A PXIe Based Scalable MRI Spectrometer for Inhomogeneous Imaging

Guang Yang

School of Engineering and Computer Science

Victoria University of Wellington

This dissertation is submitted for the degree of

Doctor of Philosophy

October 2022

Abstract

Traditional MRI plays a significant role in clinic diagnoses for providing versatility of examinations. However, the high cost and complexity of the MRI system presents a challenge for most people to access. Low-field MRI with low cost and simplicity could be used in particular sites, e.g., on the ambulance or in the rural area, which would benefit more people. A spectrometer is one of the critical parts of a low-field MRI system. Even though the commercial spectrometers are available, they are usually expensive and their closed proprietary designs frustrate adaptation to new techniques or experiments.

To tackle these challenges, a scalable PXIe based spectrometer with a multichannel transceiver was developed for MRI research and development. Key features of the spectrometer include the use of a quad-channel digital receiver for parallel imaging, a dual-channel transmitter for spatial encoding, e.g., TRASE, a high-bandwidth protocol for data transactions and an Ethernet-based user interface.

A 2D MRI system was built to verify the spectrometer. In addition to the spectrometer, an RF coil system containing a solenoid coil for excitation and a receive coil array for receiving, detuning circuits, and pre-amplifiers were built. NMR magnetometry was used to measure the field map of a Halbach magnet. Based on the 2D imaging system, parallel MR signals were captured and verified.

Acknowledgements

There has been a lot of help from workmates, friends, and family members during the research and writing this thesis. I am very grateful for what you have done for me.

Robin Dykstra deserves endless gratitude. He has been an ideal teacher, mentor, and thesis supervisor, offering advice and encouragement with insight. I'm proud of and grateful for my time working with Robin. I would like to thank Paul Teal for attending the regular meeting and being supportive of my work. Sergei Obruchkov offered useful advice to my thesis and work. I really appreciate that.

Thanks to Matthew Bourne, Andrew Ang, and Jemima Teal for their former contribution, which facilitates my work.

I want to thank Dion Thomas, Jie Wang, Zhihua Ren, and Shaoying Huang for their helpful discussion and suggestion. I would like to thank Diana Siwiak for organizing all the activities to add more fun to our life at the university. Many thanks to Cameron Dykstra, Kerry Dykstra and John Zhen, Petrick Galvosas and the lab member for their support.

I would also like to thank Suesan Huen, who supported my family a lot during writing this thesis. She brought us to her dear family (especially Eileen Knights and Lian Chan) and we had the best holiday with her big family in New Zealand. Suesan Huen and her daughter Monica Chin invited me to stay at their places in my hard time in the last few months, which I really appreciate.

I have lovely memory in New Zealand for the education in Victoria University, my diving friends in the clear ocean and also my new family with Suesan Huen.

A special thanks to my parents and my wife for always being there for me, encouraging me to fulfill my dream. And to my baby Xinghan, who brought so much joy to me.

Table of contents

List of figures

List of tables

Nomenclature

ADC Analog-to-Digital Converter

CPLD Complex Programmable Logic Device

DAC Digital-to-Analog Converter

DDS Direct Digital Synthesis

DMA Direct Memory Access

DSP Digital Signal Processor

FPGA Field Programmable Grid Array

LPF Low-pass-filter

PC Personal Computer

PCI Peripheral Component Interconnect

PCIe Peripheral Component Interconnect Express

PXI PCI eXtensions for Instrumentation

PXIe PXI Express

RF Radio Frequency

SNR Signal Noise Ratio

SoC System on Chip

TTL Transistor–Transistor Logic

USB Universal Serial Bus

VHDL VHSIC (very high speed integrated circuit) Hardware Description Language

FID Free Induction Decay

MRI Magnetic Resonance Imaging

NMR Nuclear Magnetic Resonance

Chapter 1

Introduction

1.1 Low-field MRI Applications

Traditional high field MRI (Magnetic Resonance Imaging) scanners play a significant role in clinical diagnoses, such as tumors, stroke, etc. However, accessing the MRI systems is still a challenge for most people globally, according to a review of the accessibility of MRI [1]. The complexity and high cost ($\sim$\$1M/T) of the high field MRI system are the main reasons that prevent people from accessing the health-beneficial technology.

In recent years, low-field MRI has drawn people's attention as it can be compensated with the traditional MRI system and allows the use of the MRI to be expanded. For example, using a permanent magnet [2–6] reduces the challenges of building an MRI system, especially the superconducting magnet and its cryogenic maintenance. Low-field MRI with less weight has the potential to MR imaging at sites, such as the remote areas, sports areas, ICU (intensive care unit), etc. Applying the low-field MRI to such sites could impact health globally. For instance, stroke is a severe brain attack when a burst blood vessel bleeds into the brain, and time is the most critical factor for ensuring their survival and minimizing the extent of brain damage. Radiologists could adequately treat the patient if a low-field MRI system were available in the rural area or on an ambulance.

To date, MRI research have been trending towards high channel-count receivers, transmitters, and gradients for innovative applications [7–9]. However, because of the proprietary and

high integration of the hardware and software in the commercial spectrometers or consoles, it is difficult to make any adaptations. Fortunately, with the advent of modern electronic components or modules, like the multichannel transmitter and receiver on a single chip or commercial module, which includes FPGA (Field Programmable Gate Arrays), ADC (Analog-to-Digital Converter), and DAC (Digital-to-Analog Converter), it is feasible for researchers to build custom consoles, but high synchronization, bandwidth, scalability, and cost pose challenges.

The spectrometer plays a significant part in a low-field MRI system, as it determines the performance in an MRI system. For example, the receiver number affects the scanning time. However, preparing a proper spectrometer for a particular application has several obstacles. Such as commercial spectrometers are usually expensive and hard to modify due to the property; it needs the expertise of electronics and considerable time if researchers tend to build their spectrometers.

In this thesis, the focus is on the development of an MRI spectrometer with a multichannel transceiver based on a PXIe (PCI eXtensions for Instrumentation) chassis.

1.2 Research Objectives and Contributions

1.2.1 Research Objectives

1. Develop a multichannel transceiver board, including dual-channel transmitter and four-channel receiver. The dual-channel transmitter is used for encoding the axial direction and the four-channel receiver is used for parallel imaging. Develop the FPGA firmware, including the NMR (Nuclear Magnetic Resonance) or MRI pulse program and digital receiver to process the NMR/MRI signal.

2. Integrate the PXIe standard transceiver board to the existing PXIe system controller and develop software or user interface on a host computer to control the NMR/MRI experiments.

3. Develop the MRI RF coils, including the transmitting coil and receive coil array. Implement the active/passive detuning/decoupling to the RF coils.

1.2.2 Contributions

The candidate contributed the following:

1. Developed the multichannel transceiver board and FPGA firmware.
2. Integrated the developed transceiver board to the previous system controller on the PXIe chassis. The integrated PXIe chassis was referred to the MRI spectrometer; software was developed on the host computer to allow users to control the NMR/MRI spectrometer.
3. Solenoid transmitter coil and receive coil array were constructed. Active/passive detuning/decoupling was implemented for MRI experiment.

In this project, some work was finished by the co-workers in the laboratory.

1. The PXIe system controller was designed and constructed by Robin Dykstra [10] and his former students [11, 12], Andrew Ang and Matthew Bourne.
2. The pre-amplifier was originally designed by Robin Dykstra.
3. The image reconstruction was done by Paul Teal [13].

1.3 Outline

The following subsections outline the structure of this thesis:

1.3.1 Chapter 2

Chapter 2 firstly presents the fundamental MRI basics and the MRI system. It then reviews the designs of the key components of the prior spectrometers, such as analog receivers, digital

receivers, transmitters, and backplanes. The motivation and architecture of our spectrometer and the overall plan of the MRI system are also illustrated.

1.3.2 Chapter 3

Chapter 3 details the designs of the multichannel transceiver board, which was built from scratch. It starts with the expected functions and design specifications of the spectrometer. Then, the key components, such as ADC, DAC, and FPGA, are introduced regarding the selection considerations, features. After that, the implementation of the hardware components is described. Once the hardware components were implemented, the FPGA firmware was developed part by part to realize individual functions in conjunction with the particular hardware component. Finally, the transceiver board was tested concerning the synchronization of the transmitter and the receiver chain.

1.3.3 Chapter 4

In Chapter 4, the system controller is reviewed regarding the hardware components, FPGA firmware design, PXIe architecture, PetaLinux building, PCIe driver, and API (application programming interface). Later, the PCIe user endpoint was introduced. That user endpoint is a subsystem that builds a connection between AXI memory-mapped in the FPGA on the transceiver board and the PCIe Endpoint, which allows the data to be transferred over PCIe protocol. Then a software architecture for addressing the hardware integration problems was also elaborated on. Finally, NMR experiments were tested on the fully integrated spectrometer.

1.3.4 Chapter 5

Chapter 5 describes the preparation work for the 2D MRI system. Firstly, the rotating frame was constructed for the magnet. Secondly, the field of the Halbach magnet was mapped by building a tiny NMR probe, and sorting out an algorithm for correlating the acquired signal to the actual field. Thirdly, the transmitter coil, receive coil array, and four pre-amplifiers were

constructed. In the end, the parallel imaging experiment was conducted, i.e., four receivers simultaneously acquired the signals from four coils in a coil array.

1.3.5 Chapter 6

Chapter 6 summarizes the achievements and the points to be improved in this study.

1.4 COVID-19 Statement

Initially, we planned to collaborate with the MRI group at Singapore University of Technology and Design to develop a low-field MRI system. More specifically, we intended to focus on the multichannel spectrometer and integrate it with the RF coils and rotating magnet [4] built by that MRI group in Singapore. However, we could not use the coils and magnet due to COVID-19 lockdown and border control in both New Zealand and Singapore. In order to verify the functionalities of the spectrometer, we had built the rotating frame for the existing Halbach magnet, mapped the fields of the magnet, and constructed the excitation coil and the receive coil array in New Zealand.

Chapter 2

Background

2.1 MRI Basics

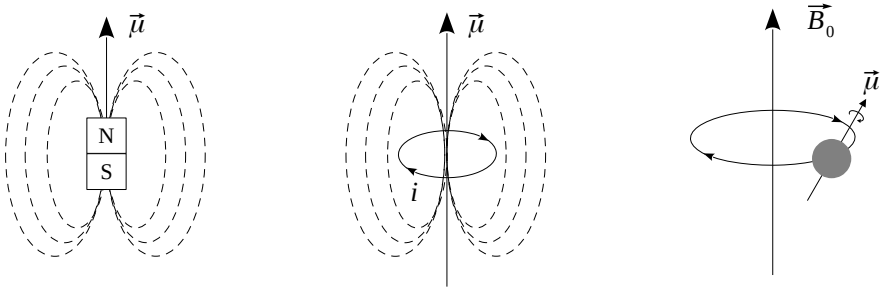
2.1.1 MRI Concepts

Fundamental Interaction

The interaction of a spin with an external magnetic field (B_0) is the fundamental basis of magnetic resonance [14]. A simple explanation can be provided using a classical approach. The object's intrinsic magnetic properties are often seen as coming from a tiny bar magnet with a north and south pole and is called the magnetic dipole moment (μ), as shown in Figure 2.1 a. Alternatively, the magnetic moment can be modeled as the small current going around the loop edge of area (A), as shown in Figure 2.1 b. From the right-hand rule, the vector direction of the magnetic moment is perpendicular to A. That vector quantity is used to measure the tendency of an object to interact with the external magnetic field.

The interaction of the protons in hydrogen with the external magnetic field, B_0, results in the precession of the proton spin about the field direction, as illustrated in Figure 2.1 c. The angular frequency of the precession for the proton magnetic moment vector is given by

$$\omega_0 = \gamma B_0 \tag{2.1}$$



(a) (b) (c)

Fig. 2.1 Interaction of a spin with the magnetic field.

where γ is a constant called the gyromagnetic ratio. For hydrogen proton, the value is about $2.68 \cdot 10^8$ $rad/s/T$, equals 42.58 MHz/T. For instance, for a 1 T magnetic field, the spins precession frequency referred to as the Larmor frequency is 42.58 MHz. Equation 2.1 is referred to as the Larmor equation.

System Magnetization Detection

To initiate precession in the x-y plane, the magnetization vector aligned with the B_0 direction must be tipped away by a perturbing, transient RF field orthogonal to the B_0 direction. Immediately after the RF field, the transverse magnetization, M_0, starts to precess around B_0 in the x-y plane. The complex magnetization can be defined [15] as

$$M_+(t) = M_x(t) + iM_y(t) = M_0 e^{-i\omega_0 t + i\phi_0} \tag{2.2}$$

where the M_x and M_y are the projected magnetization. ω_0 is the angular frequency of the precession. ϕ_0 is the initial phase of the magnetization.

Equation 2.2 reveals the connection between the time-dependence of the complex phase and the magnetization rotation. The initial phase angle, ϕ_0, is the rotated angle from the original direction. The RF field (B_1 created by an RF coil) that turns the magnetization with

90 degrees is called the $\pi/2$ pulse, as shown in Figure 2.2. The coil that generates the RF field is the "transmitter" coil.

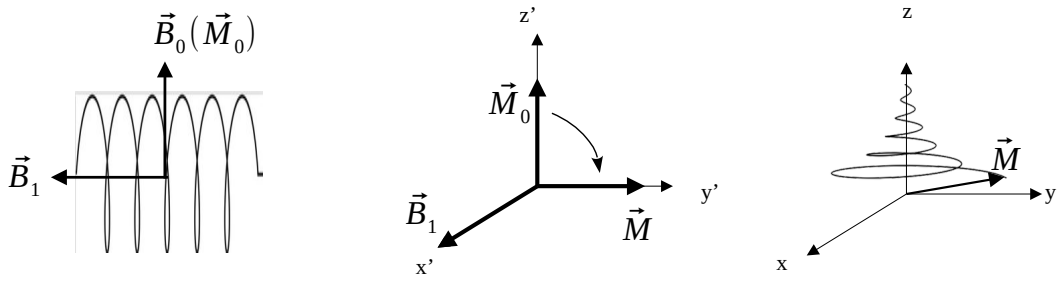


Fig. 2.2 Magnetization. The diagram is redrawn from the reference [15].

Once the M_0 has a transverse component, the net magnetization, $\overrightarrow{M}$, precessions about B_0 can be detected, which is considered as MR signal detection. The detection by a "receive" coil is derived from Faraday's Law of electromagnetic induction, the emf (electromotive force) induced in a coil by a changing magnetic flux can be calculated by:

$$emf = -d\Phi/dt \tag{2.3}$$

where Φ is the flux through the coil:

$$\Phi = \int_{coil\ area} \vec{B} \cdot d\vec{S} \tag{2.4}$$

where $\vec{B}$ is the magnetic field goes the coil area, and the $\vec{S}$ is the effective unit. The flux can be considered as a number of field lines penetrating the effective area of a coil. When applying this mechanism to the Magnetic Resonance Imaging (MRI), where the roles of the magnetization source and the detection coil are exchanged, the flux expressed in Equation 2.4 can be eventually converted to:

$$\Phi_M(t) = \int_{sample} d^3 r \vec{B}^{receive}(\vec{r}) \cdot \vec{M}(\vec{r},t) \tag{2.5}$$

where $B_{receive}$ is the magnetic field produced by the receive coil per unit current,and $\vec{r}$ is the radius vector in the spherical space. The M_0 in Equ 2.2 is the integration of the $\vec{M}(\vec{r},t)$ in the spherical space of radius r, or the $\vec{M}(\vec{r},t)$ can be considered as a volume element of the M_0. In Equation 2.5, the flux is subject to the $\vec{B}^{receive}$ that represents the "receive" field generated by the receive coil at any point where the magnetization is not zero.

That is an example of the reciprocity principle. The voltage induced in the receive coil is given by (the detailed procedures were illustrated in Appendix A.)

$$\begin{aligned} emf &= -\frac{d}{dt}\Phi_M(t) \\ &= -\frac{d}{dt}\int_{sample} d^3 r\vec{M}(\vec{r},t)\cdot \vec{B^{receive}}(\vec{r}) \end{aligned} \tag{2.6}$$

in which it indicates that the sample volume, the strength of M (determined by the static field strength), and the receiving ability of the coil can be found.

Free Induction Decay

When a $\pi/2$ pulse is applied, it rotates the magnetization from the longitudinal direction (along with the B_0 field) to the transverse plane. Then the tipped spins begin to precess freely and collectively. During that process, the time-varying coherent magnetic field, which comes from the overall precess spins, induces a small emf in the RF coil that is properly placed for detecting the corresponding changes of flux. This experiment is called a free induction decay (FID). This simplest NMR or MRI experiment is helpful in practical applications. For example, when tuning the RF coils or optimizing the system response by adjusting the RF amplitude and duration from the amplitude of the signals. In this study, FID is used for field mapping and 2-D imaging experiments, which is elaborated on in Chapter 5.

Relaxation Time

After the magnetization is rotated into the transverse plane, it will tend to recover to the original direction of the $\vec{B_0}$ field. The rate of the regrowth is called longitudinal relaxation

time, T_1, which can be described by

$$M_z(t) = M_0(1 - e^{-t/T_1}) \tag{2.7}$$

where the M_0 represents the initial magnetization. Because this process happens due to the interaction of the spins with the surrounding environment, T_1 can be used to characterize materials and monitor chemical changes. An example can be seen in Chapter 5, using copper sulfate water as a sample to accelerate the experiments.

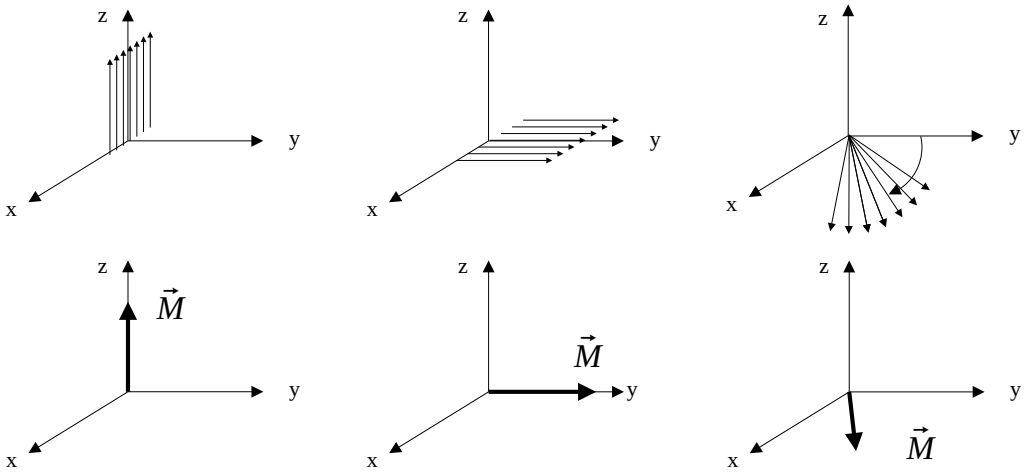


Fig. 2.3 Transverse relaxation. The diagram is redrawn from the reference [15].

Another process followed by the $\pi/2$ pulse is the transverse relaxation, as diagrammed in Figure 2.3. In the upper three figures, a set of spins (isochromats) was tipped into the transverse plane, and then they will align with the y-axis in the laboratory frame (in the up-middle figure). Once the $\pi/2$ pulse is switched off, the individual spin begins to precess in phase in the transverse plane. However, due to the local fields' variations resulting in different precessional frequencies, the protons precess with different speeds and then become out of phase. The M_{xy} decreases over time due to transverse relaxation, which can be described as

$$M_{xy} = M_0 e^{-t/T_2} \tag{2.8}$$

where T_2 is referred to the transverse relaxation time constant.

Spin Echo

The spin echo sequence is based on applying two pulses: a $\pi/2$ for tipping the magnetization to the x-y plane and another following pulse for "refocusing" the spins out of phase. The "refocusing" pulse used for tipping the magnetization by 180° is called π pulse, and the detected signal after refocusing is called an echo. This process is also known as the Hahn-spin echo experiment [16].

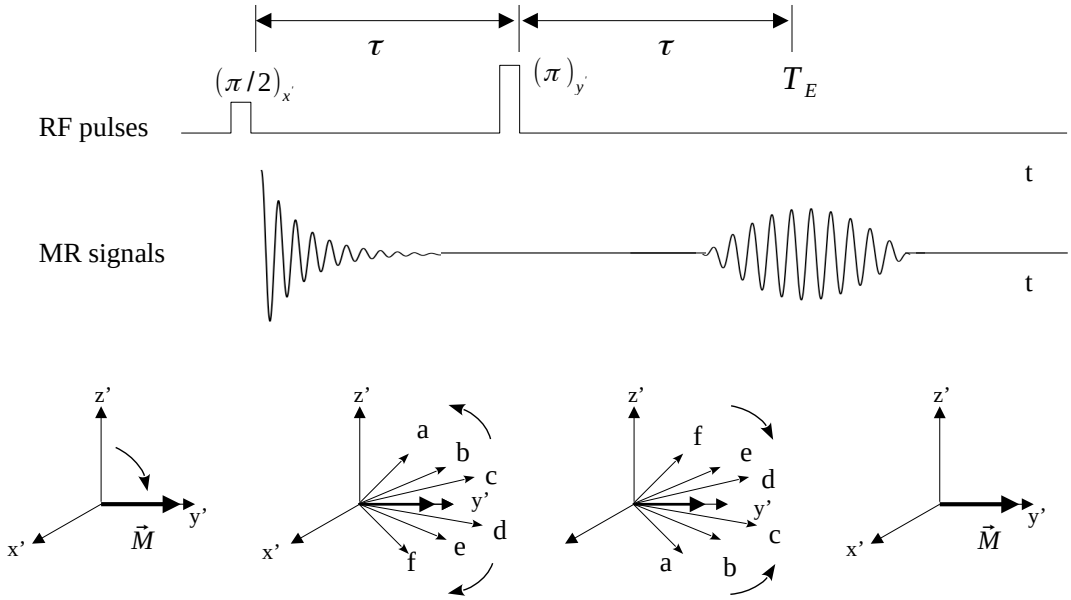


Fig. 2.4 Spin Echo diagram. The diagram is redrawn from the reference [15].

2.1.2 Magnetic Resonance Imaging

Frequency Encoding

When the static field B_0 is homogeneous, and RF excitation is performed uniformly, all protons will precess the same way-in phase, which leads to the signals derived from spins at various locations in the imaging object being indistinguishable [17, 18]. Fortunately, it is known that the spin in a magnetic field precesses about the field at the "Larmor Frequency", which, in turn, depends on the magnitude of the field itself. Therefore, the Larmor Frequencies are spatially varying when a spatially varying magnetic field is applied across the object.

The MR signals that carry different frequency components can be separated to give spatial information about the object. That is the key point of spatial encoding, which opens the door to MR imaging. Practically, adding a spatially changing magnetic field across the object produces signals which have spatially varying frequency components according to

$$\omega(x) = \gamma B(x) \tag{2.9}$$

where x represents the spatial coordinate along the direction of the changing magnetic field. That is fundamental for imaging as it builds a connection between the frequency domain of acquired signals and the physical locations. If the gradient field is linear, which is prevalent in traditional MRI systems, Fourier transformation can be used for imaging reconstruction. However, linearity is not essential to image encoding; other reconstruction methods rather than using Fourier transformation should be explored.

k-space and 1D Imaging

The k-space represents the spatial frequency information in the MRI imaging, either one, two, or three dimensions of an object.

When applying a constant gradient (G) over a time interval (0,t), k-space can be expressed:

$$k = \frac{\gamma}{2\pi} Gt \tag{2.10}$$

The simplicity of the model is used to demonstrate the 1D imaging method, as shown in Figure 2.5.

The RF pulse is applied to tip the spins into the transverse plane, and the field of gradient (G_z) is presented after the $\pi/2$ pulse. During the time when presenting the gradient, the precessional rates of the two spins will differ slightly from the Larmor frequency: the spin at z_0 rotates clockwise, and the spin at $-z_0$ rotates counterclockwise at the same rate.

Then the expected superposition of the envelope signal can be acquired. The signal is

$$s(t) = s_0 e^{-i\gamma G z_0 t} + s_0 e^{i\gamma G z_0 t} = 2 s_0 cos(\gamma G z_0 t) \quad 0 < t < t_2 \tag{2.11}$$

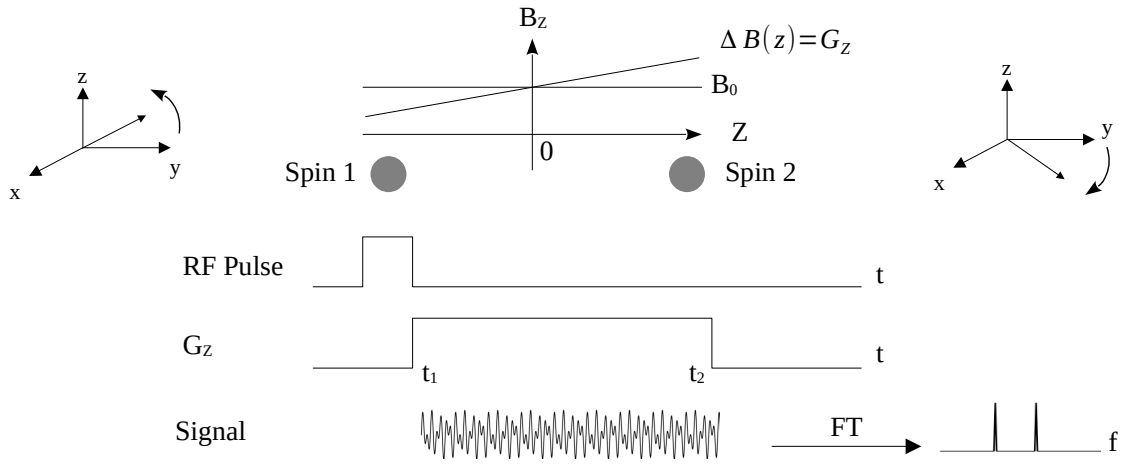


Fig. 2.5 An example of 1D MRI experiment. The diagram is redrawn from the reference [15].

By using Equation 2.10, the signal expression (2.11) becomes:

$$s(k) = 2s_0 cos(2\pi k z_0) \tag{2.12}$$

Eventually, the distance information of the spins can be revolved by Fourier transformation, which transforms the signal in the time domain to the frequency domain.

2.1.3 Parallel Imaging

Parallel imaging is a technique in which multiple receive coils are placed to assist spatial localization of the MR signal. With the additional information of the placement of receive coils, in the number of encoding steps during acquisition can be reduced resulting in a several-fold reduction in imaging time. With this mechanism, some associated image reconstruction algorithms were invented, such as SENSE [19], SMASH [20], GRAPPA [21].

The reduced imaging time brings several benefits: more comfortable for patients, fewer motion artifacts, higher resolution in a region-of-interest etc. Given these benefits, parallel imaging has become the prevalent method used in MRI systems and has become a significant concept in designing the RF coils. For example, according to the reciprocity principle, the spatial sensitivities of one coil is its local field profile describing the ability of that coil can pick up MR signal from that region.

2.2 MRI Systems

2.2.1 Traditional High-Field MRI System

A traditional high-field MRI system mainly consists of the following hardware components [22, 23]:

1. **Superconducting magnet.** This provides the static homogeneous magnetic field, B_0, which polarizes the magnetic moments of the spins. To make the field strong and uniform, numerous winding and a cooling system for the electromagnet lead to the magnet being very heavy, usually around 5-10 tons.

2. **Gradient unit.** The gradient unit, consisting of gradient coils and gradient power amplifiers, produces the linear fields, G_x, G_y, G_z along x, y, z directions. These gradient fields are varied and switched during the experiment. Due to the requirements of the gradient fields, such as the water-cooled system of the coil and high gradient strength for dominating B_0 inhomogeneity, the unit is heavy and power-hungry.

3. **Transmitting and receiving units.** The transmitting unit is compromised of transmitting coil or B_1 coil tuned to resonant frequency, and a high power amplifier is used for tipping the spins from the alignment with the B_0 field. The receiving unit made of receive coils and pre-amplifiers is used for acquiring the MR signals for further processing.

4. **Spectrometer.** A spectrometer generates the pulse sequences and TTL signals for switching the associated devices, such as high power amplifier, gradient power amplifiers, etc., and acquires MR signals during the imaging experiment.

2.2.2 Low-Field MRI System

Changing from the high field produced by a superconducting magnet to a low field usually created by a permanent magnet has the benefits of alleviating cost, siting, and operational burdens. However, that change leads to the low sensitivity of low-field MRI, which means

that the SNR becomes low, and the resolution will not be as high as the high field MRI. Considering that the low-field MRI system has the potential of being accessed by more people because of the lower cost and the portability for point-of-care due to the lightweight, it is worthy of sacrificing image quality to a certain degree to reach larger patient populations and regions.

In the academic field, several medical applications [24–26] based on low-field MRI systems were explored. Recent research about low-field MRI has been reviewed [27, 28] regarding several aspects such as the signal-noise ratio, developments in hardware (e.g., spectrometer, magnet, gradients, and RF coils), and image reconstruction.

2.2.3 Key Components for Low-Field MRI

Permanent Magnet

Compared with a superconducting magnet, a permanent magnet has two benefits: power-free because of the stored energy in the magnetized material and cryogen-free. However, temperature should be stable to ensure the static field is constant. A Halbach magnet array [29] is an arrangement of permanent magnet that augments the magnetic field of one side and cancel the field to zero on the other side. Many researchers followed the idea and presented array-based designs for either NMR or MRI. Generally, for NMR, the hole of the Halbach magnet array is designed small to achieve a higher static field. That usually applies to chemical analysis. For MRI applications, for example, head imaging, the hole needs to be larger to accommodate a head while maintaining the field at a certain strength. There are two kinds of Halbach magnets regarding the direction of the static field. One is that the B_0 direction is on the transverse plane [3, 4, 30, 31], and another one is that the B_0 direction is along with the magnet's axis [32]. The left side picture in Figure 2.7 shows the rotating magnet, and the right picture shows the magnetic field in the transverse plane.

Spectrometer

The spectrometer is a key part of the low-field MRI system. Using a commercial spectrometer suitable for a low-field MRI system is an option, but there are several obstacles. With the proprietary rights and the high integration of the commercial spectrometers, it is hard for researchers to modify it to achieve innovative experiments. For example, some spectrometers just have one receive channel; it is challenging to realize parallel imaging, which requires multiple receive channels. Another obstacle is the high price due to the nonrecurring development cost and low-volume production of special functions in the spectrometer. To overcome those obstacles, researchers have launched many spectrometer designs.

RF Coils

Similar to the traditional MRI system, the RF coils in low-field MRI consists of a transmitter coil for tipping the magnetization to the transverse plane and a receiving coil to detect the MR signals. In low-field MRI, the SNR property of receive coils is of importance for the quality of an image. There are two factors associated with the SNR of a coil: sensitivity and noise. The formula is given by [33]:

$$\psi_{rms} = K\eta M_0 \sqrt{\frac{\mu_0 Q \omega_0 V_c}{4FkT_c\delta f}} \tag{2.13}$$

where K is the numerical factor that is determined by the receiving coil geometer; η represents the how the coil volume matches the sample; M_0 is the magnetization that is proportional to the field strength B_0; μ is the permeability; Q is the coil factor; ω is the Larmor angular frequency; V_c is the coil volume; F is the noise figure of the pre-amplifier; k is the Boltzmann's constant; T_c is the probe temperature; and the Δf is the bandwidth of the receiver.

There are a number of receive coil designs adopted in MRI [34, 35], such as saddle coil, birdcage coil, surface coil, and phased array coils, as shown in Figure 2.6.

RF coil design is an important part of an complete imaging system, as shown in Equation 2.13, several factors must be considered to achieve optimal imaging result, especially in

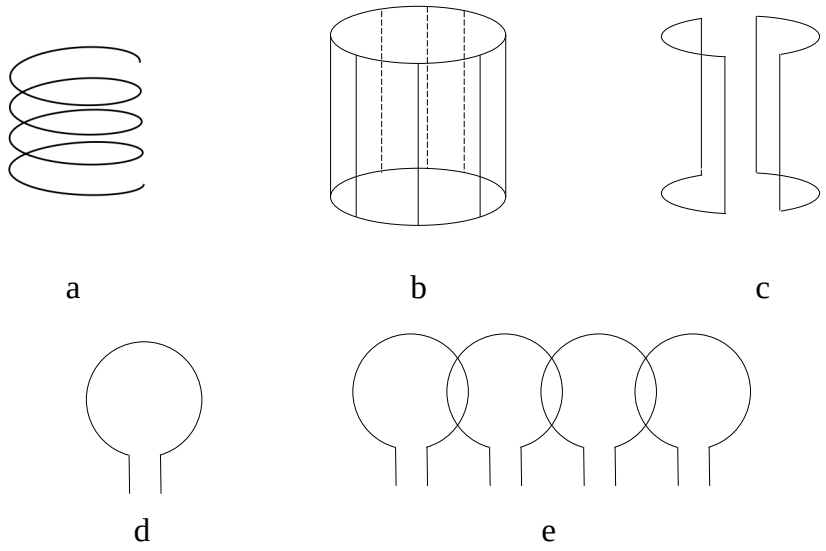


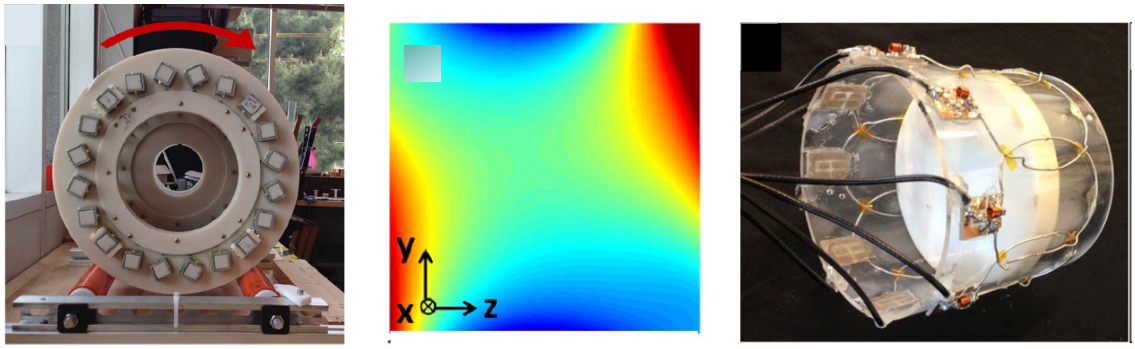
Fig. 2.6 Different MRI coils. a. solenoid coil. b. birdcage coil. c. saddle coil. d. single surface coil. d. coil array.

the low-field MRI, such as the geometry of the coil, Q, etc. In addition, when assembling the RF coils, transmit coil, and phased array coils (for parallel imaging), decoupling and detuning are needed to guarantee the individual coil could have a proper frequency response during a pulse sequence. In other words, if the resonant frequency of the transmitter coil does not match with the Larmor frequency due to the coupling from the receive coil, the transmitter coil cannot transform the power from power amplifier efficiently for stimulating the protons; if receive coil's (coils') resonant condition was spoiled by the transmitter coil or by the adjacent receive coil, then the receive coil(s) cannot detect the changing magnetization and no signal can be seen from the receive coil(s).

Spatial Encoding

Encoding for low field MRI could be the same as the high field MRI which employs the linear gradient fields for spatial encoding and uses Fourier transformation for image reconstruction. However, the initial purposes of low field MRI are to reduce cost, lower weight, etc. Eliminating the gradient unit consisting of gradient coils and power-hungry gradient amplifiers makes it closer to this purpose. Recall that locations of imaging objects

can be distinguished by introducing non-uniform fields on the static field B_0, which does not need to be linear. For example, Hennig proposed the concept for spatial encoding by non-unidirectional, nonbijective spatial encoding magnetic fields (SEMs) [36]. In this way, an image reconstruction algorithm other than Fourier transformation needs to be developed to reconstruct the image.



(a) (b) (c)

Fig. 2.7 (a) Permanent magnet [3]. (b). Filed map. (c). Receive coil array.

In 2015, Cooley introduced a 2-D imaging MRI scanner without a gradient unit [3]. In that system, the natural inhomogeneous fields of the transverse plane (Figure 2.7) were used for spatial encoding. The individual coil in the receive coil array has its own sensitivity to its location; therefore, the phase-varied MR signal can be differentiated. By rotating the magnet and stabilizing the receive coils and sample, which essentially changes the static field and leads to a change of the coil sensitivity, more data can be gathered to improve the SNR.

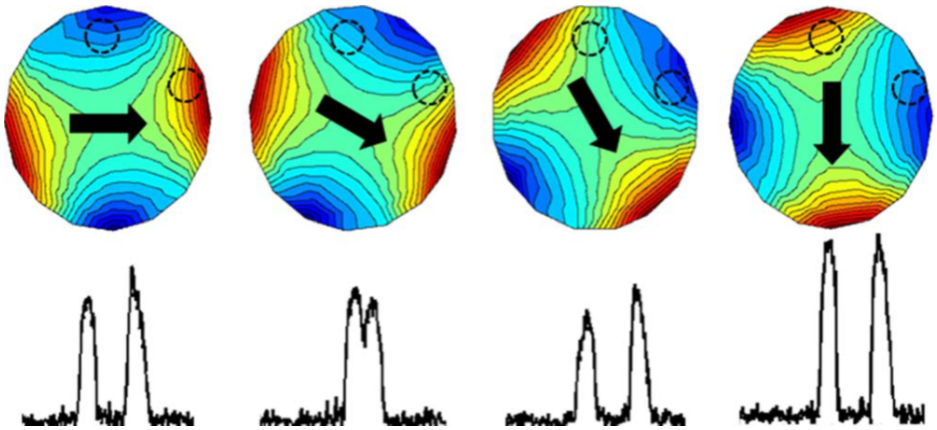


Fig. 2.8 Projection. [3]

To avoid using a gradient power amplifier for the third dimension encoding, we intended to use the transmit RF (B_1^+) phase encoding. The phase gradient is produced by the TRansmit Array Spatial Encoding coil array (TRASE) [37–41], which consists of two nested cylindrical coils: a birdcage coil and a Maxwell coil, as shown in Figure 2.9 (a). The equation in Figure 2.9 (b) illustrates that the coils can be designed to approximate the cosine and sine for a linear slope. By applying a 180-degree phase shift to the Maxwell coil, the sign of the phase slope can be flipped. Figure 2.9 c shows the ideal amplitude and phase results of designed coils: the amplitude is constant, and phases are linear over the field-of-view.

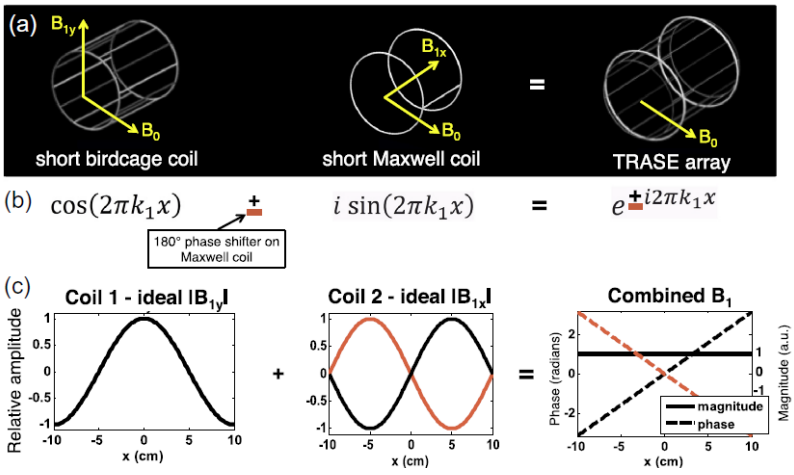


Fig. 2.9 TRASE coil array [42].

In the previous study [37], a power splitter was used to divide the power at a specific ratio to guarantee a constant amplitude. A 180-degree phase shifter was used to flip the transmitting signal phase to change the phase slope sign over the field-of-view. That method works, but the phase shifter consisting of lumped elements has a narrow bandwidth, and parameters of elements in the power splitter must be adjusted. These problems can be addressed by introducing a dual-channel transmitter because the amplitudes and the phases of the two transmitting outputs can be configured individually.

2.3 Review the Designs of Spectrometers

2.3.1 Key Components

Analog Receiver

An analog receiver is a conventional way of detecting the MR signal [43]. Figure 2.10 illustrates one of the conventional analog receivers. The amplified signals are sent to two separate mixers to be mixed with the sine and cosine waveform that is created by a complex synthesizer. The multiplied signals are amplified and filtered by low-pass-filters (LPF) for eliminating the higher frequency components from the mixing stage. The filtered signals are amplified again to match the input range of ADC (Analog to Digital Convertor) for converting the analog signals to digital for the post processing.

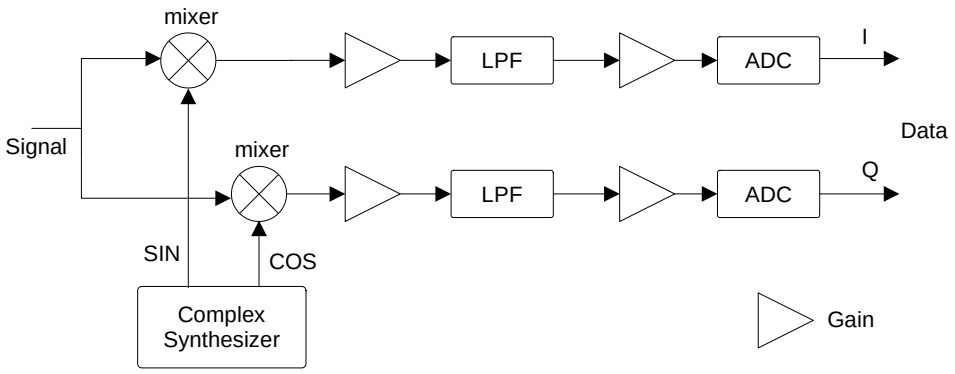


Fig. 2.10 Diagram of the analog receiver.

Digital Receiver

The digital receiver is a method that deals with the received signal digitally rather than in an analog way. More specifically, a digital receiver manages with the mixing in a numerical method, i.e., the signals to be mixed are digital signals output from ADC.

The AD6620 is a digital receive signal processor. In 2002, Michel [44] proposed to use a digital signal processor (DSP) as the digital receiver, as illustrated in Figure 2.11. In that implementation, the amplified MR signal is filtered by either a band-pass-filter (BPS) or

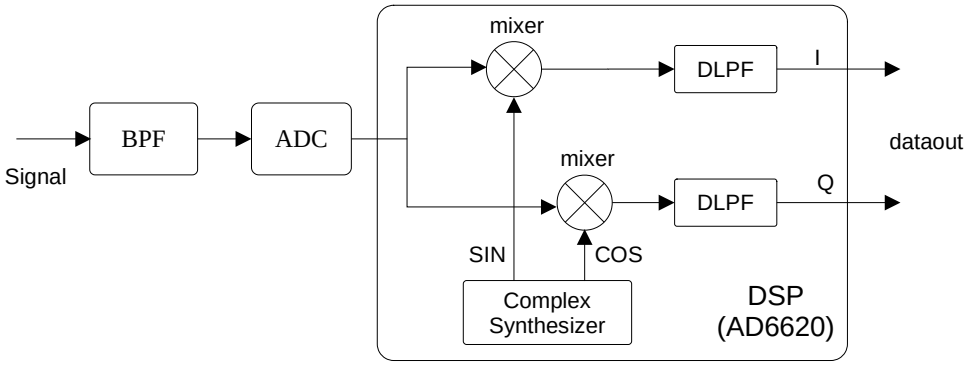


Fig. 2.11 Digital receiver in the DSP. DLPF represents the digital low-pass-filter.

low-pass-filter to eliminate the unrelated components. Then the filtered signal is digitized via an ADC. Later, in the digital receiver, the digital MR signal is multiplexed numerically with synthesized sine and cosine waveform separately to generate quadrature outputs. The low-pass-filter digitally filters the output to reduce the bandwidth and finally down-sampled to reduce the output data rates.

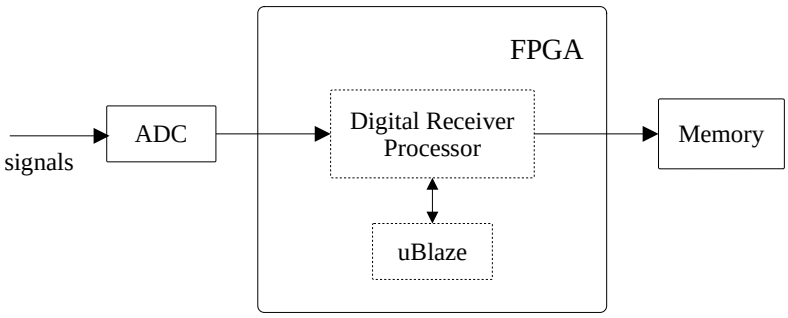


Fig. 2.12 Digital receiver in the FPGA.

FPGA developments in the last decades, such as the increment in logic gates and speed, are large enough and fast enough to implement digital functions as found in the AD6620 or other devices. FPGA-based designs have the following advantages: the processing stages can be configured easily with a soft processor embedded inside an FPGA. The raw ADC data can be accessed, which is useful to verify the correct ADC operation.

To implement the digital receiver in the FPGA, the synthesizer, mixer, low-pass-filter, and down-sampling an FPGA modules are employed in the FPGA design [45], as shown in Figure 2.12. The FPGA is not only used to implement digital receivers; other NMR/MRI-related functionalities can also be implemented, such as the pulse program generator. Then the architecture of the receiver chain is simplified to LPF + ADC + FPGA, which means that the DSP chip can be eliminated.

A digital receiver yields several design advantages over the analog receiver.

1. Firstly, with the digital receiver, the synthesizer, including the local oscillator, analog mixer, are eliminated, which simplifies the circuit design and lowers the cost.

2. A digital receiver has merits of linearity, dynamic range, gain stability, phase, DC offset, and signal distortion over the analog receiver.

Transmitter

In an MRI spectrometer, the pulses are the outputs from a transmitter mainly consisting of a DDS (Direct digital synthesis) and a DAC. The DDS is a device that generates a time-varying signal in digital form, and the DAC converts the digital signal to an analog signal. A processor must be adopted to configure the DDS for such parameters to generate pulses with tunable frequency, phase, and time interval. Therefore, the basic architecture of the transmitter would be processor + DDS + DAC.

Figure 2.13(a) shows an example of one transmitter architecture [45]. The AD9951 is a device implanted with both DDS and DAC to achieve the functionality of a transmitter in conjunction with a processor. In that implementation, an analog sine waveform with configurable frequency and phase can be generated. Now that the soft processor in FPGA is used in the receiver chain, it also could be used to develop a transmitter. In addition, the DDS can be shifted to the FPGA, as shown in Figure 2.13(b). This new architecture has the following benefits:

1. The DDS modules in one FPGA for digital receiver and transmitter bring advantage to synchronizing a pulse program in NMR/MRI experiments.

2. The DDS module for the transmitter can be configured to generate arbitrary digital waveform rather than just a sine waveform. The arbitrary digital waveform is then converted to arbitrary analog pulse via DAC, which is useful for particular pulse sequences.

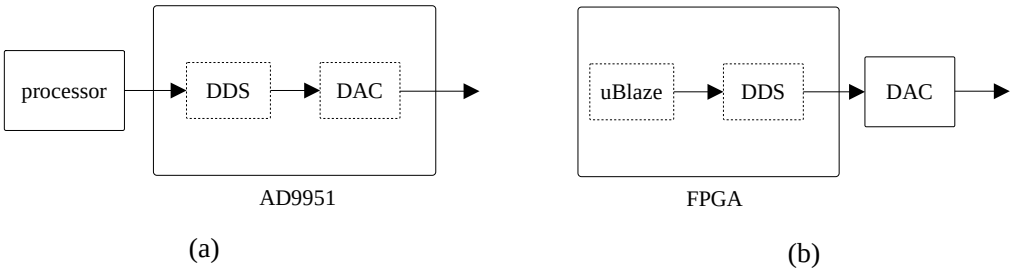


Fig. 2.13 Different approaches to implement the transmitter.

Pulse Program Generator

The pulse program generator cooperates with the mentioned transmitter, and the digital receiver to manage the pulses and acquisitions with a constrained timeline. Takeda developed the pulse program generator by using the VHDL language to describe the functionality [46, 47]. Fortunately, with the FPGA development and the associated software design kit, abundant FPGA cores can be used for free. For example, in the block design of Xilinx Vivado software, the FPGA modules of soft processor, DDS, mixer, low-pass-filter, decimator, etc., could be implemented directly without VHDL coding. Compared with VHDL coding, the block design also benefits from a clear hierarchy of the design. In addition, the embedded application for the soft processor in the implementation of the pulse program generator facilitates the flexibility of the spectrometer, such as different pulse programs can be developed for different MRI experiments. This is elaborated on in Chapter 3 and Chapter 4.

