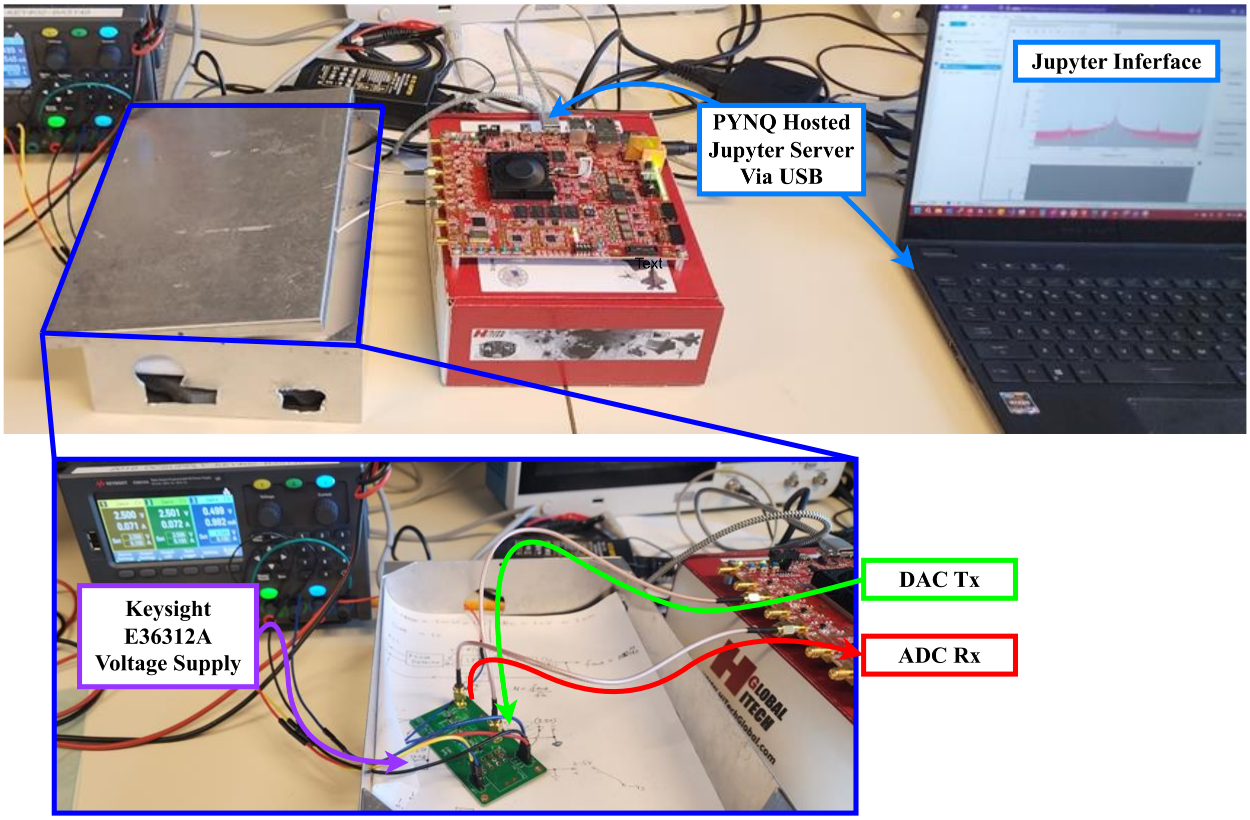


Fig. 24. Image of the physical fully integrated system.