2.3.2 Custom Spectrometers

Multiple spectrometers with different characters are presented on the market or in academic fields. This subsection introduces the key specifications of them and detail three typical home-made spectrometers, such as the OPENCORE NMR spectrometer, Medusa MRI spectrometer, and Micro Spec NMR spectrometer.

Custom Spectrometers List

Table 2.1 lists the key specifications of spectrometers of Medusa [48], OCRA [49], OPENCORE [46, 47], MiniSpec [45], gr-MRI [50], Tecmag [51], Kea [52], RS2D [53]. The specifications supply transmitter channel count, receive channel count, gradient unit, frequency range, GUI, and backplane information. The transmitter channel count determines how many transmitter coils the spectrometer could drive. The receive channel count is related to the number of receive coils; at least two receive channels are needed for parallel imaging MRI. The gradient unit drives the gradient coils via a high gradient power amplifier to encode the imaging area, which is required in traditional MRI systems. The frequency range determines the application fields of the spectrometer. GUI or user interface describes the platform through which the users could manage the experiments and view the results. A backplane is a backboard to connect other individual boards; it could supply the power, trigger lanes, and data transfer lanes.

Table 2.1 Key Specifications

Name	Description	Tx-ch	Rx-ch	Gradient	Frequency	GUI	Backplane
Medusa	Homebuilt scalable spectrometer with multichannel. Processor, ARM. Digital receiver in DSP.	2	4	Yes	0-100 MHz	Matlab	No
OCRA	Based on Red Pitaya, which has dual-channel ADC and DAC.	2	2	Yes	0-40 MHz	Python	No
OPENCORE	Open-source. Pulse program generator in FPGA to interact with peripheral analog circuits such as transmitter and analog receiver.	3	1	No	0-400 MHz	C++	No
gr-MRI	Software-based spectrometer.	1	1	Yes		iPython	No
MicroSpec	Commercial spectrometer.	1	1	No		Python	Custom
Tecmag	Red Stone. Commercial spectrometer.	up to 128	up to 512	Yes	2K-3.5GHz	Commercial	Custom
Kea2	Commercial spectrometer.	1	1	Yes	1-100 MHz	Prospa	Custom
RS^2D	Cameleon4. Commercial spectrometer.	3	up to 16	Yes	1-900MHz	PRim	No

OPENCORE NMR Spectrometer

OPENCORE NMR is an open-source toolkit for implementing an NMR spectrometer, including the source code or document and the graphical user interface software. The spectrometer has the size of a laptop computer, making it portable and low-cost compared to a commercial one.

Figure 2.14 shows the diagram of the OPENCORE spectrometer, which mainly consists of the digital part (FPGA) and analog part, including DDS, DAC, mixers, and ADC. Inside the FPGA, multiple functional modules, such as pulse program generator (PPG), digital receiver, and USB interface, are developed by VHDL language.

The analog transmitter part (the yellow region in Figure 2.14) illustrates how the RF pulses are generated in cooperating with PPG from the FPGA. DDS(I), operating at a constant frequency of 20 MHz, is phase configured by the PPG via the DDS driver. And the DDS(II) with constant phase is frequency configured directly by the PPG. In order to generate a signal with a frequency f, the output frequency of DDS(II) is configured to either f + 20 MHz or f - 20 MHz. The two signals from DDS(I) and DDS(II) outputs are sent to the mixer, AD8343, that generates a signal at a frequency of f and an additional frequency of f+/-40 MHz. The output from the mixer is then filtered through a band-pass-filter to eliminate the unwanted frequencies. In the end, the generated pulse is amplitude-modulated by another mixer and then sent out across a switch that is used for gating.

The analog receive part (the green region in Figure 2.14) shows how the NMR data is acquired and processed. The NMR signal at a frequency of $f + \Delta$ is amplified by a low-noise amplifier, and then the amplified signal is mixed with the reference signal, which is also generated by DDS(II). The higher frequency is eliminated through a low-pass-filter and the signal at a frequency of $20 + \Delta$ MHz is left. The switch controlled by the PPG is used to acquire the signal in the window of receiving. The NMR signal is amplified again to meet the voltage level of the analog-to-digital receiver, which is used for digitizing the signal for further processing in the digital receive module inside the FPGA. In the digital receiver module in the FPGA, the submodules, such as quadrature demodulator, filter, and signal

accumulator,to process the data, and the result is eventually sent to PC through the receiver interface module.

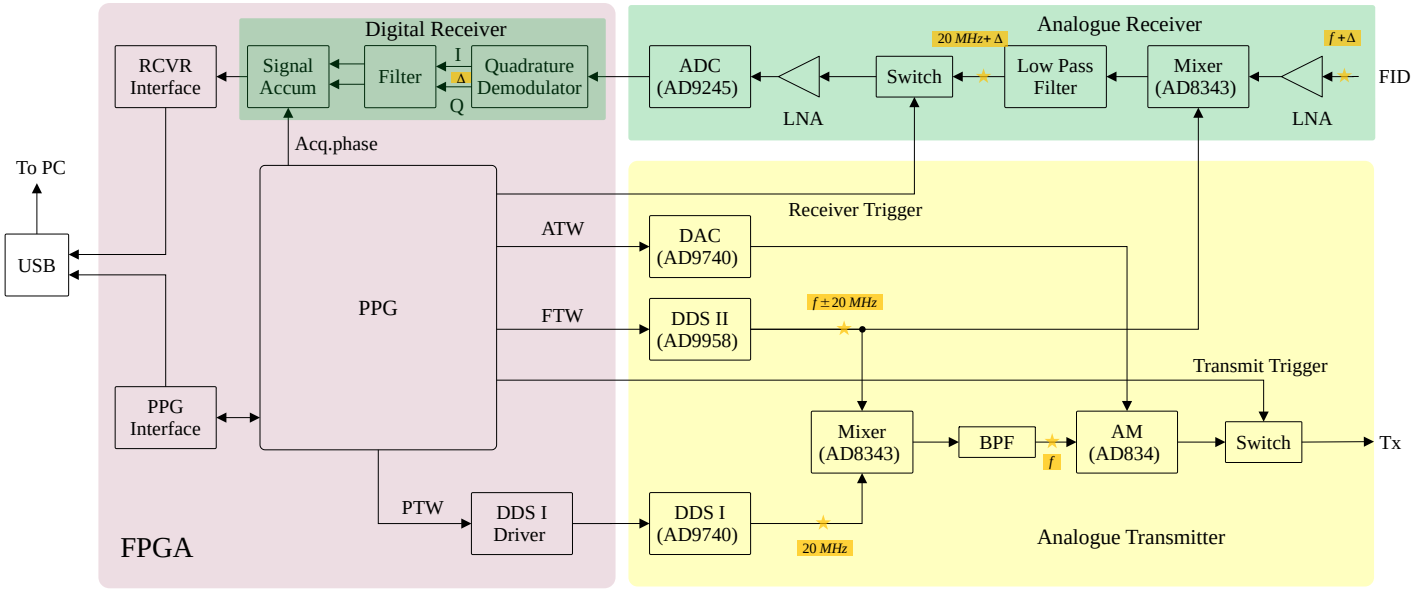


Fig. 2.14 Diagram of OPENCORE spectrometer [46, 47]. PPG, pulse program. PTW, phase tuning words. ATW, amplitude tuning words. FTW, frequency tuning words. BFP, band-pass-filter. AM, amplitude modulator. LNA, low-noise-amplifier. RCVR, receiver.

Medusa Console

The Medusa console is targeted to achieve basic MRI console functions including multichannel transmitting, multichannel receiving, gradient waveform generation, TTL signals (for controlling amplifiers), and software platform (for managing experiments by controlling the console).

Figure 2.15 shows the architectural design of the Medusa spectrometer. It used distributed processing and buffering for dealing with high data rates and stringent synchronization, which is significant for multichannel MRI. Furthermore, an USB bus bridges the PC and system controller, which mainly consists of an ARM processor and a CPLD (Complex Programmable Logic Device). To achieve multichannel transceivers, multiple individual RF modules were constructed. Each module supplies one digital receiver, two transmitters, and gating signals. The backbone bus supplies the configuration, control, clocking, and USB data link back to the host PC.

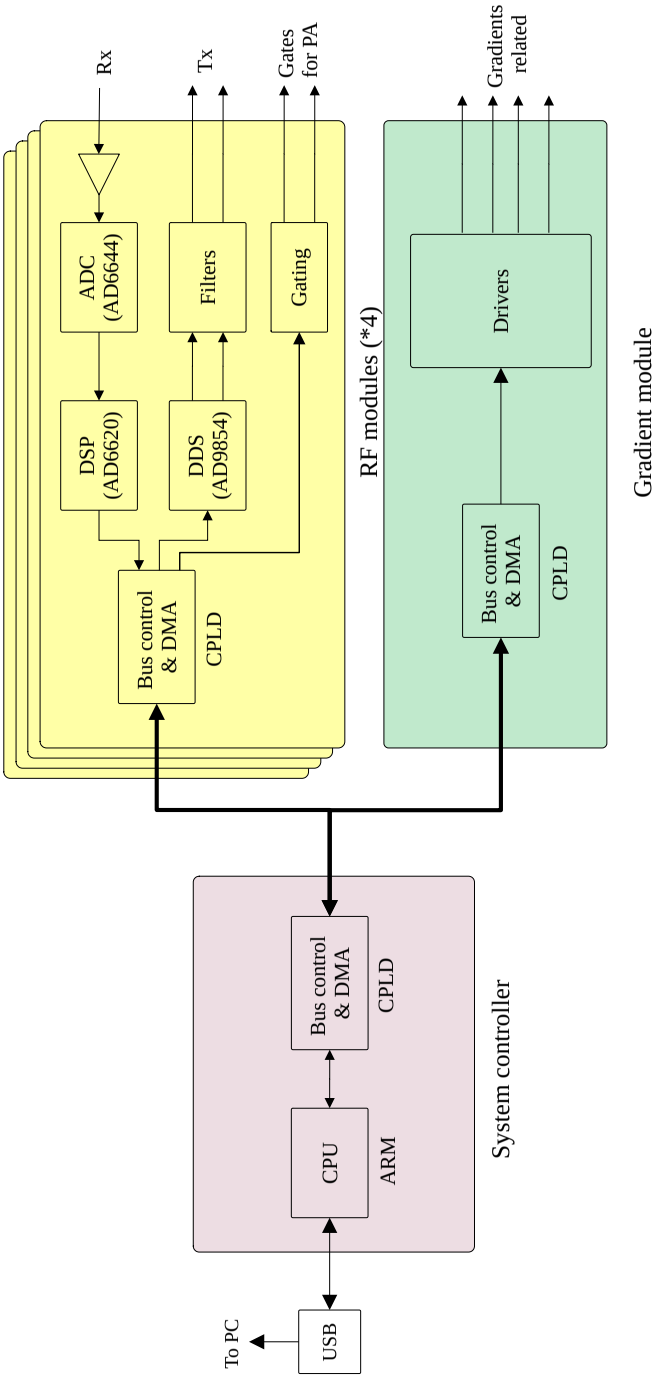


Fig. 2.15 Diagram of Medusa spectrometer [48].

MicroSpec

The "MicroSpec" spectrometer [45] was built based on the SoC FPGA module (System on Chip FPGA, Xilinx Zynq 7020, MicroZed module). A Linux OS was installed on the hardcore processor to provide a web-server system via Wi-Fi on that SoC FPGA The pulse program generator and digital receiver were implemented in the FPGA fabric. Figure 2.16 outlines the spectrometer architecture, and Figure 2.17 shows the picture of the key PCB board in the spectrometer. As seen in Figure 2.17, the SoC FPGA module is carried by a custom board on which the RF transceiver, including a single-channel transmitter and single channel receiver, is implemented. Using that architecture, a custom board to carry an FPGA module, bring several benefits, such as less effort on the PCB design of the FPGA, more compact, and reduced cost. In the next context, several more FPGA modules are used for these reasons.

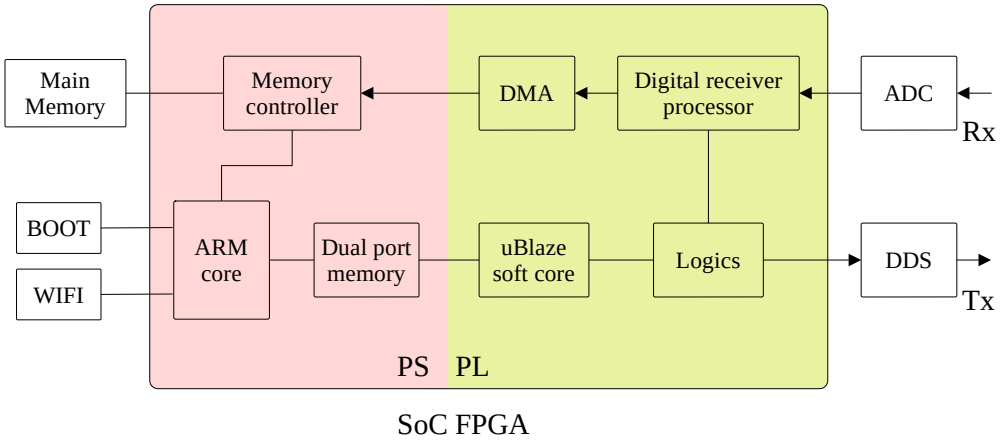


Fig. 2.16 Outline of the MicroSpec spectrometer [45].

2.3.3 Backplanes

A backplane is a bridge board or bus-wire that connects multiple function boards to achieve a scalable instrument; that strategy allows the instrument to be extendable or flexible. In NMR or MRI spectrometer particularly, the backplane can act as a platform to integrate different function modules, such as controller, amplifiers, receivers, gradient units. As listed

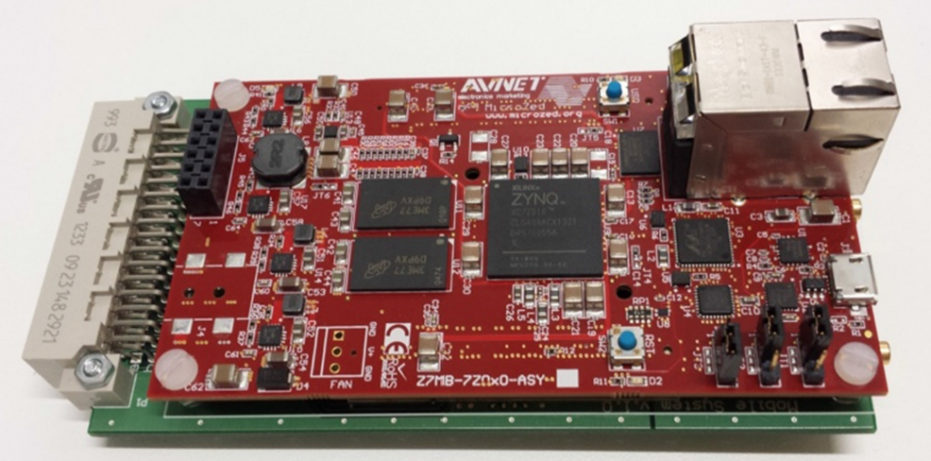


Fig. 2.17 Digital transceiver carrier board with MicroZed SOC module [45].

in Table 2.1, some spectrometers do not have a backplane, and some of them have their custom backplanes. Extension is not applicable for the spectrometers without any backplane, and the non-standard backplane creates barriers for collaboration between different groups. There are several standard backplanes, such as the GPIB (General Purpose Bus Interface), VXI(VME eXtensions for Instrumentation), and PXI (PCI eXtensions for Instrumentation). The limitation of GPIB is that the bandwidth is limited.

PXIe Chassis

The PXIe (PCI eXtensions for Instrumentation) chassis provides a backplane option as the PCIe bus supports significant data bandwidth, compatibility, and synchronization. The synchronization is facilitated by the differential system clock, differential signalling, and differential star triggering. Using this standard allows one to make use of modules provided by many vendors. That standard also enables scalability or flexibility that helps, for example, an MRI spectrometer to be extendable. The range of a PXIe chassis can be variable from 4 to 18 slots, which enables the instrument to be flexible for different purposes. Figure 2.18 shows a PXIe chassis that has eight slots, one system controller for managing the whole system, and seven peripheral slots to realize special functions. The backplane supplies the standard connectors for power supply, PCIe connection between the system controller and peripheral boards, the connection between peripheral boards, etc.

Fig. 2.18 PXIe chassis, NI 1062Q [54].

Figure 2.19 diagrams the topology of PCI Express. PCIe is a point-to-point topology, with separate serial links connecting every device to the root complex (host). The PCIe devices communicate via a logical connection called a link, which consists of variable lanes (can vary from x1, x2, x4, x8, x16, x32). As illustrated in Figure 2.20, each lane composes of two differential signaling pairs, one pair for transmitting and another pair for receiving.

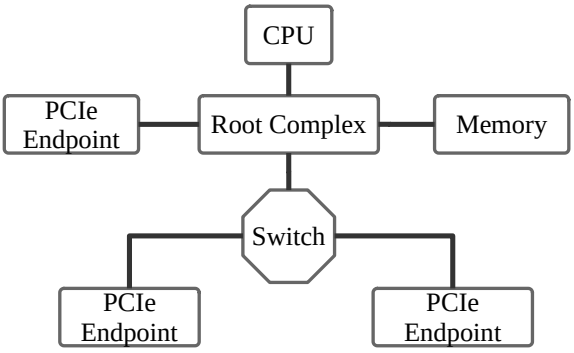


Fig. 2.19 PCI Express topology [55].

Commercial PXIe modules are usually expensive and contain proprietary Intellectual Property (IP). System developers have no choice but to purchase a complete vendor platform or if developing their hardware are still required to pay for IP. The system complexity is a potential barrier to many university-based researchers who want to develop their instrumentation hardware.

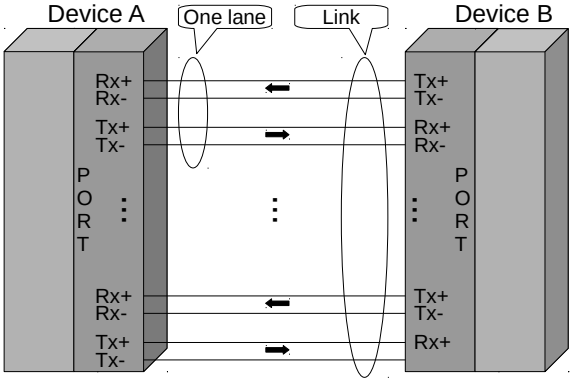


Fig. 2.20 PCI Express link [55].

2.4 Our MRI Spectrometer

2.4.1 Motivation

As mentioned in the custom spectrometer section, the "academic" spectrometers, such as Medusa, OCRA, or OPENCORE do not have a backplane, meaning that it is hard to expand the receive channels for parallel imaging. The commercial spectrometers usually have their own backplane and they are closed for custom development. So far, no spectrometer has a standard backplane that supports multichannel transceivers and allows users to develop their own units to do creative experiments.

This study aims to build a low-cost, scalable, high bandwidth multichannel spectrometer with a PXIe chassis for MRI systems. The PXIe backplane on a chassis, a non-propriety backplane or standard backplane, allows different communities to share the source and develop the spectrometer together. Focusing on a specific module by each collaborator and sharing reduces the cost and development time for completing a complex spectrometer. For example, a spectrometer has a transmitter, receive, gradient, and shim module. In addition, the PXIe backplane also empowers the users to employ the existing PXIe modules on the market to accelerate the spectrometer design. Another reason for using PXIe is that some fundamental work [11, 12] associated with that had been done by two students before. The specific work is illustrated in the next subsection.

Given these advantages, either the advantages of PXIe itself or the prior work on it, we tended to build a PXIe based MRI spectrometer with a multichannel transceiver. As shown in Figure 2.21, based on the PXIe chassis and the pre-design system controller, the main focus for this MRI spectrometer design is on the multichannel transceiver board design and integration between the two boards. The transceiver would contain quad channel receivers for parallel imaging and dual-channel transmitter for third dimension encoding by using TRASE [37, 38]. Furthermore, as discussed earlier, the scalable MRI spectrometer can be extended with gradient controller, extra receivers, etc., for different applications.

There would be several highlights of this spectrometer. Firstly, it is scalable. Based on the PXIe chassis, it allows users to expand up to 17 peripheral boards, which could include multiple receiver boards, transmitter boards, gradient boards, and shimming boards. Secondly, the high bandwidth. The throughput of the PCIe version 3.0 can reach upto 4 GB/s with four lanes, which is enough for an MRI system. Lastly, lower-cost. The cost of the hardware for building an MRI spectrometer is much lower than the commercial MRI spectrometers.

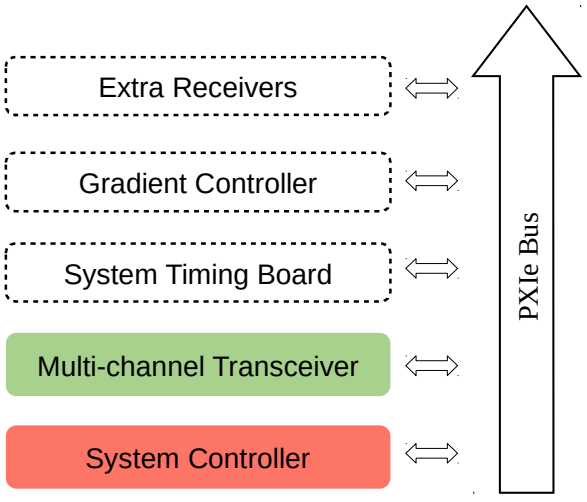


Fig. 2.21 Scalable MRI Spectrometer.

2.4.2 Prior Work on the PXIe chassis

Figure 2.23 shows the diagram of the system controller, which is based on an SoC FPGA module, PicoZed 7015. In the prior work, a PXIe system controller board was built, and the related designs were completed, including building the Linux system for the ARM in the processing system, implementing the PCIe root complex in programmatic logic, and programming PCIe drivers. More details related to the system controller are elaborated in Chapter 4.

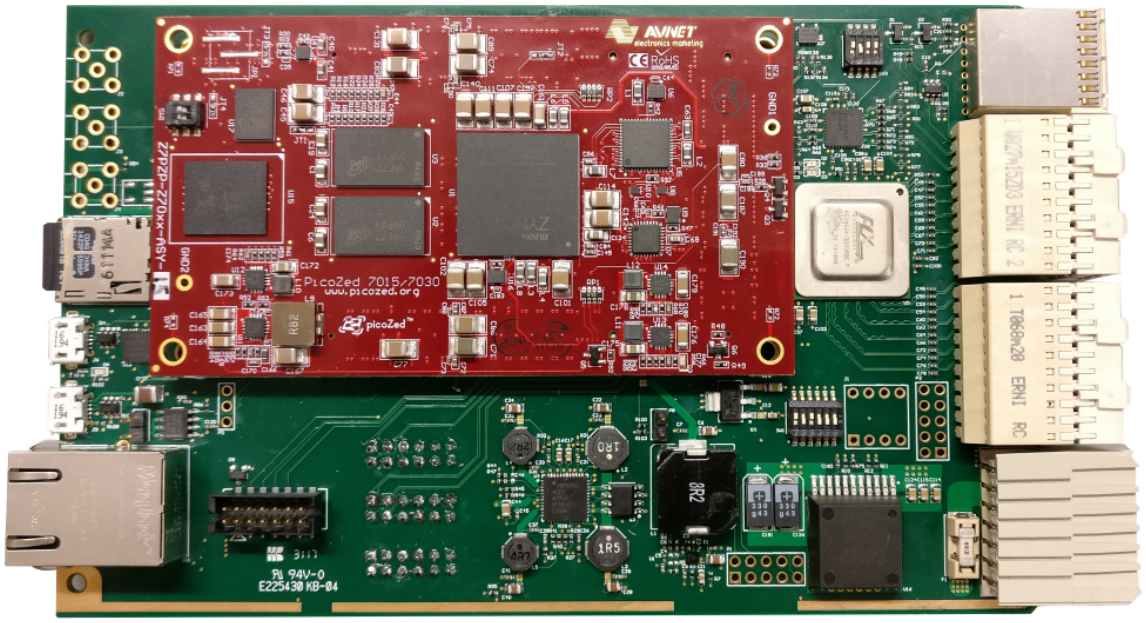


Fig. 2.22 System controller board [11, 12].

Figure 2.23 shows an application of developing a peripheral board based on the system controller on the PXIe chassis. On the custom peripheral board, there are two main parts, one is the FPGA module, and the other one is the FMC card (AD9467 evaluation board) which is connected to the FPGA. That system setup successfully verified that the system controller could control the peripheral board, and the data captured by the FMC card could be transferred to the controller over PCIe protocol.

Figure 2.24 illustrates the architecture of the PXIe application developed by Andrew Ang[11, 12]. In the peripheral FPGA on the peripheral board, the PCIe CDMA subsystem was implemented for initiating DMA transfer and performing the PCIe endpoint. The captured

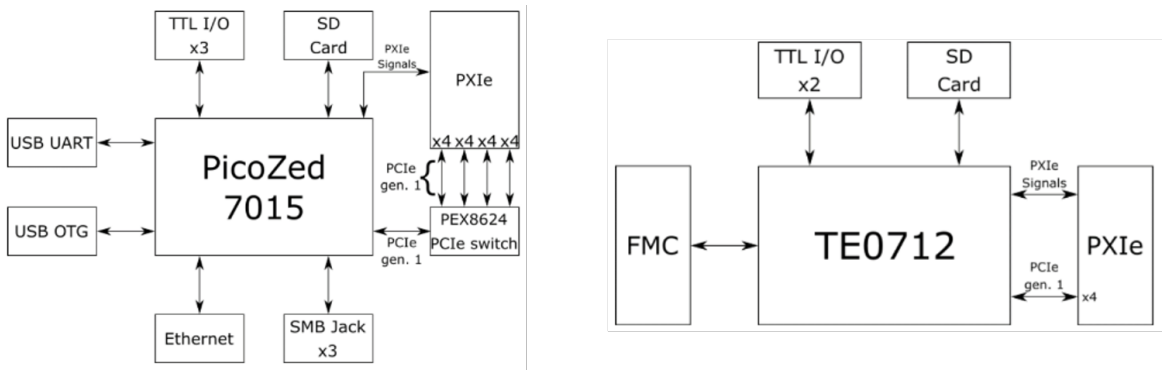


Fig. 2.23 Diagrams of system controller (left) and peripheral FMC board (right) [11, 12].

data is stored in DDR via the ADC interface module, DMA (Direct Memory Access) module, and MIG (Memory Interface Generator) module. The stored data is then transferred to the system controller through the PCIe CDMA system and the PCIe lanes on the chassis backplane.

2.4.3 Architecture of the MRI Spectrometer

The MRI spectrometer based on the PXIe chassis consists of the existing system controller and the multichannel transceiver board. The ZYNQ FPGA module on the system controller connecting with the PCIe switch can communicate to multiple PCIe endpoints through the PXI bus on the backplane of the chassis. The multichannel transceiver is comprised of three main components: An FPGA module, a quad-channel ADC, and a dual-channel DAC. The DAC configured by the FPGA is used for generating the pulse sequence for excitation. The four-channel ADC is used for digitizing the received MR signal from multiple coils through pre-amplifiers. The data processed by the FPGA goes to the system controller through the PXI bus. A Graphic User Interface (GUI) is used to transfer commands or acquire data from the system controller through the SSH protocol.

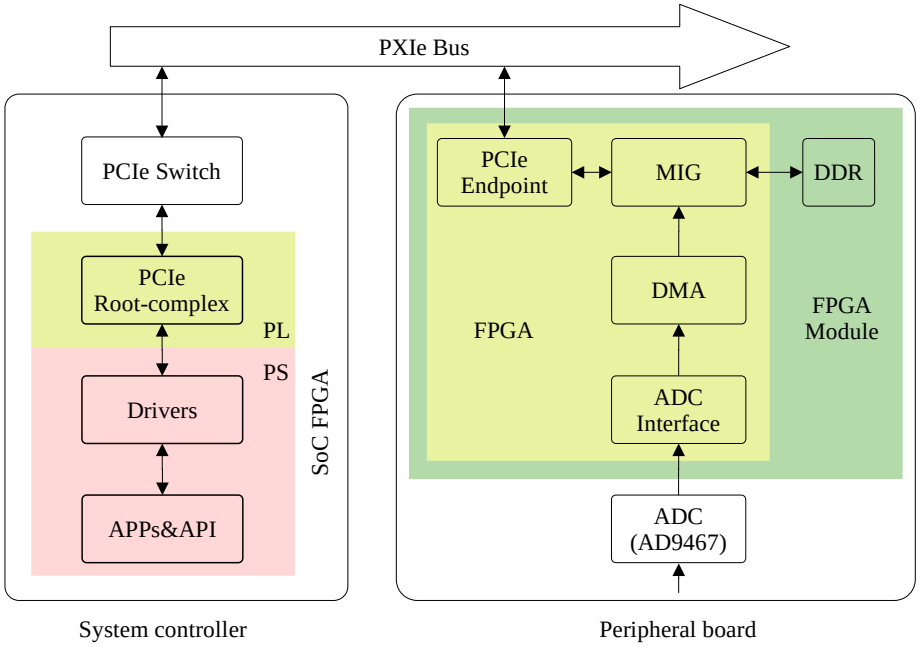


Fig. 2.24 Prior work on PXIe chassis [11, 12]. This diagram is redrawn based on the description of these references.

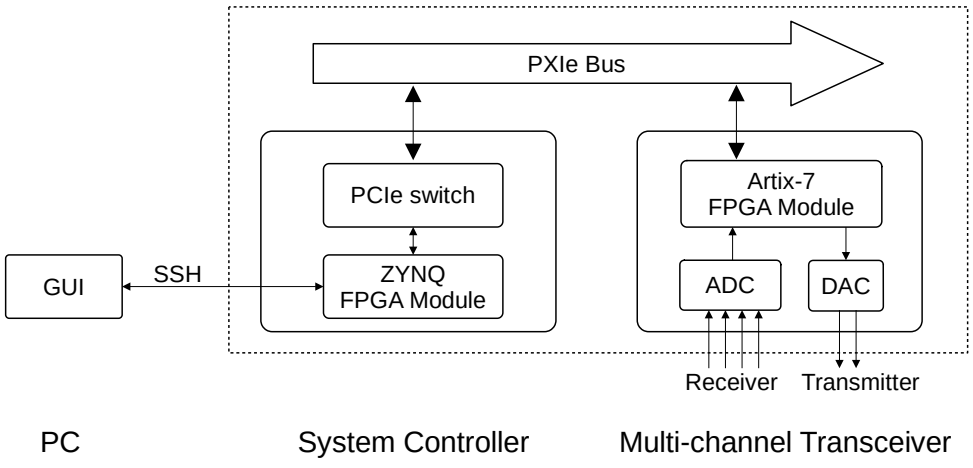


Fig. 2.25 Diagram of MRI console.

2.4.4 Overview Plan of the MRI System

The additional parts are also constructed or prepared to complete an MRI system, such as the user interface, high power amplifiers, pre-amplifiers, RF coils, and magnet unit to verify the spectrometer. Figure 2.26 shows the big picture of the MRI system.

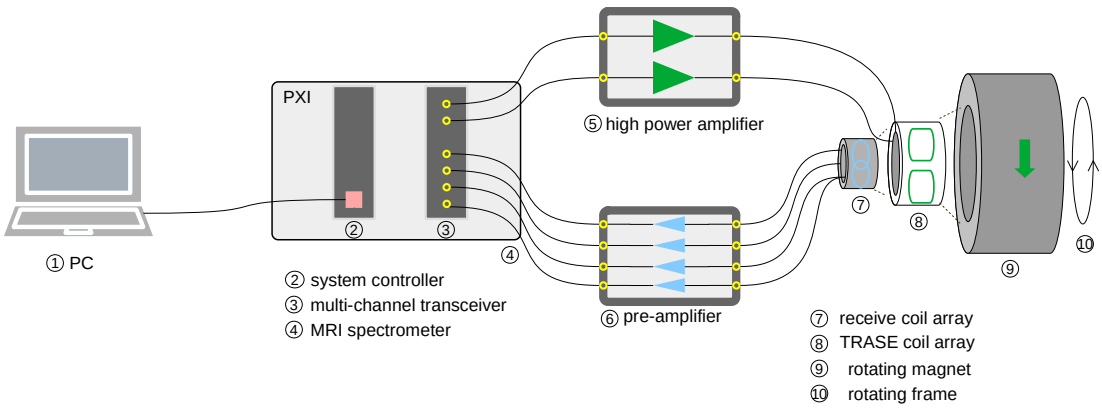


Fig. 2.26 Overview of the MRI system.

The plan to complete the whole system is illustrated in Figure 2.27.

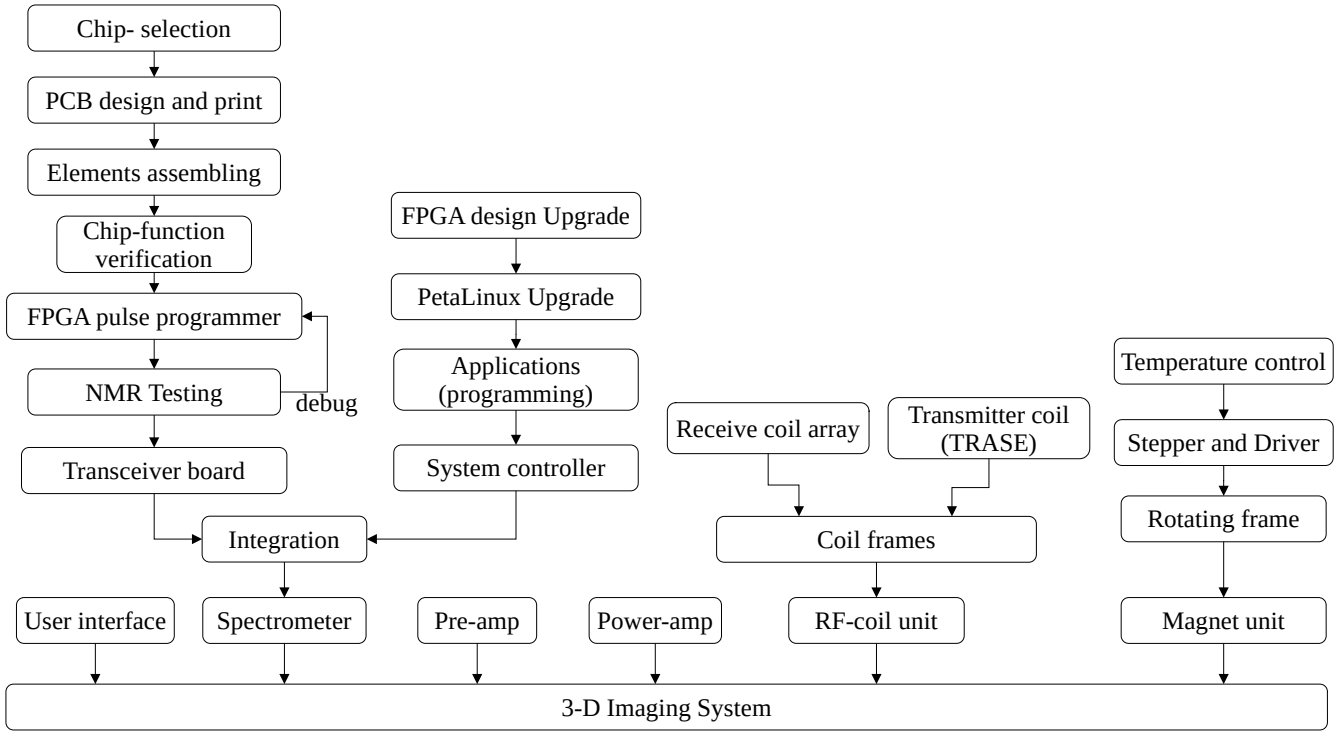


Fig. 2.27 Project plan overview.

Chapter 3

Multichannel Transceiver Board

Introduction

This chapter considers how the multichannel transceiver board was constructed. The main issues addressed in this chapter proceeds as follows:

- Proposed design requirements and specifications of the transceiver board.
- Key components selection. That includes considerations of selecting the ADC, DAC, FPGA, and clock generator.
- Hardware implementations of the key components.
- FPGA firmware design and implementation. The FPGA firmware designs are introduced part by part.
- Transceiver board testing is done to verify the synchronization.

3.1 Design Requirements

This section begins by describing the main requirements of the transceiver board and then indicates the key specifications of the transceiver board.

3.1.1 Function Requirements

As discussed in section 2.4.1, the present study is set to build a versatile spectrometer from a low field to a 3T MRI system. The spectrometer is designed to have the following functions:

1. Receiving. The board will receive or digitize the pre-amplified MR analog signal at a certain sampling rate with multiple channels simultaneously. The digitized signal can be processed numerically in the FPGA, and the processed data can be stored on the memory device on the transceiver board.

2. Transmitting. Two transmitter channels are built on the board to produce arbitrary pulses simultaneously for driving TRASE coils [37, 38] via high-power amplifiers. The phase and frequency of the two generated waveforms from the two channels are independent and programmable for the two channels.

3. TTL signals. The transceiver board will provide multiple TTL signals for controlling the external devices, such as enabling or disabling the power amplifiers, configuring the gain of pre-amplifiers, switching on/off the active de-tuning circuit on receive coil, and controlling the stepper to rotate the magnet. Given that different devices are involved, the voltage levels of the TTL signals would be 5 V, 3.3 V, and 1.8 V.

4. Pulse program generator. This generator allows the transmitter to produce the primary NMR/MRI sequence(s), such as single FID pulse, Spin Echo pulses, etc. In a pulse program, the receiver chain can acquire the MR signals.

5. Data transfer with the system controller. The system controller is able to access the memory device on the transceiver board. In other words, the system controller is

capable of transferring data to the memory device over the PCIe protocol and copying data from the memory block on the transceiver board.

3.1.2 Key Specifications

The specifications of a spectrometer are significant as it determines the application and performance of the spectrometer. The key specifications can be listed as follows: receive channel count, receive bandwidth, the amplitude of receive channel, transmitter channel count, transmitter frequency range, the amplitude of transmitter channel, and time resolution. Table 3.1 depicts the specifications.

Table 3.1 Proposed key specifications of the spectrometer.

Items	**Features**	**Notes**
Rx channel-count	4	Performing parallel imaging
Rx Bandwidth	>130 MHz	Fit a 3 T MRI system
Rx Amplitude	0 - 2 V	Match with the pre-amplifiers
Rx SFDR	90 dBc	2.0 V p-p input span
Tx channel-count	2	TRASE [37, 38]
Tx frequency range	0 - 250 MHz	Fit a 3 T MRI system
Tx Amplitude	-9 - 0 dBm	Adjustable
Memory Depth	1 GBytes	To store processed MR data
Decimation Rate	10-100	Adjustable
Time Resolution	10 ns	-
TTL Voltage	5V/3.3V/1.8V	Managing versatile devices

3.2 Key Components

After proposing the main functions and the key specifications of the transceiver on the MRI spectrometer, the issues covered in these sections are the key components to fulfill these requirements. The issued can best be treated under four headings:

- Analog-to-Digital Converter.
- Digital-to-Analog Converter.
- FPGA-based logic system.
- Clock generator.

The selection process and features of the selected components is elaborated on in the following subsections, section ADC Selection Considerations, section Digital-to-Analog Converter (DAC), section FPGA.

3.2.1 Analog-to-Digital Converter (ADC)

An analog-to-digital converter (ADC) refers to a device that converts analog signals into digital signals with a certain sample rate. On this transceiver, the ADC is used to convert the amplified analog MR signal from pre-amplifier to digital signal for further processing in the FPGA fabric. This section analyzes the ADC selection considerations, available ADC components, and the selected ADC features.

ADC Selection Considerations

To construct a multichannel transceiver spectrometer that is expected to suit a low-field to a 3T MRI system, multiple considerations must be taken into consideration for selecting an ADC device. All the considerations are about the parameters of an ADC that decide the receiving ability of the spectrometer. The specific parameters are:

1. Channel-count. Channel-count stands for the number of analog to digital converters on an ADC chip. In modern MRI systems, the number of ADC channels usually equals

the number of coils in the receive coil array that is used for parallel imaging. Parallel imaging that decreases the phase encoding times by using the receive coils' profiles reduces the acquisition time. In light of this advantage, a multichannel receiving ADC is desired to be implemented to realize parallel imaging in this study. Even though multiple single-channel ADC could realize multichannel receivers, it generally increases cost and complicates implementation. Thus, a single-channel ADC chip is not considered.

2. Analog Bandwidth. For converting an analog signal to digital and recovering it, the analog bandwidth can be defined as the frequency range in which the signal can be acquired accurately; in other words, it is defined as a frequency at which the measured amplitude is 3 dB below the amplitude of the signal. The inherent frequency response of the input path, which causes loss of amplitude and phase information, determines the limitations of analog bandwidth. To make this transceiver suitable for a 3 T MRI system, the analog bandwidth has to be at least 130 MHz, according to Equation 2.1.

3. Sampling rate. This factor determines the sampling rate at which a signal is converted from the analog signal to digital. According to the Nyquist theorem, a signal must be sampled at least twice as fast as the bandwidth of the signal. Otherwise, the aliased products in the band of interest can distort the signal and lead to loss of information. To apply the transceiver to a 3T MRI system, this study chooses an ADC with a sample rate over 260 MHz to realize a bandwidth of 130 MHz. However, it is not necessary because only several MHz bandwidth of signal around the Larmor frequency is useful. Therefore, sub-sampling can be applied in conjunction with a certain sampling frequency, say 100 MHz, to achieve that. A band-pass-filter, of which the central frequency is the Larmor frequency, is necessary to eliminate the signals outside the band of interest.

4. Resolution. Resolution is the smallest voltage that the ADC can recognize [56]. There are three common resolution bits, 12-bit, 14-bit, and 16-bit, for pipelined ADCs. Higher resolution means more data accuracy, but it is more expensive.

5. Dynamic range. It is the range of signal amplitude that can be resolved by the ADC. With the ADC resolution and LSB (least significant bit) detected by the ADC, dynamic range can be expressed [57]:

$$Dynamic\ range = 20log_{10}\frac{(2^N - 1)LSB}{LSB} \tag{3.1}$$

where N is the ADC resolution and LSB represents least significant bit.

The ADC dynamic range significantly affects the large dynamic of the receiver in the receiver chain.

6. Aperture-jitter. Aperture jitter or uncertainty is the sample-to-sample variation in the instant of the encoding process, which significantly affects the system performance, for example, degradation of the SNR. Figure 3.1 shows how an aperture uncertainty in the sampling instant results in an error in the sampled voltage. The error is related to the rate of change of the analog signal. The degradation in SNR at a given input frequency (f_A) due only to aperture jitter (t_J) can be calculated [58]

$$SNR\ Degradation = 20log_{10}\frac{1}{2\pi f_A t_J} \tag{3.2}$$

The phase jitter of the external clock for the ADC has the same type of error, which degrades the SNR. To realize a high-performance spectrometer, this study tends to choose a low aperture jitter ADC.

7. ADC architectures. ADC architectures can be classified into successive approximation (SAR), Sigma-delta ($\Sigma - \Delta$), and pipeplined ADC [56]. The successive approximation has a wide bit resolution, from 8-bit to 18-bit, while the sampling rate is low, up to several MHz. As a result, the ADC with this architecture does not meet this study's requirement. Sigma-delta has a more precise resolution, up to 32-bit but has an even lower sampling frequency, up to several hundred kilohertz. The pipelined ADC architecture is designed for high-speed applications. Hence the sampling frequency can be up to several GHz. Furthermore, the pipelined ADC can correct flash errors by

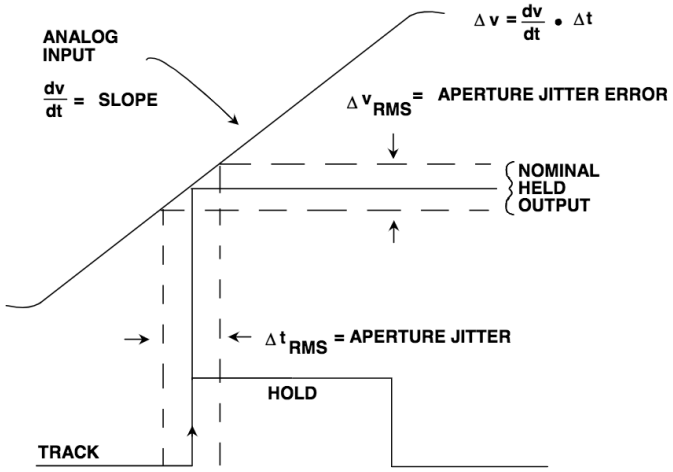


Fig. 3.1 RMS Jitter [59].

providing sufficient overlap. The calculated outputs from each stage are formed into a 6-bit result (an example in Fig. 3.2), which will then be transmitted to outputs by the digital serializer. In the end, the data is serialized with LVDS criteria and aligned to the frame and data clocks.

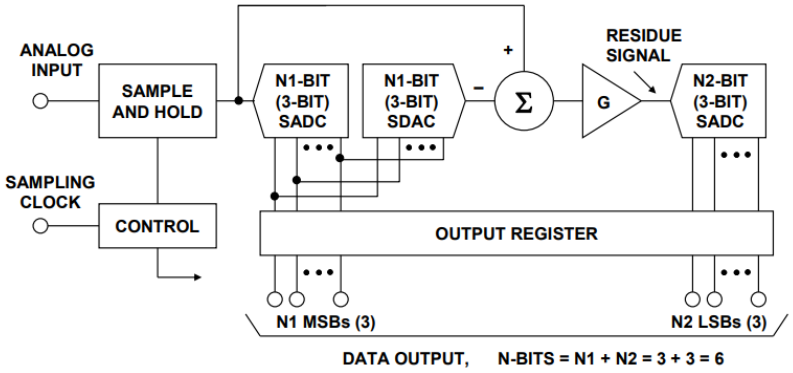


Fig. 3.2 Pipelined ADC [60]. An example of 6-bit.

8. Integral nonlinearity (INL). It revels the maximum vertical difference between the actual and the ideal curve of an ADC. However, small range of devices with typical performance. It is not a primary concern here, but need to keep an eye on it.

9. Others. Some other features are also important, including input voltage range, output interface, power supply, power consumption because they affect the designs of other

circuits connected with them. For example, the output of the pre-amplifier must match the input voltage of the input range of the ADC. Pin-package is also of particular concern. Because the transceiver board is a prototype, we prefer to use the pin-package that can be soldered manually rather than using BGA (ball grid array), which can complicate the manufacturing because they need the solder machine for mounting.

Available Components

Some available ADCs are described in Table3.2. Four channels are preferred in this study to realize parallel imaging at this stage. Despite the sample rate, bandwidth and resolution, the AD9653 and AD9656 are advantageous as their jitter is lower than the others. The difference between the AD9653 and AD9656 is that the output of AD9653 is serial LVDS while the output of AD9656 is JESD204B. Compared with serial LVDS output, JESD204B protocol has the advantage of fewer pin connections between the ADC and the FPGA, higher frequency, and simplified interface timing. In contrast to the AD9653, it is more expensive to use the AD9656 as the interface IP core costs around 10,000 NZD and requires more time to develop the JESD204B interface. Therefore, we selected AD9653 that has the serial LVDS output.

Table 3.2 Some available ADC devices.

ADCs	Channels	Sample rate	Bandwidth	Resolution	Jitter	SNR	Output	SFDR	Pin-package	Cost
LTC2284	2	105	575	16	200	74	Parallel	84	QFN	154
AD9653	4	125	650	16	135	77	Serial-LVDS	89	LFCSP	676
AD9656	4	125	650	16	135	78	JESD204B	86	LFCSP	697
ADS5263	4	100	700	16	220	79	JESD204B	74	QFN	585
ADS52J90	4	65	70	14	500	73	Serial-LVDS	73	NFBGA	339
Units	-	MHz	MHz	Bit	fs(rms)	dB	–	dB	-	NZD

–

AD9653 Features

The AD9653 is selected in this design. It is a multistage pipelined ADC with quad-channel, 16-bit, 125 MSPS, 650 MHz analog bandwidth, and 90 dBc dynamic range at a 2.0 V peak-peak input span.

As shown in Figure 3.3, there are 4 ADC chains in that AD9653. In each ADC chain, the input from a differential pair is sent to the first stage of the pipeline, and the flash errors are corrected in the preceding stages. The quantized outputs from each stage are combined into 16-bit data, and the data is transmitted through a digital serializer. The timing diagrams of the ADC output, frame clock, and data clock are presented in subsection 3.3.2. For the ADC, the common voltage to the center of the input transformer ensures the input signal remain above the ground. VCM is the analog output at mid-supply voltage, which is used to set the common voltage of the analog input. For the ADC, the commom-mode voltage to the center of the input transformer input ensures the input signal reminds above the GND. And the differential inputs are used to reduce susceptible interference.

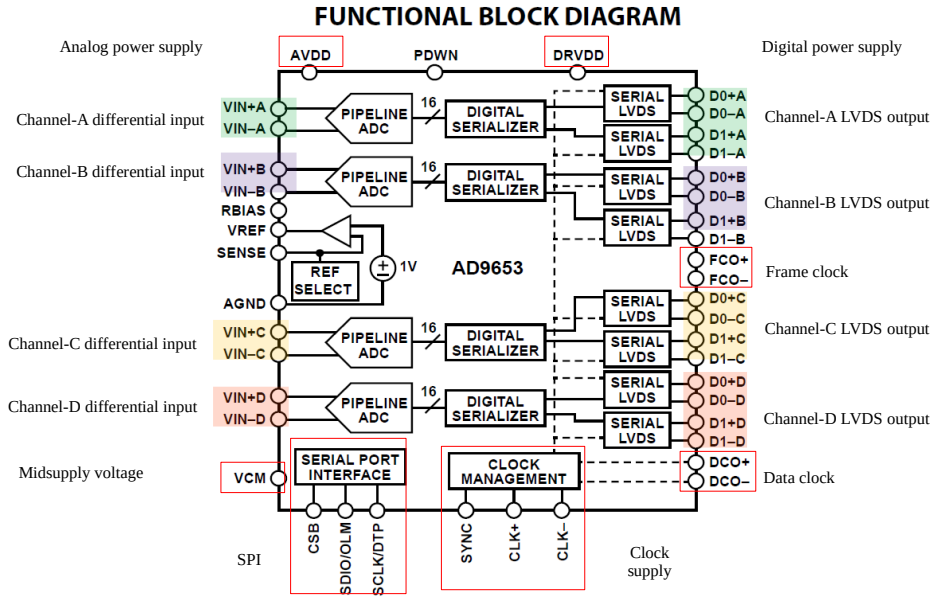


Fig. 3.3 ADC function diagram [61] with added information.

More features and descriptions are shown in Table 3.3.

Table 3.3 Other features of the ADC(AD9653). LFCSP represents Lead Frame Chip Scale Package, which exposes the pins on the edge of the chip and allows it to be soldered manually.

Features	Description
Output type	Serial LVDS. This interface type decreases pin-count.
Power supply	1.8 V supply operation. As indicated in Figure 3.3, the analog power supply (AVDD) and the digital power supply (DRVDD) must be provided separately.
Pin-package	48-lead LFCSP, which can be soldered manually.
Input voltage	Maximum 2 V peak-peak.
SPI control	• Built-in and custom digital test pattern generation. This feature facilitates the ADC testing, which is discussed in section 3.4.3. • Programmable output clock and data alignment. • Digital reset.

3.2.2 Digital-to-Analog Converter (DAC)

A Digital to Analog Converter (DAC) is a device that converts digital signals into analog signals. On this transceiver board, DACs are implemented, followed by some auxiliary circuits, such as low-pass-filter (LPF), gain block, and RF switch, to produce RF pulses.

DAC Selection Considerations

There are several considerations for choosing a DAC device, such as resolution, sample rate, interface type.

1. Channel-count. Channel-count determines the number of transmitter channels. Two channels are required to realize TRASE[38].

2. Resolution. A minimum 14-bit resolution is required to guarantee the accuracy of the signal for amplitude modulation and level control.

3. Sample rate. This factor determines the maximum RF pulse frequency to be produced. To make this transmitter available for a 3 T system, the sampling frequency must be at least 260 MHz.

4. Data input interface type. There are two main types of data input interface, parallel LVDS, and JESD204B [62]. As is discussed in the ADC selection section, the JESD204B interface type is expensive to use. Also, a LVDS interface with two channels at 500 MSPS means that the frequency of the signal would be 1 GSPS, which is possible for an FPGA.

5. Interpolation. Interpolation is a technique to increase the sample rate without affecting the signal itself. However, the input bandwidth must be limited.

DAC Selection

Table 3.4 shows some available DAC devices. Again, to make the transceiver of the spectrometer suitable for a 3 T MRI system, the sampling rate must be at least 260 MHz to produce 130 MHz RF pulses. In this study, AD9783 is selected.

Table 3.4 Some available DAC devices.

DACs	Channels	Sample rate	Resolution	SNR	Interpolation	Cost
DAC5682	2	1000	16	73	Yes	119
DAC3174	2	500	14	76	No	58
DAC3823	2	800	16	76	Yes	53
AD9783	2	500	16	68	No	73
Unit	-	MHz	Bit	dB	-	NZD

AD9783 Features

This dual-channel DAC, AD9783 (Analog Devices), is adopted for driving a transmitter coil array, TRASE[38]. The AD9783 is a high dynamic range, dual DAC with 16-bit resolution,

and its sample rate is up to 500 MSPS. Full programmability provided through a serial peripheral interface (SPI) port enables the users to configure the current output, which results in adjustable pulse amplitude.

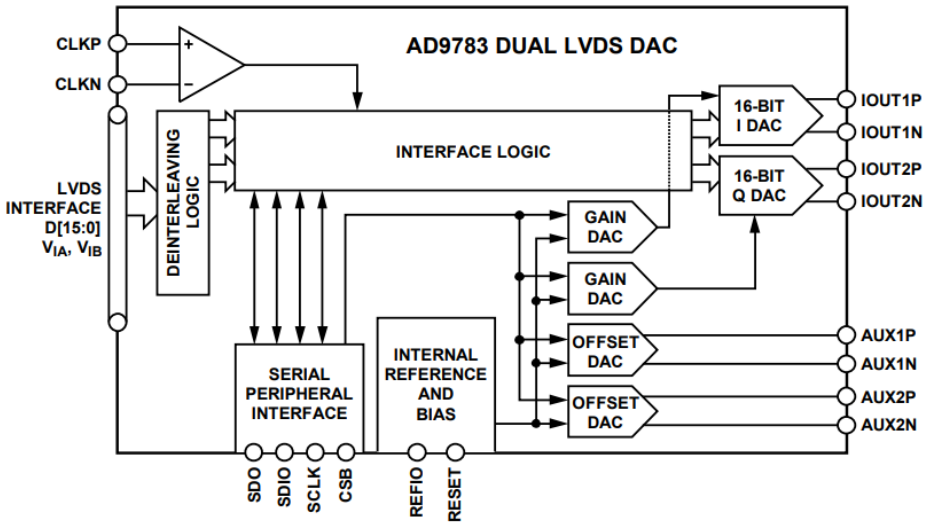


Fig. 3.4 DAC functional block diagram [63].

Figure 3.4 shows the function block diagram of the DAC (AD9783). Recall that the Figure 2.13 b in subsection 2.3.1, the architecture of FPGA + DDS would be adopted to generate the arbitrary waveform. In the interface of the dual DAC, the interleaved data from the FPGA is sent to the interleaving logic, and the output goes to the interface logic, which is clocked by the main clock, sample clock, CLKP/CLKN. The individual digital signals from the interface logic are converted into analog signals separately. The SPI port is used to configure the other features, such as the output current, internal logic alignment, etc.

3.2.3 FPGA

Options

An FPGA is a key device for transmitter design and digital receiver design, as discussed in subsection 2.3.1. There are two options to implement an FPGA: to implement an FPGA directly onto our transceiver board; to use an FPGA module to be mounted onto the transceiver board. The latter one was chosen.

Advantages of Using an FPGA Module

There are several advantages of adopting a commercial FPGA module. Firstly, it is time-saving and economical because connecting several hundred FPGA pins to the peripheral circuits on a compact PCB board is burdensome. The possible failure of the complex design costs time and money. The price of a commercial FPGA module is just slightly higher than a homemade one (if successfully implemented) because of the volume production. Secondly, using the commercial FPGA module saves space for the transceiver board because the FPGA module is sitting above the top layer. Thirdly, the vendor-supplied reference designs could familiarize the customers with the resources on the FPGA module and facilitate customer designs.

FPGA (Module) Selection

Selecting an FPGA module, essentially, is choosing a proper FPGA. Several brands of FPGA, such as Xilinx, Intel, Microchip, etc., are available. This study continues to use a Xilinx FPGA because, in our NMR laboratory, the previous students or mentors have developed NMR related FPGA projects by using Xilinx FPGAs before. Therefore, referring to the projects or discussing the design details under the same platform with the designers could facilitate the following FPGA designs. Before selecting an FPGA, another concern to be dealt with is to determine how many IO pins are needed. Table 3.5 illustrates the required IOs.

There are six families of FPGA from Xilinx, and each family has several series, which are differentiated by the number of resources such as logic cells or DSP slices, etc. Table 3.6 shows the available FPGAs from Xilinx company [64–66]. The prices are referred to Digikey [67].

Some available FPGAs have a hardcore processor, e.g., ARM. The others have a softcore processor, e.g., MicroBlaze. In this study, the ZYNQ FPGA module on the system controller already has a hardcore processor (ARM). It is unnecessary to have another hardcore for the transceiver as MicroBlaze can be used in a softcore processor.

Table 3.5 The required I/Os of an FPGA for the key components. (DIFF = Differential)

Devices	Number of IOs	IO type	IO voltage (V)
ADCs output	16	DIFF LVDS	1.8
ADCs configuration	5	single-end	1.8
DACs input	32	DIFF LVDS	1.8
DACs configuration	5	single-end	1.8
Clock generator	5	single-end	1.8
FPGA TTLs	10	single-end	3.3/1.8
Buffered TTLs	8	single-end	1.8
Others	20	single-end	3.3/1.8
Total	106	–	–

MicroBlaze Introduction

The MicroBlaze is an embedded soft processor core, which is a reduced instruction set computer (RISC) and optimized for embedded applications for Xilinx devices. Figure 3.5 shows the functional block diagram of the core. As a softcore processor, it can be implemented by employing the general-purpose memory and logic fabric of Xilinx FPGAs. The highly configurable capability and versatile interconnections of the MicroBlaze, such as AXI interconnection, primary I/O bus, allows users to select the combination of peripheral, memory, and interface features to build the exact embedded system on a single FPGA.

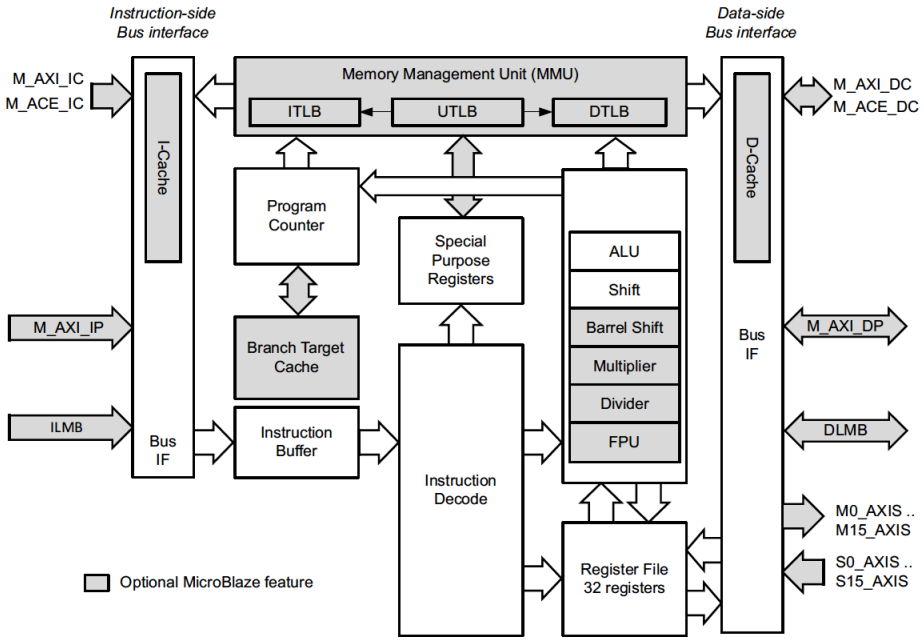


Fig. 3.5 Block diagram of MicroBlaze core [68].

Table 3.6 FPGA selection.

FPGAs	**Logic cells**	**DSP Slices**	**I/O Number**	**PCIe**	**Processor**	**Cost**
Spartan-6	3.8∼147	8∼180	132∼540	No	MicroBlaze	29∼1,000
Spartan-7	$6 \sim 102$	10∼160	$100 \sim 400$	No	MicroBlaze	28 ∼300
Artix-7	$12 \sim 215$	40∼740	150 ∼500	Yes	MicroBlaze	50 ∼840
Kintex-7	$65 \sim 477$	240∼1,920	300 ∼400	Yes	MicroBlaze	335 ∼14,308
Virtex-7	$582 \sim 1,954$	1,260∼3,600	300 ∼1,200	Yes	MicroBlaze	$6,394$ ∼65,000
ZYNQ-7000	$23 \sim 444$	$66 \sim 2,020$	$138 \sim 528$	Yes	ARM & MicroBlaze	90 ∼7,860
Units	k	1	1	1	-	NZD

TE0712 FPGA Module

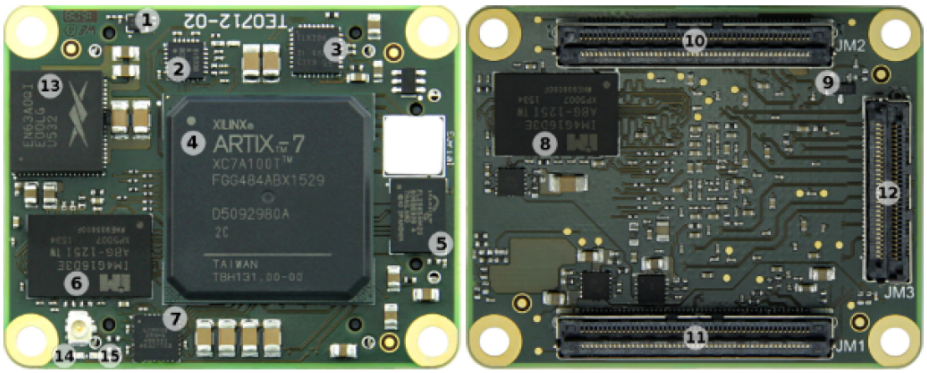


Fig. 3.6 Top and bottom view of TE0712 FPGA module [69]. The size is 4 cm by 5 cm.

The Xilinx 7-series FPGA, Artix-7, part xc7a200tfbg484-2, was selected as it meets all the demands and the FPGA modules are available on the market. Figure 3.6 shows the front and back view of the Artix TE0712 FPGA module from TRENZ [69]. Table 3.8 shows the main resources of the FPGA module. Figure 3.7 reveals the diagram of the FPGA module, TE0712.

Table 3.7 lists the voltage supply information and available I/O pins of the FPGA on the module.

Table 3.7 Available IOs on TE0712 FPGA module.

Bank	Voltage	IO types and amount
13	Supplied by user	15 LVDS pairs
14	3.3 V	11 LVDS pairs, 8 MIO
15	Supplied by user	24 LVDS pairs, 2 IOs
16	Supplied by user	24 LVDS pairs

3.2.4 Clock Generator

Synchronization of the transmitting and receiving is critical because NMR/MRI is a resonant phenomenon that correlates frequency and phase. On the FPGA module, a clock generator,

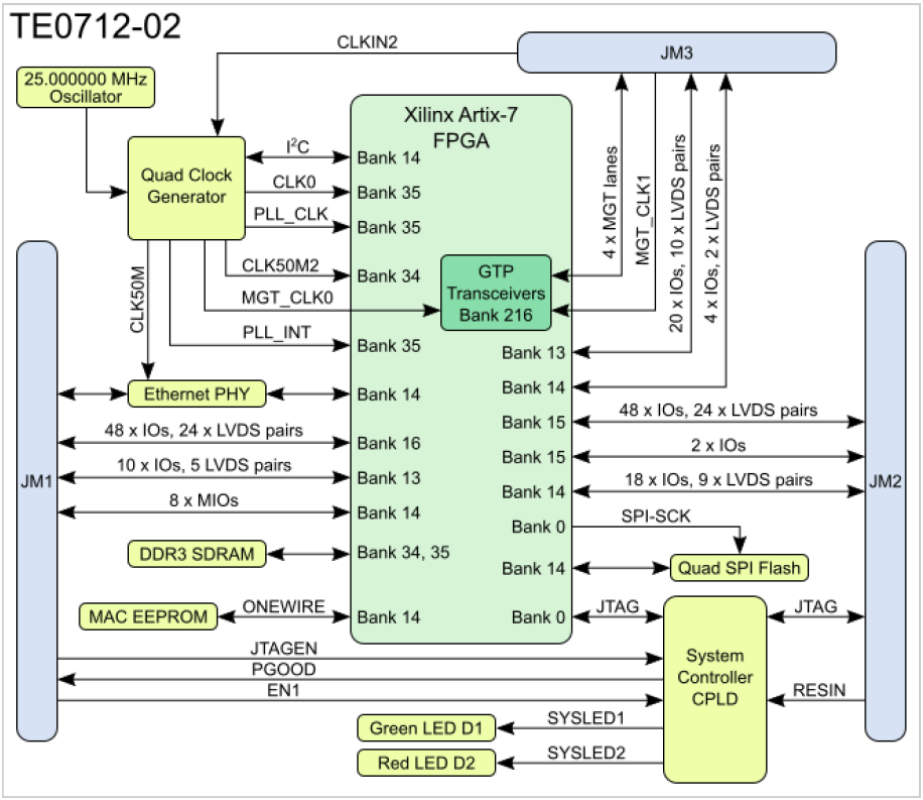


Fig. 3.7 Diagram of TE0712 FPGA Module [69]. The clock source of the on-module clock generator can be configured to the external clock input (CLKIN2 on JM3) rather than the on-module oscillator.

Table 3.8 Main resources on TE0712 FPGA module.

Resources	Functions
FPGA	
DDR	On the one hand, it stores the processed MR data and be accessed by the system controller for fetching the data. On the other hand, it stores the machine code of the pulse program to be copied to local memory in FPGA and executed in MicroBlaze.
PLL	As is shown in Figure 3.7, this clock generator clocks the FPGA and GTP transceiver. But this clock generator is fed by another clock generator on the carrier board rather than the oscillator on the module.
QSPI	To store initial FPGA configuration, i.e., the bitstream file of FPGA design, including the executable file for the microprocessor.
GTP	The GTP transceiver is used to transmit and receive the PCIe data from/to the system controller.

Si5338, is dedicated to clock the FPGA. The input of the clock generator can either be the local oscillator or another clock source. Another clock generator with multiple outputs is needed to clock the ADC, DAC, and the existing clock generator on the FPGA module for synchronization.

Clock Generator Selection

There are several considerations for choosing a clock generator. Firstly, the frequency reference should be stable (should be less than 50 Hz for imaging) and high resolution. In this spectrometer, a stable 100 MHz clock source that comes from the chassis' backplane is used. Secondly, phase control is also important for imaging which employs phase cycling to remove artifacts. Thirdly, phase noise or jitter, which relates to MR signal and noise, could reduce the effectiveness of signal averaging. For sub-sampling, the jitter is expected to be less than that contributed by the ADC of which the jitter about 100 fs.

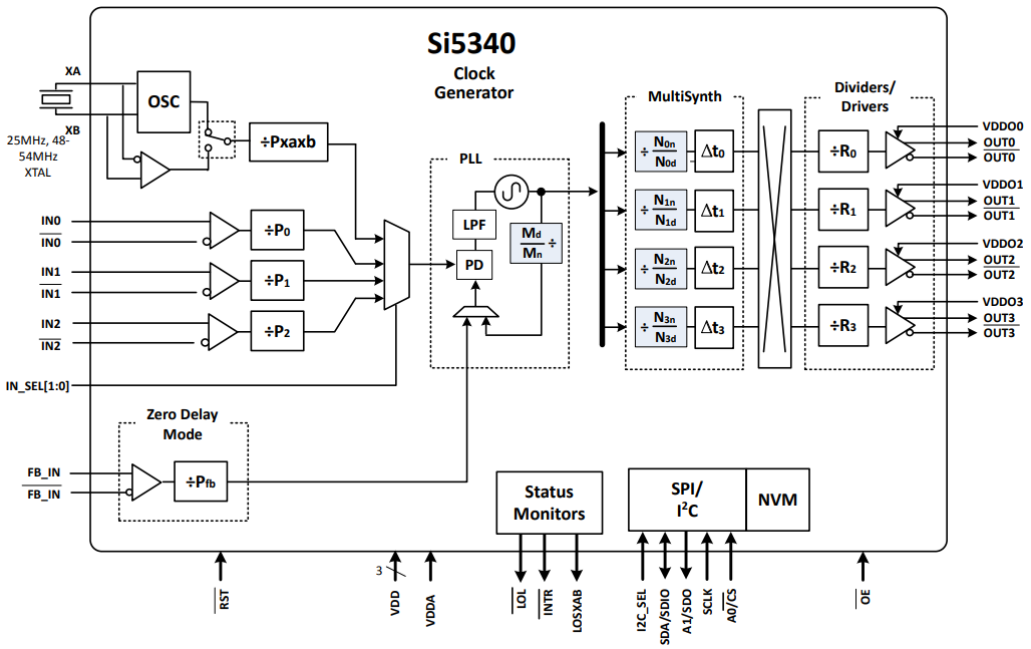


Fig. 3.8 Diagram of the clock generator [70].

Available Clock Generators

Table 3.9 shows some available clock generators. The main parameters to be considered while selecting the clock generator are listed as follows: the number of input channels, the range of input frequency, the number of output channels, the range of output frequency, jitter, and package. This study chose the Si5340 (Silicon Lab). This clock generator can synthesize a wide range of integers and non-integer-related frequencies up to 1 GHz on four differential clock outputs while delivering sub-100 fs RMS phase jitter performance.

Clock Tree table

Finally, a clocking system needs to be designed to produce the necessary clocks to the ADC, DAC, and FPGA, as shown in Table 3.10.

Table 3.9 Available clock generators.

Clock generators	LTC6951	ADC9518	Si5340	Units
Input channels	1	1	4	-
Input frequency range	1-425	0-250	10-750	MHz
Output channels	4	6	4	-
Output frequency range	2-2500	0-2950	0-1028	MHz
Jitter	115	225	90	fs(rms)
Package	QFN	LFCSP	QFN	-

Table 3.10 Clock tree table.

Devices	Frequency (MHz)	Source
Si5340	100	Chassis
ADCs	100	Si5340
DACs	200	Si5340
FPGA	100	Si5340

3.2.5 Support Components

Power Supply

A power supply architecture is needed as different devices need different levels of voltage. The backplane of the chassis supplies three power rails, 12 V/4A, 5V/1A, and 3.3V/6A. A different power regulator powers each device, i.e., LP3878, of which the input ranges from 2.5 $\sim$ 16 V and the output ranges from 1 $\sim$ 5.5 V. Table 3.11 shows the power supply requirements.

Voltage Level Translators

The eight trigger lines (Figure 3.9) are shared among all the slots on the PXI chassis, allowing users to use the trigger to synchronize the operation of different PXI peripheral modules. The

Table 3.11 Power supply table.

Devices	**Current@3v3**	**Current@1v8**	**Power(mW)**
AD9653	-	330 mA	594
AD9783	96 mA	83 mA	466.2
Si5340	190 mA	120 mA	843
FPGA	-	-	210
Si5338			300
Total	-	-	2413.2

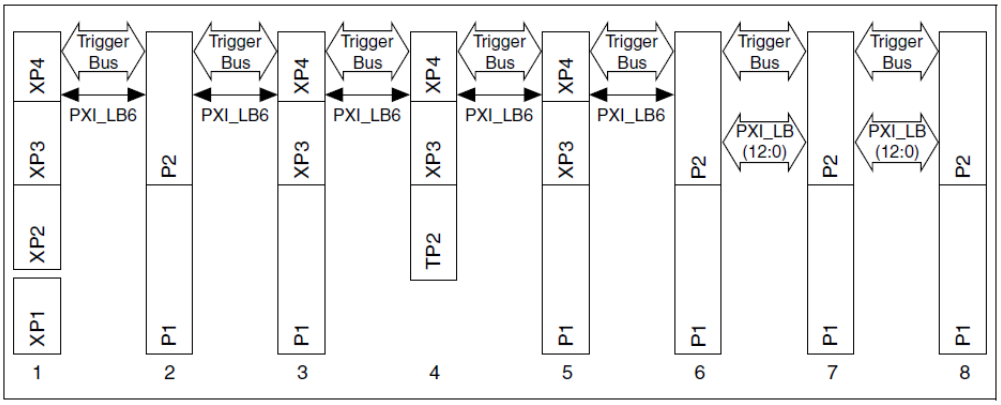


Fig. 3.9 Diagram of the PXI trigger bus [54].

translator, SN74CB3T16210 (Texas Instruments), is a high-speed TTL-compatible FET bus switch with low ON-state resistance, allowing for minimal propagation delay.

Another voltage level translator is used to translate the 1.8 V TTL signals from the FPGA to 5 V TTL signals which are required by the power amplifier.

3.3 Hardware Implementation

3.3.1 Overview

In the last section, Key Components, the key components are reviewed, and a more detailed account of the key components is given in the following sections, 3.3.2, 3.3.3, 3.3.4, 3.3.5. The main issues addressed in this section are:

1. Interconnections between the key components. The interconnections between ADC and FPGA, DAC, and FPGA need to be considered. For example, the logic voltages of the FPGA must match with the ADC or DAC. In addition, the logic type, say differential LVDS, of ADC/DAC, must also match with the FPGA.

2. Circuit board interfaces. Circuit board interfaces describe the connections with outside parts, such as power amplifiers, pre-amplifiers, and the chassis backplane. The detailed description is illustrated in Table 3.12.

3. Clock system details. It specifies how the clock tree is built, including the coupling circuits for the inputs and outputs of the clock generators.

4. Power supply. A number of regulators are used to support a variety of voltage levels.

Figure 3.10 shows the block diagram of the multichannel transceiver board. This is a standard PXI Express peripheral module that fits the PXI chassis.

3.3.2 ADC Implementation

In section 3.2.1, AD9653 was chosen to achieve optimal analog signal to digital signal conversion performance. Implementing the ADC, including analog input, clock input, voltage reference, and digital output, is critical for its performance. The following few subsections will elaborate on how those individual parts are implemented.

Table 3.12 Multichannel transceiver I/O array legend.

Name	**I/O**	**Comments**
Tx-CH1	O	Transmitter output, to feed the power amplifiers
Tx-CH2	O	
Ext–CLK	I/O	External clock input, optional
Rx-CH1	I	ADC Channel, from the output of pre-amplifier
Rx-CH2	I	
Rx-CH3	I	
Rx-CH4	I	
XJ4	I/O	Connection with the chassis backplane.
XJ3	I/O	More details are shown in Table 3.15
TTLs	O	BLANK/UNBLANK for the power amplifiers, gain control of pre-amplifier

Analog Input

As shown in Figure 3.3, the input of the ADC is a differential pair. According to the ADC datasheet [61], the ADC input must be set to a differential configuration to achieve optimal performance. Another consideration is the DC-bias. The ADC does not provide an internal DC-bias, so we must supply the bias externally in this application. According to the datasheet of AD9653 [61], setting the DC-bias voltage as the common-mode voltage ($V\_CM = AVDD/2$) is recommended. The common-mode voltage is provided by the ADC and can be used directly.

Figure 3.11 shows the schematic design of the analog input. In the diagram, R25 matches the signal impedance to 50 Ω. The transformer converts the single-ended signal into a differential pair of signals that fulfill the ADC's input requirement. The 33-ohm serial resistor provides isolation from the transient current in the input circuit of the ADC. The

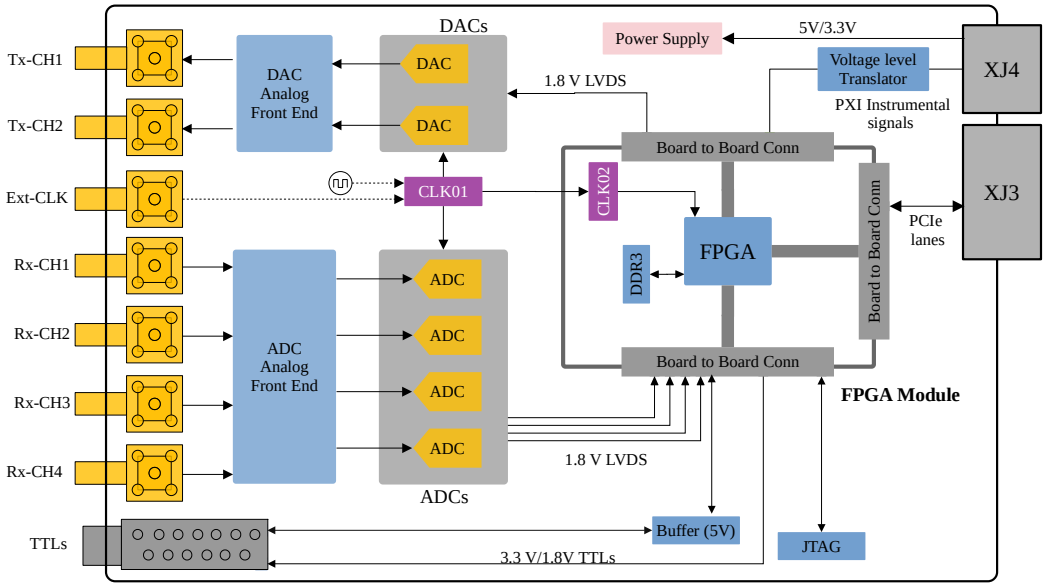


Fig. 3.10 Overview of the transceiver board diagram.

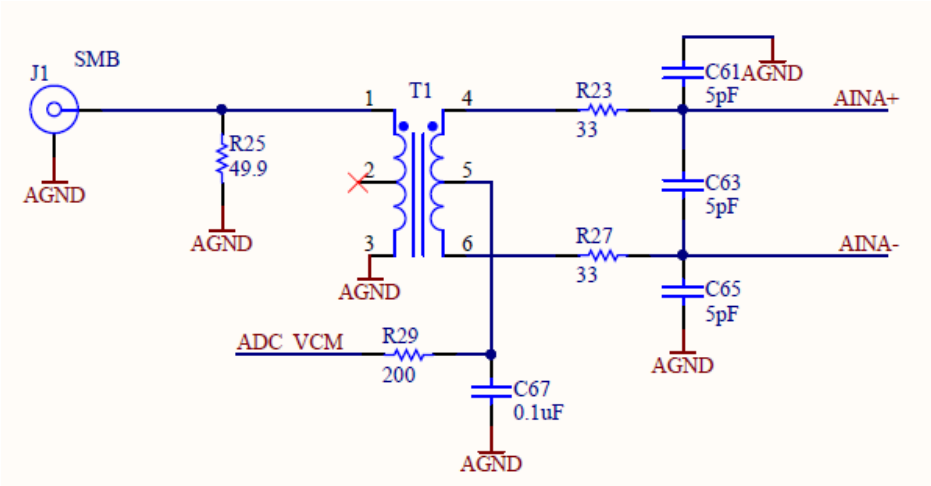


Fig. 3.11 Schematic of the ADC input circuit.

two single-ended capacitors are placed on the inputs to provide a matching passive network, creating a low-pass filter at the input to limit the broadband noise.

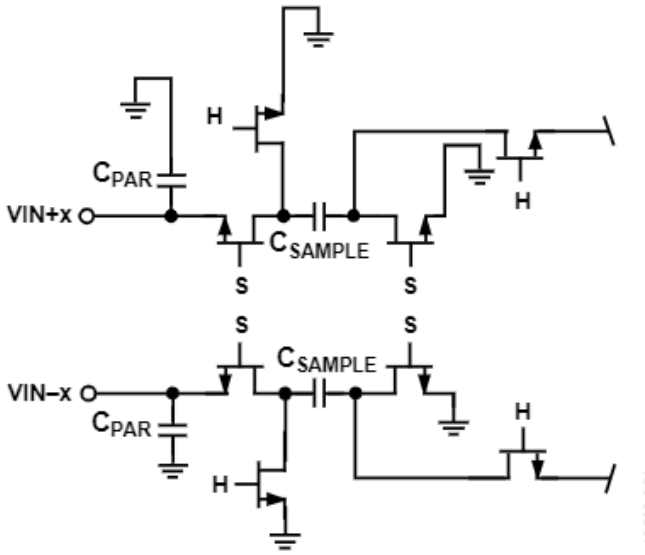


Fig. 3.12 ADC switched-capacitor input circuit [61].

Figure 3.12 shows the differential switched-capacitor circuit (inside ADC chip) designed for processing differential analog input signals [61]. The two modes in the input circuit, sample mode and hold mode, are switched by the clock signal alternately. The signal source must be able to charge the sample capacitors and settle within one-half of a clock period. Preferably, a small resistor in series with the input is helpful to reduce the peak transient current from the driving source.

Clock Input

A differential clock input is preferred for optimal performance. The clock signal is typically ac-coupled into the CLK+ and CLK- through a transformer or capacitors. The AD9653 supports a flexible clock input structure, such as CMOS, LVDS, LVPECL, or sine wave signal. In this design, capacitors are used for ac-coupling the source clock. Despite the optional types of clock input structure being used, the input clock jitter affects the eventual performance significantly because the high speed and resolution ADCs are very sensitive to

the quality of clock input. As discussed in ADC Selection Considerations in section 3.2.1, the clock source's jitter also influences the SNR performance.

Digital Output and Timing

As shown in Figure 3.3, the output of each channel has two serial LVDS lanes. The analog signal is transformed into a digital, serial LVDS data stream with a 16-bit resolution provided along with a high-speed data clock (DCO) and frame clock (FCO). Figure 3.13 shows the timing diagram. CLK(+/-) is the input differential clock operating at 100 MHz in this application. Sample N of the analog signal is converted to a digital format and presented at the output after a delay period. The outputs of DCO, bit-data, and FCO are synchronized on the terminal of the ADC. Several timing modes are optional through the SPI configuration. The DCO clock can be configured to either SDR (single data rate) mode or DDR (double data rate) mode. In SDR mode, data is clocked on the rising edge of the clock, which means that the clock bit rate needs to be twice as much as the transmission rate. While in DDR mode, data is transferred on both the rising and falling edges of the clock, which means that the clock bit rate is the same as the transmission rate. Figure 3.13 and Figure 3.14 show two-lane and one-lane modes of data transmission separately.

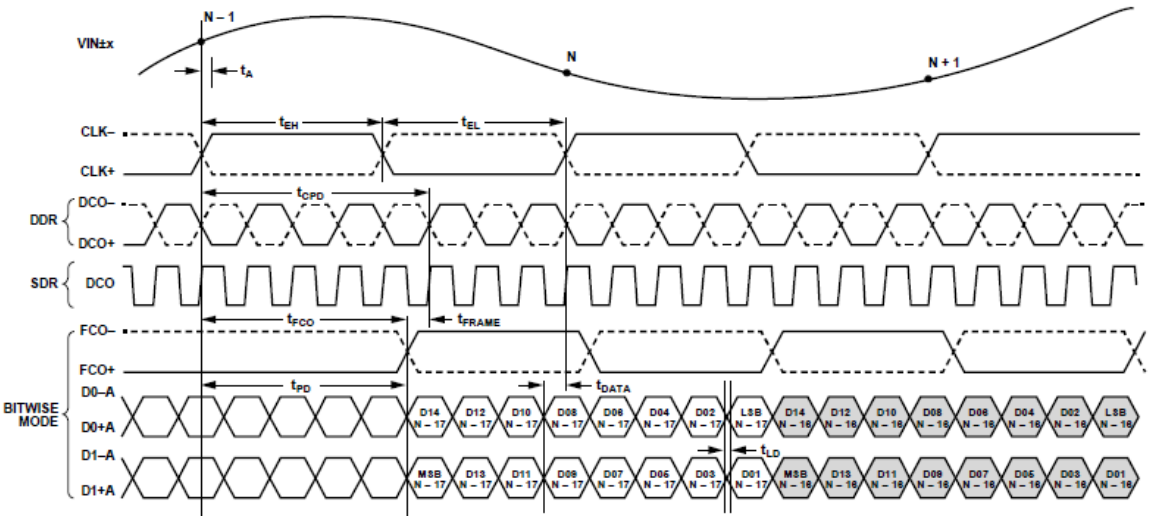


Fig. 3.13 ADC timing diagram, two-lane mode [61].

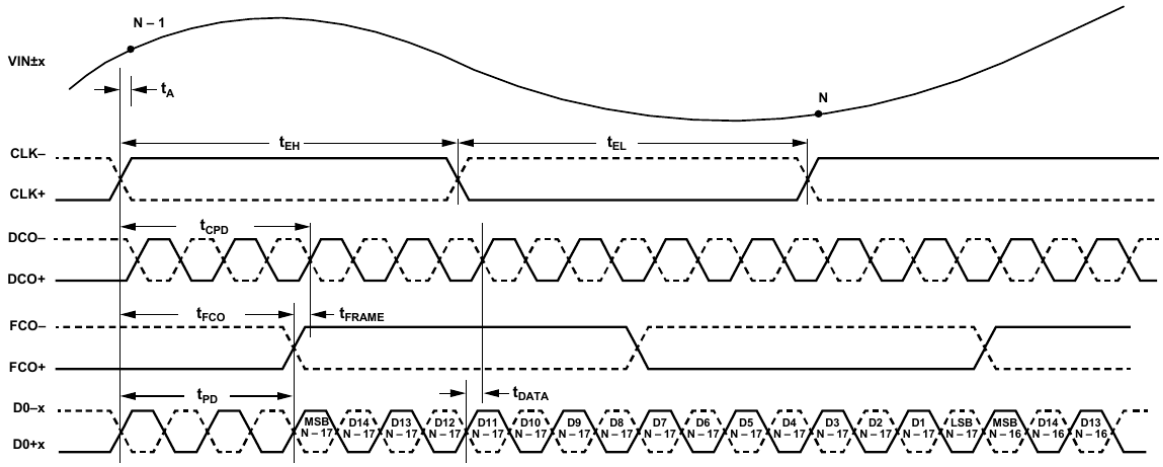


Fig. 3.14 ADC timing diagram, one-lane mode [61].

Table 3.13 Data clock frequency of 16-bit ADC under different configurations.

Mode	Number of lanes	Bit clock (MHz)
DDR	2	400
DDR	1	800
SDR	2	800

The bit clock frequency is determined by the ADC resolution, sample rate, data clock modes (SDR or DDR), and lane mode. When the DDR mode is selected, the bit clock can be calculated by

$$Bit\_clock(MHz) = \frac{ADC\_resolution * Sample\_rate}{2 * Lane\_number} \tag{3.3}$$

In this application, the 16-bit ADC is operating at 100 MHz. If the SDR mode is selected, the data clock frequency is 800 MHz.

Table 3.13 indicates the data clock frequency under different configurations. The one-lane mode has the advantage of less connections between the ADC and the FPGA, while the 2-lane mode doubles the number of connections. But the two-lane mode contributes to dividing the data clock by 2. According to the datasheet [65] of the chosen FPGA, the maximum speed of the digital signal processing metrics is around 550 MHz, lower than 800 MHz. In addition, the maximum rate of the data clock is 500 MHz, as is stated in the datasheet [61] of

the selected ADC. Therefore, the one-lane mode can not be used here, so the DDR mode and 2-lane mode are configured in the application. Lower frequency also simplifies the PCB design for the ADC part because more considerations need to be taken into for high-speed RF circuits. Bit-mode, byte-mode, MSB (most significant bit), or LSB (less significant bit) first mode is also configurable. However, these are the least important as the FPGA ADC interface module could match that data type.

Even though the data outputs are aligned with the bit clock and frame clock properly at the output ports, more specifically, the data clock is aligned with the frame clock, and data is 90 degrees out-of-phase to the data clock, as shown in Figure 3.13. These alignments should be maintained from the ADC output to the FPGA by using a quality PCB layout to guarantee that the delays of all the signals are equal. However, the clock signal and the serial data go through different buffers in the FPGA, which distorts the alignment. Thus, an ADC interface FPGA module must be implemented to tackle that problem. That FPGA module is discussed in subsection 3.4.3.

3.3.3 DAC Implementation

Digital Data Port and Timing

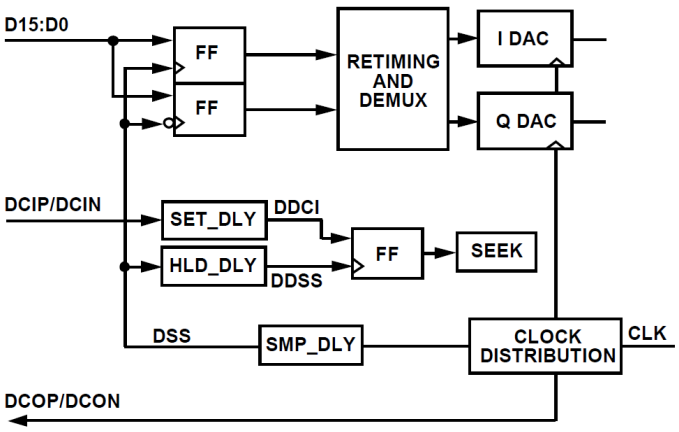


Fig. 3.15 DAC digital data port diagram [63].

As shown in Figure 3.15, the parallel data contains two interleaved data words, I and Q, targeting an individual DAC channel. DCIP/N are the clock signals aligned with the

interleaved data, which provide timing information of the parallel data, and indicate the destinations, i.e., I DAC or Q DAC, of the data. DSS, a delayed version of the main clock, samples the interleaved data on its rising and falling edge.

Table 3.14 lists the external and internal clocks of the DAC. Increasing HLD and SET results in the clock input being sampled later and earlier in its cycle separately. The result of the sampling is recorded in the SEEK block, which can be accessed by adjusting the two delays and reading the SEEK bit. The accurate timing information can be extracted to set the related register values for optimal performance.

Table 3.14 DAC related clocks.

Clock	**Notes**
DCIP/N	Input clock aligned with the input data.
DDCI	Internal clock generated by DCIP/N input clock. It can be configured by the parameter "SET_DLY" via SPI.
DDSS	Internal clock, a delayed version of DSS. It can be configured by the parameter "HLD_DLY" via SPI.
DSS	Internal clock, a delayed version of the DAC main clock. It is used for sampling the interleaved data on both edges.
DCOP/N	Output clock. Clock at the DAC sample rate. They are not used in this application.

The clock input signal should be aligned with the data signals by using an alternating bit sequence (010101...) to maximize the opening eye in the clock input and data signals, which improves the reliability of the data port interface. Also, it can be seen that the DATA rate should be twice as much as the clock rate of the DCIP/N and DSS, as shown in Figure 3.16.

Front-End Circuits

Following the DAC output, the front-end circuits are needed to complete the transmitter implementation. As shown in Figure 3.17, the circuits are listed:

1. Transformer. To transfer the differential signal to the single-end signal.

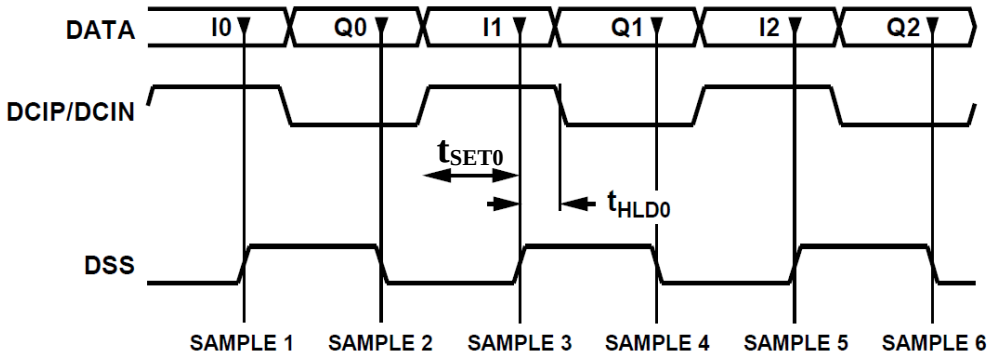


Fig. 3.16 DAC timing diagram [63].

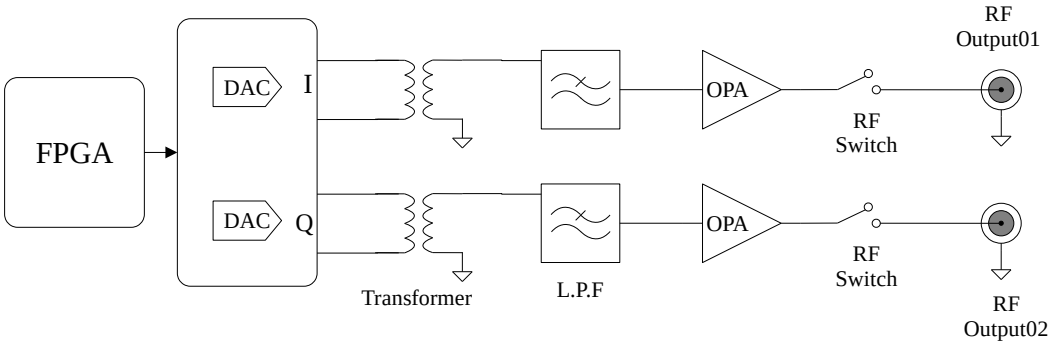


Fig. 3.17 DAC front-end configuration.

2. Filter. To eliminate the high-frequency components or sampling artifacts.

3. Gain. To adjust the amplitude of the output signal to match the input of the high power amplifier.

4. Gate. A TTL signal is used to turn on/off the gate of RF output.

Figure 3.18 and Figure 3.19 show the schematic design of the front-end circuits.

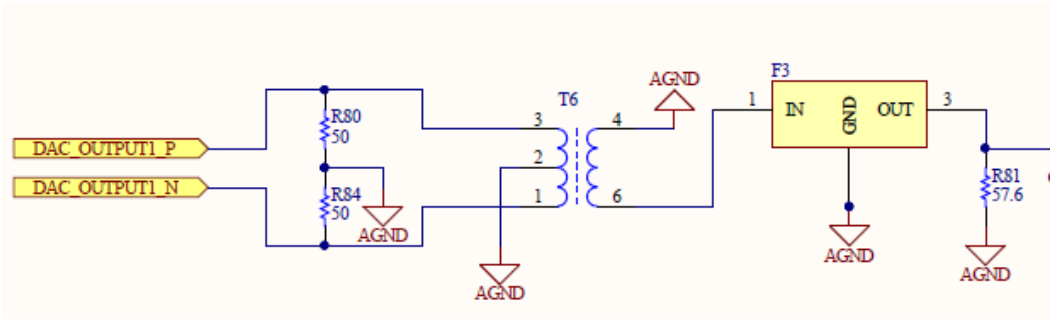


Fig. 3.18 Schematic of the DAC output circuit.

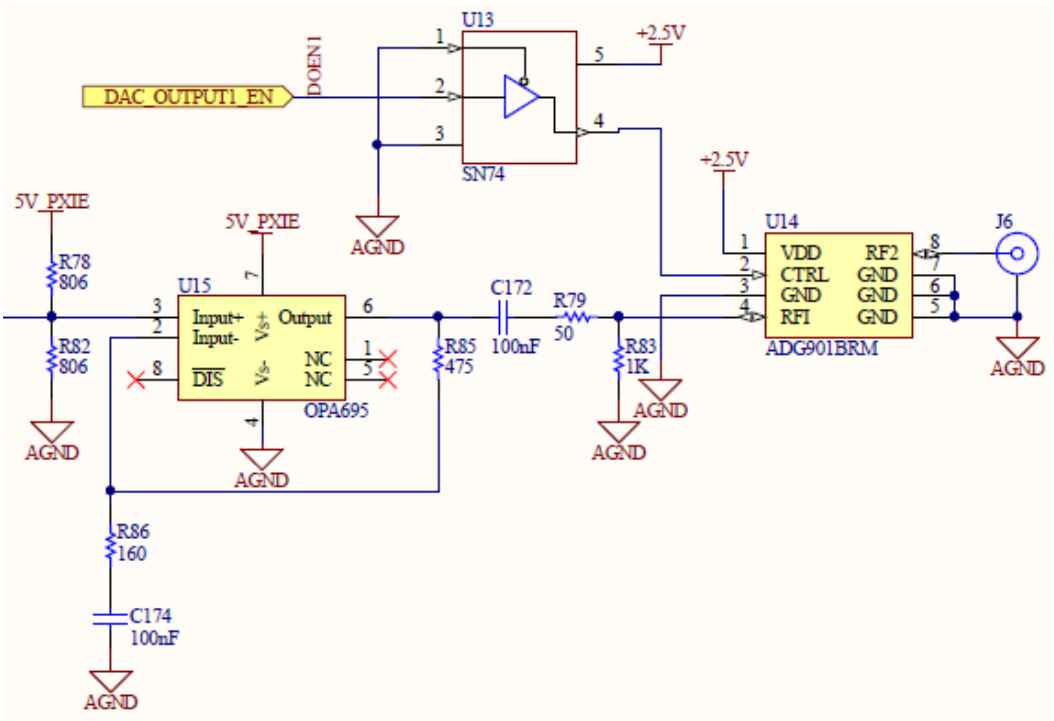


Fig. 3.19 Schematic of the gain and RF switch circuit in the front-end circuits.