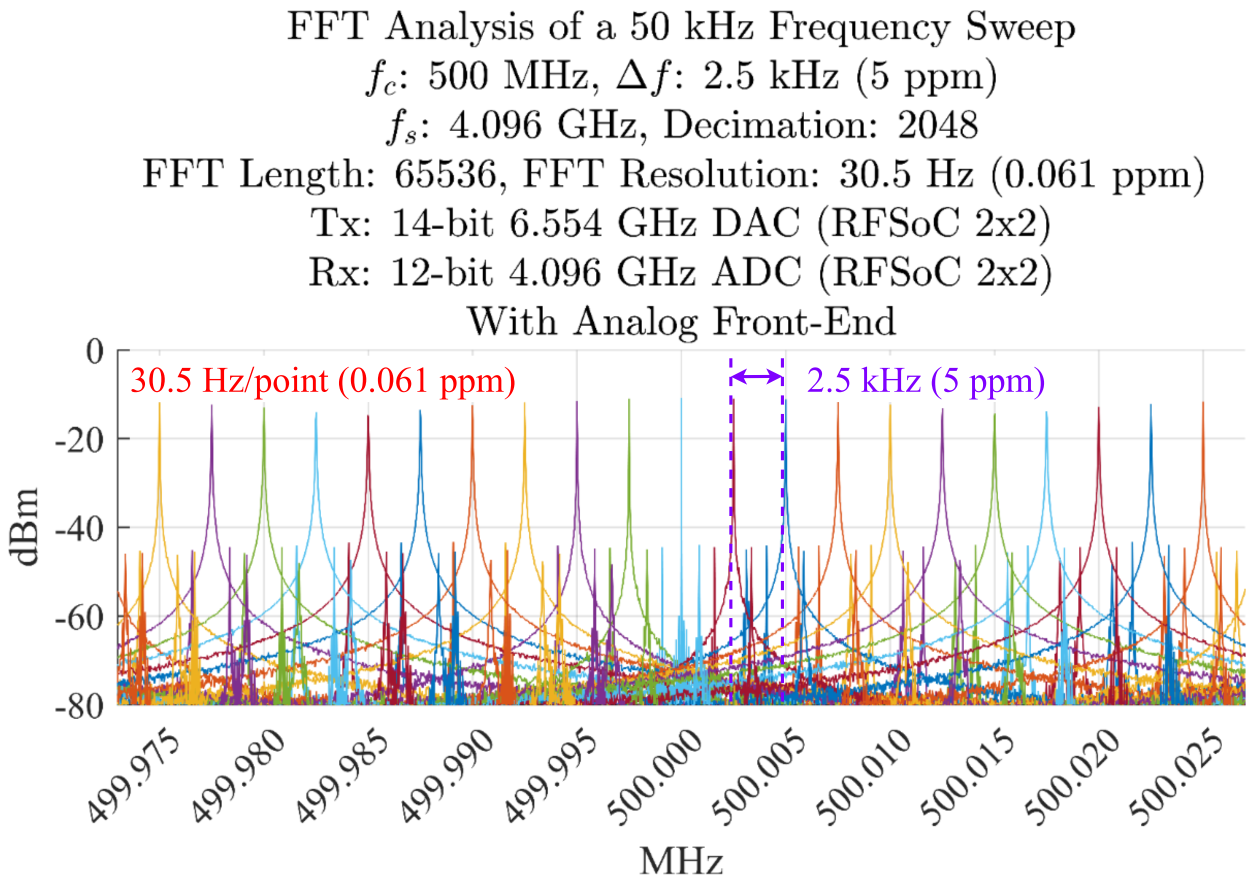


Fig. 25. Superimposed FFTs from fully integrated system.

Fig. 25 shows the frequency sweep of an identical Tx signal as Fig. 22, but with the analog front-end installed between the DAC Tx and the ADC Rx of the RFSoC 2x2 board as illustrated in Fig. 25 and Fig. 24. The sweep revealed significant attenuation of the in all tested frequencies compared to that of Fig. 22. Furthermore, there is a presence of a secondary row of peaks between the primary peaks around -50 dBm.

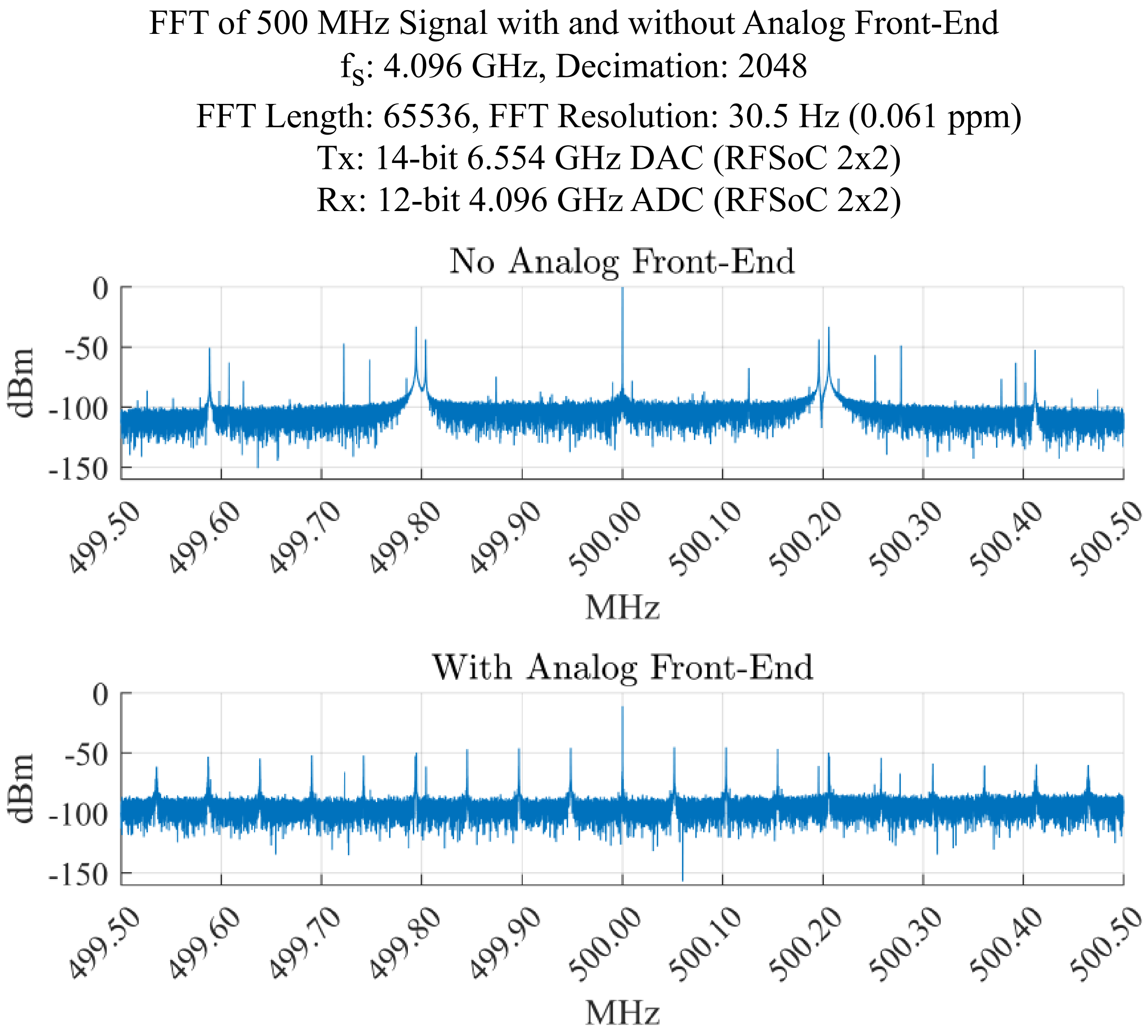


Fig. 26. FFT from the back-end with and without integration of analog frontend. 500 MHz input signal.

The source of the secondary row of peaks observed in Fig. 25, the plots of individual FFT transforms of a 500 MHz Tx signal with and without the front end. While the analog front end seems to attenuate the noise slightly around 500 $MHz \pm 200\ kHz$, it introduces additional peaks. The closest peaks to 500 MHz are the source of the rows of peaks seen in Fig. 25. As previously discussed, it is unclear if these peaks are a result of noise, distortion, or intermodulation.

These secondary peaks likely jeopardize the performance of the proposed system in FFT analysis as they may interfere with legitimate chemical shifts. They are likely related to the errant noise signal as discussed in Section 4e. Pre-ADC Filter Result Discussion.

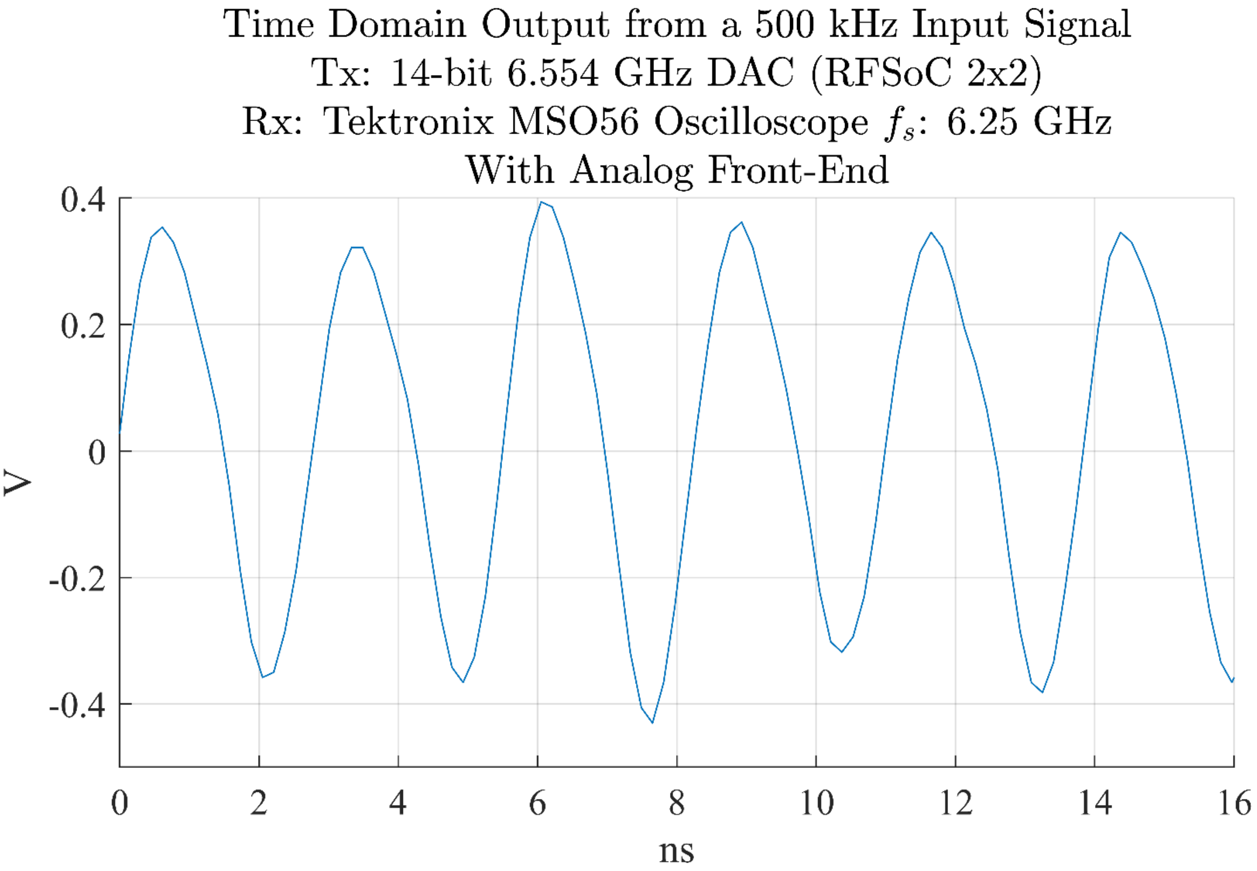


Fig. 27. Time domain signal from the analog front-end, input 500 kHz, Tektronix MSO56 oscilloscope Rx.

Looking at the time domain output of the analog front-end with the Tektronix MSO56 oscilloscope, it is even more evident that the analog front-end is under-performing required specifications. At 500 MHz input, there should be observable periodicity at intervals of 2 ns. Instead, the dominant periodicity appears to be ~2.75 ns, leading to a dominant frequency of ~360 MHz.

It is clear that while the implemented digital back-end meets the requirements for the NMR experimentation, there is significant work that needs to be done for the analog front end to reduce the noise introduced into the system.

8. Conclusion

This thesis aimed to implement a spectrometer capable of fine-grain NMR spectroscopy. The aim was to utilize a simplified Rx to reduce distortion and direct sample the signal at frequencies orders higher than the minimum Nyquist requirement, leveraging modern advances in high frequency ADCs. This would preserve the integrity of the signal, allowing the implementation of a runtime deep-FFT to perform fine-grain analysis.

A pre-ADC filter was designed to act as an analog-front end. The goal of this filter would be to attenuate non-relevant frequencies enough to minimize intermodulation and distortion being transferred into the digital domain, where further analysis of the signal would occur.

The implementation of this circuit has encountered difficulties due to a distortion of unknown nature at around 1 GHz. The amplitude of this distortion is significant compared to the input, and further work needs to be done to address it.

The digital back-end was implemented on the Xilinx RFSoC 2x2 FPGA board which features a 12-bit 4.096 GHz ADC. The back-end was a modification of the spectrum analyser implemented by the ‘StrathSDR’ Lab. The analyser implements a mixer-based zoom FFT in hardware, which was modified to enable deeper lengths of FFTs and higher orders of decimations to implement FFT analysis with resolutions of 0.06 ppm allowing detailed characterization of the peaks.

While the implemented back-end satisfies the target requirements, the distortion from the analog front-end proved too significant for the proposed system to be used reliably for high resolution NMR spectroscopy.

The analog front-end needs significant work to reduce this distortion and should be redesigned in the future to minimize high frequency noise, with a higher Q factor to eliminate interference from higher band frequencies.

With the goal of a weaker β_0 field for portability in mind, another approach would be to target a significantly lower f_L. The digital back-end is already capable of fine-grain spectral analysis. With the current back-end at the highest f_L discussed in Section 2, where f_L = 48 MHz [7], the achieved $\Delta ppm = 0.64$ based on Eq. (7), much finer than that performed by the Tektronix MSO56 in Fig. 20. Accuracy could also be further increased by increasing the level of decimation factor past 2048.

This would allow for a design of an Rx at a lower f_c, reducing the impact of parasitics in the circuit.

References

[1] C. A. Lepre, J. M. Moore and J. W. Peng, "Theory and applications of NMR-based screening in pharmaceutical research," *Chemical Reviews,* vol. 104, pp. 3641-3676, 2004.