3.3.4 Clock Generator Implementation

Architecture of the Clock Tree

Figure 3.20 describes the architecture of the clock tree.

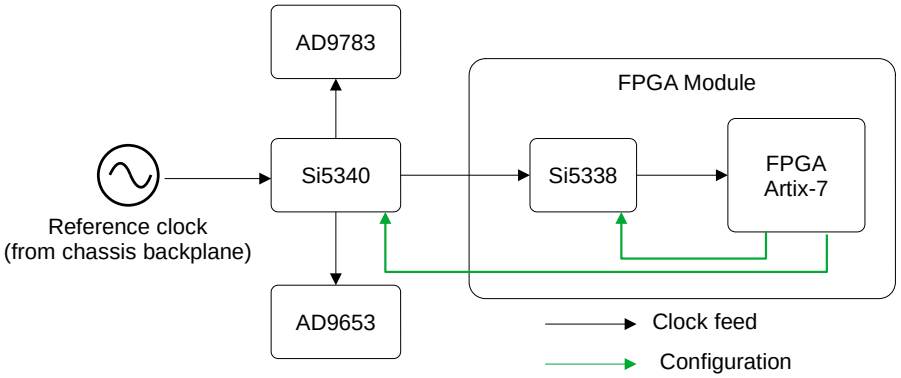


Fig. 3.20 Diagram of the clock tree and its configuration.

Input Clock Termination

In this design, the reference clock (PXIe_CLK100) from the backplane was the clock tree input, as is shown in Figure 3.20. The specification [54], states that this clock must be terminated on the peripheral with the LVPECL termination for the buffer to drive PXIe_CLK100. Otherwise, there is no clock being driven on the pair to that slot. Figure 3.21 shows the schematic of the clock input termination.

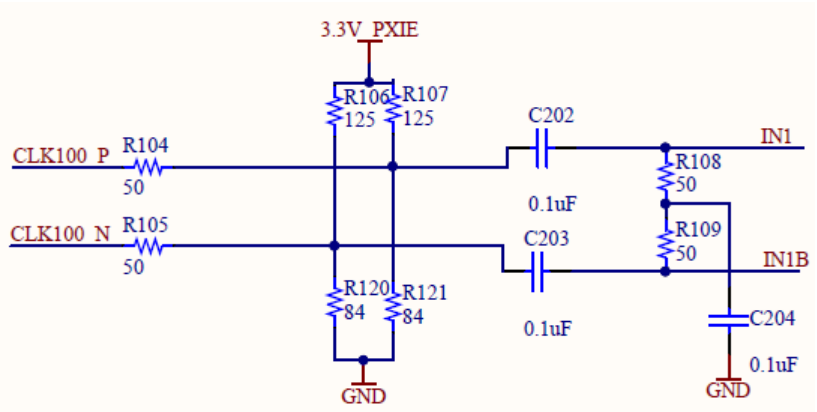


Fig. 3.21 Schematic of clock input termination.

Clock Generator Configuration

The two clock generators, the Si5340 and Si5338, are not configured upon powering up. To generate the desired clocks for the devices, e.g., the ADC and the DAC, the two clock generators need to be configured by a processor, the MicroBlaze, in the FPGA. However, during configuration, the clock generators do not have outputs, which means that the FPGA can not rely on the output clock of any clock generator for configuring them. Fortunately, the internal clock source inside FPGA can be used for configuring the clock generators. The first step is to configure Si5340, and one output of Si5340 feeds Si5338. The second step is to configure Si5338. Once the configured clock generators operate, the FPGA could rely on them for the rest of the FPGA firmware. More details of the configuration are presented in the FPGA design in section 3.4.9.

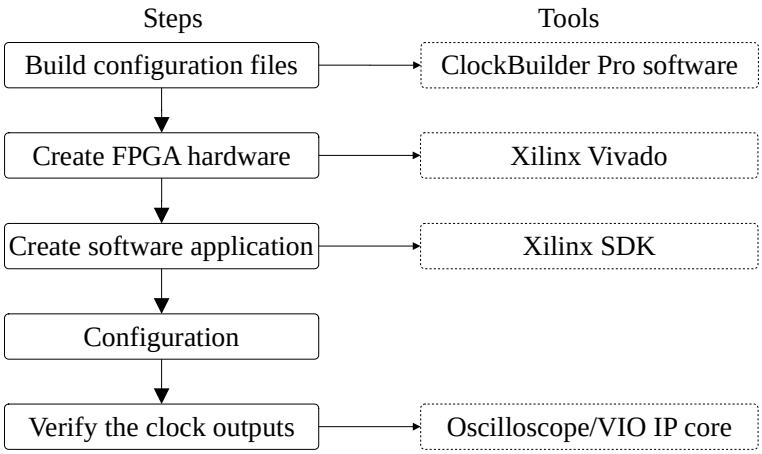
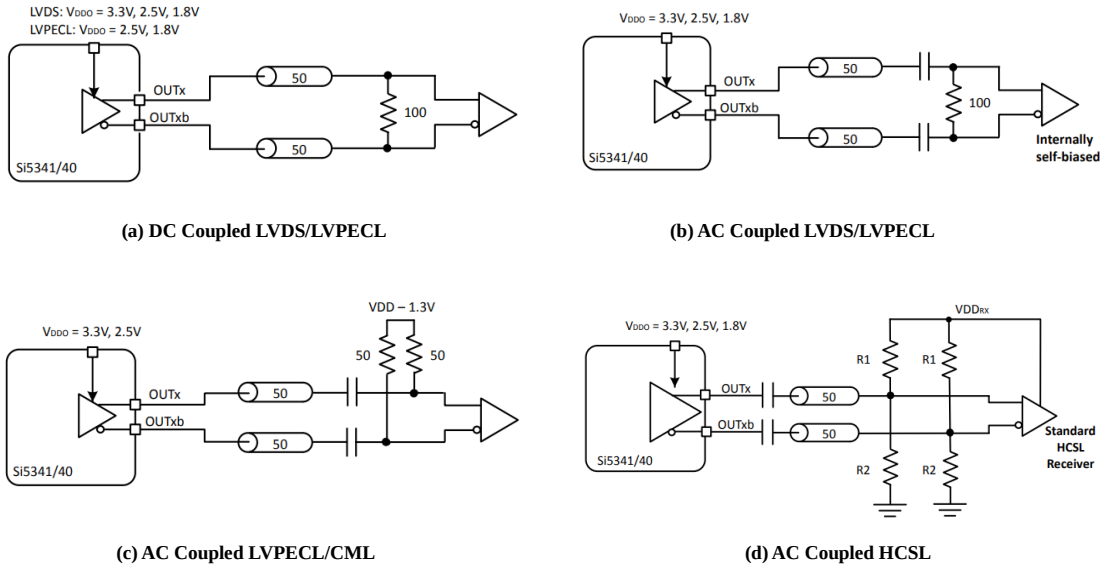


Fig. 3.22 Clock configuration steps.

Figure 3.22 shows the flow chart for configuring the two clock generators. Firstly, the software tool is used to generate the configuration file. Secondly, the soft processor writes the values to the specific addresses in the clock generators.

The software tool, ClockBuilder Pro [71], allows designers to customize the clock generators in terms of output frequencies, voltage, input channel, etc. Once all the parameters are determined on the graphical user interface, the tool produces a configuration file consisting of the values and addresses for the related registers in the clock generators.

Terminations



(a) DC Coupled LVDS/LVPECL

(b) AC Coupled LVDS/LVPECL

(c) AC Coupled LVPECL/CML

(d) AC Coupled HCSL

Fig. 3.23 Supported differential output terminations of Si5340 [70].

The terminations deal with the output types of Si5340 and the clock input types of the devices to be clocked. Figure 3.23 illustrates the four types of output terminations of the clock generator, Si5340. One of the output terminations must be matched with these devices' recommended clock input terminations when implementing the connections between the clock generator and the devices to be fed. According to the datasheet of the AD9653 [61], the optional clock input termination could be differential ac-coupled LVDS or differential PECL. The ac-coupled LVDS was adopted as the termination, as drawn in the schematic in Figure in the appendix. The recommended clock input termination is ac-coupled LVDS with a 200 mV - 400 mV common-mode voltage for the DAC clock input termination.

3.3.5 Power Supplies and Buffers

The components on the transceiver board need a different level of voltage supplies. Figure 3.24 shows the architecture of the power supply.

A low-noise adjustable voltage regulator, LP3878 (Texas Instruments, Texas, U.S.), was selected for translating voltages. Its input supply voltage ranges from 2.5 V to 16 V, and

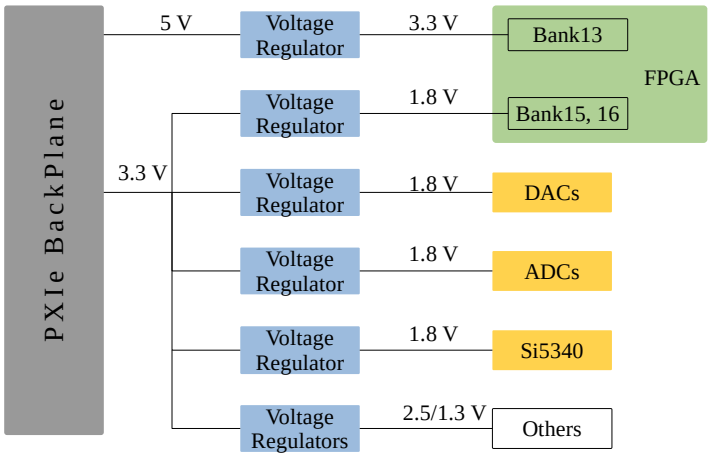


Fig. 3.24 Power supply architecture.

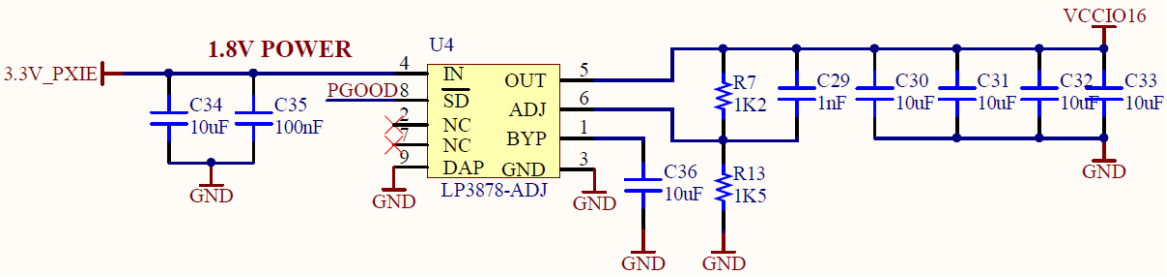


Fig. 3.25 Schematic of one of the power supply circuits.

the output ranges from 1 V to 5.5 V. Figure 3.25 shows one of the voltage regulators; it transforms the 3.3 volts to 1.8 volts for ADC. The voltage of ADJ determines the output voltage. The capacitors are used to eliminate the small spurs of the voltage to make the power supply stable. R7 and R13 are used to adjust the output voltage.

$$Vout = Vadj * (1 + (R7/R13)) \tag{3.4}$$

In this design, Vadj = 1 V. The PGOOD is a control signal from the FPGA. When the FPGA detects that the power supply for itself is proper, it will enable the rest power supply chips by asserting the PGOOD high.

The TTL signals with 3.3 V or 1.8 V could be generated by the FPGA directly, but a 5 V TTL signal for enabling or disabling the power amplifier needs to be buffered from a 3.3 V or 1.8 V FPGA signal. This application uses an 8-Bit dual-supply bus transceiver with 3-state outputs, the buffer diagram was shown in 3.26.

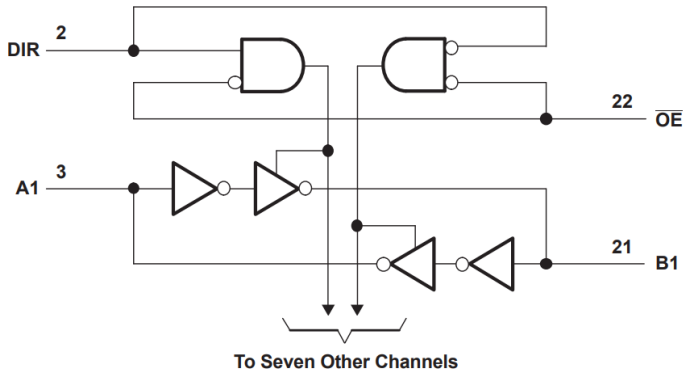


Fig. 3.26 Buffer diagram, SN74LVC8T245 [72].

3.3.6 Connection with the Backplane

The connection between the transceiver board and the chassis backplane are described in Table 3.15.

Table 3.15 Key signals on the connectors.

Key signals	**Description**
PXI_TRIG(0∼7)	PXI trigger signal.
PXIe_DSTAR	PXIe star signals are used for synchronization. Usually, it is cooperating with the timing board.
PXIe_CLK100+/-	This differential pair is the system reference clock.
5 V, 3.3 V	These supply the power for the whole transceiver board.
1PETn0 1PETp0 1PERn0 1PERp0	One-lane signal of PCIe. Transmitter includes a differential pair, and so does the receiver.

3.3.7 PCB Design Considerations

Figure 3.27 shows the board layout of the transceiver board. The stackup design followed the recommendation and design by Altium designer [73] and Analog Device [74].

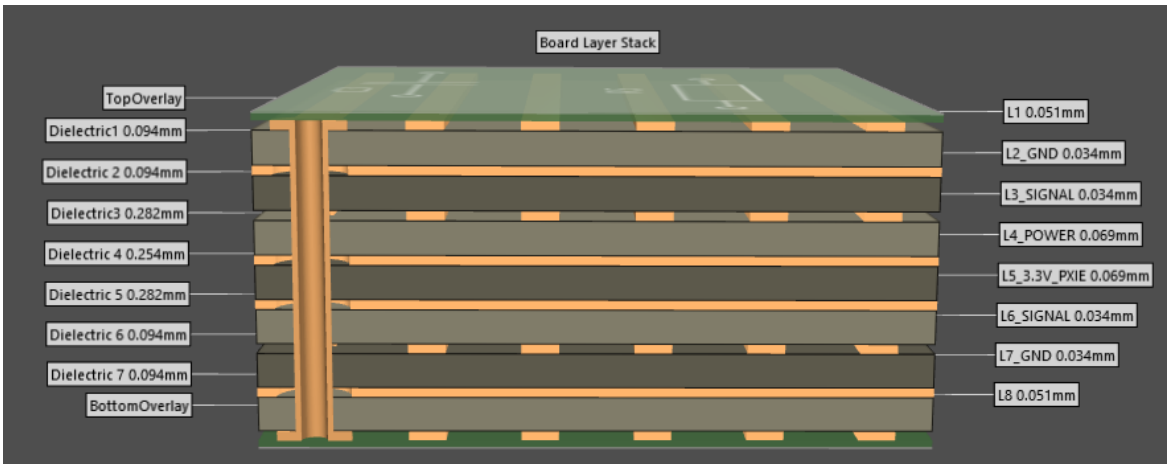


Fig. 3.27 Transceiver board layer stack.

Table 3.16 Considerations for PCB layout and assembling.

Items	Notes
Grounds	Analog Ground and Digital Ground.
Differential Signals	Differential signals are applied in several parts on the transceiver board. During the PCB layout, one critical issue is to maintain the same length of two single signals in a differential pair.
Layer Stack	To consider the electromagnetic interference.
Coupling capacitors	Multiple coupling capacitors are mounted around the power connecting points.
Assembling order	The transceiver board was constructed in the order of power supply, FPGA module, DAC, and ADC.

Analog Ground and Digital Ground

The front-end circuits of ADC and DAC are analog circuits on the transceiver board, and the rest are digital. To avoid interference between the two parts, analog circuits have separate grounds and connect to the digital ground at one point. This strategy prevents the currents from the digital section from interfering with the analog sections.

3.3.8 Transceiver Board Overview

Figure 3.28 shows the picture of the multichannel transceiver board. Because the prototype of the transceiver board has not been verified, it is necessary to assemble the components on the board part by part to make sure there is no fatal error in the design. Firstly, the power regulators were mounted and verified. Secondly, the DAC and the auxiliary circuits were tested. The ADC part was verified at last as it is the most expensive component on this board.

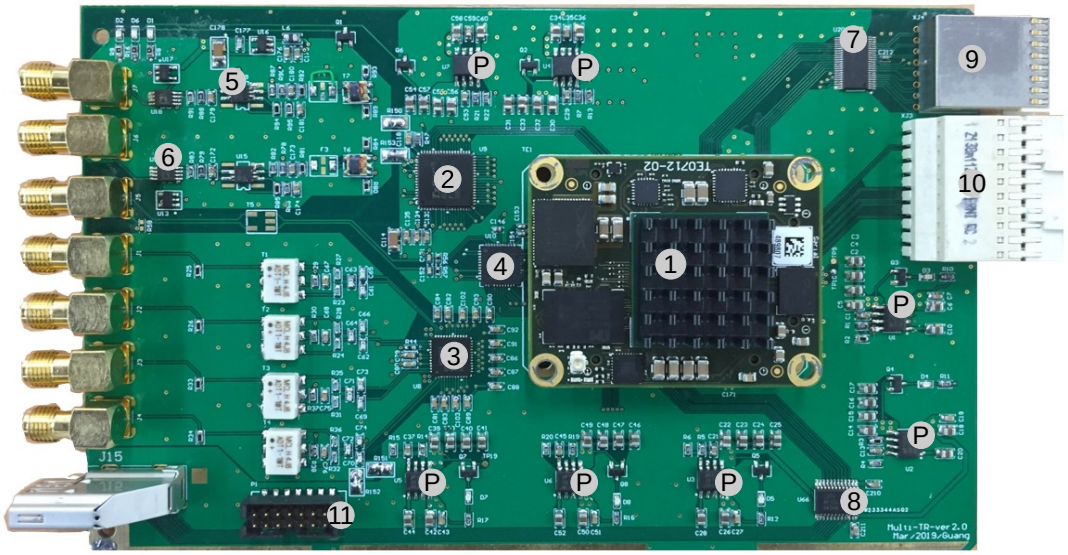


Fig. 3.28 Multichannel transceiver board.

3.4 FPGA Firmware Implementation

3.4.1 Design Tools and Flow

Vivado Design Suite

In the Vivado Design Suite, Xilinx provides abundant IPs, such as MicroBlaze (a soft-core microprocessor), DDS Compiler, DMA, GPIO, etc., which allow users to use them directly in the block design without coding. Users also can build their own IP blocks to be added and cooperate with the provided blocks.

Figure 3.29 shows the view of the block design in Vivado. In the block design, AXI (advanced extensive interface) [75] is an interface protocol, which is a fundamental bus used in the FPGA design. There are three types of the AXI4-interface:

- AXI4. It is used for high-throughput memory-mapped requirements, for example, the AXI DMA [76].
- AXI4-Lite. It is used for low-throughput memory-mapped communication, for example, AXI GPIO [77].

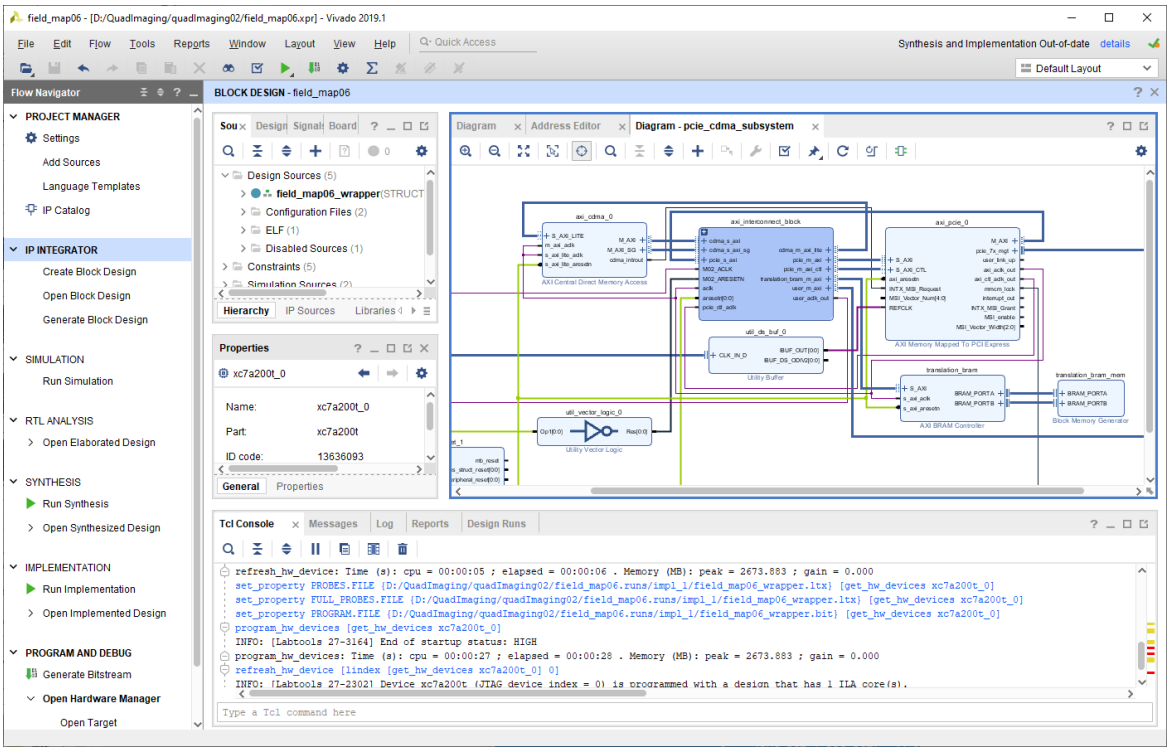


Fig. 3.29 Vivado IP Integrator view. A screenshot from Vivado which is used for developing the FPGA firmware.

- AXI4-Stream. It deals with high-speed streaming data, and it is not involved in memory-mapped cores, for example, the DDS Compiler [78].

As shown in Figure 3.30, AXI4 and AXI4-Lite protocols defines five channels: read address channel and read data channel in a read transaction, write address channel, read address channel and response channel in a write transaction. Each channel is an independent collection of AXI signals, such as VALID and READY signals. In the AXI read transaction, the AXI master sends the address read channel to the slave to set the address, then the data at that address is transferred from the slave to the master via the read data channel. In the AXI write transaction, the address is first sent from the master to the slave to set the address. Then, the data for that address is sent from the master to the slave through the write data channel. Lastly, a write response is sent from the slave to the master to indicate that the data transfer is successful.

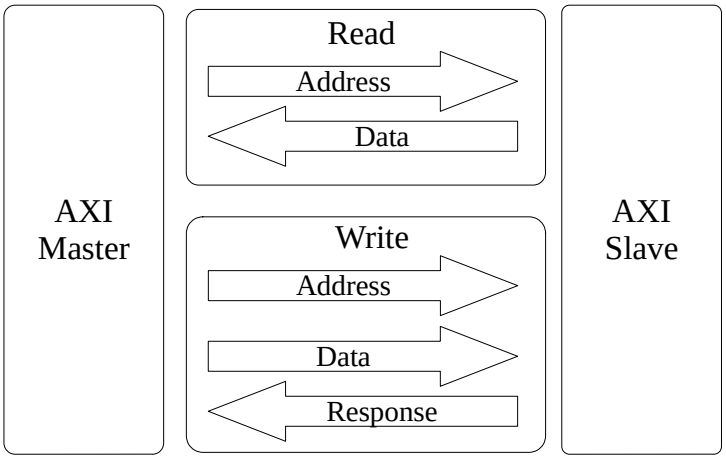


Fig. 3.30 AXI bus protocol [75].

Software Development Kit

The Xilinx SDK is the Integrated Design Environment (IDE) for creating embedded applications on Xilinx's microprocessors, such as Zynq-7000 SoCs, MicroBlaze. XSDK provides user-customizable drivers for the hardware IPs from Vivado. Figure 3.31 shows the XSDK software environment.

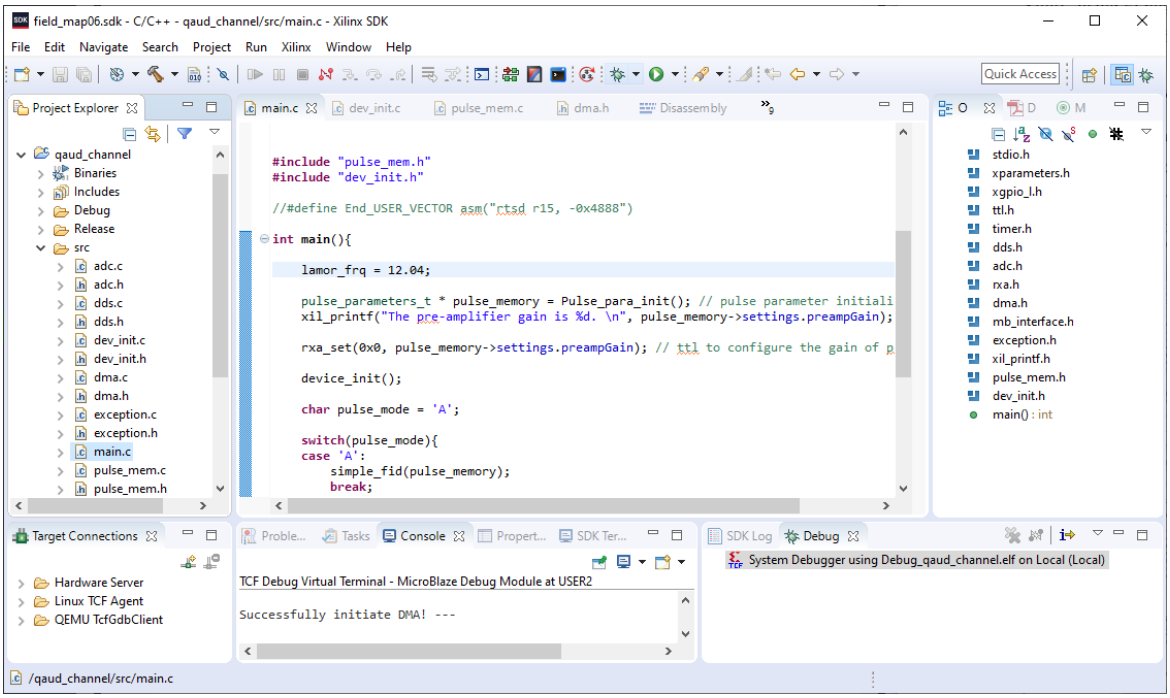


Fig. 3.31 XSDK software environment. A screenshot from the XSDK software which is used for developing the embedded application.

Debugging

Figure 3.32 shows the flow chart of the FPGA project design with Xilinx tools. In Vivado, a .hdf (hardware) file is created after finishing the block design. XSDK uses the hardware file that contains addresses or configuration of the logic cores in the block design for software development. During developing the FPGA firmware and the software, Vivado generates the bitstreams and programs the FPGA via JTAG cable. Then XSDK debugs the software with the created elf file (debug version) via JTAG cable. Once the development is completed, the elf file (completed version) is added to Vivado to generate the bitstreams file to be stored in the flash memory. In this way, the FPGA can be automatically booted, i.e., the firmware is implemented into the FPGA, and then the application is operated.

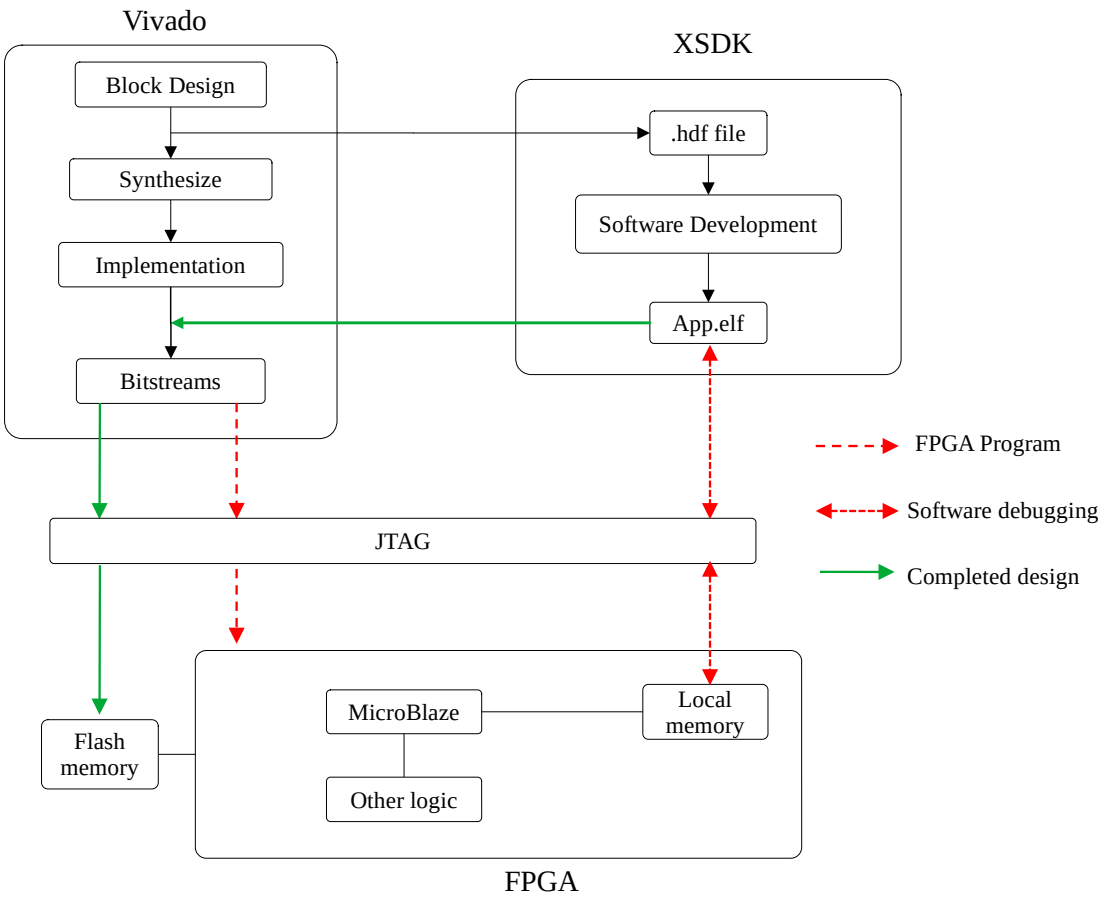


Fig. 3.32 Xilinx FPGA design flow.

3.4.2 FPGA Firmware Design Overview

The FPGA design provides the required functionalities for an NMR or MRI spectrometer. Figure 3.33 shows the FPGA design diagram, and it can be broken down to the following parts:

1. MicroBlaze-01, MicroBlaze-02 are softcore processors. MicroBlaze-01, in conjunction with configure interface FPGA module, is used for configuring the two clock generators. Once the configuration of the clock generators is finished, MicroBlaze-01 will reset the MicroBlaze-02 for other FPGA firmware, e.g., the pulse sequence generator. The configuration of the clock generators are illustrated in section 3.4.9.

2. Time management. For NMR or MRI experiments, time management is important for accurate synchronization. An AXI Timer is used to realize time management. This part is elaborated on in section 3.4.8.

3. TTL signals producing. TTL signals are produced by the AXI GPIO core configured by MicroBlaze-02. The TTL outputs of the GPIO core, including internal and external TTLs, are used for enabling or disabling the functional cores inside the FPGA or the devices outside the FPGA. This is the key strategy for the synchronization of the pulse program generator.

4. DAC driver. This block configured by MicroBlaze-02 feeds the digital signals to DACs for producing an analog signal. This part is presented in DAC Driver in section 3.4.6.

5. ADC interface. This block realigns the distorted alignments of the ADC outputs and deserializes the serial data into parallel data for further processing in the FPGA. This part is discussed in section 3.4.3.

6. Digital receiver processor. Instead of using analog mixing, a digital receiver processor is implemented. This design is detailed in section 3.4.4.

7. Data movement. The processed data is moved into DDR memory through a subsystem, Data Movement block. This design is detailed in section 3.4.5.

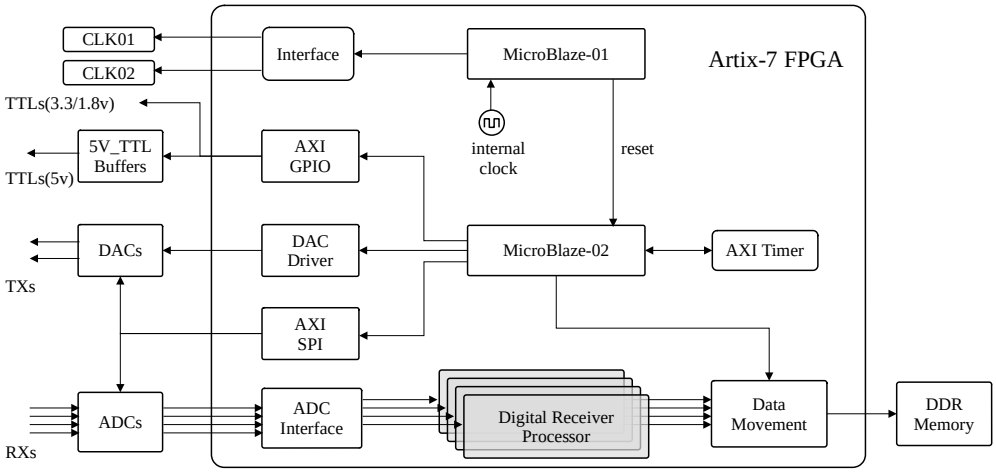


Fig. 3.33 FPGA design overview.

3.4.3 ADC Interface

Background

As mentioned earlier, an ADC interface FPGA module needs to be implemented to deal with the forthcoming signals (data clock, data, and frame clock): realign the phase-distorted signals and deserialize the serial data to parallel data for further processing. Before digging into the details of the ADC interface module, a few points need to be reviewed for this application:

1. As indicated in the ADC implementation, the ADC needs to be configured to DDR and two-lane mode, which means that 16-bit data is split into two 8-bit data, and each lane will carry 8-bit data.

2. The sampling rate of the ADC is set up as 100 MHz, which leads to the data clock operating at 400 MHz, as indicated in Table 3.13.

3. The data clock provided by the ADC is 90 degrees out-of-phase with respect to the data signals and frame clock, as shown in Figure 3.14.

The ADC interface FPGA module is implemented based on the Xilinx XAPP524 Application, Serial LVDS High-Speed ADC interface application note[79]. It is designed for applications

for different situations, such as 12-bit, 14-bit, or 16-bit, or different lane numbers or data rate modes (SDR/DDR). In a particular application, the bit-number and lane number must be configured in the FPGA module to match with the ADC setup. The following subsections introduce the interface design in our specific configurations of ADC, say, two-lane mode, DDR mode, 16-bit, etc., to make it easier to comprehend the design.

Key Resources

The two primitive components, ISERDESE2 and IDELAY2 [80], are important resources for the ADC interface FPGA module, and they are available in both high-range (HR) and high-performance (HP) I/O banks.

ISERDESE2, which is diagrammed in Figure 3.34, is an input serial-to-parallel logic or deserializer. In the diagram, the serial data D, C, B, A go to input (D) of the ISERDESE2 at the rate of CLK, because the deserializer is used for dealing with the data from the ADC chip, and the frequency of CLK is equal to the data clock which is 400 MHz. The CLK_DIV is operating at 100 MHz, of which the frequency is equal to the frame clock. Basically, for acquiring 16-bit data, 4 ISERDESE2s is used.

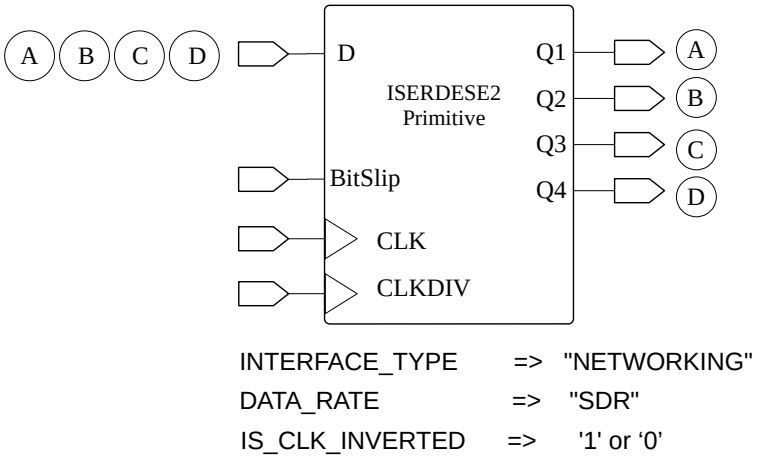


Fig. 3.34 Diagram of the ISERDESE2 [80]. The "INTERFACE_TYPE" must be set as "NETWORKING" mode under which the Bitslip function is available (this function is discussed in subsection "Discovery of Frame Clock" The "DATA_RATE" is set to "SDR" mode.). IS_CLK_INVERTED is set to either "1" or "0" to match the differential data clock.

IDELAY2 is a programmable delay primitive in each I/O bank. It allows the incoming signals to be delayed by a certain time by increasing or decreasing the calibrated tap. In this application, it is used for adjusting the ADC data clock to a proper position. When using IDELAY2, the IDELAYCTRL submodule must be instantiated to provide the taps.

The IDELAYCTRL submodule continuously calibrates the individual delay taps of IDELAY in its region to reduce the effects of the process, voltage, and temperature variations. The IDELAYCTRL calibrates the IDELAY2 using a 200 MHz reference clock, which results in the tap with a resolution of 78 ps.

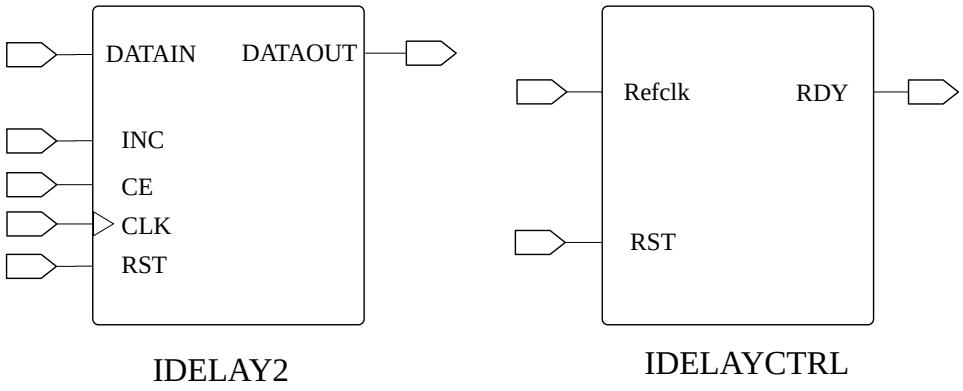


Fig. 3.35 Diagrams of IDELAY2 and IDELAYCTRL [80]. In IDELAY2, DATAIN is the input to connect with the clock source, which needs to be adjusted. DATAOUT is the output of processed clock. The adjustment is realized by increasing or decreasing the taps which are configured by the inputs of CE and INC, when INC = 1, increments, INC = 0, decrements. The tap resolution is determined by IDELAYCTRL, which must be instantiated when using IDELAY2. The designated reference clock for IDELAYCTRL in Artix-7 FPGA is 200 MHz. In addition, during instantiating IDELAY2, it is necessary to tell IDELAY2 that the frequency of the reference clock is 200 MHz by setting "REFCLK_FREQUENCY => 200.0".

Table of the clocks

Table 3.17 lists the different clocks running in the ADC interface module.

Table 3.17 Related clocks for the ADC interface module.

Name	Frequency (MHz)	Notes
DCLK	400	Data clock from ADC chip.
FCLK	100	Frame clock from ADC chip.
Bit_CLK	400	Delayed version of DCLK.
Bit_CLKDIV	100	New frame clock, phase aligned with Bit_CLK.
Refclk	200	A reference clock for IDELAYCTRL module.

Delay of the Data Clock

Ideally, the already properly aligned signals (data, frame clock, and data clock) go to the related logic without any skew. However, the high-speed data clock needs necessary buffers, such as IBUFO or BUFR, to drive the following logic, ISERDESE2, for deserialization. Figure 3.36 shows the connection between the ADC and the ADC interface inside FPGA. Inevitably, the alignment is spoiled. However, the data and frame clock go to the inputs of ISERDESE2 directly, which means that they are still aligned without phase shifting. Therefore, the problem can be simplified as adjusting the phase of the data clock only.

Figure 3.37 diagrams the delays in a more precise way. The top two signals represent the data, and data clock (DCLK) from the output of ADC and 90-degree phase aligned. The center (or the eye) of the data is located on the rising or falling edge of the data clock for ensuring data validity. When these signals are sent to the FPGA, the data signal is delayed due to the routing but the data clock, which becomes Bit_clock, experiences more delay because of the buffer. Thus, the center of the data might shift away from the edges of the data clock, making the data not valid for Bit_clock anymore.

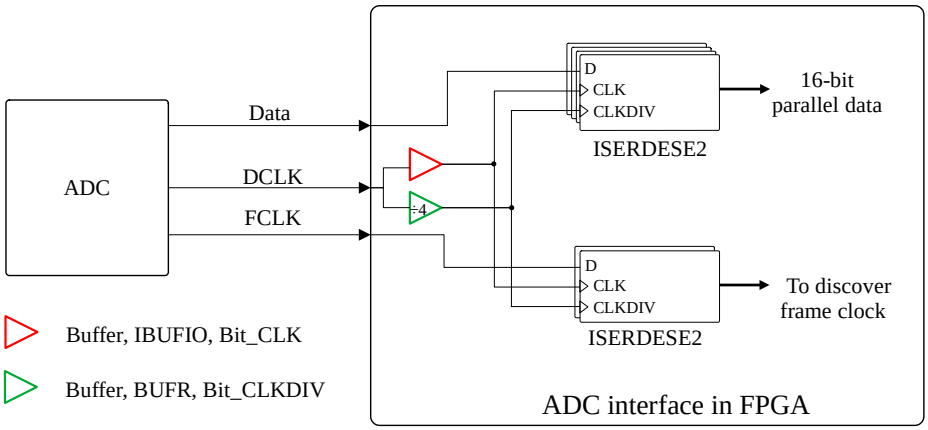


Fig. 3.36 Diagram of connection between ADC and ADC interface FPGA module. At the output of ADC chip, data, frame clock and data clock are properly aligned. More specifically, there is no phase shift between the data and frame clock, but the data clock is 90-degree phase shifted to data signals. When these signals go to the ADC interface in the FPGA, the data signal and frame clock reach directly to inputs (D) of ISERDESE2, which are clocked by the data clock through buffers. The buffers do spoil the alignments, particularly the relative position between data and data clock.

Data Clock Alignment

Figure 3.37 shows the diagram that the spoiled alignment can be tackled by adjusting the relative position between Data* and Bit_CLK. Adjustment of Bit_CLK is realized by implementing IDELAY2 primitive. In that diagram, the green vertical line shows that the data and DCLK are perfectly aligned at the output of the ADC. After entering the FPGA, the data only encounters a routing delay. However, the DCLK becomes Bit_CLK for driving ISERDESE through clock buffer, IBUFIO, which encounters more delay than the data. The red vertical line indicates that the data center is not aligned with either edge of the data clock, which invalidates the data. One solution is to reposition the clock to the middle of the data valid eye, but it is unnecessary as the clock is already in the middle. Thus, the solution is to realign the edge of the Bit_CLK to either edge of the DCLK by implementing IDELAY2 to adjust the Bit_CLK position relative to Data*. Therefore, DCLK is continuous, which means just a delayed version can be used to sample the delayed data.

Furthermore, an ISERDESE2 primitive and the "Bit Clock Phase Alignment State Machine" are also adopted for detecting and adjusting the relative position between Data* and

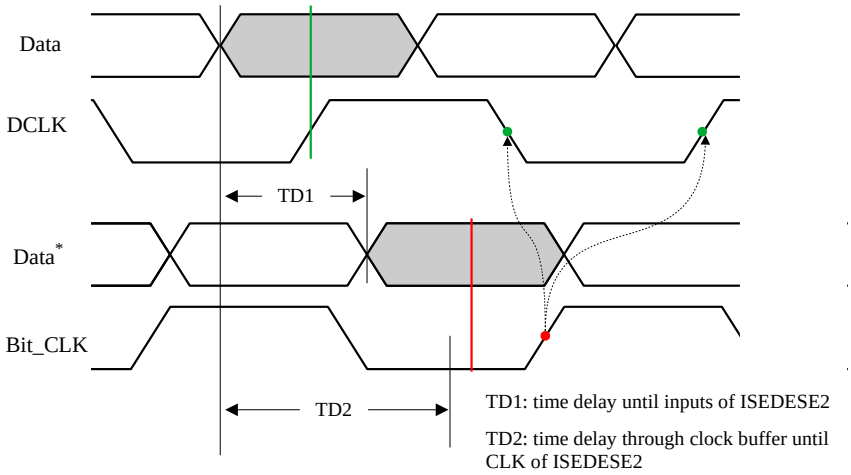


Fig. 3.37 Clock skew in the ADC interface [79].

Bit_CLK, as shown in Figure 3.38. After the alignment is finished, "Done" is asserted, which indicates that the Bit_CLK and Bit_CLKDIV are available for all ISERDESE2s used in this interface, including the ISERDESE2 component, which uses DCLK as data input. The DCLK registers itself in the ISERDESE2 that uses a delayed version of itself as the clock. That approach allows the user to be able to detect the position of the rising or the falling edge of DCLK, and therefore put the CLK and CLKDIV clocks of ISERDESE2 in any position in a period of DCLK by using "Bit Clock Phase Alignment State Machine".

The state machine monitors the parallel outputs of ISERDESE2, which are the eight serial bits captured from DCLK. When all the 8-bit parallel data are equal, all 1s or zeros, the state machine will decrease or increase the taps to postpone the DCLK by a certain period. By decreasing or increasing the taps, the state machine will obtain an opposite result, which means the edges of DCLK and Bit_CLK are close. The eventual position can be optimized by analyzing the practical situation.

When the reference clock of IDELAYCTRL is 200 MHz, the resolution of each tap is 78 ps. Therefore, within the tap number up to 32, the span of IDELAY2 is 2.5 ns, which is half of a DCLK period. There are several scenarios when detecting the edge of the DCLK. All the cases are illustrated in Figure 3.39. Initially, the state machine operates with a preset IDELAY2 delay at 16 taps. After sampling eight clock periods, which results in generating

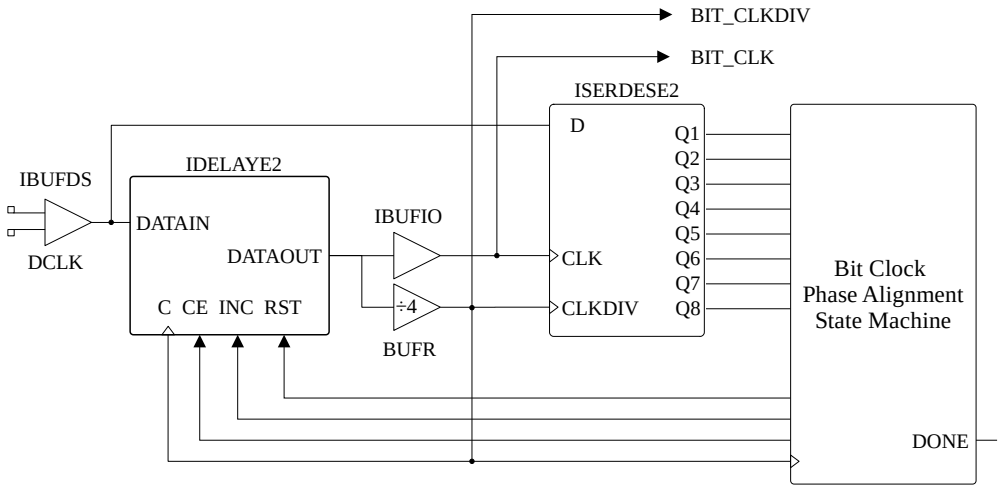
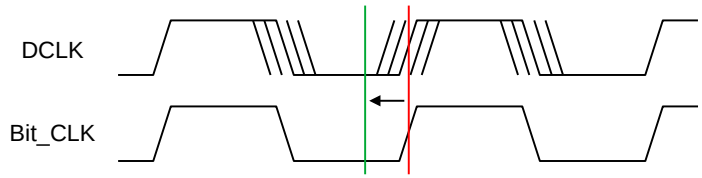
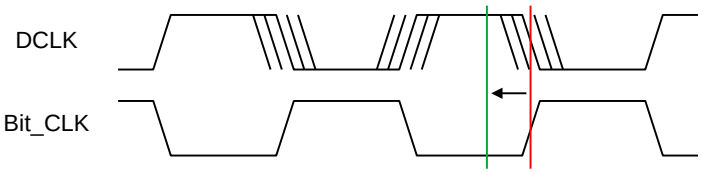


Fig. 3.38 Bit clock alignment in ADC interface [79]. The IDELAY2 is used in variable delay mode under which the delay value can be changed after configuration by manipulating the control signals CE and INC. IBUFIO, BUFR are phase-matched. All ISERDESE2 used in this interface need to be driven by clock buffers. CLK is driven by BUFIO, and CLKDIV is driven by BUFR.