[2] L. Lagh, G. Picone and F. Capozzi, "Nuclear magnetic resonance for foodomics beyond food analysis," *TrAC Trends in Analytical Chemistry,* vol. 52, pp. 93-102, 2013.

[3] D. Ha, J. Paulsen, N. Sun, Y.-Q. Song and D. Ham, "Scalable NMR spectroscopy with semiconductor chip," *PNAS,* vol. 111, no. 33, pp. 11955-11960, 2014.

[4] J. Handwerker, M. Pérez-Rodas, M. Beyerlein and F. Vincent, "A CMOS NMR needle for probing brain physiology," *Nature Methods,* vol. 17, pp. 64-67, 2020.

[5] D. Ariando, C. Chen, M. Greer and S. Mandal, "An autonomous, highly portable NMR spectrometer based on a low-cost System-on-Chip," *Journal of Magnetic Resonance,* vol. 299, pp. 74-92, 2019.

[6] A. S. Mukesh Kumar Singh, "Nuclear magnetic resonance spectroscopy," in *Characterization of Polymers and Fibres,* Woodhead Publishing, 2022, pp. 321-339.

[7] H. Bürkle, T. Klotz, R. Krapf and J. Anders, "A 0.1 MHz to 200 MHz high-voltage CMOS transceiver for portable NMR systems with a maximum output current of 2.0 App," in *ESSCIRC 2021*, Grenoble, France, 2021.

[8] S. Hong and N. Sun, "Portable CMOS NMR system with 50-kHz IF, 10-µs dead time, and frequency tracking," *IEEE Transactions on Circuits and Systems,* vol. 68, no. 11, pp. 4576-4588, 2021.

[9] Xilinx, "RFSoC 2x2 Kit," [Online]. Available: https://www.xilinx.com/support/university/xup-boards/RFSoC2x2.html. [Accessed 14 10 2022].

[10] Texas Instruments, "OPA855," [Online]. Available: https://www.ti.com/product/OPA855. [Accessed 26 Jan 2023].

[11] TI Designs, "High-Q active differential band-Pass filter reference," March 2017. [Online].

[12] StrathSDR, "University of Strathclyde SDR Lab," [Online]. Available: https://sdr.eee.strath.ac.uk/. [Accessed 10 04 2023].

[13] Xilinx Research & Open Source Projects, "SDR with the ZYNQ RFSoC; section 7: spectrum analyzer overview," [Online]. Available: https://www.youtube.com/watch?v=PqPdfnbNxyY. [Accessed 10 04 2023].

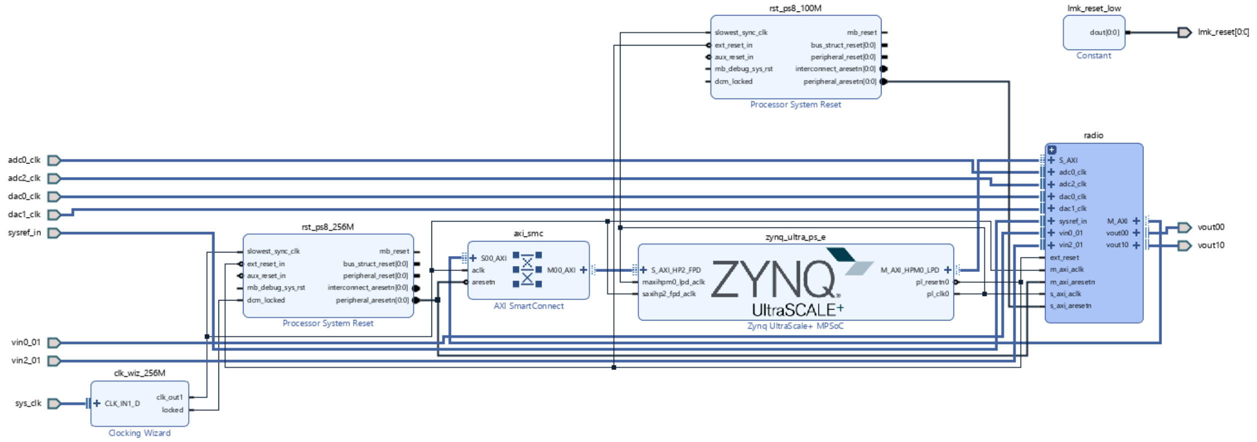
[14] ARM, "AMBA AXI protocol specification," March 2023. [Online]. Available: https://developer.arm.com/documentation/ihi0022/j/?lang=en.

[15] Xilinx, "Zynq UltraScale+ RFSoC RF Data Converter," 21 10 2022. [Online]. Available: https://docs.xilinx.com/r/en-US/pg269-rf-data-converter.