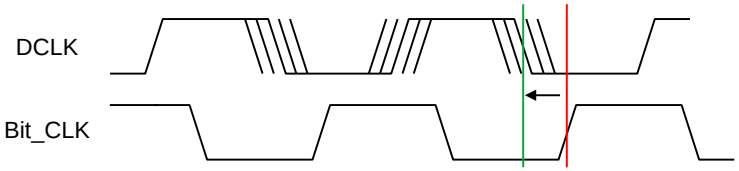
8-bit parallel data. Then the state machine steps backward by decreasing the number of steps and measures the outputs of each step. By comparing the results, the relative position between the two clocks can be found, and then the delay the Bit_CLK should have to align its edge with the DCLK's edge can be determined. The first four cases are straightforward, and the edge can be found in changing three steps. Case 5 is different from the last 4 cases, as both of the detection of the initial point and three taps step point are stable. Then the state machine should increase the taps to find the unstable point and then verify the result by increasing the tap until the stable point is discovered. Once the data clock is aligned, the Bit_CLK and Bit_CLKDIV is available for CLK and CLKDIV inputs of all the ISERDESE2s in this interface.



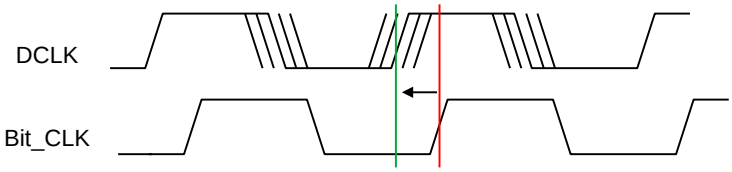
Case 1: -3 taps, from Q[8:1] unstable to Q[8:1] stable at 0. Rising edge found.



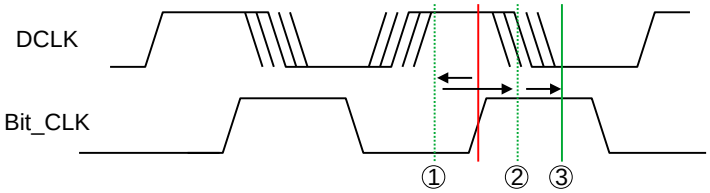
Case 2: -3 taps, from Q[8:1] unstable to Q[8:1] stable at 1. Falling edge found.



Case 3: -3 taps, from Q[8:1] stable at 0 to Q[8:1] unstable. Falling edge found.



Case 4: -3 taps, from Q[8:1] stable at 1 to Q[8:1] unstable. Rising edge found.



Case 5: -3 taps, from Q[8:1] stable to another stable status. Then increase the taps, until an unstable status is detected, then continue to increase the taps, an opposite status will be found.

Fig. 3.39 Data clock edge detection [79]. In the diagrams, the slashes around the DCLK edge stand for the jitter, which is unstable. The red vertical line is the start point of sampling DCLK. The green vertical line represents the endpoint of delay.

Discovery of Frame Clock

The frame clock that comes out from the ADC is a slow-running clock. Therefore, the clock is a digital version of data of the Bit clock with a known and regular pattern. Since the frame clock is phase-aligned with the data, the frame clock can be used to train and re-align the captured data in the FPGA. Figure 3.40 shows the diagram of the frame discovery circuit. The complementary signals of the frame clock are sent to two individual ISERDESE2, and the 8-bit data of the frame clock can be detected. However, it is unknown which complimentary signal (positive or negative) is detected first due to phase between Bit_CLK and DCLK, which could be 0 degree or 180 degrees.

Because the captured frame clock data, say 1111_0000, which is symmetric, it is unknown whether the data is swapped. The state machine sends a bitslip signal to one of the ISERDESE2, and then the 1111_0000 will shift to 0111_1000 to address that problem. The captured data is 1011_0100. The "compare" function block compares the swapped output with a constructed pattern to see if they are matched.

ADC Data Interface

Now that the bit clock and frame clock is valid, it is time to deal with the data. Figure 3.41 diagrams the data interface which mainly consists of 4 ISERDESE2s, 2 Multiplexers, and several slips or swaps commands from the frame clock discovery circuit.

Testing the ADC Chip and the ADC Interface

Since the ADC converts the analog signal into a digital signal, it is hard to measure the output directly from the output of the ADC with an oscilloscope. However, the Test Mode configured via the SPI port in the ADC chip can be applied for the testing.

In the Test Mode, user-defined values can be written to the ADC via SPI, and the ADC will output the values to be captured in the ADC interface module. Either expected outputs can be either captured by an ILA (Integrated Logic Analyzer) core [81] or read by the MicroBlaze core through a GPIO block to verify the ADC and the ADC interface. The two

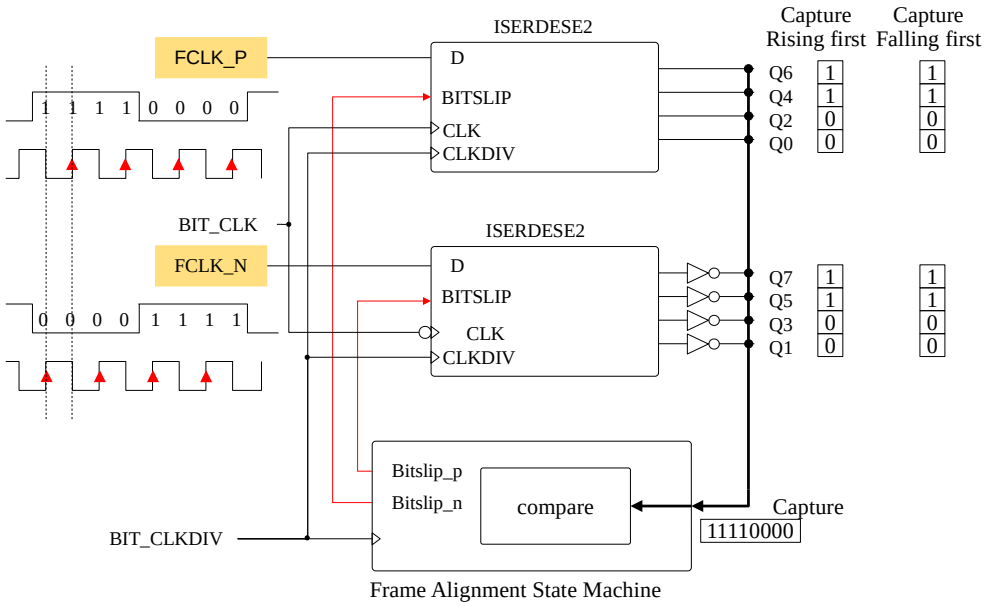


Fig. 3.40 Frame clock discovery diagram [79]. The two ISERDESE2s are set as "NETWORKING" mode under which the BitSlip function is enabled. Both of them are used in SDR mode, but the mode of "IS_CLK_INVERTED" is different because only in this way the two SDR mode ISERDESE2s can sample the whole data of the frame clock, like the red arrows in the Bit_CLK (When ISERDESE2 is in SDR mode, sampling happens on the rising edge. But the CLK input can be reversed.). In addition, the output of the ISERDESE2 that deserializes the negative signal of the frame clock must be inverted. The inverted results are combined to 8-bit data, which is sent to Frame Alignment State Machine.

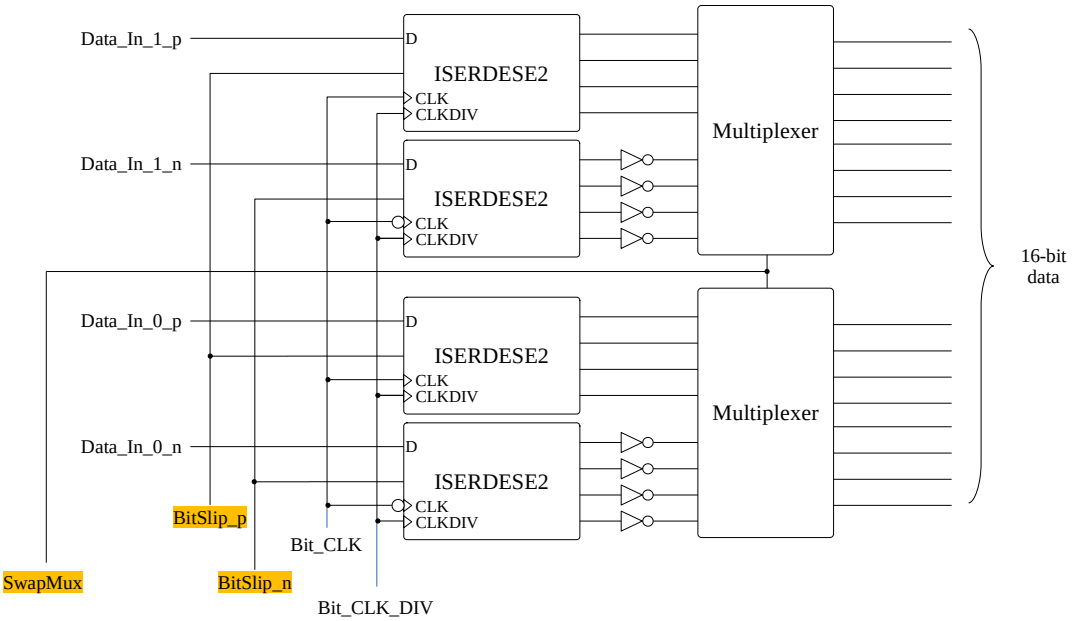


Fig. 3.41 Diagram of ADC data interface [79]. This submodule uses 4 ISERDESE2s to produce 16-bit data in one frame clock period. The BitSlip_p(n) signals are from the state machine in the frame discovery submodule. The SwapMux signal is used to swap the slipped data back to the correct order.

methods are illustrated in Figure 3.42 and Figure 3.43 separately. Once the ADC chip and the ADC interface are verified successfully, another step is to use a sine waveform, usually generated by a signal generator, to feed the ADC to test the front-end circuits.

The method shown in Figure 3.43 can be embedded into the final application for verifying whether the ADC is operating properly or not. In the final application, the soft processor can switch between the two modes: Test Mode for verifying the ADC before conducting an experiment and Running Mode for normal data acquisition after the ADC verification.

3.4.4 Digital Receiver Processor

A digital receiver processor unit is used to process the MR data digitized by an ADC, as shown in Figure 3.44. Table 3.19 lists the logic cores used in the digital receiver.

These raw signals are further processed in a digital receiver processor: mixing, decimating, filtering, and combined with being sent to the memory device through the AXI DMA for post-process for the final MR image. The 16-bit raw data from the ADC interface mixes

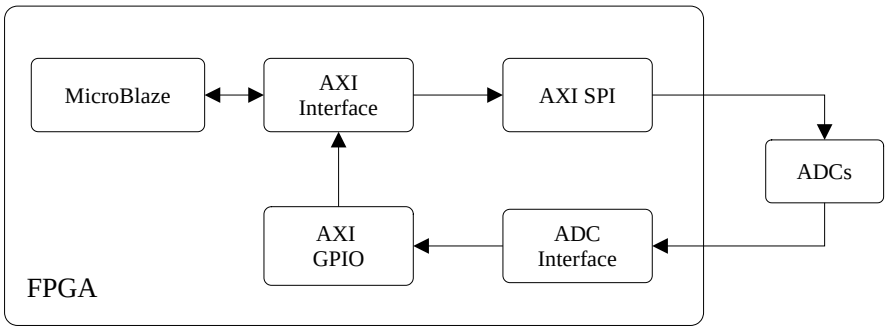


Fig. 3.42 ADC testing diagram, the ADC is configured at Test Mode at which the ADC generates repeated data. Xilinx logic IP cores, MicroBlaze [68], AXI Interface [75], AXI GPIO [77], and AXI Quad SPI [82] are utilized. ADC Interface is a custom IP core based on the application note [79].

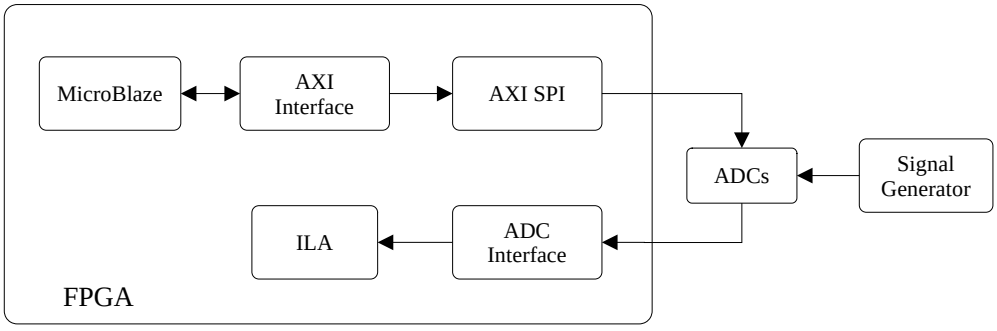


Fig. 3.43 ADC testing diagram, the ADC is configured at normal mode. Xilinx logic IP cores, MicroBlaze [68], AXI Interface [75], AXI Quad SPI [82], and ILA [81] are utilized. ADC Interface is a custom IP core based on the application note [79].

Table 3.18 IP cores used for verification of ADC interface.

IP Core	**Configurations and functions**
MicroBlaze	To configure ADC mode through AXI SPI and to read captured ADC data.
AXI SPI	Implementing the ADC mode configuration.
ADC Interface	To deserialize the captured serial ADC data to parallel.
AXI GPIO	To receive the ADC data and send it to the processor. As shown in Figure 3.42
ILA	Internal Logic Analyzer. To capture the output of the ADC interface in Figure 3.43 and to show it in analog mode.

with 16-bit sine and cosine waveforms separately in the mixer. The mixed signals (32-bit) are then simultaneously low pass filtered and decimated using a CIC filter. The decimation function in the CIC filter narrows the bandwidth with a ratio of 10:1, which reduces the large data volume while maintaining the key information. The 48-bit output data is then sliced to keep the high 32-bit data via a slice logic core. There are two reasons to keep the high 32-bit: one is that the 32-bit data is easier to transfer in the following stages, e.g., the DMA supports 32-bit, 64-bit data transfer; another is that the 32-bit high data already keeps the key information of the filtered data.

For multichannel receiving, the digital receiver processor can easily be replicated to four receivers in the FPGA design. They can operate parallel in synchronization, which is one of the advantages of using the FPGA.

Digital Receiver Processor Verification

A separated FPGA project is built to verify the design of the digital receiver processor. As shown in Figure 3.46, signals at several points, including the input of the mixer, an output of the mixer, and the sliced output from the CIC unit, are captured to verify the design of the digital receiver processor in the FPGA. The results captured from the ILA logic core are shown in Fig 3.46.

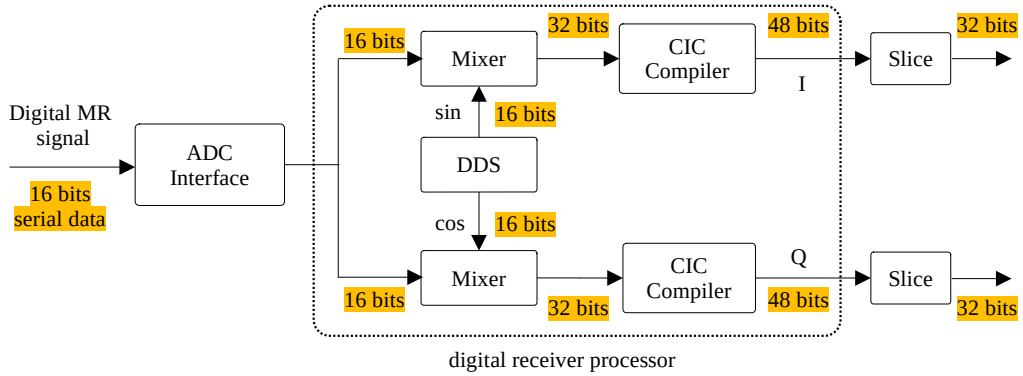


Fig. 3.44 Digital receiver processor. Xilinx logic cores, multiplier [83], DDS compiler [78], and CIC Compiler [84], are utilized.

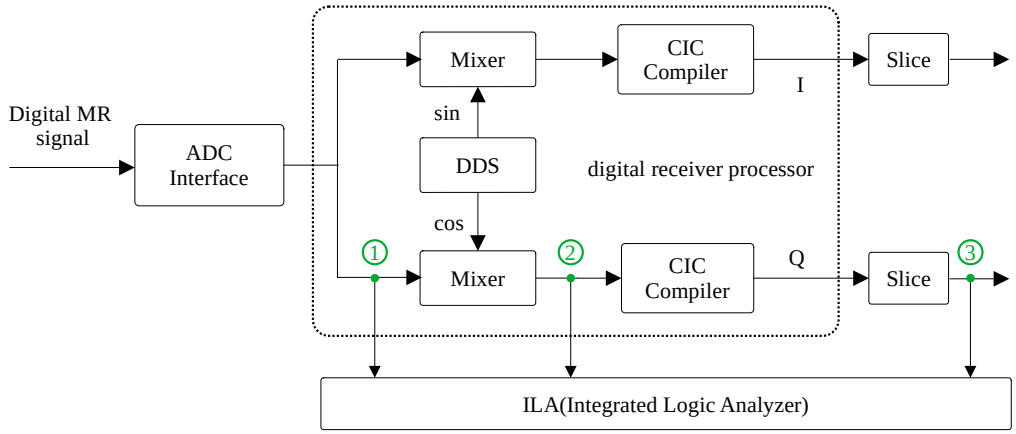


Fig. 3.45 Verification for the digital receiver processor.

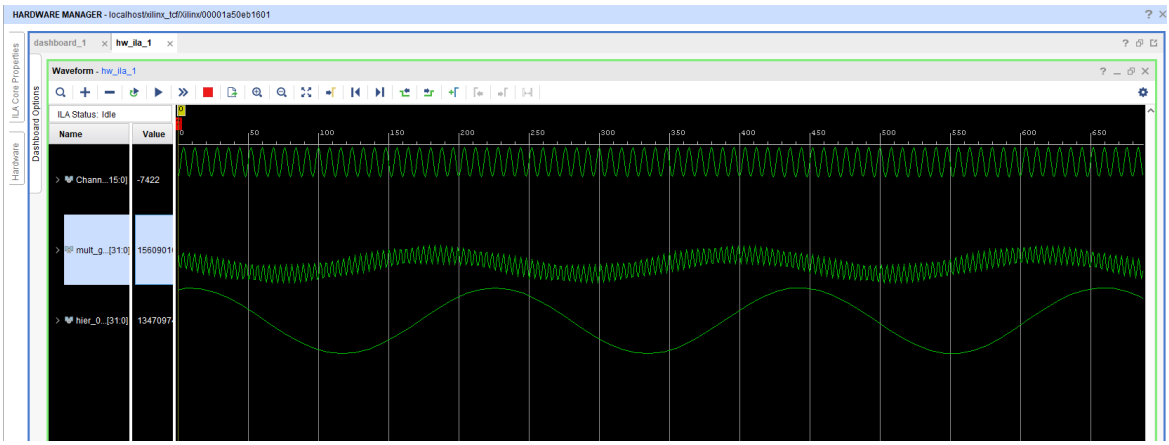


Fig. 3.46 Signal captured from ILA core.

Table 3.19 Digital receiver processor.

IP Core	Configuration and functions
Mixer	A multiplier. To multiply the digitized signal and reference sin or cos signal.
DDS	Direct Digital Synthesizer. To generate the sin or cos waveform. The IP core can easily alter the frequency and phase via MicroBlaze.
CIC	Cascaded Integrated Comb consists of multi-rate filters used for implementing large sample rate changes in digital systems. The filtered type is configured as decimation with a rate of 10:1. Normally, an FIR composition filter would be used to flatten the pass-band. In this application, the data is still over-sampled, and it can be further processed later in the host computer. The quantization mode is selected as full precision, which results in a 48-bit output.

3.4.5 Data Movement

Data Movement Diagram

After the raw MR signal is processed in a digital receiver processor, this stream data needs to be saved in the memory device for further process, as depicted in the FPGA firmware overview in Figure 3.33. As shown in Figure 3.46, this data movement subsystem transfers the stream data to the memory-mapped device, DDR memory, on the transceiver board. The data before going to the AXI DMA is stream data, and it becomes memory-mapped data after that. In that subsystem, multiple logic cores, such as Concat [85], Data Width Converter [86], AXI DMA [76], and AXI MIG [87], are utilized. Table 3.20 describes the functions or configurations of these logic cores.

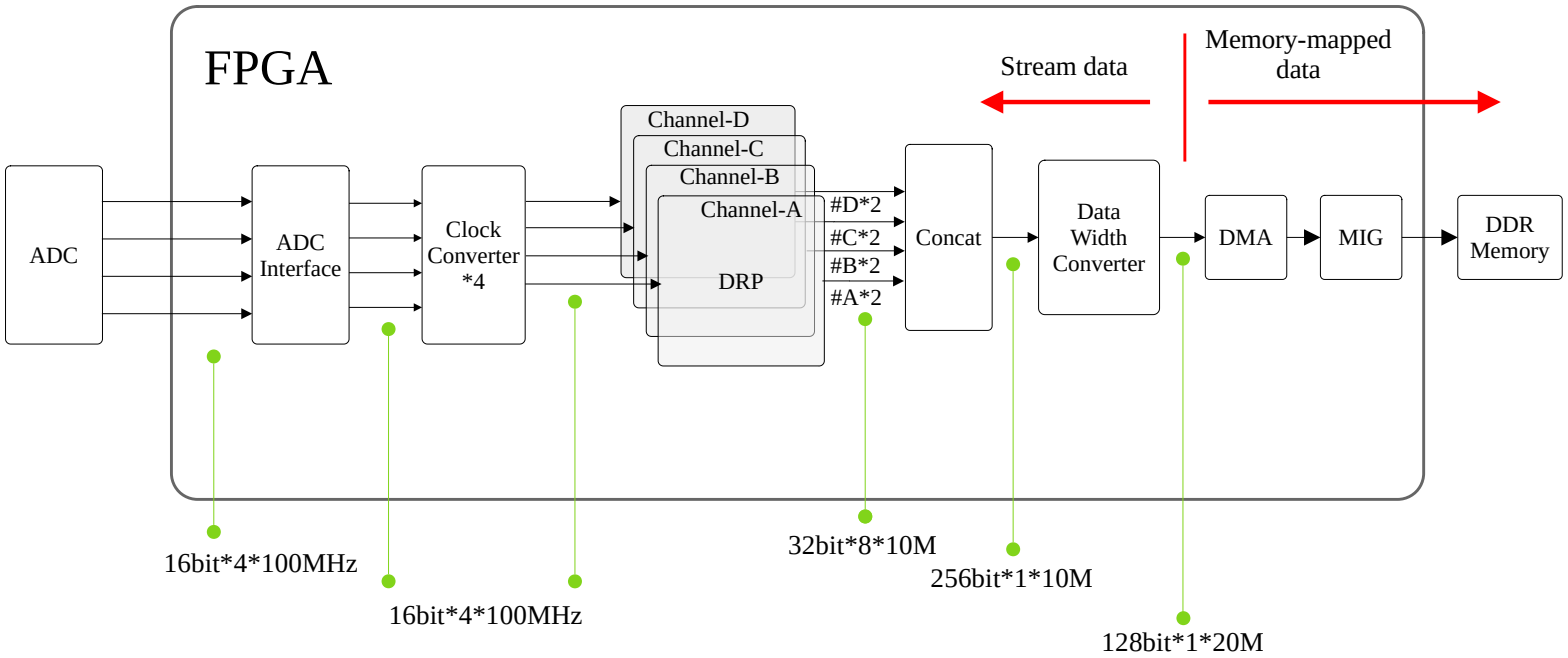


Fig. 3.47 Data flow from the ADC device to the memory device on the transceiver board.

Table 3.20 Logic cores used in the data movement subsystem.

IP Core	Configurations and functions
Concat	Concatenate. To combine all the outputs from the four-channel digital receiver processors. Each digital receiver processor has two outputs of 32-bit data. Then in total, the combined data is 256-bit.
Data Width Converter	There are two reasons for using the data width converter IP core here. One is that the output standard of the data width converter matches the AXI DMA; another one is that this core is used as a valve to acquire data or not. A TTL signal controls the turn on/off condition of the "valve".
AXI DMA	Direct Memory Access. This IP core moves the stream data to memory-mapped data for data saving. MicroBlaze configures the core in initialization, destination addresses, and data amount.
MIG	Memory Interface Generator. This IP core creates memory controllers in FPGA. MIG also generates multiple clocks, such as a general clock (100 MHz) for the micro processor and some other function blocks, a 200 MHz clock for supplying the reference clock of the ADC interface, and a 400 MHz clock for clocking out interleaved data in the DDS interleave core.

The AXI DMA (Direct Memory Access) is a device that provides high-bandwidth direct memory access between memory and AXI4-Stream type target peripherals. Using DMA has the advantage of offloading the data movement task from the microcontroller unit. In this practical application, the DMA core is configured to S2MM (stream to memory map) mode. Initialization, status, and management registers are accessed by the soft processor through an AXI4-Lite slave interface. In this application, the DMA transfers the stream data from the outputs of the digital receiver processor to the memory-mapped data, which is eventually sent to the DDR memory.

Data Movement Verification

The verification of the data movement subsystem is performed by an individual FPGA project, as shown in Figure 3.48. Eight constants or fixed values, instead of the outputs from the four digital receiver processors, are concatenated and sent to the DDR memory via the Data

Width Converter, DMA, and MIG. In this data flow, only the DMA needs to be configured regarding the DMA initialization, setting the amount and destination of the to be transferred data. Other logic cores in that flow operate automatically. After the "desired data" is stored in the DDR memory, the software, XSDK, could verify if the data matches with the fixed valued data by accessing the memory via a Debug Module [88].

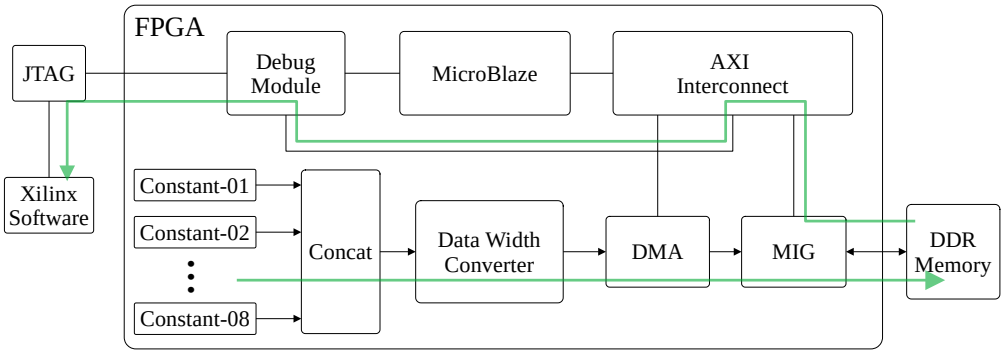


Fig. 3.48 Diagram of data movement testing in FPGA.

Figure 3.49 shows the flow chart of the configuration of the AXI DMA. Basically, the first step is to initialize the DMA and kick off it by giving the associated configuration information. However, during testing this subsystem in the FID experiment, the MR signals are not acquired after the $\pi/2$ pulse. The reason is that the starting DMA transfer is performed immediately after the DMA initialization, and initialization took several ms to complete. The initialization time is even longer than the FID signal which can be observed. Therefore, the DMA initialization needs to be performed before the pulse program loop.

3.4.6 Transmitter

DAC Driver

The DAC driver refers to the FPGA logic, for example, the DDS, to feed the DAC with the digital waveform to be converted. In this application, the DAC driver is expected to drive the dual-channel DAC. In order to generate two independent outputs that are frequency and phase configurable, two individual DDS cores configured by the processor via the AXI GPIO

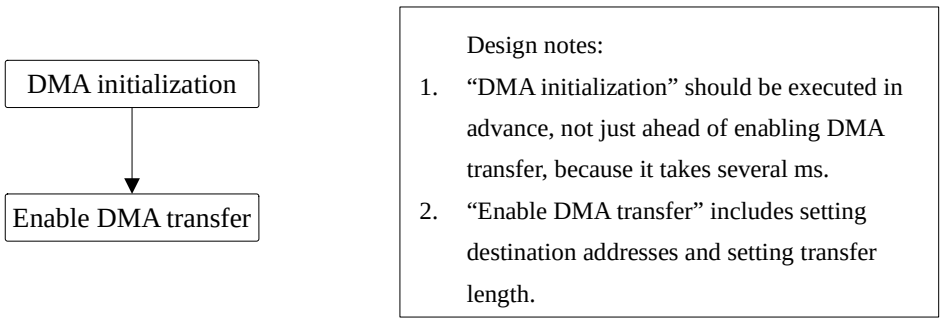


Fig. 3.49 Workflow of the DMA transfer configuration.

are employed. Figure 3.50 shows the diagram of the DDS driver, and Table 3.21 lists the adopted logic cores with description and configuration information. The AXI GPIO was configured to use dual 32-bit outputs, and each of the 32-bit outputs is split into two 16-bit for configuring the frequency and phase separately for the two output channels.

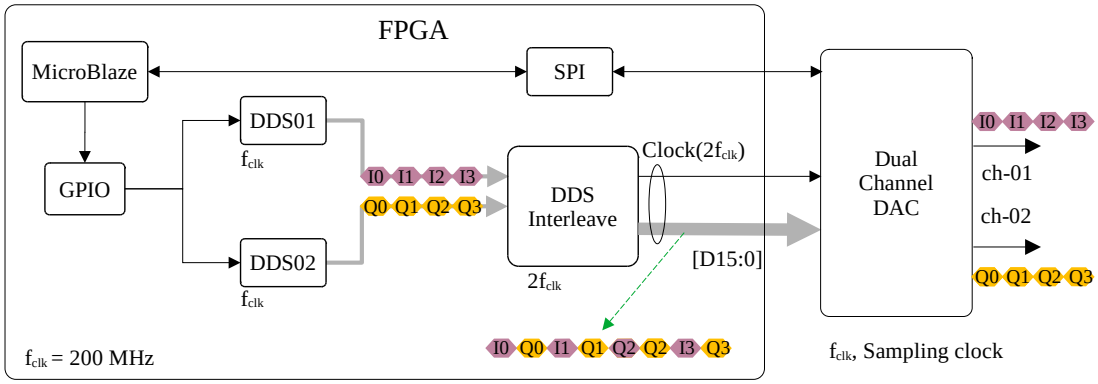


Fig. 3.50 The FPGA design diagram for driving the dual-channel DAC.

Recall that in Figure 3.15 and Figure 3.16, the 16-bit data for feeding the DAC needs to be interleaved. Therefore, a DDS interleave custom core was developed to fulfill the input requirement of the DAC device.

In the DDS interleave core, the two digital waveform data are interleaved and sent out with a frequency of 400 MHz, which is twice the frequency as the DDS core. In addition,

Table 3.21 IP cores description in DACs FPGA driver.

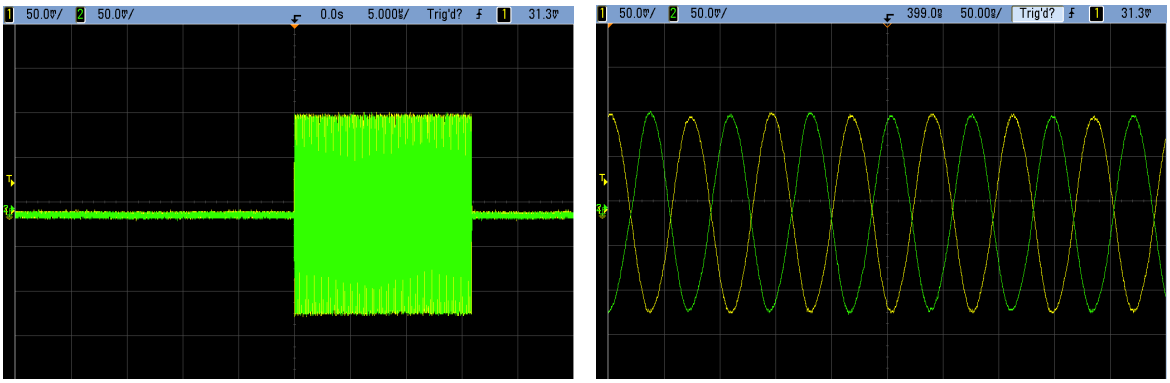
IP Core	Configurations and functions
MicroBlaze	Soft core processor. To configure the phases and frequencies of the DDS cores through AXI GPIO.
AXI Interface	To connect the AXI GPIO with the MicroBlaze through the AXI bus.
AXI GPIO	General Purpose Input and Output. It is configured as dual 32-bit outputs to feed DDS with high 16-bit for phase configuration and low 16-bit for frequency configuration.
DDS	Direct Digital Synthesizer. Two DDS cores are used for generating individual digital waveforms. The phase and frequency of the two DDS IP cores are configured as programmable.
DDS Interleave	This custom IP core interleaves the outputs of the two DDS cores. The single-end 16-bit data and the aligning clock are transferred to differential signals sent to a dual-channel DAC to produce analog signals.

another differential pair of clocks (DCIP/N) are created to align with the output data to improve the reliability of the data port interface, as shown in Figure 3.16.

Figure 3.15 shows the digital data port diagram. The parallel data (D15:0) are clocked by the DSS clock, which is a delayed version of the main DAC clock (CLKP/N). The parallel multiplexer data moves to the Retiming and Demmux blocks, at which the pre-designed individual digital data goes to the I or Q channel separately to create individual waveforms. However, the rising and falling edges of the DSS clock, which samples the data, as shown in Figure 3.16, are not at the optimal positions. The phase of the DSS clock should be shifted by configuring the parameters of SMP, SET, and HLD to make the multiplexer data more reliable, especially at a higher sampling frequency.

Transmitter Verification

Transmitter verification includes verifying the DAC driver, DAC chip, and the related front-end circuits. An oscilloscope was used to measure the outputs from the transmitter, which is



(a) Pulses output from the two DAC channels. (b) Pulse view in a smaller timescale.

Fig. 3.51 Outputs from transmitter on the transceiver.

shown in Figure 3.51. In this test, the frequencies of the two waveforms are designed to be the same, but the phases are set with 180 degrees shift.

At this stage, only one single channel is needed. Two-channel outputs with configurable frequency and phase are designed to drive the TRASE [37] coil array.

3.4.7 SPI Configurations

Because both the ADC and DAC support SPI bus for their configuration, an AXI Quad SPI logic core is employed to configure the two devices. Figure 3.52 details the FPGA design diagram, and Table 3.22 features the logic core used in the SPI configuration design. In the design, the ADC and DAC share the SPI bus, including clock and SDI/SDO signals. The 2-bit slave bus is split into two individual signals by a slice logic core for the two devices separately. When one device is to be configured, the associated bus is asserted to a low state to activate the configuration.

3.4.8 Pulse Program Generator

A pulse program generator is a core functional block for managing an NMR/MRI experiment, i.e., controlling the transmitter output and the acquisition in a stringent timeline. More specifically, a pulse program generator combines digital and analog circuits to produce pulses with defined parameters, such as frequency, pulse width, and repetition time for the

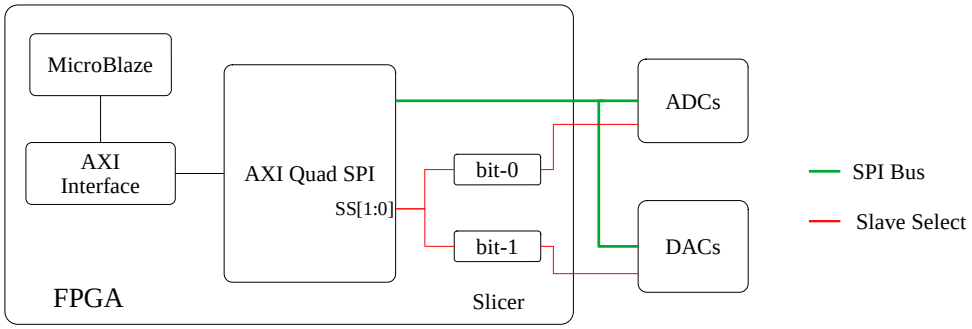


Fig. 3.52 Diagram of SPI configuration. Xilinx logic cores, MicroBlaze [68], AXI interface [75], AXI Quad SPI [82], and Slice are used.

Table 3.22 IP cores description in the SPI configuration.

IP Core	**Configurations and functions**
MicroBlaze	Soft core processor. To configure the AXI Quad SPI via the AXI bus.
AXI Interface	To connect AXI GPIO with MicroBlaze through the AXI bus.
AXI Quad SPI	It is configured in standard SPI mode, is a full-duplex synchronous channel that supports a four-wire interface (receive, transmit, clock, and slave-select) between a master and a selected slave.

experiments. The digital circuit of the pulse generator is designed and implemented inside the FPGA.

Recall section 2.3.1 which reviewed the section Transmitter that the combination of the DAC driver and DAC can generate the continuous waveform. And the data movement is also able to move data continuously. In order to apply these functions properly in the NMR/MRI experiments, which need strict time management, the continuous operation must be broken into desired sections. For example, in an FID experiment, a certain width of the pulse is needed for excitation and during which the acquisition should not be enabled until the end of the pulse. Therefore, a time management subsystem is needed to tackle the time issue.

Time Management Subsystem

Time management can be performed by software, i.e., programming the soft processor to perform time control. However, that method cannot achieve a high-resolution time because the time of executing individual functions for the processor is uncertain. In this application, a hardware solution is adopted for realizing accurate time management, as shown in Figure 3.53. The employed logic cores are listed in Table 3.23 with descriptions.

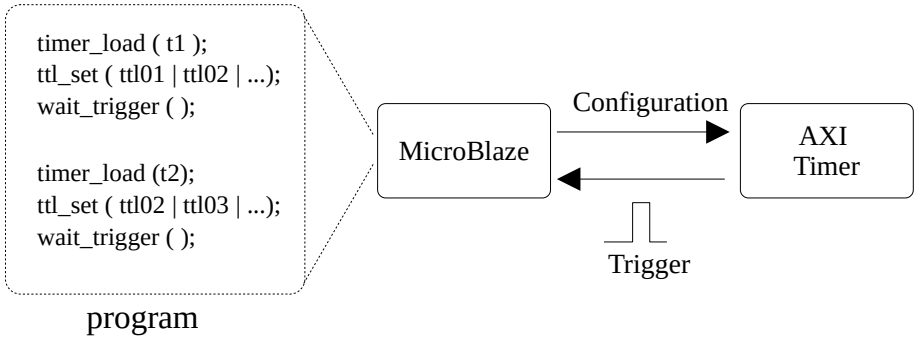


Fig. 3.53 Time management diagram in the FPGA.

Table 3.23 IP cores used in time management.

IP Core	Configurations and functions
MicroBlaze	It is used as a softcore processor. In the core configuration, at least one stream interface (link) must be added. In addition, the additional stream instructions must be enabled.
AXI Timer	It is configured as the generate mode under which the counter inside the timer counts down or up based on the value in the load register which is manipulated by a soft processor. Once the timeout value is reached, the Timer generates a single pulse for one clock cycle. The counter will then start counting based on another value from the load register. The resolution of the generated pulse depends on the clock frequency. In this application, the clock frequency is 100 MHz, so the related resolution is 10 ns.

In the configuration of the MicroBlaze core, one stream link interface is enabled, and the additional stream instructions are enabled, as shown in Figure 3.54. The additional

stream instruction allows users to provide functionalities, such as dynamically accessing the instructions GETD and PUTD with a low latency interface to the processor pipeline. That is ideal for extending the processor execution unit with custom hardware accelerators.

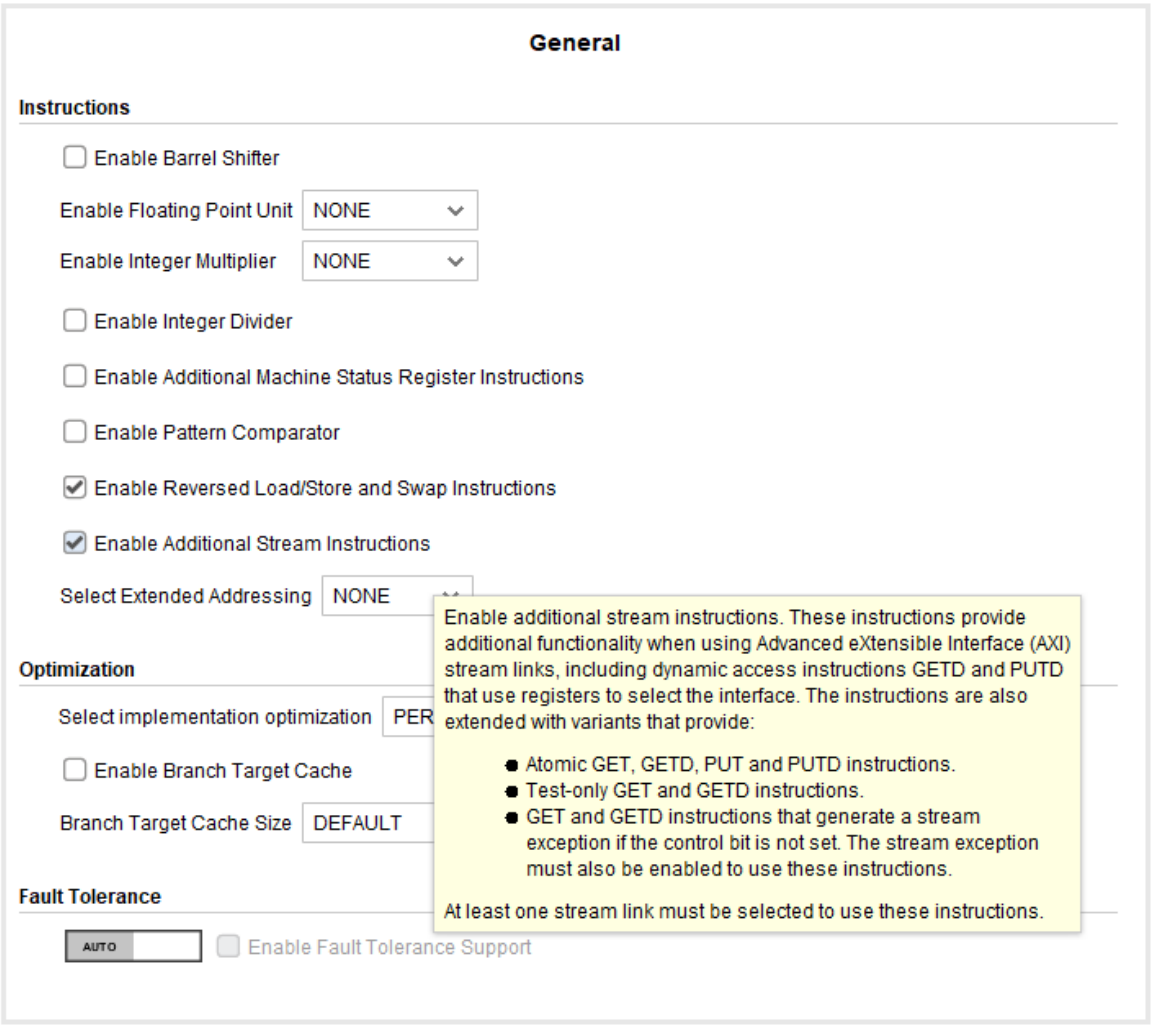
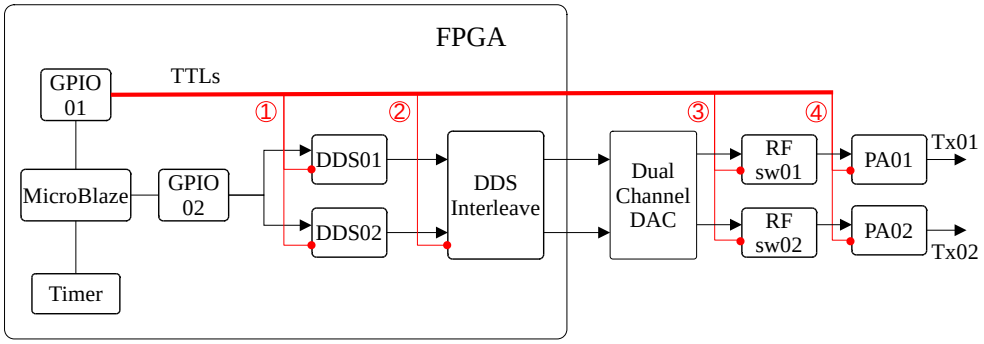
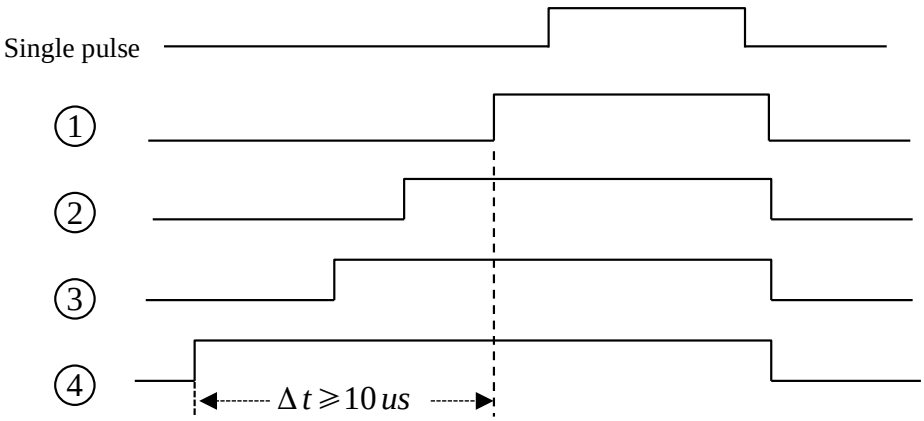


Fig. 3.54 MicroBlaze configuration.

The AXI Timer cooperating with the MicroBlaze is a hardware acceleration solution for achieving accurate time management. The processor, MicroBlaze, configures AXI Timer [89] by loading a value, and the Timer starts to count. Before waiting for the response from the Timer, the processor will set the TTLs for enabling or disabling the internal logic cores or external devices. Once the time expires, the Timer generates a pulse to inform the processor,



(a) Diagram of the pulse program generator.



(b) Timelines for the TTLs.

Fig. 3.55 Pulse program generator and the TTLs.

and it loads another value to the Timer. The loaded values determine the length of the TTLs, which eventually manages the time in the NMR/MRI experiments.

Pulse Program Testing

Figure 3.55a shows the diagram of the pulse program generator, and Figure 3.55b shows the timelines of the TTLs for controlling the related logic or devices. The power amplifiers, RF switches, DDS interleave, and DDS cores are enabled in sequence. However, the power amplifiers should be enabled at least 10 μs before enabling the DDS cores because the power amplifiers need several microseconds to warm up.

3.4.9 Clock Generators Configuration

This subsection details how the clock generators are configured and how the output clocks are verified. Figure 3.20 shows the diagram of the clock tree. The default setting of Si5338 adopts the local oscillator as the input clock source. To achieve the clock synchronization on the transceiver board, the Si5338 is configured to use the alternative input other than the default one. The alternative input is the CLKIN2, as shown in Figure 3.7.

Configuration Strategies

Recall that the clock generators are not able to output clocks during configuration. Therefore, an internal clock in the FPGA is implemented to provide the clock to drive the logic of configuring the clock generators. Configuring the clock generators, essentially, is to write the values to the registers on the clock generators. The values created by the clock builder [71] consist of several hundred lines of data formatted as a C file. Using VHDL to build a module to configure the clock generators is possible, but it is complex and inflexible for changing configuration. Another way to realize clock configuration is to use a microprocessor, MicroBlaze. It is easier to build the C-based application to put the C-type file of register values to the clock generators. The MicroBlaze MCS (Microcontroller system) [90] was chosen instead of the general MicroBlaze because the MicroBlaze MCS uses fewer resources. Figure 3.56 shows the block design for clock configuration.

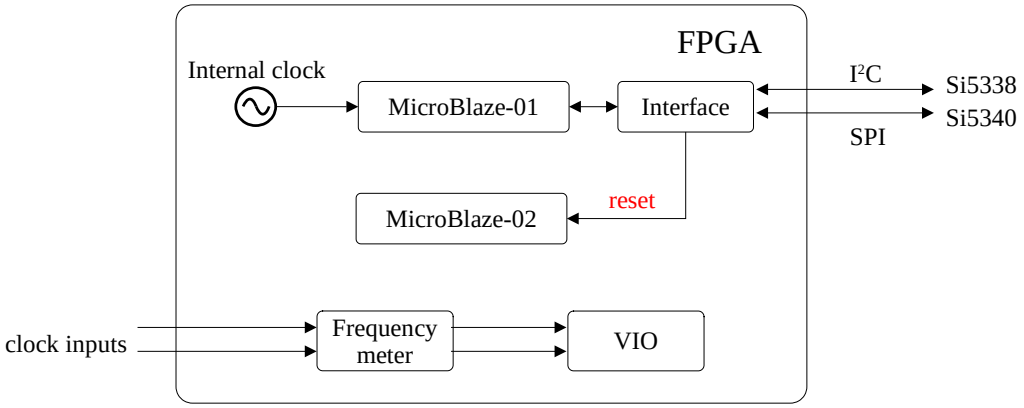


Fig. 3.56 Clock configuration diagram.

Table 3.24 IP cores description in FPGA design for clock generators configuration.

IP Core	Configuration and functions
MicroBlaze MCS	MicroController System is a standalone processor system intended for controller applications. It contains the MicroBlaze with a fixed configuration, optimized for minimal area.
Interface	A custom logic core for transferring the GPIO signal to special buses, such as IIC or SPI, for configuring the related clock generators.
Frequency meter	A custom logic core to measure the frequency created by the clock generators.
VIO	Virtual Input or Output, a debugging tool in Vivado software. To show the measured frequency results from the frequency meter.

Verification of the Clock Tree

The verification of the clock generators is conducted by an oscilloscope and frequency meter module in the FPGA. The output results from the clock generators are identical as planned.

3.5 Transceiver Board Testing

The main issue covered in this section is the verification of the transceiver board, i.e., synchronization of the transmitting and receiving.

3.5.1 Synchronization Testing

For signal averaging, for example, averaging eight scans of FID signal for higher signal-noise ratio, the phase of the signal must be strictly aligned. That is related to two things: one is that the initial phase of the transmitting pulses should be identical each time; the other is that the start time of acquisition should be constant during each scan.

Figure 3.57 shows the loop test: the receiver acquires the signal, which is the pulse during transmitting. And the transmitting or acquisition is repeated four times with a constant time interval.

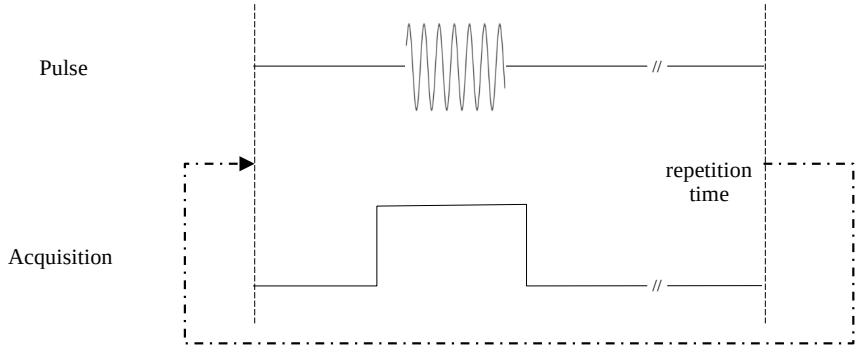


Fig. 3.57 Diagram for synchronization testing, the pulse was acquired by the receiver during transmitting.

Figure 3.58 shows the diagram of the FPGA firmware and the related devices for transmitting or receiving. Still, the TTL signals controlled by the time management unit manage the internal logic and external devices. The timeline for the TTL signals is illustrated in Figure 3.59. Number 1, 2, 3 TTL signals are managed for generating a single pulse, and number 4 TTL signal is a switch for acquiring the data. In this experiment, that switch is turned on in advance of the pulse. Number 5 TTL signal kick-off/stop the DMA transfer and with a short pulse separately. The single pulse is repeated at a certain repetition time. In practical NMR/MRI experiments, the number 4 and 5 TTL pulses are shifted to acquire the real signals after the $\pi/2$ or π pulses.

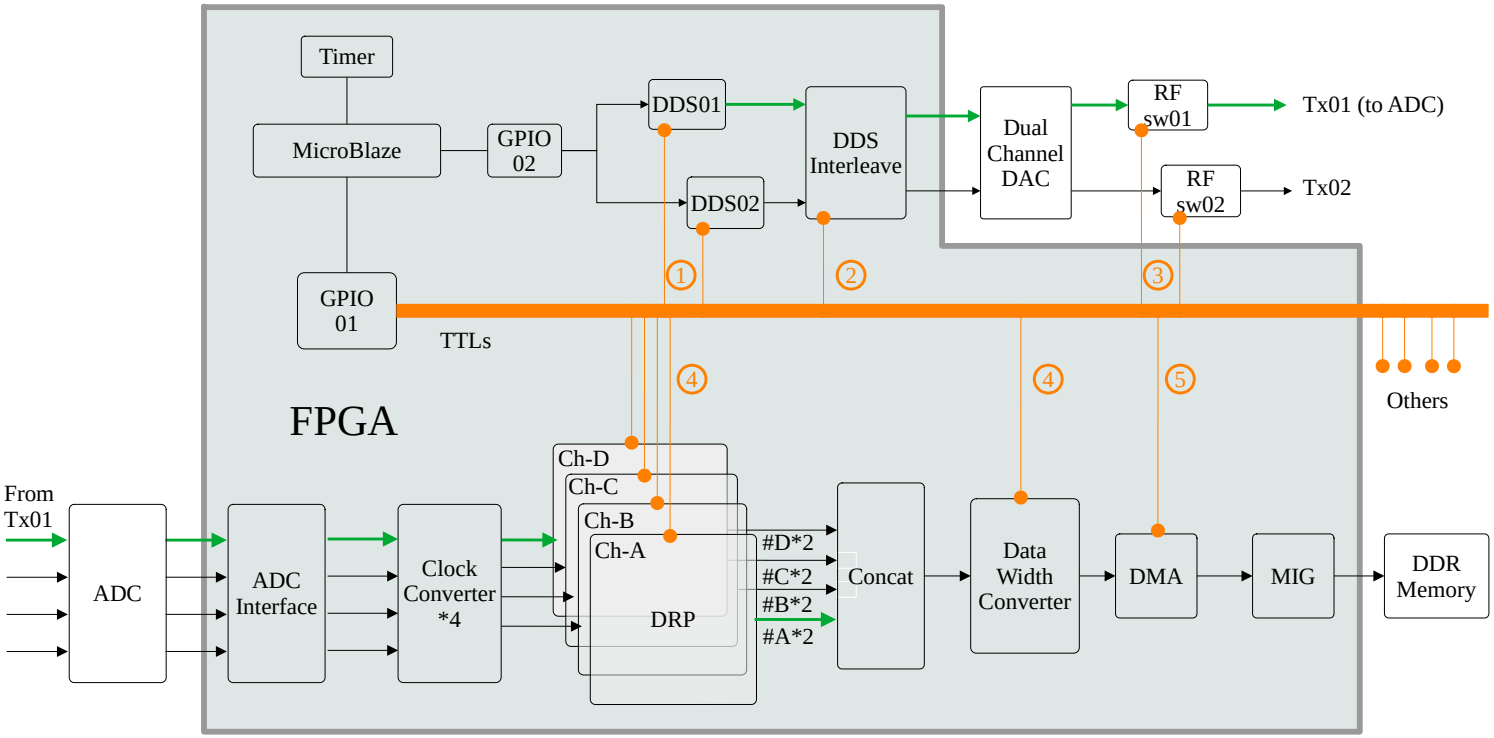


Fig. 3.58 Overview of the synchronization testing.

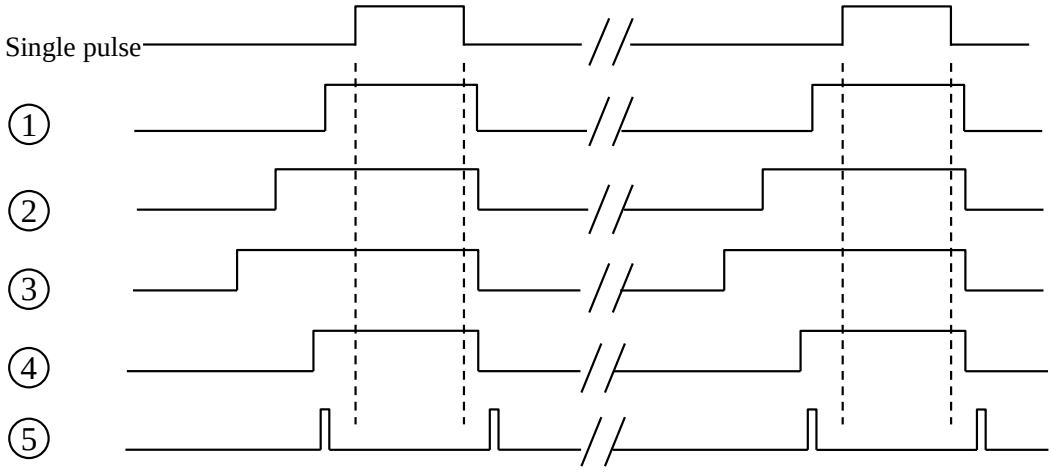


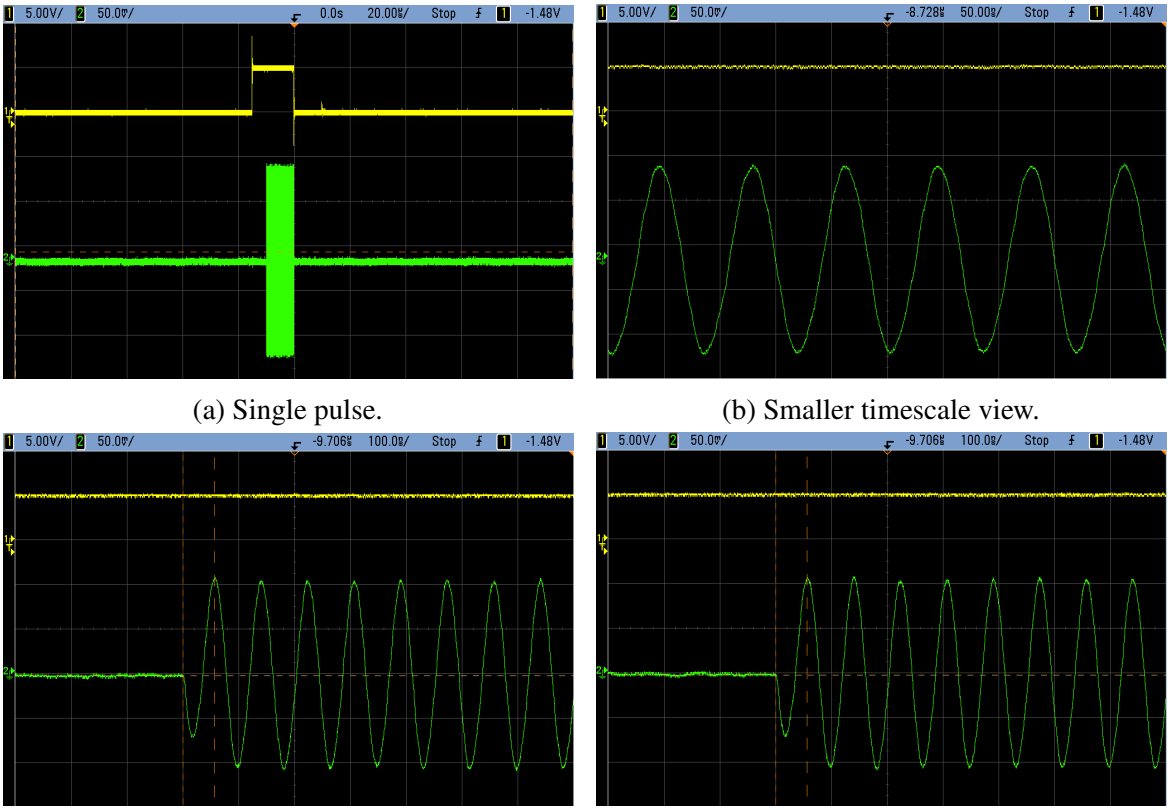
Fig. 3.59 TTL signals.

Figure 3.60 shows the single pulse from the transmitter captured with an oscilloscope. In Figure 3.60a, the yellow channel shows the TTL signal for the power amplifier, and the green channel shows the single pulse in a larger time scale. Figure 3.60b shows the smaller time scale of the pulse. Figure 3.60c and Figure 3.60d are the two separated pulses, and it can be seen that the initial phases are identical.

In this experiment, as shown in Figure 3.58, it is a loop test, i.e., the generated single pulses are digitized by the receiver, then processed in the digital receiver processor, and eventually, the processed data are moved to DDR memory. Figure 3.60c and Figure 3.60d illustrate the synchronization of the transmitting part.

The synchronization of the receiver chain also needs to be verified. In the practical NMR/MRI FPGA firmware design, the DDS used in the transmitter subsystem (Figure 3.55a) and the DDS used in a digital receiver processor (Figure 3.44) are the same DDS. To see a waveform output rather than a DC output from the digital receiver processor, separated DDS cores with a certain frequency offset are used in the transmitter subsystem and the digital receiver processor.

In this experiment, the transmitter generated four single pulses with a certain repetition time, then the ADC digitized the pulse, and the digital receiver processor processed the data. The processed data was eventually sent to the DDR memory. Figure 3.61 shows the plotted graph with the processed data from the DDR memory. The first four rows of the graph show



(a) Single pulse.

(b) Smaller timescale view.

(c) Initial phase view of one pulse.

(d) Initial phase view of another pulse.

Fig. 3.60 The single pulse measured with an oscilloscope.

the four processed data from the four single pulses, and the last row shows the overlap of the above four rows. This loop experiment illustrates that the transmitter and the receiver chain, including the ADC and the digital receiver processor, are synchronized. In this experiment, no observable jitter could be found.

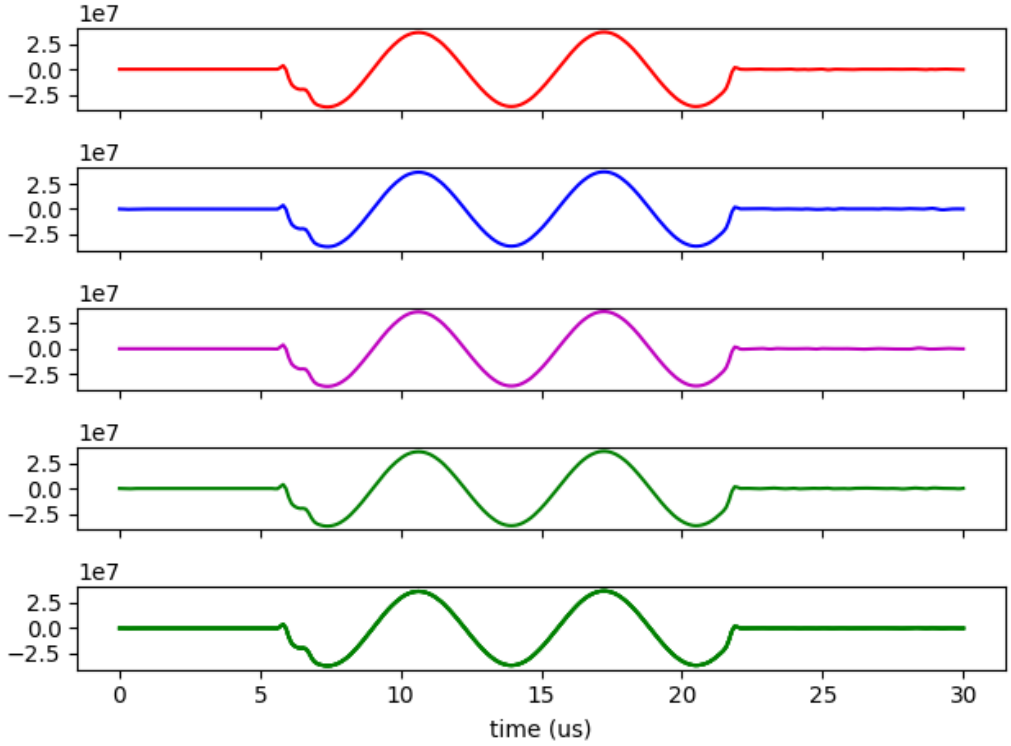


Fig. 3.61 Captured signal from the synchronization testing.

3.5.2 Quantification of the Transceiver Board

The specifications of the MRI spectrometer are provided in this section. Agilent E4411B was used as a signal generator to provide the signals with different amplitudes and frequencies. One of the input receive channels was used for analyzing the specifications. After the signal was digitized by the ADC, the digital data was processed via the digital receiver processor as shown in Figure 3.44. Because the sampling frequency of the ADC is 100 MHz and the decimation rate of the CIC compiler is 10:1, the bandwidth of each measurement is 10 MHz. Therefore, multiple measurements were conducted to reveal the receive bandwidth of the

spectrometer. The frequency of the DDS in the digital receiver processor can be adjusted to match the input frequency.

Figure 3.62 shows the acquired noise and spectrum when the input channel is terminated with a 50 ohm terminator. Figure 3.63 shows the amplitude of the noise.

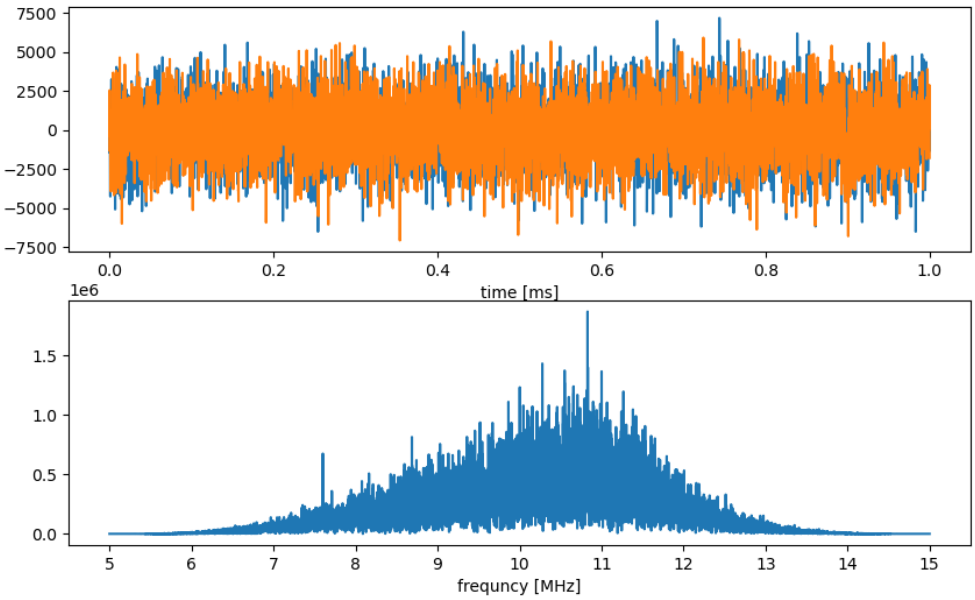


Fig. 3.62 Noise scan.

When the receive channel was fed by the sine wave signal with a certain frequency, Figure 3.64 shows the signal and the spectrum. The digital signal was windowed by the Hanning window. Specifically, the input frequency of the sin wave is 10.022 MHz and the mixed frequency is 10 MHz. The spectrum shows the input frequency and the amplitude.

According to the AD9653 datasheet, the maximum input voltage is 2 Vpp (10 dBm with 50-Ohm system). Figure 3.65 shows the maximum amplitude when the input receiver was fed with the maximum input power. The experiment was conducted when the center frequency of the input signal was 10 MHz.

After calibrating the maximum amplitude (full scale), the noise level can be figured out. As shown in Figure 3.63, the noise level is about -110 dB when compared to the full-power signal.

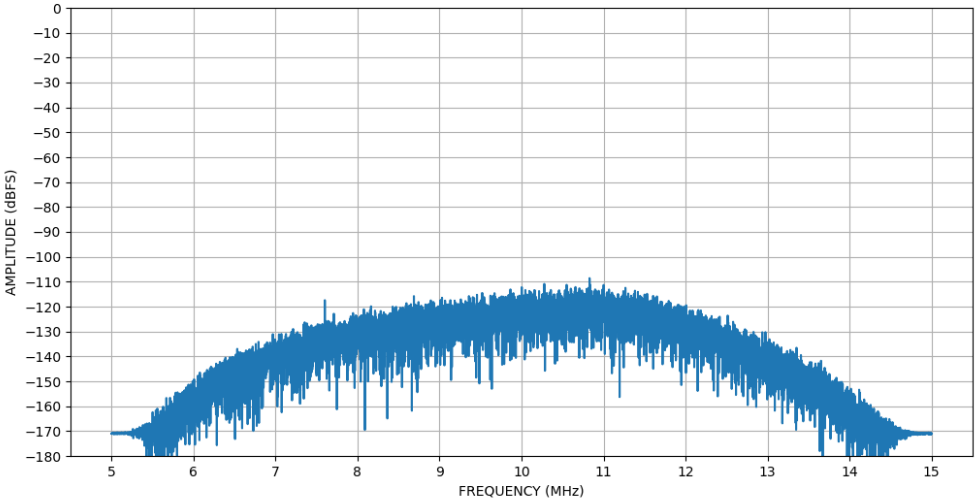


Fig. 3.63 Noise amplitude.

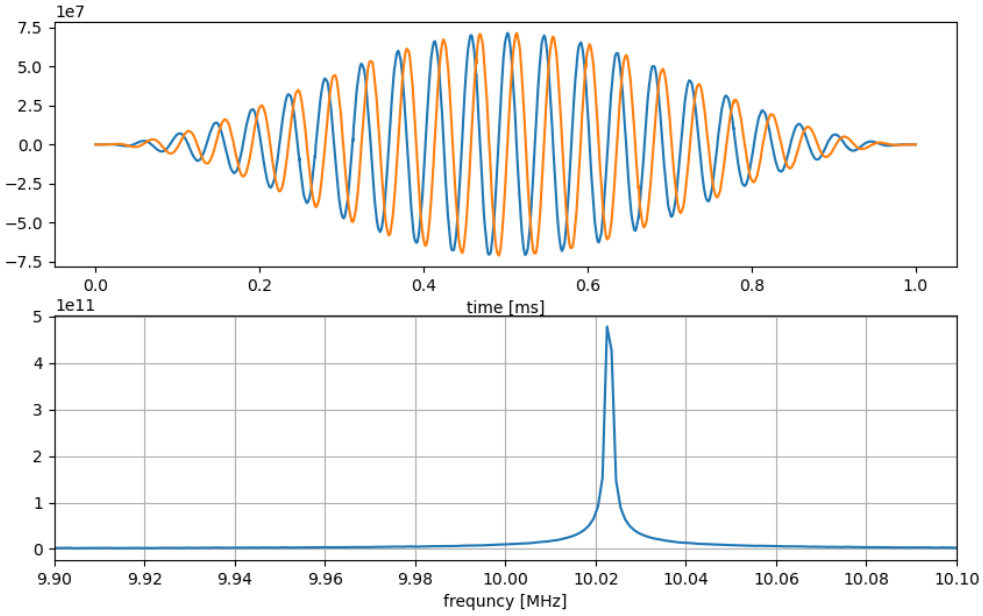


Fig. 3.64 Frequency Response.

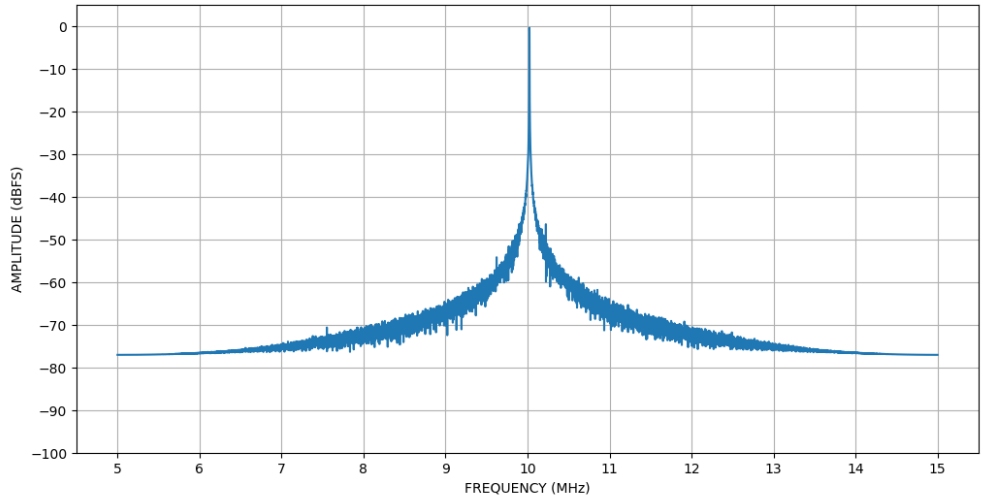


Fig. 3.65 Maximum input power.

Figure 3.66 shows the linearity of the spectrometer when the center frequency is at 10 MHz.

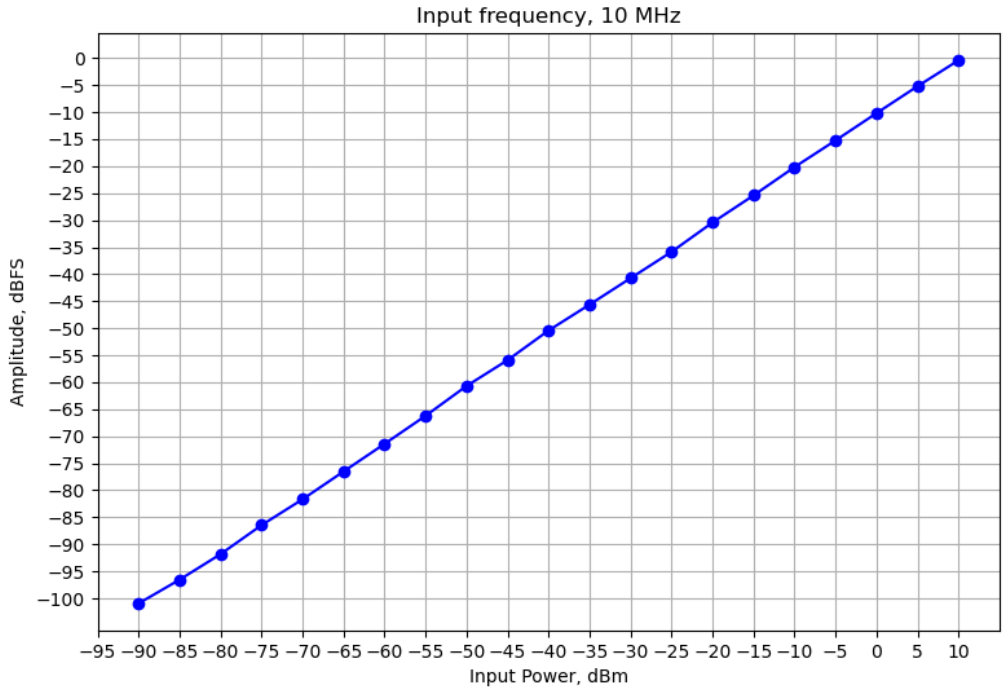


Fig. 3.66 Linearity test.

Figure 3.67 shows the frequency response centered at 12 MHz. The frequency of the output of the DDS in the digital receiver processor was 12 MHz and the input signal was adjusted from 7.5 MHz to 12.5 MHz. The frequency response is well matched with the magnitude response of the CIC compiler output.

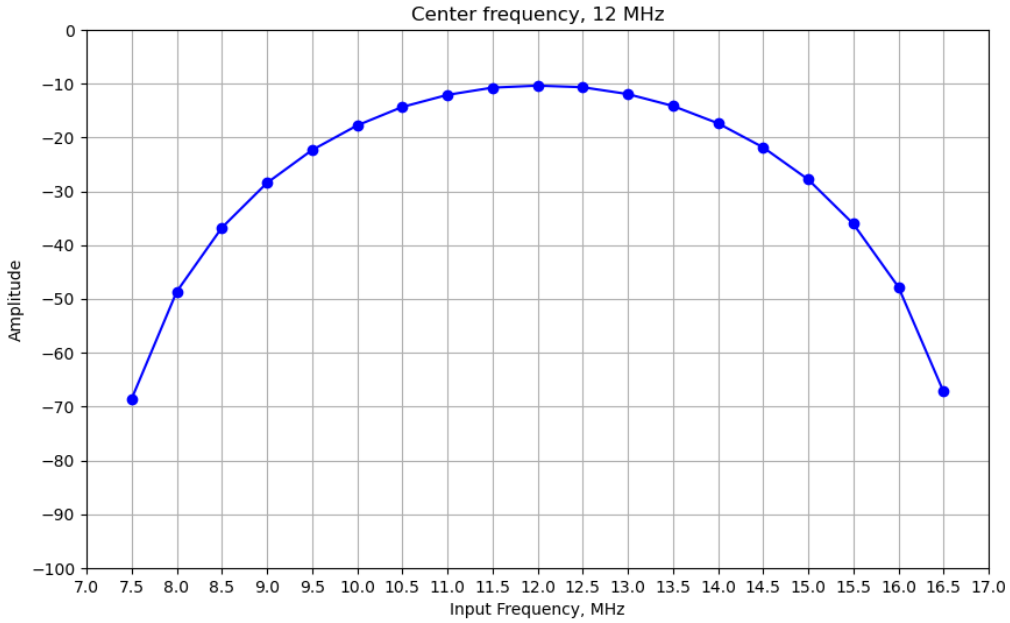


Fig. 3.67 Frequency Response.

Figure 3.68, Figure 3.69, and Figure 3.70 show the frequency and amplitude responses when the input was fed by different amplitude of power and frequencies.

Table 3.25 shows the comparison with other spectrometers.

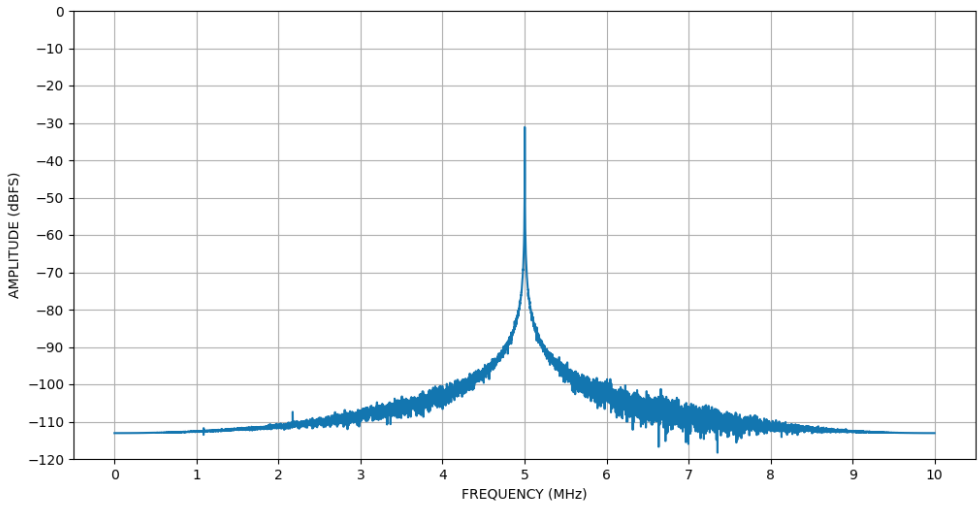


Fig. 3.68 -20 dBm input signal at 5 MHz.

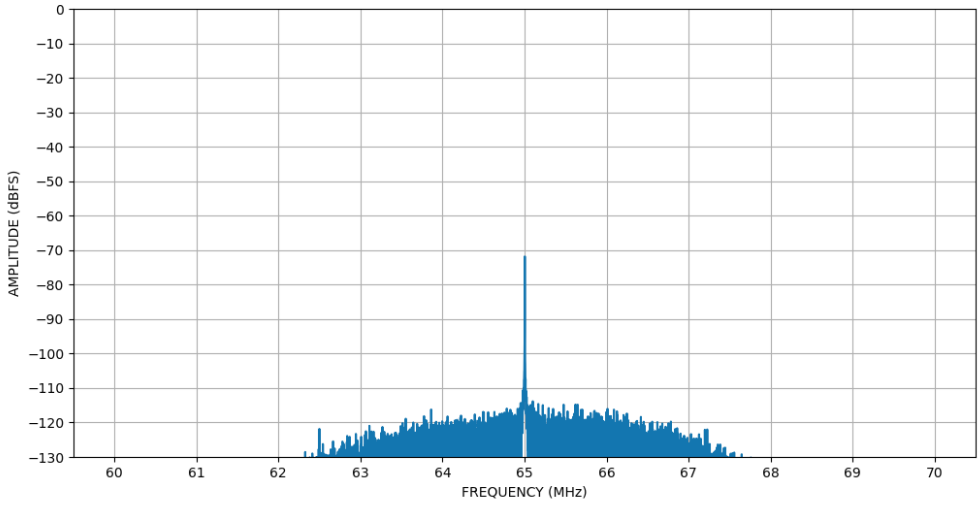


Fig. 3.69 -60 dBm input signal at 65 MHz.

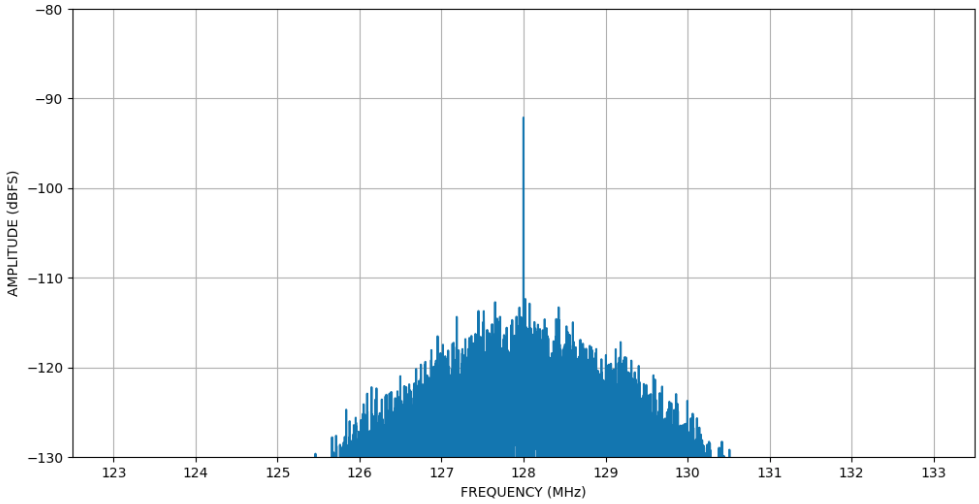


Fig. 3.70 -80 dBm input signal at 128 MHz.

Table 3.25 Comparison with other spectrometers.

Name	**Tx-ch**	**Rx-ch**	**Frequency (MHz)**	**Notes**
Medusa	2	4	0-100	–
OCRA	2	1	0-40	–
OPENCORE	3	1	0-400	–
MicroSpec	1	1	–	commercial
RS^2D	3	up to 16	1-900	commercial
PXIe Spectrometer	2	4	0-200	–

Chapter 4

Spectrometer Integration and Testing

Chapter 3 discusses the implementation of the key components on the transceiver board and the FPGA firmware on the FPGA module. The processed MR data can be stored on the DDR memory device on the transceiver board, and it needs to be shown on the user interface. In addition, to manage the experiment more flexibly regarding different pulse programs or variable parameters, such as pulse width, repetition time, etc., the spectrometer needs to be integrated. The integrated spectrometer means that the NMR/MRI experiments can be performed in the user interface rather than adopting the debug cable used for developing the FPGA firmware and embedded applications. The chapter then moves on to how the spectrometer is integrated with the system controller on the PXIe chassis. The main issues addressed in this chapter proceeds as follows:

- Overview of the system controller.
- PCIe-DMA Engine on the transceiver.
- API and application development.
- Integration of the spectrometer.
- NMR testing.

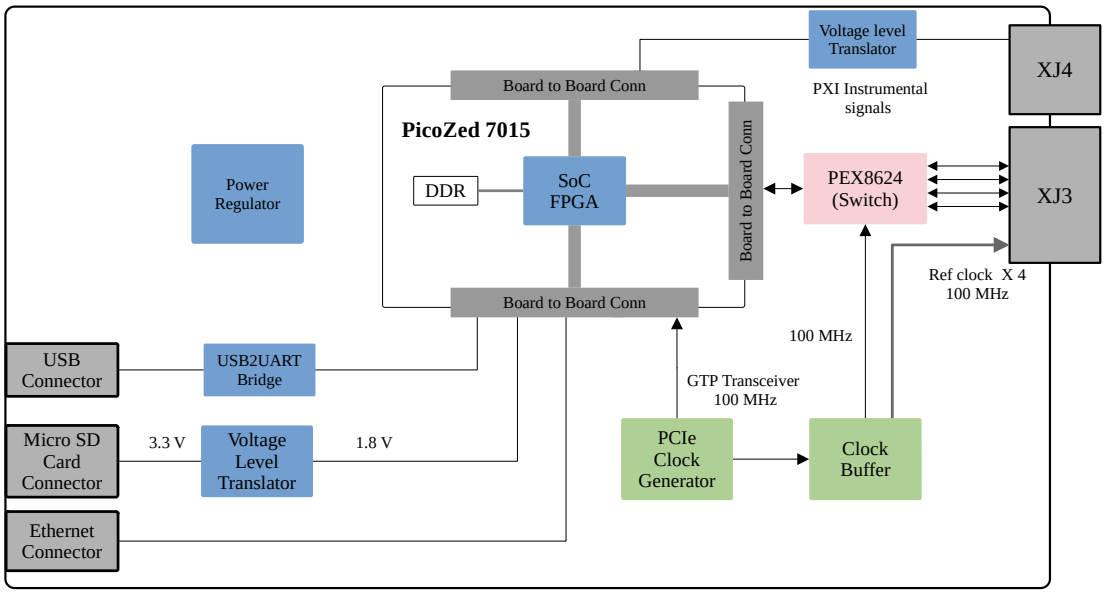


Fig. 4.1 Overview of the system controller.

4.1 System Controller Review

On a PXIe chassis, a system controller mounted in slot 1 is used for managing the peripheral boards. The system controller used in this study was designed by former students [11, 12]. The former work consists of a PCB board and pre-configured operating system with a device driver and API to support multiple peripheral boards.

4.1.1 Hardware

The system controller comes with standard features such as an integrated CPU (ARM), RAM, Ethernet, serial USB, and other peripheral I/O. Figure 4.1 is the big picture of the system controller. A ZYNQ SoC FPGA module, PicoZed7015, was adopted to simplify and accelerate the development. Multiple support components were implemented on the carrier board to realize its functions. Table 4.1 shows the main components and their functions.

As shown in Figure 2.22, the ZYNQ FPGA module was mounted on the top of the carrier board. Figure 4.2 shows the block diagram of the main resources on the ZYNQ FPGA module. ZYNQ SoC consists of Arm Cortex-A9 cores, many hard intellectual property components (IPs), and programmable logic (PL). This offering can be used in two ways:

Table 4.1 Key components on the system controller.

Units	Description
FPGA module	The system-on-chip FPGA module provides the hardcore processor and programming logic. The hardcore processor is used for installing the operating system, PetaLinux. The programming logic is used for implementing the PCIe root-complex for data transfer over PCIe protocol.
Clock system	It supplies the clocks for the other devices. The PCIe clock generator generates two clock outputs. One feeds the GTP transceiver on the FPGA module directly. Another feeds a clock buffer that clocks the PCIe switch and potential PCIe devices on the peripheral modules.
PCIe switch	This switch allows up to 4 PCIe Endpoint devices to be connected to the PCIe root-complex on the system controller.
SD card	The booting files stored in the SD card can be loaded to boot the operating system on the ARM hardcore processor.
USB	It allows users to access the operating system via the serial USB protocol.

The ZYNQ SoC PS can be used in a standalone mode without attaching any additional fabric IP. Logic design can be instantiated in fabric and attached to the ZYNQ PS as a PS+PL combination. A PCIe root complex can be implemented using a high-speed gigabit transceiver and an IP block(AXI Memory Mapped to PCIe) in the FPGA fabric.

PCIe Connectivity

As we desire to support multiple peripheral boards, a PCIe switch was implemented to provide dedicated lanes to each peripheral board. Figure 4.3 shows how the PCIe root-complex and the PCIe Endpoint are connected through the PCIe switch, PEX8624 (Broadcom Inc.).

Clock System

The system controller has its internal reference clock independent of the PXIe chassis clock. Figure 4.4 illustrates the clock system on the system controller.

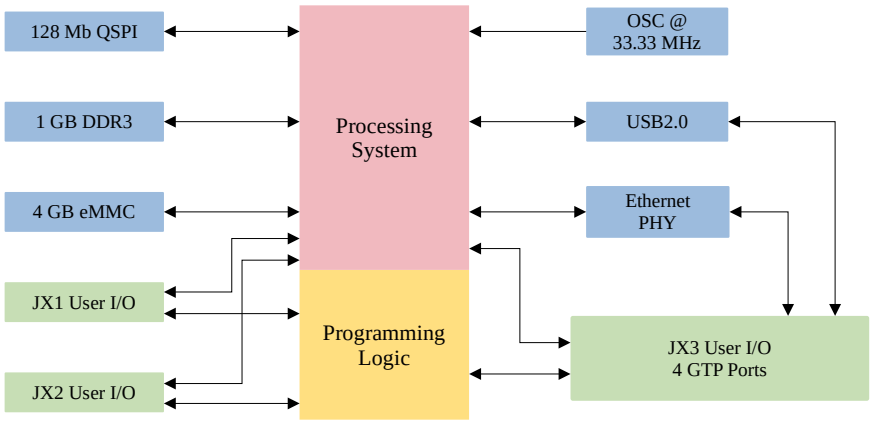


Fig. 4.2 Block diagram of PicoZed module [91].

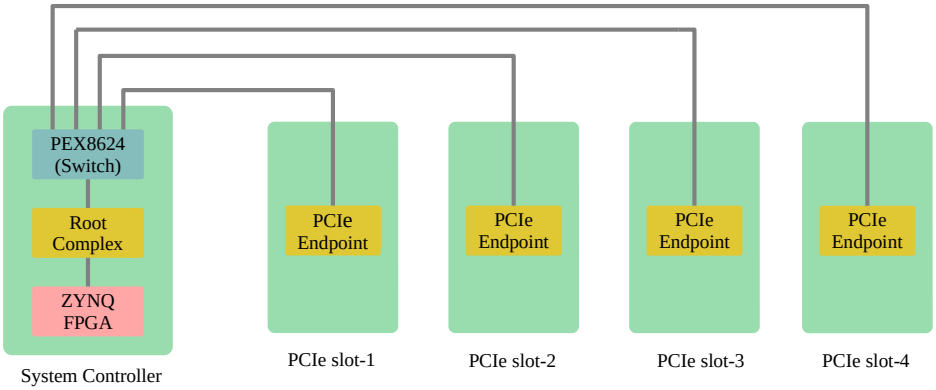


Fig. 4.3 PCIe Connectivity with PCIe switch [92].

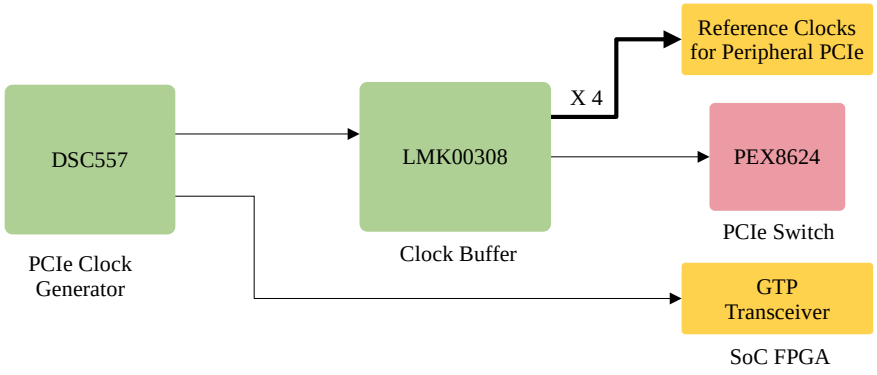


Fig. 4.4 Clock system on system controller.

4.1.2 FPGA Firmware Design in Vivado

The Vivado Design Suite [93] is used for configuring the ZYNQ core for the programming system and implementing FPGA firmware in the programming logic, as shown in Figure 4.5. Table 4.2 shows the main adopted logic cores and their particular configurations. After completing the design and configuring the logic cores, Vivado synthesis, implements the configured logic and then generates a hardware file. This file is used for creating the PetaLinux operating system. Figure 4.6 shows the overview of the ZYNQ FPGA design.

Table 4.2 IP cores description in FPGA design for the system controller.

Logic cores	Description
ZYNQ PS	• Enables GP0 AXI Master Interface. • Enables HP0 AXI Slave Interface, 64-bit width. • Enables Ethernet0. • Enables USB0. • Enables SD0. • Enables UART0.
AXI PCIe	The AXI PCIe logic core device type is configured as a root-port of PCI Express Root-Complex.
AXI Interconnect GP	General-purpose AXI interconnect. In this design, this logic core connects the other logic cores in programmable logic with the hardcore processor unit via the AXI GP port, as shown in Figure 4.5.
AXI Interconnect HP	High-performance AXI interconnect. High performance (HP) port is used for fast access to the PS-DDR memory. In this application, HP port transfers the high throughput data from AXI PCIe to DDR memory with a fast speed.

An essential logic core, AXI Memory Mapped to the PCI Express core, is adopted to transfer data over PCIe protocol. That core provides an interface that allows the mutual interaction between the user logic in the AXI4 domain and the PCIe. The core also supports

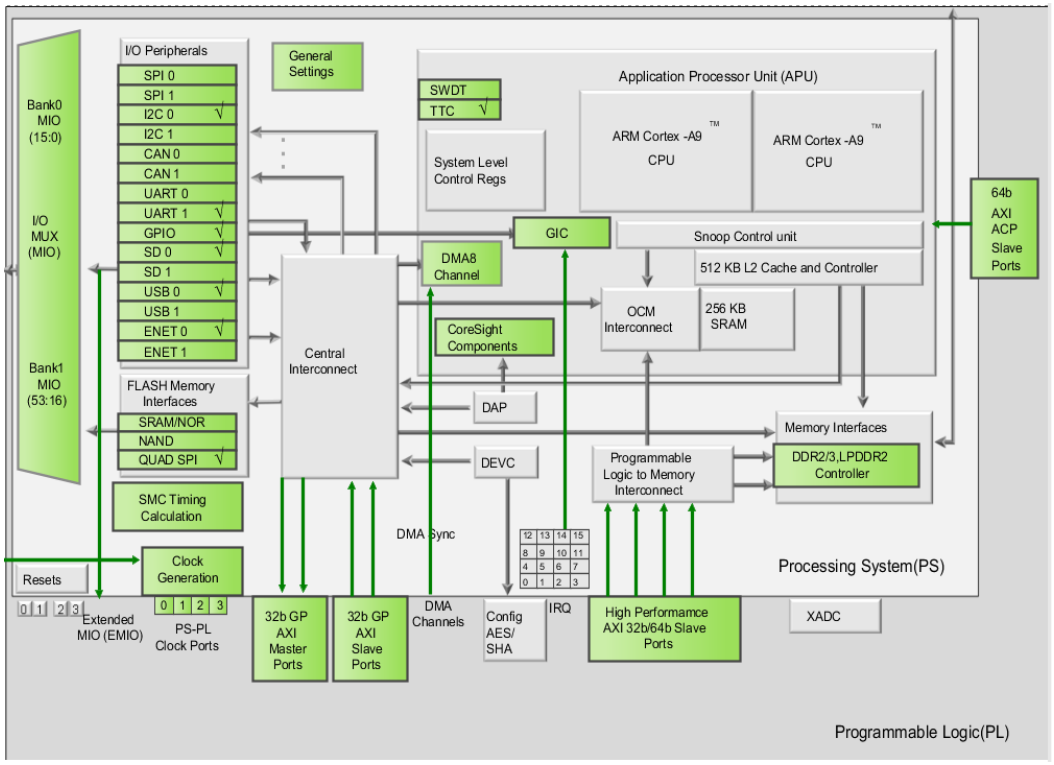


Fig. 4.5 Re-customized IP dialog box of ZYNQ.