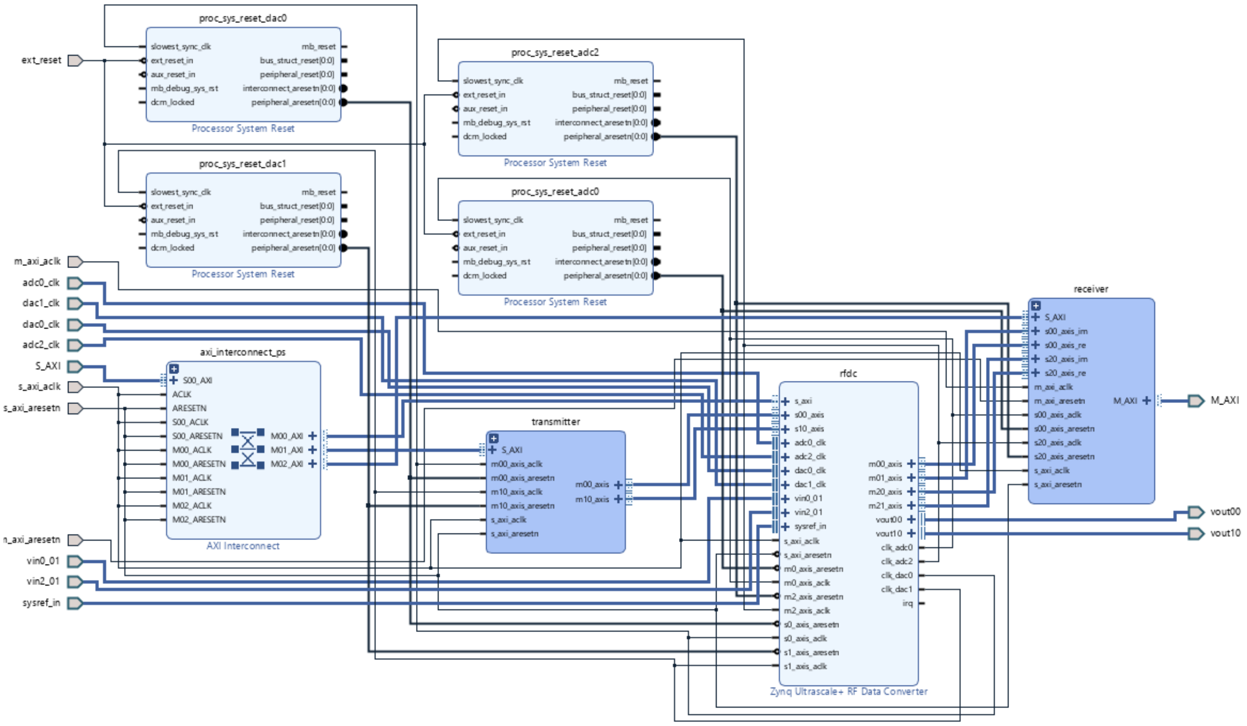
[16] StrathSDR, "Spectrum analyser on PYNQ," [Online]. Available: https://github.com/strath-sdr/rfsoc_sam. [Accessed 10 04 2023].

Appendix

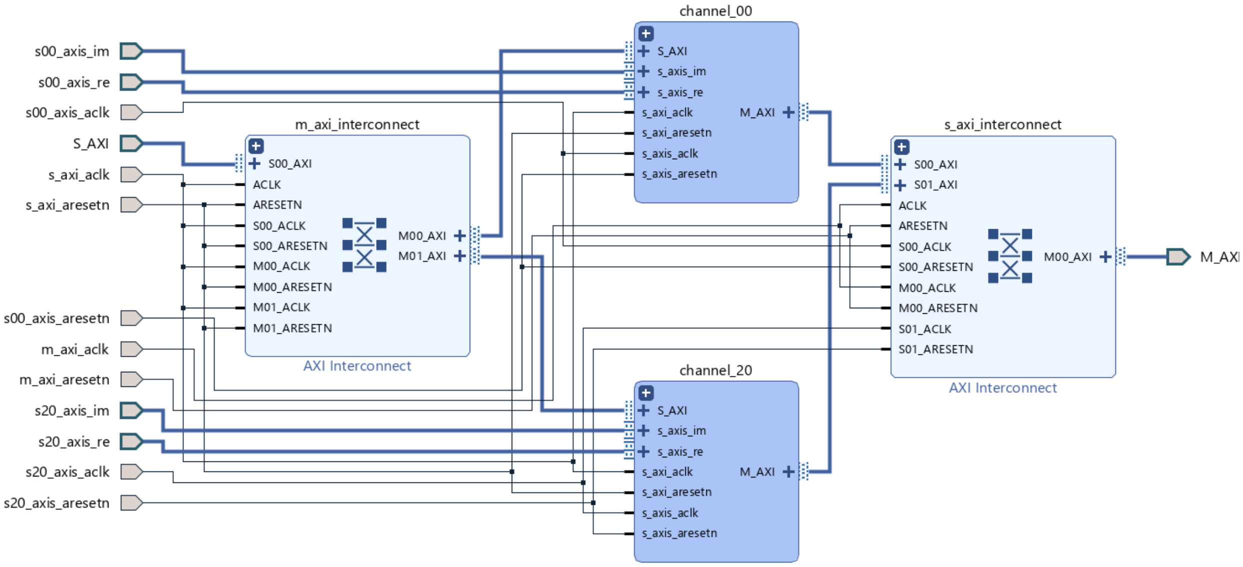
Appendix I: Vivado – High Level Block Design



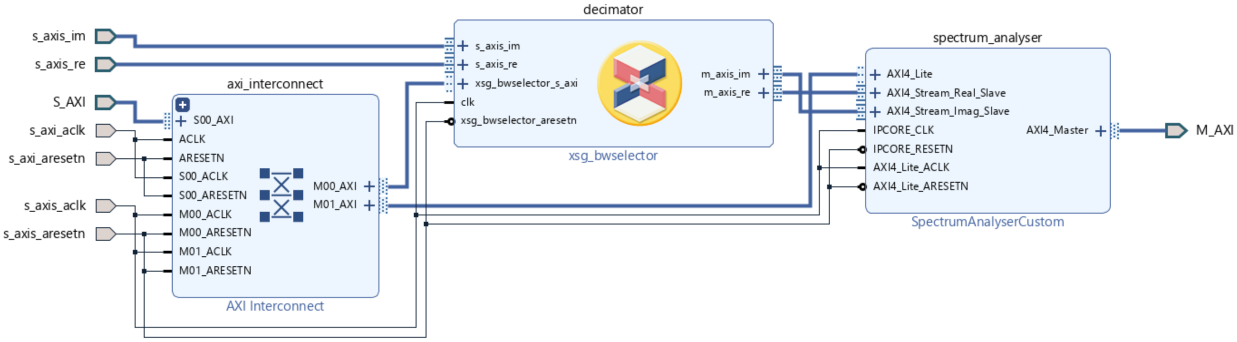
Appendix II: Vivado – Radio Block Design



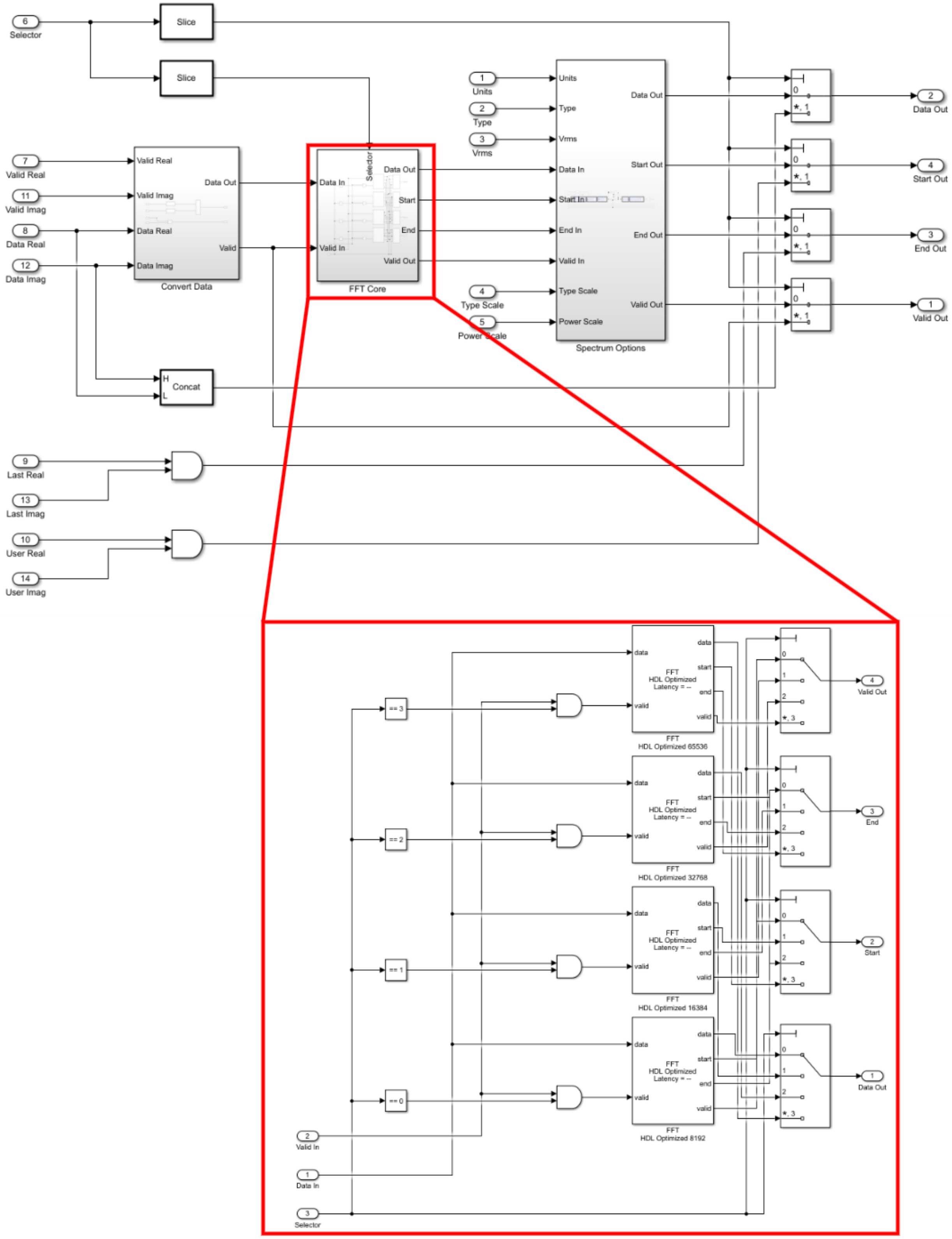
Appendix III: Vivado – Receiver Block Design



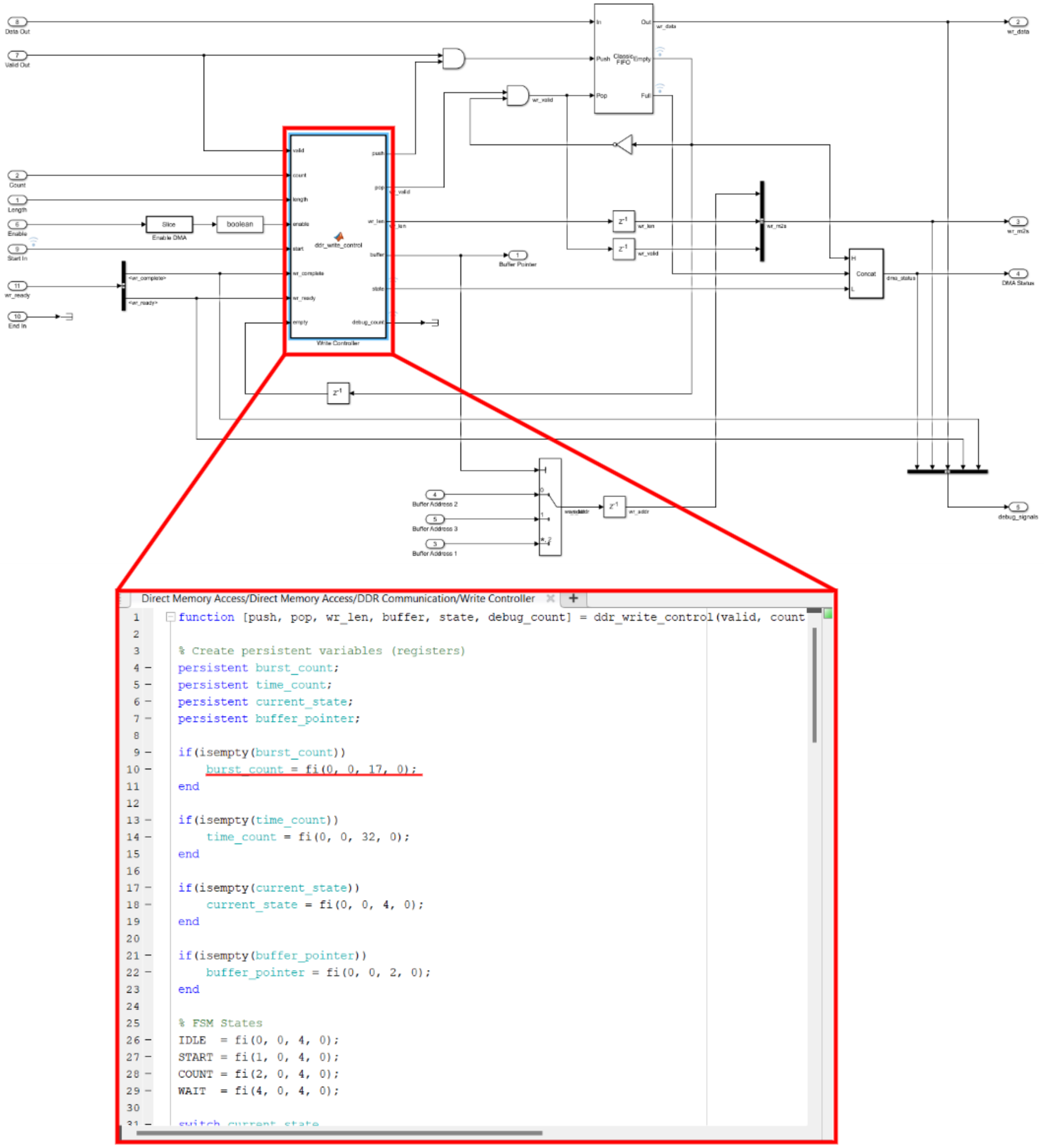
Appendix IV: Vivado – Channel Block Design