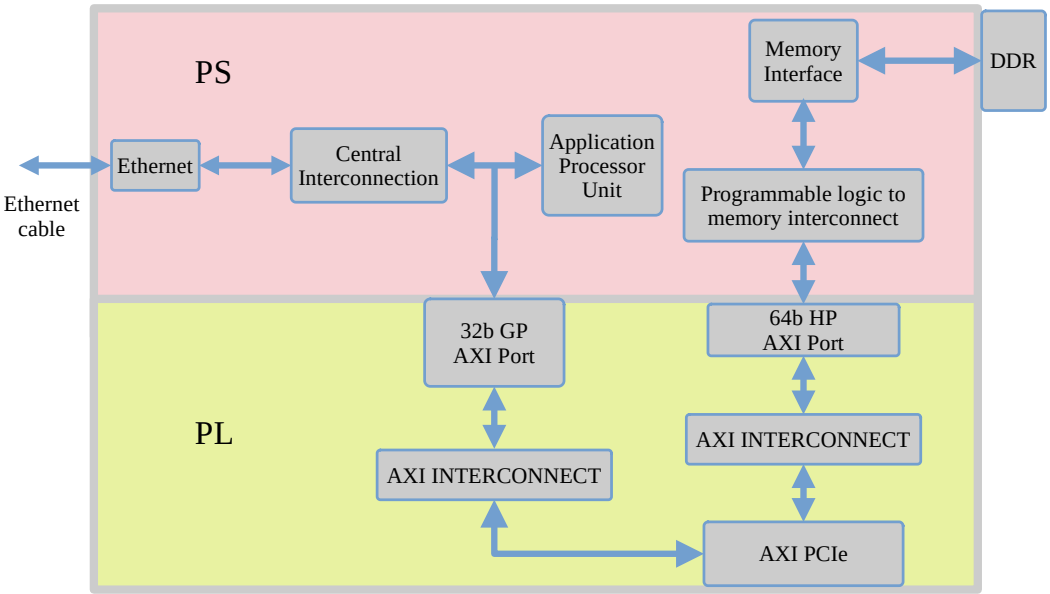


Fig. 4.6 Block diagram of ZYNQ design.

the translation level between the AXI4 embedded system to the PCIe system by translating the AXI4 memory read or write to Transaction Layer Packet(TLP) packets and translating PCIe memory read or write request TLP packets to AXI4 interface instructions.

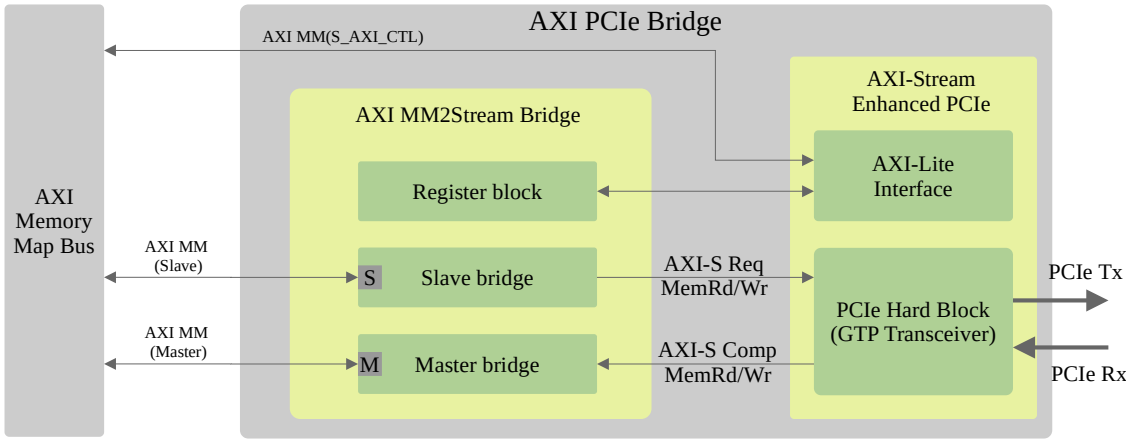


Fig. 4.7 Architecture of AXI memory mapped to PCI Express core [94].

As shown in Figure 4.7, the core contains two parts: the Memory Mapped AXI4 to AXI-4Stream Bridge and the AXI4-Stream Enhance Interface Block for PCIe. The memory-mapped AXI4 block is made up of the register block, slave bridge, and master bridge. The register block encompasses registers, such as status, control, interrupt, used for dynamically mapping the addresses in AXI4 memory mapped domain to addresses in PCIe domain. The slave bridge, which connects the AXI4 Interconnect, functions as a slave device to process the read or write requests by AXI4 masters. The master bridge, which connects to the AXI4 Interconnect, behaves as a master device to handle the read or write TLPs from the PCIe realm. The detailed functions of the two above bridges are described in Table 4.3.

The AXI-Stream Enhanced PCIe block contains AXI-Lite Interface used to interact with the Register block in memory-mapped AXI bridge and PCIe hardcore. In the Artix-7 FPGA on the transceiver board, the PCIe hardcore is the GTP transceiver to interact with the two bridges through Requester/Completer interfaces in the AXI-Stream Enhanced PCIe block.

Table 4.3 Main functions performed by AXI memory mapped to PCI Express.

Functions	Description
AXI-master-Wr	When the slave bridge receives a write transaction initiated by a remote AXI master, the write destination address and qualifiers are apprehended. The write data is then converted to MemWr TLPs, which are transferred to the integrated block for PCI Express(GTP Transceiver).
AXI-master-Rd	When the slave bridge receives a read transaction initiated by a remote AXI master, the read address and qualifiers are apprehended. A MemRd request TLP is sent to the core, and read data is returned to the AXI master.
PCIe-MemWr	When the PCIe block initiates a write request, an address and qualifiers created by the individual PCIe MEmRe request TLP header are used for the memory-mapped AXI4 bus. The related write data is then sent to the addressed-memory mapped AXI4 slave.
PCIe-MemRd	When the PCIe block initiates a read request, an address and qualifiers created by the individual PCIe MEmRd request TLP header are used for the memory-mapped AXI4 bus. The collected read data from AXI4 memory mapped AXI4 slave is converted to completion TLPs in the slave bridge. The TLPs are then passed to the integrated block for PCI Express(GTP transceiver).

4.1.3 PXIe Architecture

Figure 4.8 shows the architecture of the PXIe system with a system controller and peripheral board. The userspace contains the application and API (Application Programming Interface), which users can develop. The applications developed by the C/C++ programming language perform specific functions to copy data from the DDR memory on the transceiver board and temporarily store it in the operating system. The API refers to the higher level functions over the drivers. These functions can be called in the application programs. DMA/PIO and PCIe drivers are in the Kernel space of the operating system. Data transfer is performed by the PCIe logic core configured with Root-complex in the programmable logic in the system controller. The PCIe logic core is configured with the Endpoint in the peripheral module over

the PCIe link on the PXIe chassis. The PCIe switch permits more peripheral modules to be enabled for data transfer over the PCIe protocol. At this stage only one is used, but we intend to have multiple transceiver boards to support more channels.

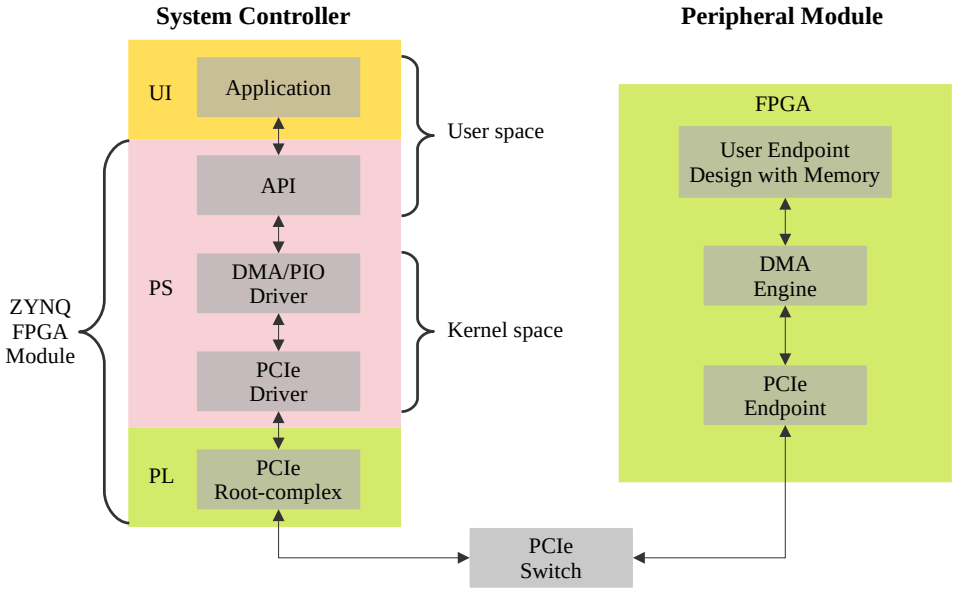


Fig. 4.8 PXIe architecture.

4.1.4 PetaLinux Build-up

Once the ZYNQ core is defined and compiled in Vivado, an operating system is to be built to support it. Figure 4.9 shows the flow chart of building the PetaLinux operating system for the ZYNQ FPGA module on the system controller. In Vivado Design Suite [93], the logic in the programming system unit and programming logic is configured and compiled, which results in generating the .hdf file. XSDK software [95] is used to develop the PCIe driver and the applications. With the PetaLinux Tool [96], .hdf and .c files are used to build the PetaLinux image files, BOOT.BIN and Image.ub. Table 4.4 shows the key configurations during building the PetaLinux operating system. Booting is performed by loading the image files to the hardcore when powering up.

There are multiple options to boot the PetaLinux image on hardware, such as using an SD card, JTAG, or TFTP (Trivial file transport protocol). Booting the PetaLinux image on

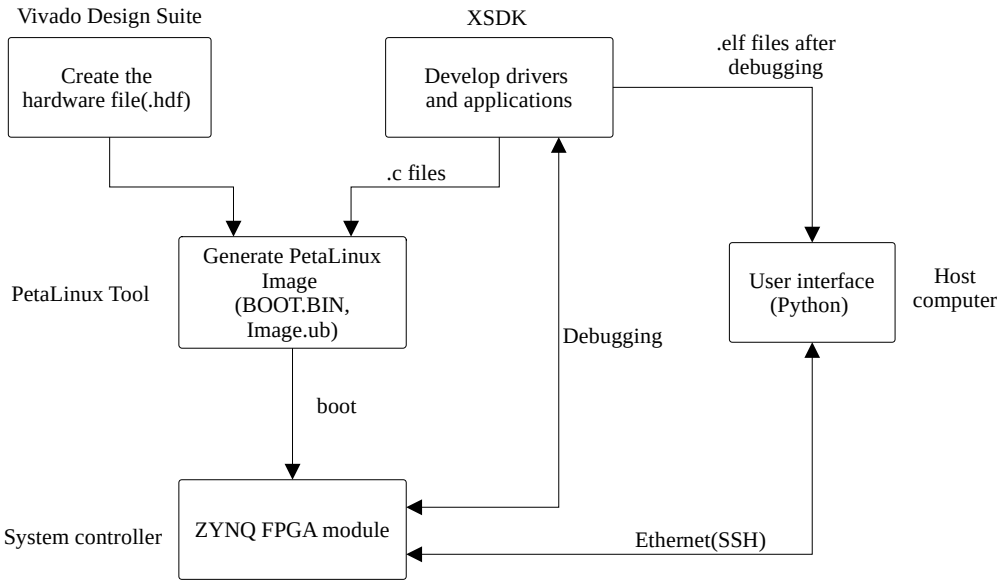


Fig. 4.9 Flow chart of PetaLinux building. In this study, the versions of the former work had been updated from 2015.3 to 2018.3.

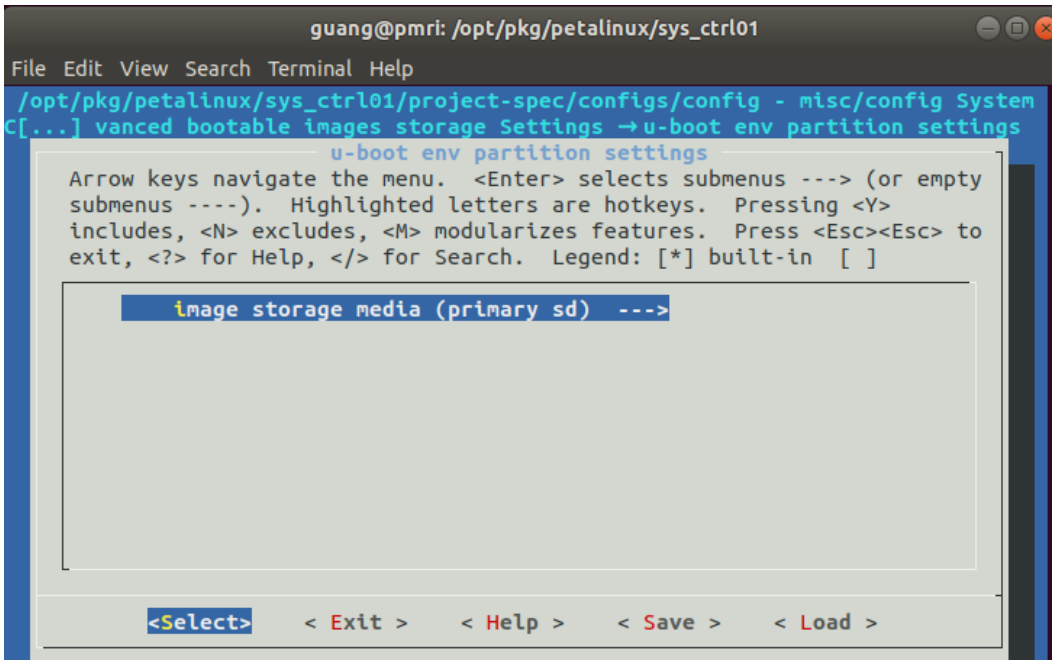


Fig. 4.10 A screenshot of PetaLinux building in the PetaLinux tool.

Table 4.4 Key configurations during PetaLinux build-up.

Configurations	Description
Booting	This configuration determines how the developed PetaLinux operating system booting.
C++ Library	The C++ library must be enabled manually for the associated C++ program used for parsing the ELF file.
IP Address	To define a static address for the PetaLinux operating system.
Driver	Install the PCIe driver.
Application	The applications can be developed after building the PetaLinux.

hardware with an SD card uses the generated image files saved in the SD card. Therefore, once the PetaLinux image is created and copied to the SD card, it can boot the image to the hardware automatically. While booting the PetaLinux image on hardware with JTAG or TFTP needs the JTAG cable each time. Therefore, the SD card for booting PetaLinux image is used. Figure 4.10 shows the configuration window for choosing the SD card as the image storage media while building the PetaLinux operating system.

After the PetaLinux operating system is installed, other applications can be developed and debugged by the software development kit (XSDK). The ELF files used for debugging or final application are different. The debug ELF file contains additional debug information for the debugger. However, the final developed ELF file for the application does not carry debug information. Once the applications are tested successfully, they would be used in the user interface developed by Python on Jupyter [97], on the host computer.

4.1.5 PCI Express Driver

To perform data transfer to/from the peripheral board (which refers to the transceiver board in this study), a PCIe driver was previously developed [11, 12] for the PetaLinux operating system on the system controller. The driver class is a character (char) driver, which allows PetaLinux on the system controller to access the user endpoint on the peripheral module as a file. Other driver classes, such as block devices or network interfaces [98], are not chosen for

the driver design as they do not match the functionalities of the peripheral module, which are described below:

1. Initialising access to a device. With this driver, the file system of the PetaLinux operating system is capable of accessing the memory block on the peripheral module as a char device. In the old version of the PetaLinux (before 2015), a terminal command **insmod + <driver name>** must be conducted to initiate the driver to be mounted under **/dev**. The later version of the PetaLinux could install the drive automatically without any terminal command.

2. Direct Memory Access (DMA). After initializing access, the peripheral device or an instance is created as a field of struct xpcie_dev on the operating system and the DMA engine could be enabled. Through programming on the system controller, the DMA on the peripheral Endpoint can be manipulated for reading and writing by setting the source and destination registers. Then transfers can be performed after writing the number of bytes to the transfer register.

4.1.6 API

The primary function of the prior API [11, 12] provided by earlier driver development is to access the DDR memory on the peripheral board, e.g., sending data to and copying data from the peripheral board. Table 4.5 indicates the main functions used in the API programming.

Table 4.5 API references.

Command	Description
open_dev(const char *path)	open and initialize device
close_dev(const char *path)	close and return allocated memory
read_data(int dev, unsigned char *read_buf, int addr)	Read data from the device at address addr and then store it in read_buff
write_data(int dev, unsigned char *data, int addr)	Write data to the device at the destination address, addr

4.2 PCIe User Endpoint on the Transceiver

Recall that the processed MR data can be stored in the DDR memory on the transceiver board (Figure 3.58), and the PCIe with Root-complex mode and the PCIe driver in the kernel space in the PetaLinux operating system are implemented. In order to access the DDR3 memory either for sending data or copying the processed MR data over the PCIe protocol between the system controller and the transceiver board, another FPGA fabric subsystem is needed to be implemented for initiating data transfer between them.

4.2.1 PCIe-Endpoint-DMA Subsystem

To perform the high speed data transfer over the PCIe link on the PXIe chassis, the PCI Express Endpoint-DMA Initiator Subsystem [99] was implemented in the FPGA fabric design on the transceiver board. The subsystem mainly consists of the AXI-PCIe block, a Central Direct Memory Access(CDMA) unit, and a translation block RAM. Figure 4.11 diagrammed the block design, and table 4.6 shows the description of the used logic cores.

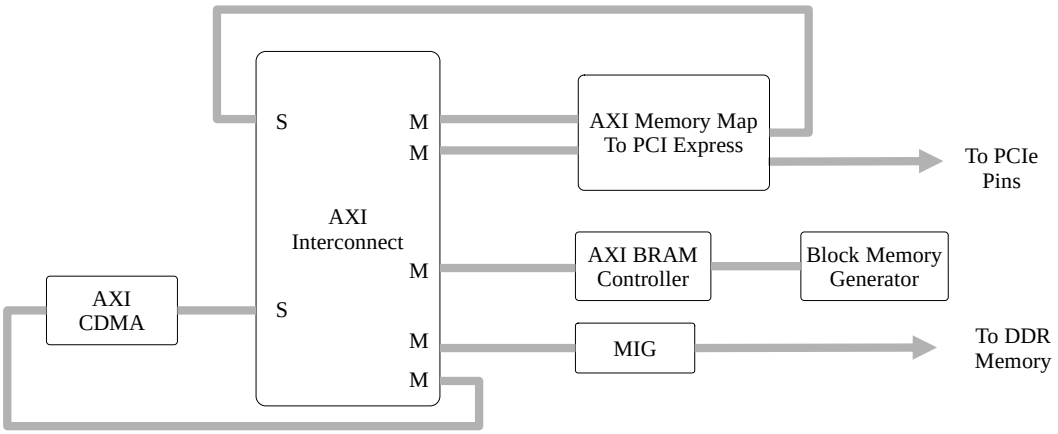


Fig. 4.11 PCIe CDMA subsystem.

The subsystem design is based around the AXI Memory Mapped to PCI Express logic core. As illustrated in Figure 4.7, this core provides slave and master AXI ports that allow the Root Complex device to send/receive data to/from the user endpoint over the PCIe protocol. By configuring the PCIe based address registers (BARs) properly, it provides the ability of

Table 4.6 IP cores description in PCIe CDMA subsystem.

IP Cores	Configuration and functions
AXI PCIe	AXI memory mapped to PCIe. It is configured as a PCI Express Endpoint device for decoding the packets. It is an interface between the AXI4 and PCI Express. It also provides the translation level between the AXI4 embedded system to the PCIe system.
Translation BRAM	The AXI Block RAM unit temporarily stores the PCIe address translation vectors facilitating dynamic translations during DMA Scatter Gather operation.
AXI CDMA	This logic core is used for performing DMA transactions, including the scatter gather(SG) operation, between the external host device (system controller) and the internal AXI connected peripherals over PCI Express.
AXI Interconnect	The multiple master ports and slave ports allow the peripheral cores to interact with the subsystem. In this application, the DDR3 memory is the user target for the subsystem via the Memory Interface Generator logic core.

the host device to access any hardware location in the AXI memory mapped realm. Also, the provided AXI to PCIe address translation enables the hardware on the user endpoint, e.g., a DMA unit, to transfer data upstream to the host device.

In the AXI PCIe core configuration, PCIE:BARs refer to the PCIe BARs as seen on the PCIe link, and it receives downstream PCIe data from the PCIe Root Complex side or host device. PCIe:BAR0 is configured to perform the address translation for PCIe to the base address of the Translation BRAM. It can be used to perform read or write transactions to the translation BRAM, AXI_PCIe_CTL, and AXI_CDMA_LITE interface. AXI:BARs refers to the AXI interface on the AXI PCI Express core. It can be seen from the AXI embedded system that initiates upstream PCIe traffic to PCIe Root Complex. As shown in Figure 4.12, two BARs are enabled. AXI:BAR0 is dedicated for the Scatter Gather port for Scatter Gather DMA transactions. That allows the DMA to read and process the descriptors from the Root Complex side. AXI:ABR1 is designated for the Data Move port for simple

DMA transactions. The DMA reads/writes from/to the PCIe Root Complex side through that port. However, because of the capability of the AXI CDMA core, the maximum allowed configuration address space for AXI:BAR0 is 8 MB.

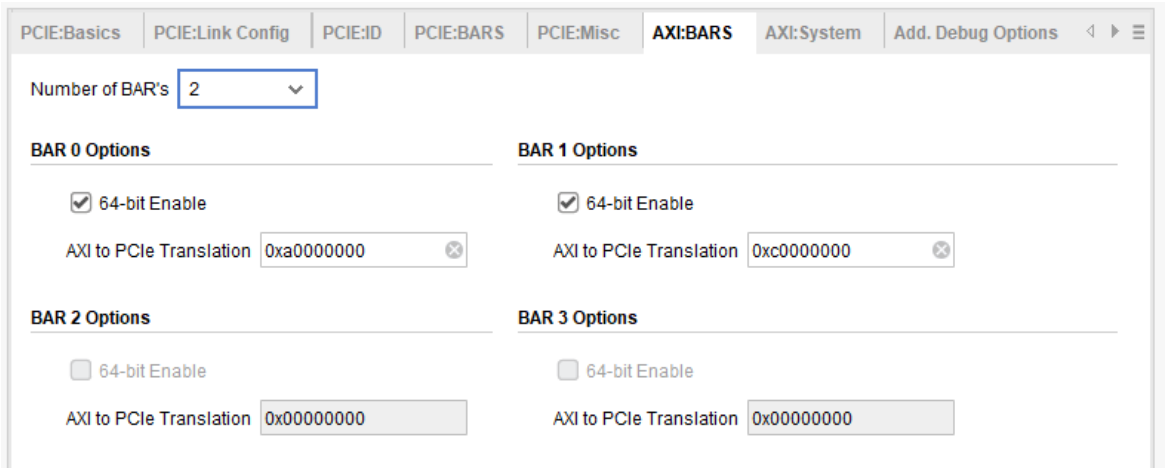


Fig. 4.12 AXI-PCIe logic core configuration.

The CDMA logic core, which has the same advantage of using the DMA in section 3.4.5, allows the FPGA to control the data transfer over the PCIe link to increase throughput and decrease processor engagement on the Root Complex side. In the subsystem, the data move is conducted in the CDMA logic. By giving the memory-mapped DMA the source address and destination address, it will transfer a block of data up to a maximum size of 8 MB. To change the transfer mode, the Scatter/Gather mode can be enabled by setting up multiple descriptors, which allows the DMA to continue to transfer data until it reaches the end of the list of the descriptors.

So far, the hardware integration of the spectrometer is completed, as shown in Figure 4.13.

4.2.2 Memory Mapping

Memory-mapped devices or logic can be accessed by PetaLinux or MicroBlaze or both, as shown in Figure 4.14. In the memory map, the limited local memory, which only could be accessed by the MicroBlaze, locates inside the FPGA to store the instruction code for the soft processor. The DDR memory, which is an external memory, can be accessed by both the MicroBlaze and the PetaLinux. The other logic, such as GPIO, DMA, which have

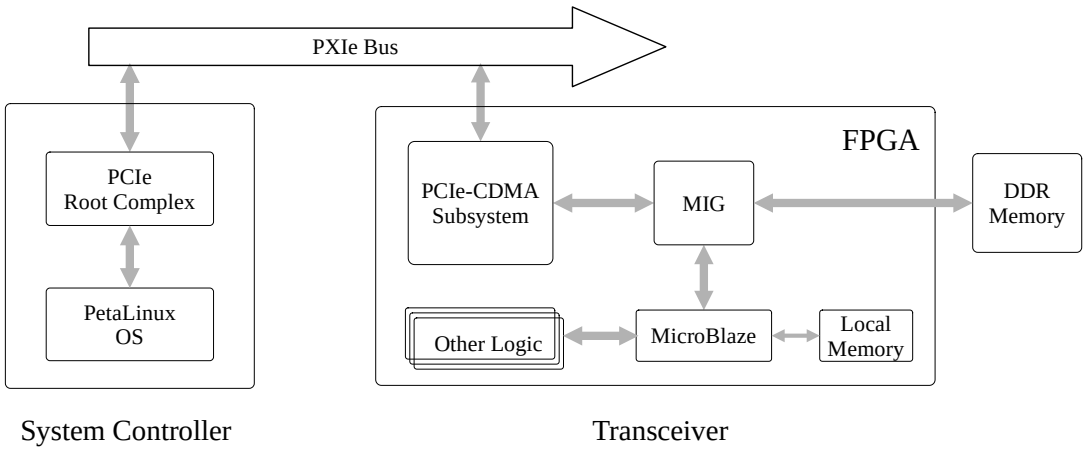


Fig. 4.13 Hardware integration of the spectrometer.

different memory addresses and ranges, could interact with MicroBlaze for performing specific functions.

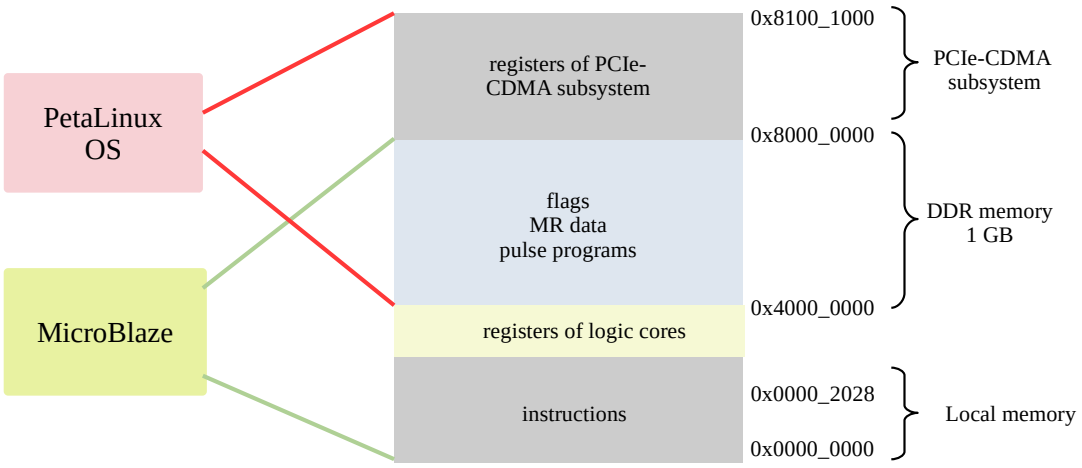


Fig. 4.14 Memory map of the spectrometer.

4.2.3 Memory Access Testing

After the hardware integration of the spectrometer is completed, we have to verify the integration. For the integration, the FPGA is programmed by the .bit file in the first place, and then the system controller starts booting. That sequence allows the PCIe End-point on the transceiver board to be detected by the operating system on the system controller.

To detect the PCIe Endpoint, a command line, **"lspci"**, for listing all the installed PCIe devices, can be used. As shown in Figure 4.15, the operating system detect both the PCIe Root-complex and the End-point.

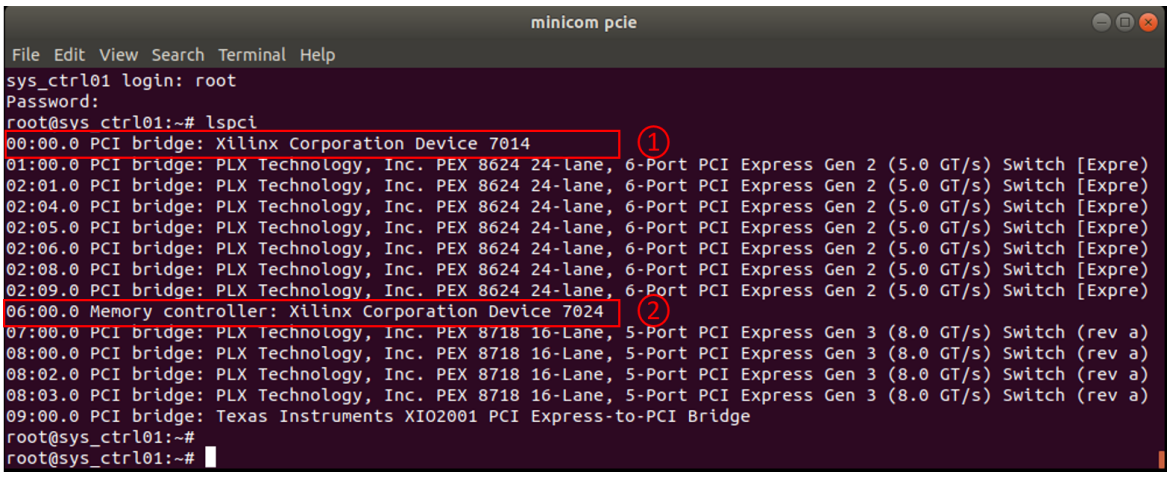


Fig. 4.15 PCIe Root-complex and PCIe End-point are detected in the PetaLinux operating system.

4.3 Integration of the Spectrometer

This section deals with spectrometer integration. Even though the data transfer can be performed over the PCIe protocol on the PCIe chassis, several issues in the spectrometer (for example, it is not easy for users to run the experiments) need to be addressed by introducing a software architecture. This section starts with the problems of the hardware integrated spectrometer.

4.3.1 Hardware Integrated Spectrometer Setup

Recall that the implementation of the transceiver board was detailed in Chapter 3, which includes the key components implementation, the FPGA fabric designs, software application development, and testing. However, during the implementation and verification of the transceiver board, a JTAG cable was used to temporarily program the FPGA with the compiled hardware file (bitstreams) provided by the Vivado Design Suit [93]. The development of the software applications for the softcore processor (MicroBlaze) also relied on the JTAG cable. The framework for these procedures is illustrated in Figure 4.16.

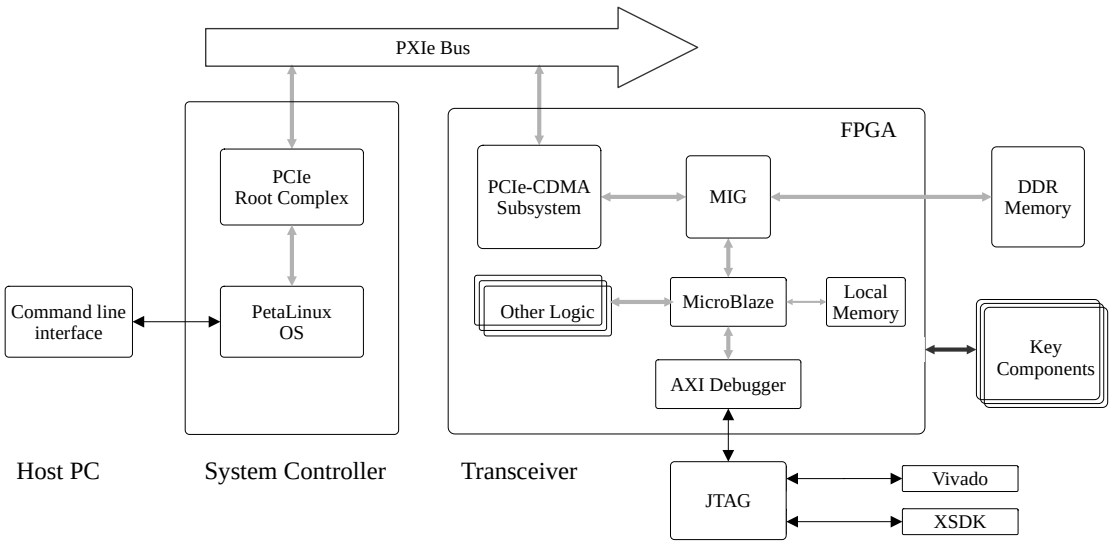


Fig. 4.16 Hardware Integrated Spectrometer Setup.

Because of the implementation of the PCIe-CDMA subsystem, in addition to the prior FPGA design (Figure 3.33), on the FPGA on the transceiver board, the saved data on the DDR memory can be accessed by the system controller. In the prior work [11, 12], a simple PCIe application provided with the PCIe driver was used to copy/send a chunk of data from/to the destination address on the DDR memory via a command-line interface on a host computer, as shown in Figure 4.17. The command-line interface allows users to send Linux commands or inputs followed by the prompts in the PCIe application to PetaLinux for execution. The inputs, including read, write, read destination address, write value and write destination address, navigate the PetaLinux to initiate data transfer with the DDR memory on the transceiver board over PCIe link. The copied data is shown by printing out; and it has to be copied to another platform for analysis.

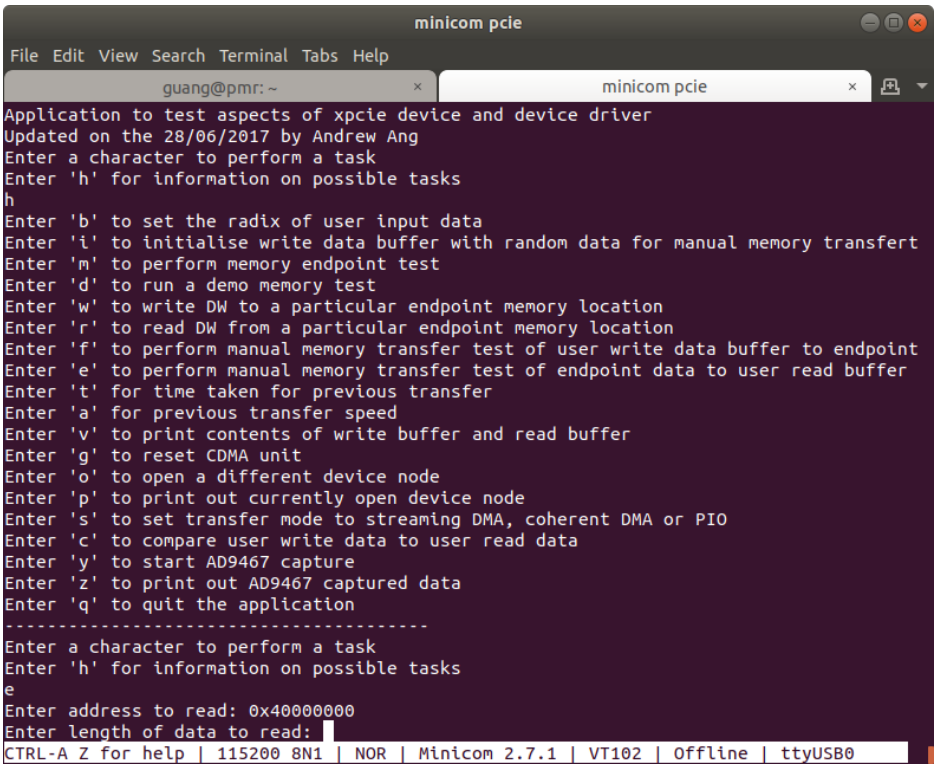


Fig. 4.17 Prior command line interface.

Obviously, the hardware integrated spectrometer is not user-friendly for MRI applications and should be optimized with the following aspects:

1. The command-line interface on the host computer can not manipulate the MRI experiment properly. On the one hand, it is not convenient for users to set up the parameters easily. Furthermore, the data-process and graphical data-display can not be done on it. Therefore, a new interface is needed for the spectrometer for operating MRI experiments.

2. The Xilinx design tools, Vivado and XSDK, which are professional, should not be used in each experiment once the FPGA development is completed. That will be an obstacle for users who are not familiar with them.

3. Following point 2, to make the spectrometer flexible, the software application e.g., the pulse program, running in the softcore processor (MicroBlaze on the transceiver board) should be adjustable and re-loadable for different kinds of MRI requirements. Such as FID and Spin Echo.

4.3.2 Software Architecture Overview

To tackle the mentioned problems of the hardware integrated spectrometer, the software architecture of the spectrometer is proposed and diagrammed in Figure 4.18. There are three parts: user interface on a platform, programs developed on the PetaLinux system on the system controller, and a Micro_OS (Micro operating system) running on the MicroBlaze on the transceiver board.

Overview

Figure 4.18 describes the software architecture of the spectrometer.

Pulse Program Thread

To execute an experiment on the transceiver board, in addition to the hardware components themselves, two other files are needed: one is the FPGA firmware file, and another one is the software application running on the soft processor(MicroBlaze). In this study, the software application is referred to as a pulse program dedicated for NMR or MRI experiments. To

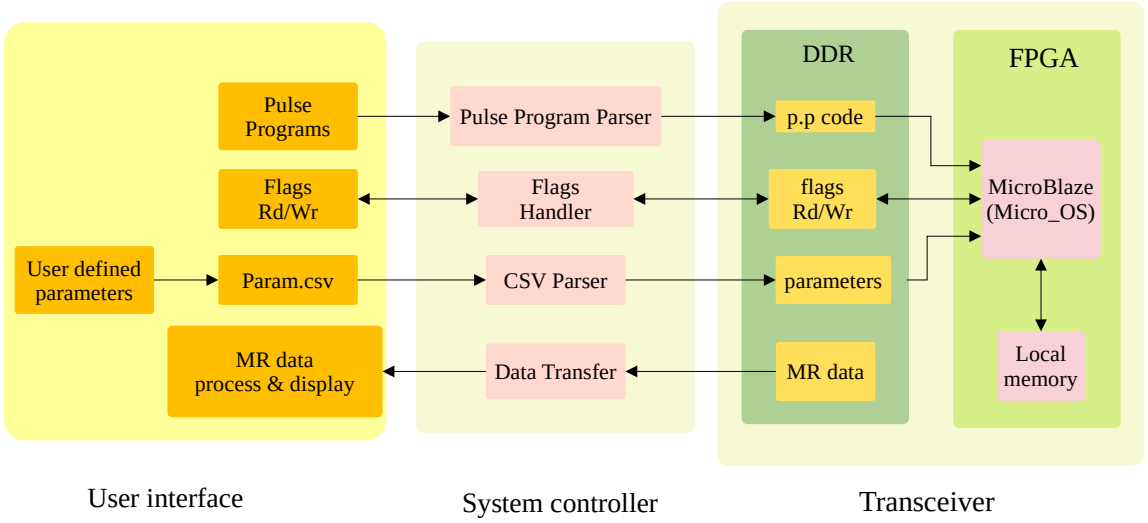


Fig. 4.18 Software architecture of the spectrometer. The PCIe- CDMA endpoint in the FPGA on the transceiver is not shown to simplify the diagram. p.p code refers to the pulse program.

execute an experiment, the firmware file should first program the FPGA, and then the pulse program operates on the soft processor. In the course of developing the software architecture, the firmware file still programs the FPGA through the JTAG cable temporarily. Once the software architecture is finished, the firmware file will be permanently stored in a flash memory to program the FPGA each time upon powering up. The strategy of storing that file is detailed in section 4.3.6.

As mentioned in section 4.3.1, to make the spectrometer flexible for different experiments, various pulse programs should be running on the softcore processor (MicroBlaze). In the software architecture, to achieve flexibility, the pulse programs are managed in the user interface and sent to the DDR memory on the transceiver board via a pulse program parser in the PetaLinux system on the system controller. Because the pulse program file can not be passed to the transceiver board directly, the pulse program parser extracts the core machine code (hexadecimal) of the pulse program to be sent. The machine code is then copied to local memory for executing through a Micro_OS, which is a Micro operating system referred to as a pulse program manager. More details about the extraction are elaborated in Pulse Program Parser and Micro_OS on MicroBlaze in section 4.3.4.

Flags Thread

In addition to the pulse program thread, the flag thread is created to control the process of the experiments. The thread allows users to control the process of an experiment by accessing the flag status registers in DDR memory. The registers are also monitored by the softcore processor for processing an experiment. Overall, the PetaLinux operating system and the Micro_OS running on the softcore processor cooperate to conduct an experiment by accessing the shared flag status registers.

Parameter Thread

The parameter thread is used for defining the experimental parameters, such as pulse width, pulse amplitude, echo time, etc., for an individual pulse program. In the software architecture (Figure 4.18), the user-defined parameters are transformed into a .csv file which is then sent and parsed by a CSV parser program in the PetaLinux operating system. Finally, the parameters are sent to DDR memory and copied to local memory.

MR Data Thread

Once the experiment is executed, and the data is stored in the DDR memory, the MR data thread transfers the data to the user interface via a system controller. During that process, the soft processor changes the "pulse program executed" register status, which is then seen by users from the flags thread. Then, users operate the copy data command to ask the system controller to copy data by the Data Transfer program running on the system controller.

4.3.3 User Interface Development

To develop a new user interface, the workflow is listed as below:

1. As the ZYNQ module does not support HDMI interface displaying for the installed PetaLinux operating system. Therefore, an appropriate platform needs to be selected to develop the proposed user interface.

2. By relying on the selected platform, a connection program needs to be built to bridge the PetaLinux operating system on the system controller.

3. Over the connection, some functional programs are to be developed to allow a user to set experimental parameters and to show the result.

Platform Selection

There are several options for the platform for the proposed user interface. A desktop PC or laptop with Ethernet has several advantages, such as the powerful computing ability and multiple available development tools for the connections or the functional programs on the Application side. Another method is to use an iPad as a platform to connect with the system controller via Wi-Fi The wireless platform looks more flexible, but it complicates the implementation because another Wi-Fi module needs to be mounted on the system controller. A desktop was chosen over a laptop as the desktop is also the platform for developing the software applications, which allows the user interface to use the developed application files (ELF file, developed by the SXDK).

Connection Protocol

Starting with the previously developed API, a new user interface was developed to produce some basic experiment control as well as provide an interface to a host computer. A connection program must be built to connect the operating system to the system controller based on the desktop PC. There are two connection ports on the system controller: USB serial port and Ethernet port. The USB port was the port for accessing the PetaLinux system, and it is still an option for connection for building the application. The Ethernet connection was employed as its bandwidth is higher than USB for transferring data. The bandwidth of the connection protocol should be as high as possible because the data transition could be intensive once more receive channels are added in the future.

The SSH (Secure Shell) was selected for implementing the Ethernet connection. The SSH protocol is a method for remote login from one operating system to another within a

network through a client/server architecture. A program developed for the SSH client runs the SSH protocol from one system in order to access another server system, issue commands, initiate data transfers, etc. In this application, the client is the host system to access the server, which is the PetaLinux operating system on the system controller, as diagrammed in Figure 4.19. The component for the SSH connection on the server/PetaLinux must be enabled in the root filesystem while building the PetaLinux operating system by the Xilinx PetaLinux Tool.

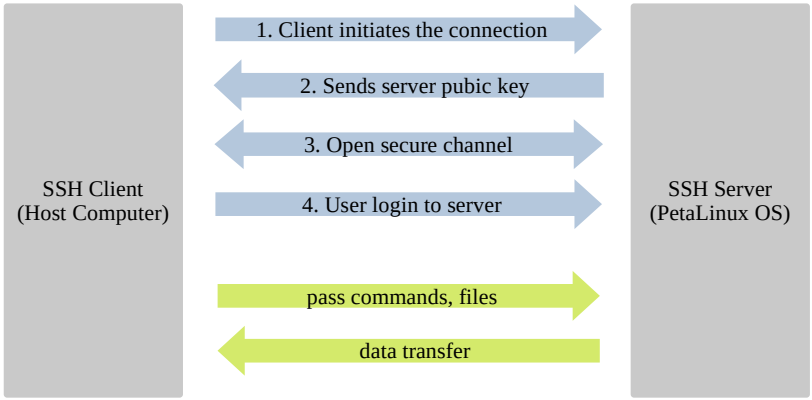


Fig. 4.19 Setup flow of a SSH connection. The blue arrows refer to the connection between the two system and the green arrows represent the desired functions.

Development Tool, JupyterLab

On the host computer, a development tool is needed to implement the SSH connection. JupyterLab [97] was adopted in this study. JupyterLab provides a web-based interactive development environment, and the associated Python libraries can accelerate the development. In this case, the Paramiko package [100], which is dedicated to the SSH implementation, was referred to develop the connection program. Listing 4.1 shows the Paramiko based function for SSH connection between the two systems. With the imported library, it needs to define the IP address, username, and password of the PetaLinux system in the **remote.connect()** function. This information can be defined when building the PetaLinux. After building the connection, that program allows the files to be transferred between the systems by calling the

Transport(host, port) function. In the last, **execute_commands**(Linux Command) is called to issue commands towards the server, PetaLinux system.

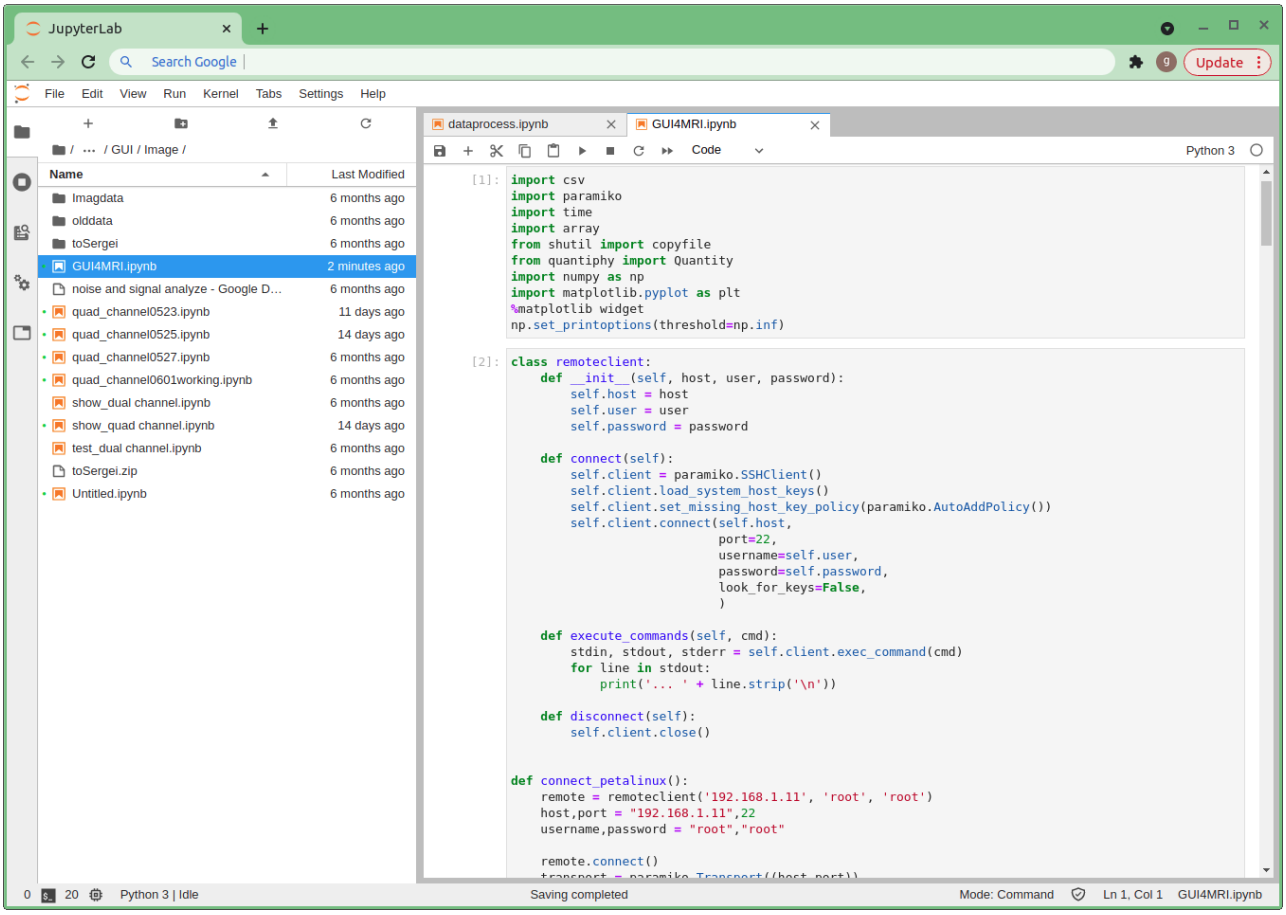


Fig. 4.20 Screenshot of the user interface.

```
1
2  def connect_petalinux():
3      remote = remoteclient('192.168.1.11', 'root', 'root')
4      host,port = "192.168.1.11",22
5      username,password = "root","root"
6      remote.connect()
7      transport = paramiko.Transport((host,port))
8      transport.connect(None,username,password)
9      sftp = paramiko.SFTPClient.from_transport(transport)
10     linuxCommand = 'lspci'
```

```
    remote.execute_commands(linuxCommand)
```

Listing 4.1 Python code to connect PetaLinux system on host computer.