Appendix V: Simulink – Block Diagram of the Modified FFT Block



Appendix VI: Simulink – Modifications to DMA DDR Write Controller

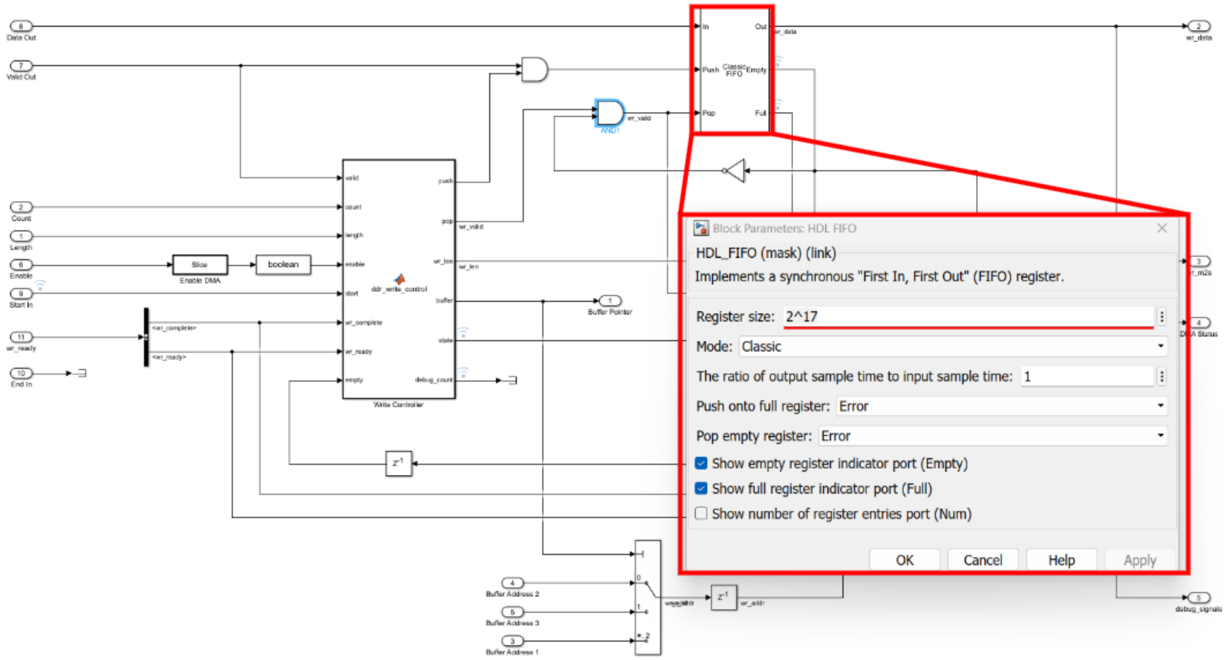


```
function [push, pop, wr_len, buffer, state, debug_count] = ddr_write_control(valid, count

% Create persistent variables (registers)
persistent burst_count;
persistent time_count;
persistent current_state;
persistent buffer_pointer;

if(isempty(burst_count))
    burst_count = fi(0, 0, 17, 0);
end

if(isempty(time_count))
    time_count = fi(0, 0, 32, 0);
end

if(isempty(current_state))
    current_state = fi(0, 0, 4, 0);
end

if(isempty(buffer_pointer))
    buffer_pointer = fi(0, 0, 2, 0);
end

% FSM States
IDLE  = fi(0, 0, 4, 0);
START = fi(1, 0, 4, 0);
COUNT = fi(2, 0, 4, 0);
WAIT  = fi(4, 0, 4, 0);

switch current_state
```

Full changes to the code are found in:

https://github.com/CarbonicKevin/2022-Year-ESC499-Thesis/blob/main/digital-back-end/boards/ip/hdlcoder_custom/slib_spectrum_analyser.slx

Appendix VII: Simulink – Modifications to DMA DDR Write FIFO Block



Block Parameters: HDL FIFO

HDL_FIFO (mask) (link)

Implements a synchronous "First In, First Out" (FIFO) register.

Register size: 2^17

Mode: Classic

The ratio of output sample time to input sample time: 1

Push onto full register: Error

Pop empty register: Error

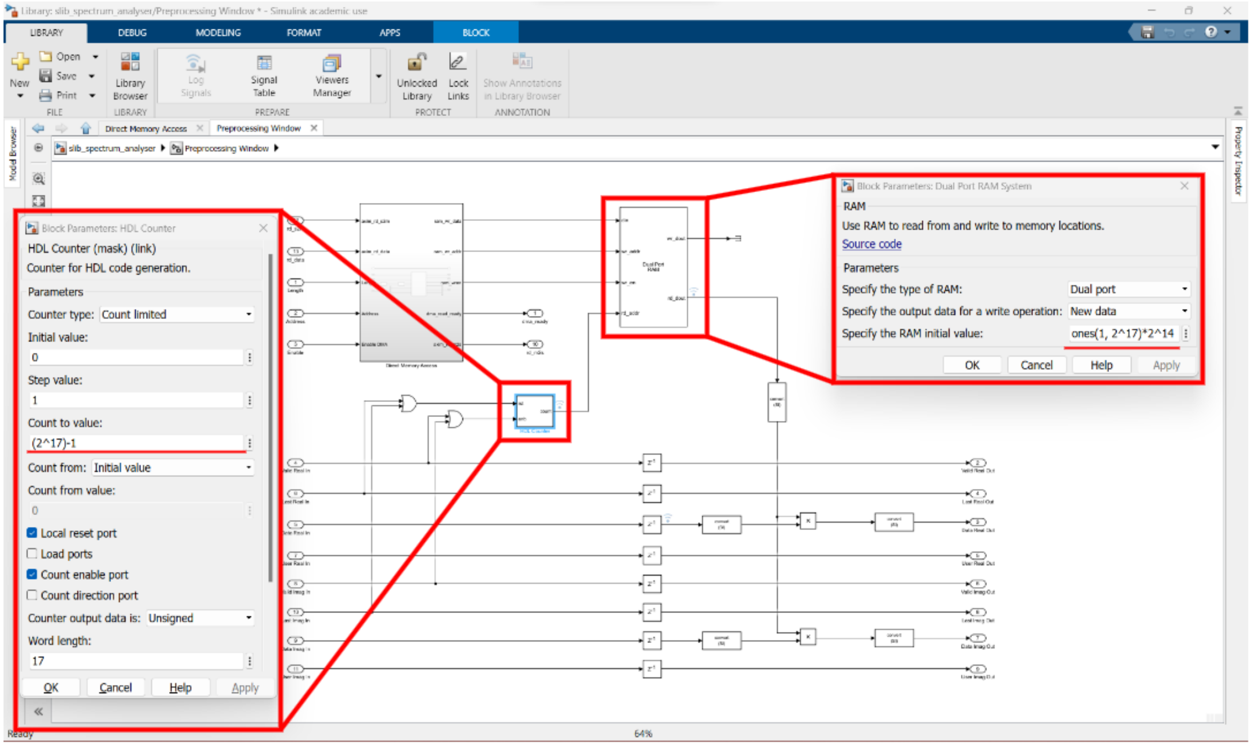
☑ Show empty register indicator port (Empty)

☑ Show full register indicator port (Full)

☐ Show number of register entries port (Num)

OK Cancel Help Apply

Appendix VIII: Simulink – Modifications to Pre-Processing Window



Block Parameters: HDL Counter

HDL Counter (mask) (link)

Counter for HDL code generation.

Parameters

Counter type: Count limited

Initial value: 0

Step value: 1

Count to value: (2^17)-1

Count from: Initial value

Count from value: 0

☑ Local reset port

☐ Load ports

☑ Count enable port

☐ Count direction port

Counter output data is: Unsigned

Word length: 17

OK Cancel Help Apply

Block Parameters: Dual Port RAM System

RAM

Use RAM to read from and write to memory locations.

Source code

Parameters

Specify the type of RAM: Dual port

Specify the output data for a write operation: New data

Specify the RAM initial value: ones(1, 2^17)*2^14

OK Cancel Help Apply

Appendix IX: Python Script to gather Data from Analyser.

```
import csv
from time import sleep
from rfsoc_sam_custom.overlay import Overlay

overlay = Overlay(init_rf_clks=True)

overlay.spectrum_analyzer().children[1]

# Define Transmission Parameters
fc = int(500e6)  # 500 MHz Center Frequency
bw = int(100e3)  # 100 kHz Sweep Bandwidth
df = int(2.5e3)  # 2.5 kHz Sweep Granularity

# Channel Used
# Channel 0: ADC2, DAC2
# Channel 1: ADC1, DAC1
ACTIVE_CHANNEL = 0

# Configure Spectrum Analyser
overlay.radio.receiver.channels[ACTIVE_CHANNEL].frontend.config = {
    'centre_frequency': fc/1e6,
    'nyquist_stopband': 100,
    'decimation_factor': 2048,
    'fftsize': 65536,
    'post_process': 'none',
    'number_frames': 1,
    'window': 'rectangular',
    'spectrum_units': 'dBm',
    'spectrum_enable': True
}

CSV_FILE_NAME = 'sweep.csv'

# Create CSV file and write meta data
with open(CSV_FILE_NAME, 'w') as file:
    writer = csv.DictWriter(file,
fieldnames=overlay.radio.receiver.channels[ACTIVE_CHANNEL].frontend.config.keys())

    # record meta data
    writer.writeheader()
    writer.writerow(overlay.radio.receiver.channels[ACTIVE_CHANNEL].frontend.config)

# write header for body of data
with open(CSV_FILE_NAME, 'a') as file:
    writer = csv.writer(file)

    writer.writerow([])  # write empty row for formatting
    # write x_data as well as a column header for target frequency
    x_data =
overlay.radio.receiver.channels[ACTIVE_CHANNEL].frontend.analyser._spectrum_analyser.
plot._x_data
    writer.writerow( ["Target Frequency"] + list(x_data) )
    # Turn on Transmitter
    overlay.radio.transmitter.channels[ACTIVE_CHANNEL].frontend.config = 
{ 'centre_frequency': (fc-bw)/1e6, 'amplitude': 0.5, 'transmit_enable':True }

    # Start Sweep & Collect Data
    for freq in [(fc+d)/1e6 for d in range(-bw, bw+df, df)]:
        overlay.radio.transmitter.channels[ACTIVE_CHANNEL].frontend.config = 
{ 'centre_frequency': freq, 'amplitude': 0.5 }
        sleep(1)  # Give some time for transmitter to re-configure

        y_data = 
overlay.radio.receiver.channels[ACTIVE_CHANNEL].frontend.analyser._spectrum_analyser.
plot._y_data
        writer.writerow( [freq] + list(y_data) )

# Turn off transmission
overlay.radio.transmitter.channels[ACTIVE_CHANNEL].frontend.config = 
{ 'centre_frequency': (fc-bw)/1e6, 'amplitude': 0.5, 'transmit_enable':False }
```

Appendix X: Github link with project files

https://github.com/CarbonicKevin/2022-Year-ESC499-Thesis