Listing 4.2 shows the command code to execute the copydata function. In the function, the program source location on the host computer and destination location on PetaLinux system are defined. The sftp.put() function transfers the copydata ELF file to the PetaLinux. The transferred ELF file, of which the permission is changed to executable by Linux command: chmod + x, is then executed to copy data from DDR memory on the transceiver board. The copied data is then sent back to the host computer for further process via the function sftp.get() in the copydata function.

```
def copydata(numFiles):
    copydataCom = '/mnt/copydata.elf 160000 {0:d} '.format(numFiles)
    loc2petaScr = "/home/guang/Documents/programs/copymultdata2.elf"
    loc2petaDes = "/mnt/copydata.elf"
    sftp.put(loc2petaScr, loc2petaDes)
    remote.execute_commands('chmod +x /mnt/copydata.elf')
    remote.execute_commands(copydataCom)
    petaFilePath = '/mnt/fidata{0:d}.txt'
    localFilePath = '/home/guang/Videos/GUI/data/new/fidata{0:d}.txt'
    while True:
    try:
        for i in range(0,numFiles):
            sftp.get(petaFilePath.format(i), localFilePath.format(i))
        print("{0:d} files copied!".format(i+1))
        break
    except FileNotFoundError:
        time.sleep(3)
        print("wiat fid data!")
```

Listing 4.2 Copy data function.

4.3.4 Program Development

To fulfill the new requirements of the new user interface for controlling the MRI spectrometer, the programs in PetaLinux system on the system controller need to be developed.

Pulse Program Parser

The function of the "pulse program parser" program running on the system controller is to convert the pulse program file (ELF) to machine code and then send it to the transceiver board. Eventually, the softcore processor on the transceiver board can run the code to execute the pulse program for NMR/MRI experiments.

Before transferring the pulse program file (ELF) from the user interface to the transceiver board, it is necessary to understand what information is included in that file and which part is needed to be sent. Figure 4.21 shows the high-level header of the ELF file by using the Linux command: **readelf -h pulse.elf**.

```
(base) guang@pmr:~/Desktop/parse_elf$ readelf -h pulse.elf
ELF Header:
  Magic:   7f 45 4c 46 01 01 01 00 00 00 00 00 00 00 00 00
  Class:                             ELF32
  Data:                              2's complement, little endian
  Version:                           1 (current)
  OS/ABI:                            UNIX - System V
  ABI Version:                       0
  Type:                              EXEC (Executable file)
  Machine:                           Xilinx MicroBlaze
  Version:                           0x1
  Entry point address:               0x0
  Start of program headers:          52 (bytes into file)
  Start of section headers:          51680 (bytes into file)
  Flags:                             0x0
  Size of this header:               52 (bytes)
  Size of program headers:           32 (bytes)
  Number of program headers:         3
  Size of section headers:           40 (bytes)
  Number of section headers:         29
  Section header string table index: 28
```

Fig. 4.21 Listing of the highest level header in the ELF file.

Figure 4.22 lists the different sections of the process' address space that is specified from the pulse program file (ELF). The .text section header contains the instructions of the pulse program. It indicates that the instructions should be loaded at address 0x00002028 in the address space in the local memory, with which the softcore processor communicates.

```
(base) guang@pmr:~/Desktop/parse_elf$ readelf -S pulse.elf
There are 29 section headers, starting at offset 0xc9e0:

Section Headers:
  [Nr] Name              Type            Addr     Off    Size   ES Flg Lk Inf Al
  [ 0]                   NULL            00000000 000000 000000 00      0   0  0
  [ 1] .vectors.reset    PROGBITS        00000000 000094 000008 00  AX  0   0  4
  [ 2] .vectors.sw_excep PROGBITS        00000008 00009c 000008 00  AX  0   0  4
  [ 3] .vectors.interrup PROGBITS        00000010 0000a4 000008 00  AX  0   0  4
  [ 4] .vectors.hw_excep PROGBITS        00000020 0000b4 000008 00  AX  0   0  4
  [ 5] .text             PROGBITS        00002028 0000bc 00561c 00 WAX  0   0  4
  [ 6] .init             PROGBITS        00007644 0056d8 00003c 00  AX  0   0  4
  [ 7] .fini             PROGBITS        00007680 005714 000020 00  AX  0   0  4
  [ 8] .ctors            PROGBITS        000076a0 005734 000008 00  WA  0   0  4
  [ 9] .dtors            PROGBITS        000076a8 00573c 000008 00  WA  0   0  4
  [10] .rodata           PROGBITS        000076b0 005744 0005c4 00   A  0   0  4
  [11] .sdata2           NOBITS          00007c74 005d08 000004 00  WA  0   0  1
  [12] .data             PROGBITS        00007c78 005d08 0006d0 00  WA  0   0  4
  [13] .sdata            PROGBITS        00008348 0063d8 000000 00   W  0   0  1
  [14] .sbss             PROGBITS        00008348 0063d8 000000 00   W  0   0  1
  [15] .bss              NOBITS          00008348 0063d8 0007e4 00  WA  0   0  4
  [16] .heap             NOBITS          00008b2c 0063d8 000804 00  WA  0   0  1
  [17] .stack            NOBITS          00009330 0063d8 000400 00  WA  0   0  1
```

Fig. 4.22 Listing of the different sections from the ELF file.

Actually, it is the hexadecimal code that needs to be extracted from the ELF file and sent to the transceiver.

Figure 4.23 shows program adress, instruction (or machine) code, and assemble code extracted from the .text section header. Actually, it is the instruction code that needs to be extracted from the ELF file and sent to the transceiver.

```
Disassembly of section .text:

00002028 <_start1>:
    2028:       b0000000        imm     0
    202c:       31a08348        addik   r13, r0, -31928 // 8348 <_SDA_BASE_>
    2030:       b0000000        imm     0
    2034:       30407c78        addik   r2, r0, 31864   // 7c78 <_SDA2_BASE_>
    2038:       b0000000        imm     0
```

Fig. 4.23 Listing of the .text section header.

Overall, as shown in Figure 4.18, "pulse program parser" program extracts the instruction code from the ELF file created by XSDK and sends it to the DDR memory on the transceiver

board. Another program running on the MicroBlaze copies the code from the DDR memory to the local memory and finally runs the code.

The user interface sends the pulse program ELF file generated by the XSDK to the PetaLinux operating system via an SSH connection.The elf parser then parses the file to extract the required instructions, which results in a .txt file. The pulse program txt format file is processed by data transfer, and the instructions are sent to the DDR memory over the PCIe link. The program was developed by referring to the ELFIO library [101].

CSV File Parser

The CSV file parser program running in the PetaLinux system extracts the experimental parameters in a .csv file created from the user interface and then transfers the parameters to the DDR memory on the transceiver board via the **write_data()** function in the API in the PetaLinux system. The softcore processor uses the parameters in NMR/MRI experiments.

Flag Status

Flag status is the pre-defined registers in the DDR memory to indicate the status in a process. Such as, when the pulse program file is loaded to the DDR memory, or the pulse program is executed. The flag status registers can be accessed by both the system controller and the softcore processor on the transceiver. The PetaLinux operating system on the system controller can notice the status by initiating a reading program, or change the status by initiating a writing program. The soft processor on the transceiver board also could access the status by using the related functions, such as **Xil_In32(addr)** and **Xil_Out(addr, value)**.

Copydata

The Copydata program is developed for copying the MR data on DDR memory via the **read_data()** function in API in the PetaLinux system. The amount of data to be copied is defined by experiment parameters, such as the number of FIDs, rotation times (will be introduced in Chapter 5). The copied data is stored in multiple files, which are then transferred to the user interface through SSH protocol.

Micro_OS on MicroBlaze

Micro_OS (micro operating system) on the MicroBlaze is a program that manages the status of a process and loads the pulse program instruction code from the DDR memory to the local memory to be executed. Before loading, it scans the associated status flag to see if the instruction code is loaded into the DDR memory. Once the instruction copy is finished, the Micro_OS will change the flag status, which the user interface can see via the PetaLinux on the system controller Listing 4.3 shows the Micro_OS program.

```
#include <stdio.h>
#include "xil_printf.h"
#define  COMMAND_ADDR    0x40000014
#define  RUN_COMMAND     0x12345678
#define  STOP_COMMAND    0x87654321
#define  MEM_SIZE_ADDR   0x40000010
#define  SCR_ADDR        0x40001000
#define  TAR_ADDR        0x00002000
#define CALL_PULSE_PROGRAM asm("bralid r15, 0x2028; nop");
int main()
{
  unsigned char *des_addr; unsigned char *tar_addr;
  des_addr = (unsigned char *) TAR_ADDR;
  tar_addr = (unsigned char *) SCR_ADDR;
  while(1){
    if(   ((int)(Xil_In32(COMMAND_ADDR))) == RUN_COMMAND   ){
      int mem_size = Xil_In32(MEM_SIZE_ADDR);
      Xil_MemCpy((void *)des_addr, (void *)tar_addr, mem_size);
      xil_printf("Run Pulse!!!\n\r");
      CALL_PULSE_PROGRAM; // call subroutine
      xil_printf("Pulse executed!\n\r"); // a mark!
      Xil_Out32(COMMAND_ADDR, 0x10100101);
      break;
      if(((int)(Xil_In32(COMMAND_ADDR))) != RUN_COMMAND){
        break;
      }
    }
  };
    return 0;
}
```

Listing 4.3 Copy pulse program code.

The following paragraphs details how the experiment status and the pulse program are processed. C code and assemble code was listed for illustration. The program address, instruction code, and operations are listed, as shown in Figure 4.24. **bralid** [68] in the first

marked line is an assembly instruction, which allows the program address to jump to a physical address to cause a system call exception. In this program, it put "8232(0x2028)" into register r15 to be pointed in the next operation. "0x2028" is the entrance address of the pulse program function. In other words, that assemble function makes the pulse program executed in the processor, as illustrated in Figure 4.26. The "eec" is a marker in the Micro_OS, which allows the program address point back to the Micro_OS after executing the pulse program.

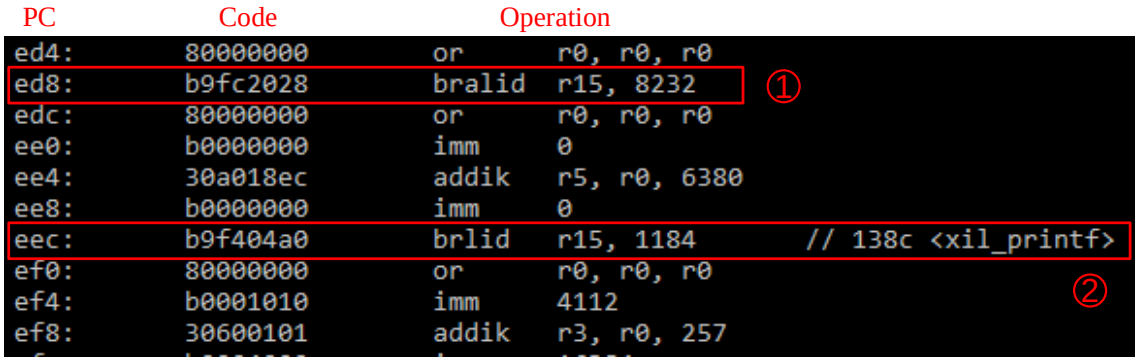


Fig. 4.24 Micro_OS disassembled code. Bash Shell was launched in XSDK and command is used to dump the ELF file of Micro_OS: **$ mb-objdump -S <file name>**.

In Figure 4.25, the function **xil_printf()** positioned after **pulse_run()** acts as a marker to find out the program address after the pulse function is executed. Then, another assemble function **rtsd** [68] is used to return to subroutine branch to the location specified by the contents of r15 plus the IMM field:

$$PC \leftarrow (r15) + sext(IMM)$$

This function allows the program address to go back to the Micro_OS program after the pulse program is executed. The flow of the mechanism and the calculation are shown in Figure 4.26. Figure 4.27 shows more details about copying the pulse program instruction code to the local memory, such as the memory addresses for the Micro_OS program and the pulse program subroutine and the functions in the Micro_OS program.

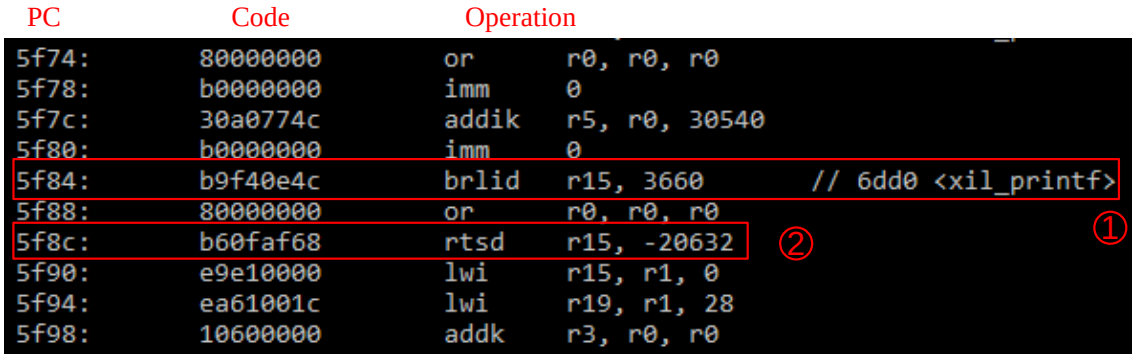
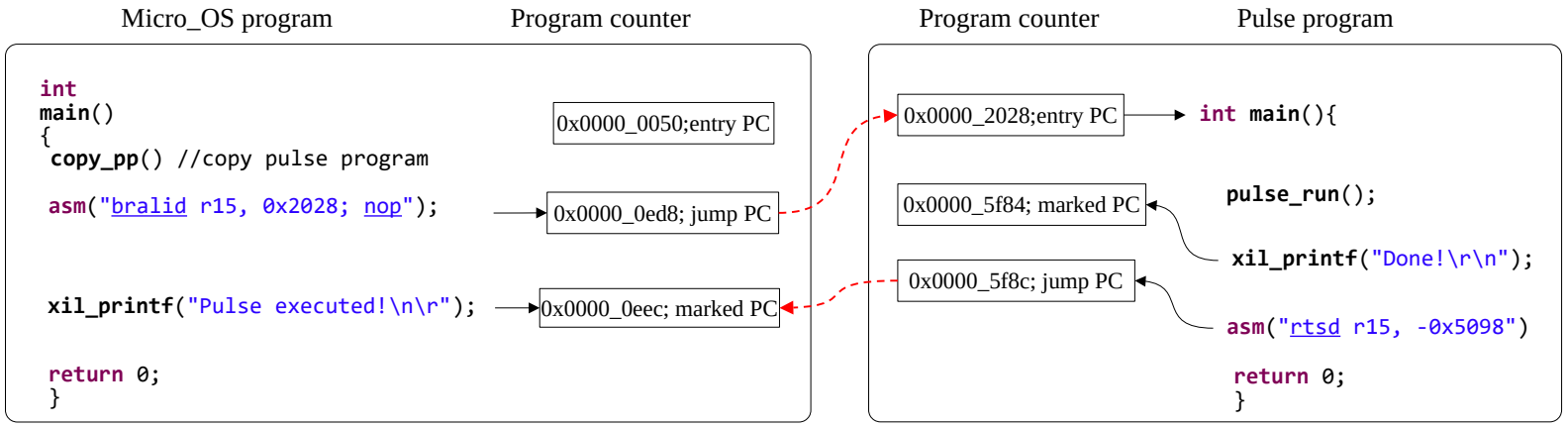


Fig. 4.25 Disassembled code of pulse program.



Calculation: 0x0eec = 0x5f84 - 0x5098

Fig. 4.26 Program address calculation.

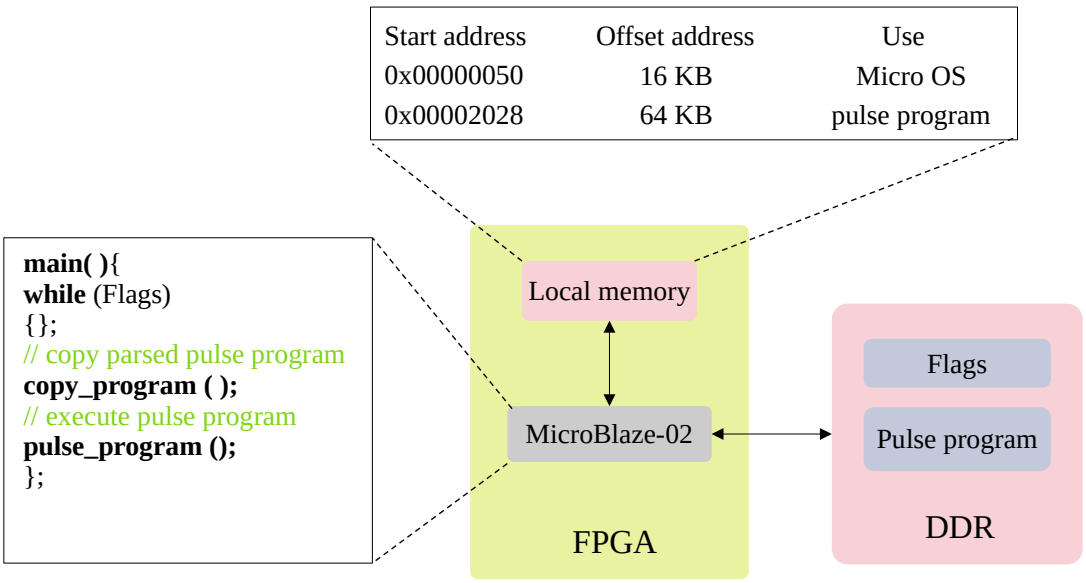


Fig. 4.27 Copy pulse program from DDR memory to the FPGA local memory.

4.3.5 Initiate a Pulse Program

The flow chart of the main functions running on the MicroBlaze is shown in Figure 4.28. Basically, four steps are made after scanning the flag status registers: copy pulse program, copy parameters, run the pulse program, and modify the "pulse program executed" register. Modifying that register allows the system controller to know that data is ready to copy.

4.3.6 Storing Program File for the FPGA

In the previous sections of developing the software architecture, the FPGA was programmed by the bitstream file created by Vivado through a JTAG cable. Once the designs are completed, the JTAG cable should be avoided to make the spectrometer user-friendly. The strategy is to store the FPGA program file into the flash memory device, which allows the FPGA can be configured when powered up. In Vivado, the bitstream file is converted to an .mcs file for the flash memory device, as shown in Figure 4.29.

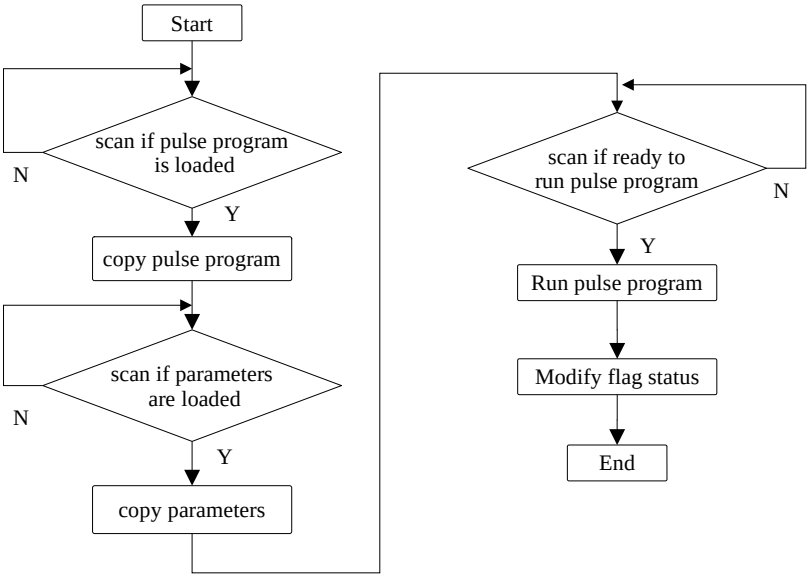


Fig. 4.28 Flow chart of the Micro_OS program.

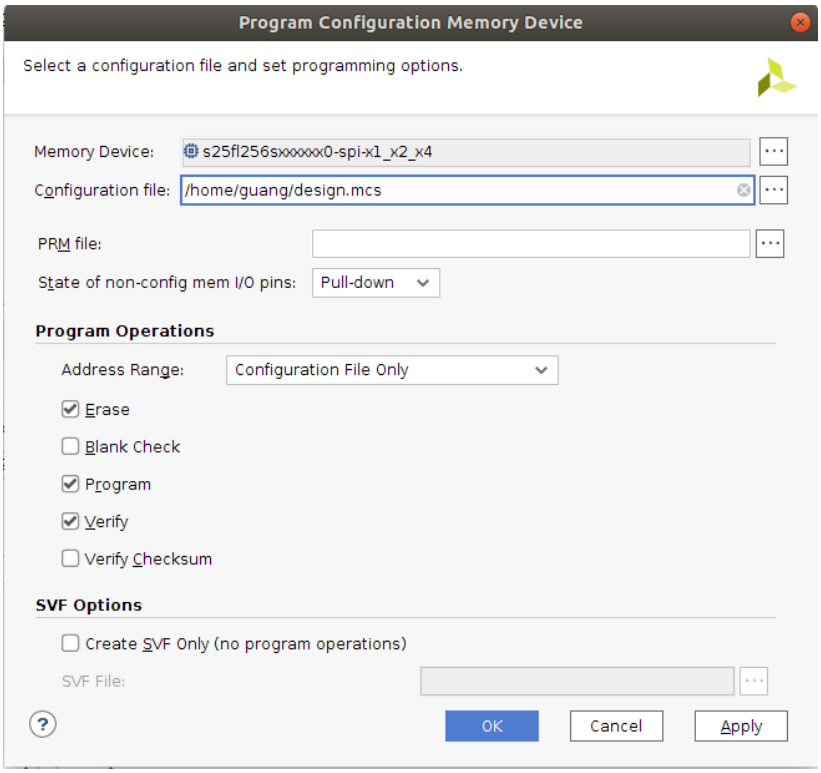


Fig. 4.29 Configuration of the memory device.

4.3.7 Completely Integrated Spectrometer

Figure 4.30 shows the completely integrated spectrometer. The user interface with pulse program, flags interacts with system controller on the MRI spectrometer through SSH protocol. The multichannel transceiver executes the pulse program and processes the acquired data. The system controller and the multichannel transceiver communicate through the PXIe bus.

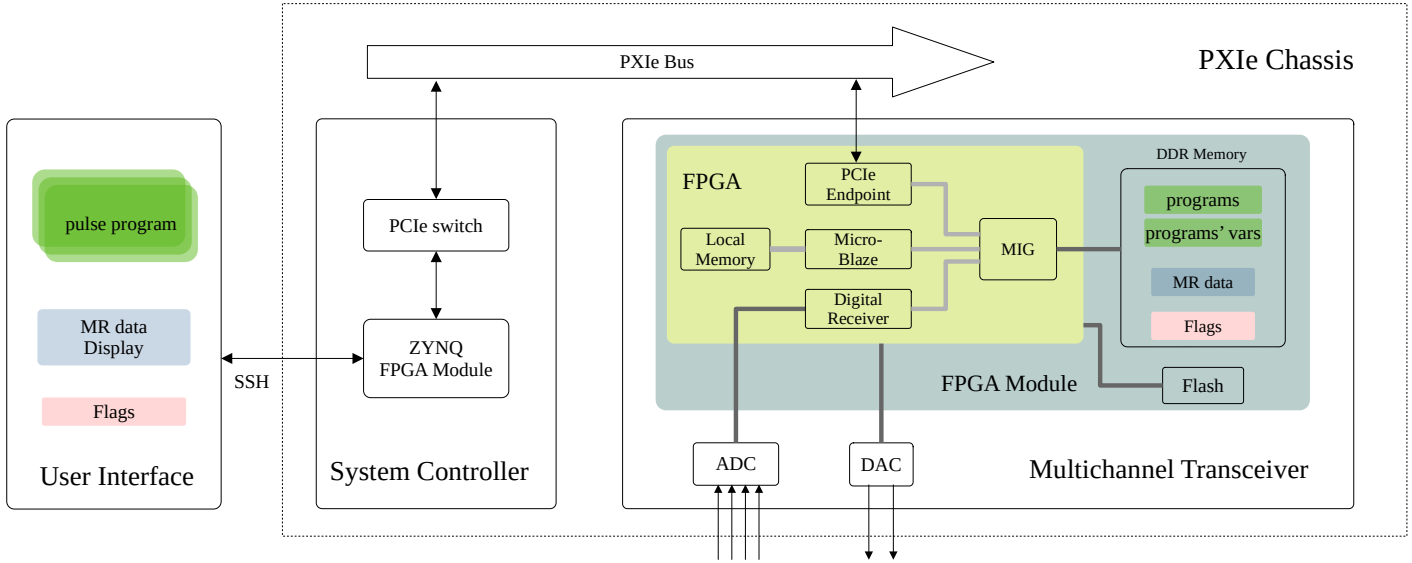


Fig. 4.30 Diagram of the spectrometer integration.

4.4 NMR Experiment Testing

4.4.1 Multichannel Testing

Before doing the NMR/MRI experiment, the four channels were tested regarding to the phase and amplitude. Figure 4.31 shows the system setup for testing the multichannel receiver. A single-channel signal generator generates a sine waveform signal with -60 dBm amplitude to feed a power splitter. The power splitter splits the signal into four channels for four individual pre-amplifiers. Then the quad-channel receiver digitized the analog signal, and the processed data was shown on the user interface.

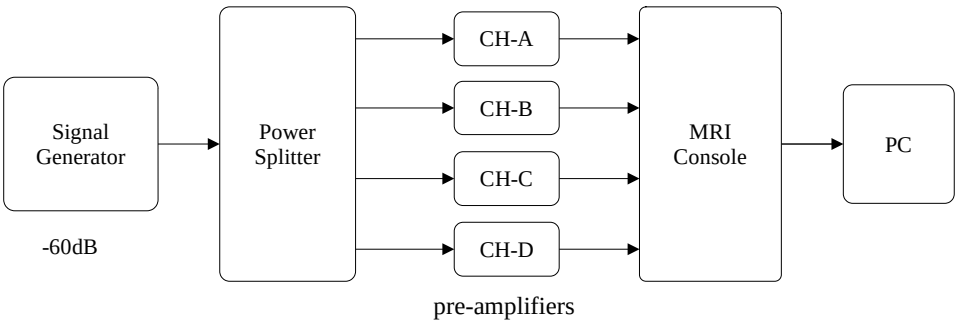
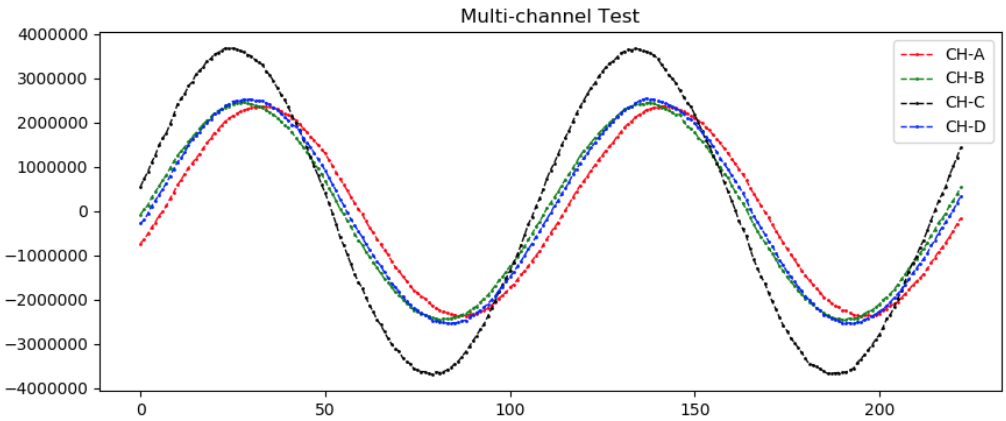


Fig. 4.31 Multichannel receiver testing setup.

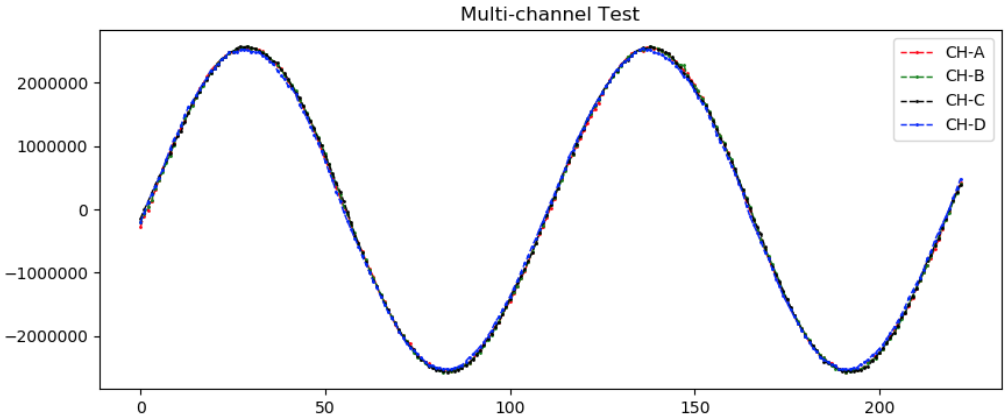
Figure 4.32a shows data from the four-channel receivers, which reveals that the amplitude and the phase are not identical. The reasons that cause the difference might be the non-identical amplifiers or tolerance of the capacitors. However, data from the four receive chain can be calibrated by shifting the phase and multiplying coefficients. As a result, the data from the four channels are identical after the calibration, as shown in Figure 4.32b. Imaging reconstruction will use the coefficients for calibration.

4.4.2 System Setup

This subsection details the system setup for single-channel NMR testing. Table 4.7 shows the elements for the system setup.



(a) Non-calibrated signal.



(b) Calibrated signal.

Fig. 4.32 Four channel data from the multichannel receiver. The interval time between two points is 100 ns.

Table 4.7 Elements of system setup

Elements	**Description**
System controller	To receive the user command.
Transceiver	To execute the pulse program and process the received data.
PXIe chassis	Commercial PXIe chassis, PXES-2301 [102], which provides power supply and associated connectors for these two boards.
Magnet	The Halbach magnet provides the static field. The direction is on the transverse plane.
Probe	To produce the B_1 field and detect the MR signal.
Pre-amplifier	To amplify the small MR signal.
Duplexer	To protect the pre-amplifier during transmitting.
Power amplifier	To amplify the RF pulse for excitation.
PC	To send commands and show results from the user interface.
Cables	Multiple RF cables; and an Ethernet cable to connect system controller and PC.

The RF front-end circuit is shown in Figure 4.33. The lumped elements of the cross diodes D1 and D2, and the π-network functions as a quarter wavelength transmission line. The quarter wavelength behaves as a transmit/receive switch [103–105] to protect the pre-amplifier during transmission. During transmitting, the high power goes to the probe through the crossed diodes, D1. During that period, the crossed D2 is conducting, and the pi-network is at the resonant condition. These lead to a high impedance and protects the pre-amplifier from being overwhelmed. Both D1 and D2 are open during receiving due to the weak signal. Under that circumstance, D1 blocks the noise from the high power amplifier, and the π-network behaves as a low pass filter.

Figure 4.34 shows a picture of the pre-amplifier (originally designed by Robin Dykstra [10]). The first two stages are shielded by a shielding boxed in the pre-amplifier to

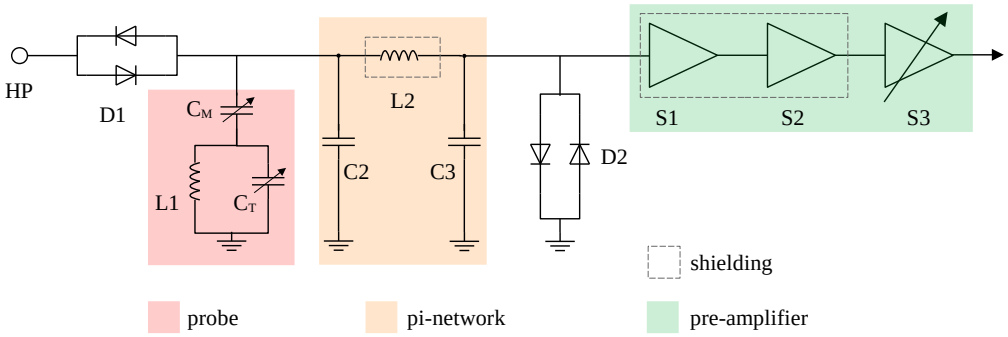


Fig. 4.33 RF front-end circuit.

prevent noise from interfering. The inductor, L2, is also shielded in the box as the inductor might pick up noise.

A variable gain amplifier in the last stage was implemented to amplify the signal up to a level that is suitable for the digital receiver input. The FPGA on the transceiver supplies the TTL signals for controlling the gain.

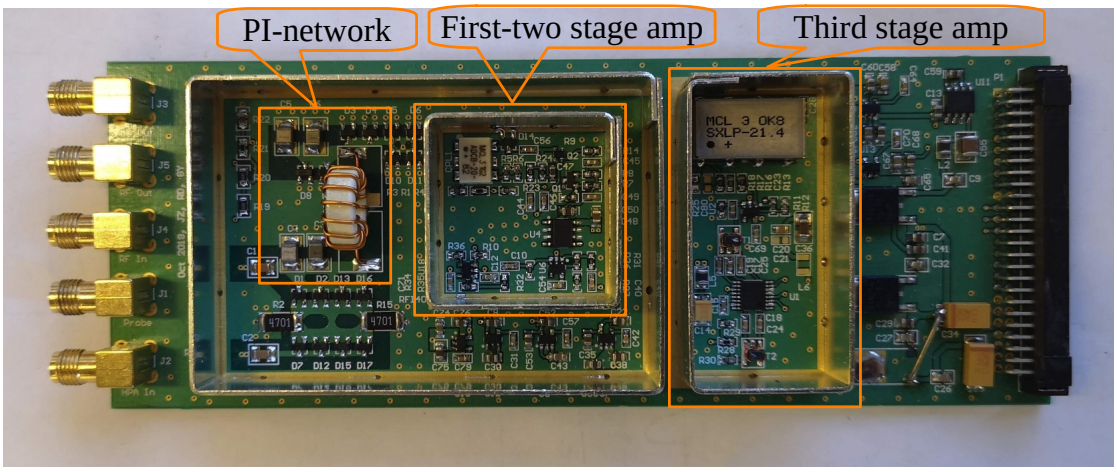


Fig. 4.34 Pre-amplifier.

Figure 4.35 shows the picture of the system setup. This work [106] was presented in the 29th ISMRM conference.

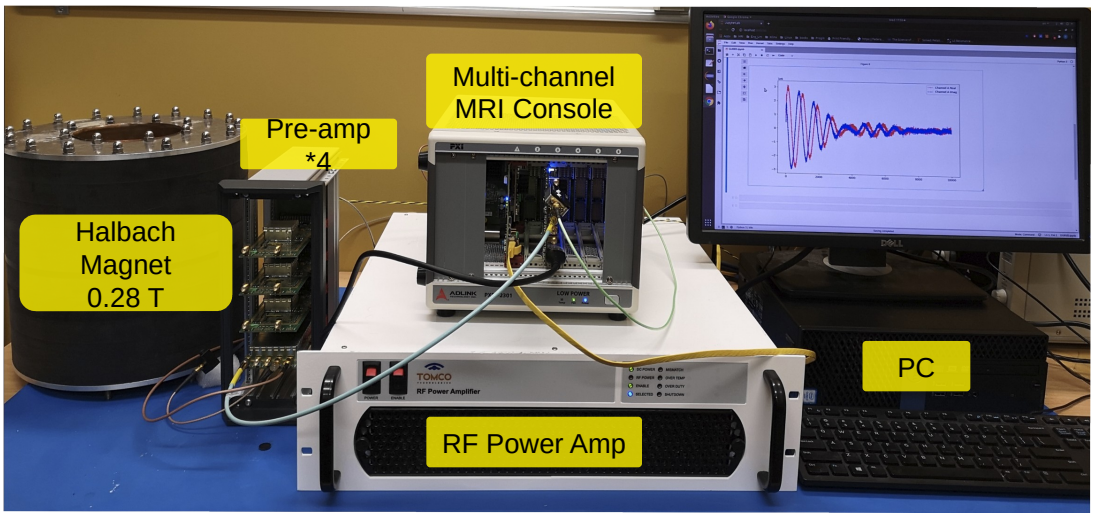


Fig. 4.35 System setup for NMR testing.

4.4.3 FID Testing

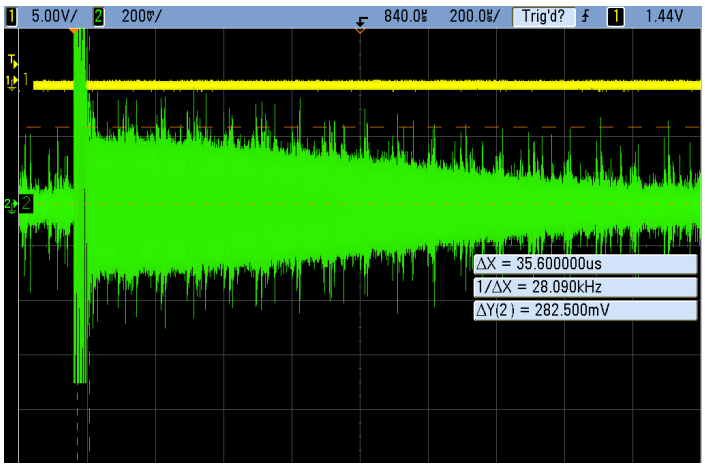
An FID (Free Induction Decay) experiment was conducted to verify the NMR system. In Figure 4.36b, the first row shows the raw FID signal, which comes from the output of the ADC interface module; the second row shows the mixed signal in the digital receiver processor; the last row reveals the signal output from the CIC filter. Figure 4.36c shows the FID data on the user interface.

4.4.4 Spin Echo Testing

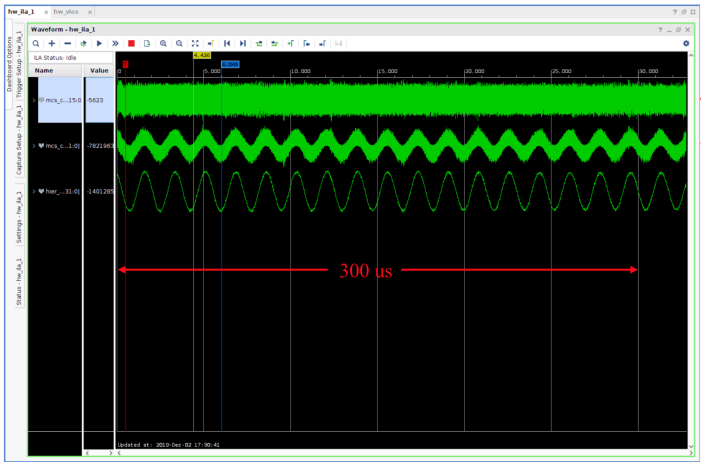
Figure 4.37 shows the spin echo pulse sequence, including a $\pi/2$ pulse and π pulse. Figure 4.38 shows the spin echo signal on the user interface. In this experiment, the acquisition is continuously operating during the sequence; as seen in the figure, the receiver chain acquired the noise caused by the π pulse.

4.5 Discussion

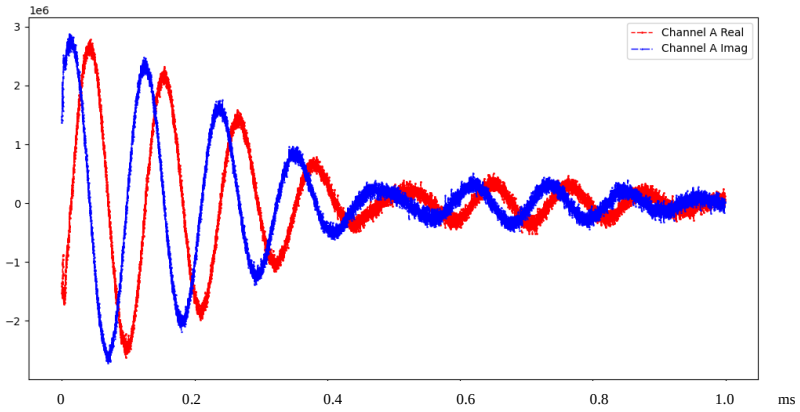
As shown in the quantification section (3.5.2), the spectrometer does cover a large range, from 1 MHz to 130 MHz. The high bandwidth of the spectrometer benefits from the bandwidth



(a) FID signal captured from oscilloscope.

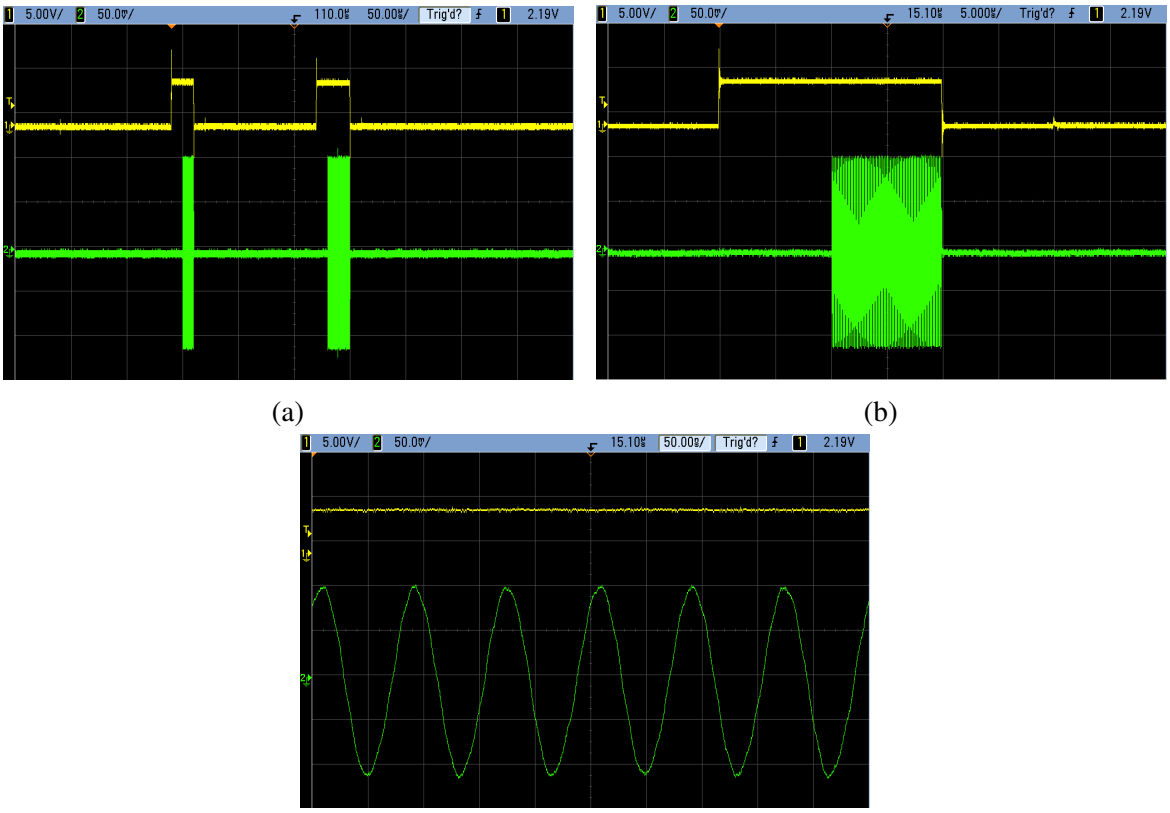


(b) FID signal captured from the ILA.



(c) FID signal on the user interface.

Fig. 4.36 FID signals.



(a) (b)

(c)

Fig. 4.37 Spin echo pulse sequence.(a), (b), (c) showing the transmitter pulse in different timescale.

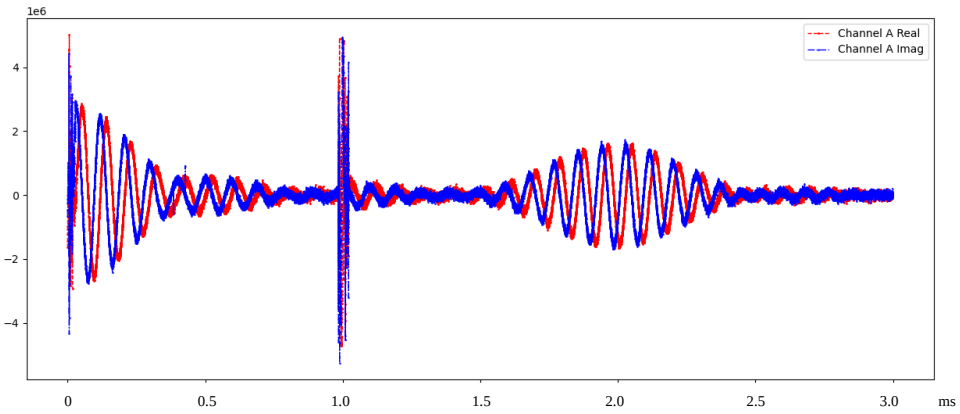


Fig. 4.38 Spin echo on the user interface.

of the ADC and DAC, which is broad. The spectrometer can be applied to different MRI systems, from a low-field (e.g., 0.05 T) system to a 3T system by only using the proper pre-amplifiers and power amplifiers.

Compared with the total cost of the multichannel spectrometer, the cost of the high speed ADC ($500 USD per chip) with 4 channels does not matter too much. In addition, another considerable cost is the PXIe chassis, which cost about $2000 NZD. However, as mentioned in 2.4.1 (Motivation), the advantages of the PXIe chassis outweigh its cost, such as the scalability, high bandwidth, and the open standard platform that allows other researchers to work together to develop a more complex MRI system.

For the current design, the architecture can be easily expanded from 4 channels to 32 channels by duplicating other seven transceiving boards. In the future, an 8-channel receiver can be used on one transceiver board, then only four transceiver boards will be used for a 32-channel receiver.

In this design, the pulse program is executed by a microprocessor. Therefore, we just need to modify the software to realize different pulse programs. As mentioned in section “Integration of the Spectrometer”, the pulse program can be loaded from the user interface to the microcontroller on the transceiver board.

Chapter 5

2-D Imaging System

The chapter 4 presented the spectrometer integration and single-channel NMR testing. This chapter moves forward to preparing the 2-D imaging system based on the multichannel transceiver spectrometer. The main points are discussed as follows:

- Field mapping of the magnet. The field map of the Halbach magnet was measured via a small NMR probe. A frame for rotating the magnet was constructed to allow the NMR probe to scan all the points in the target area. In addition, the methodology, NMR probe details, calculation, and temperature effect are discussed.
- The RF coils, such as the transmitter coil, and the receive coil array, are constructed for the 2-D imaging system. Decoupling and detuning strategies are also elaborated.
- The surface coil sensitivity is calculated with the changing of the B_0 direction.
- Signal capture and verification are illustrated by presenting the four acquired FID signal from the receive coil array (with four surface coils).
- Image reconstruction and third-dimension encoding are discussed.

5.1 Overview of the 2-D Imaging System

As proposed by Juergen Hennig [36], the ambiguity of the non-unidirectional, nonbijective spatial encoding magnetic fields (SEMs) can be solved by using multiple receiver coils and parallel reconstruction. In the statement, the fields are changed by inverting the direction of the current in the coils. In this study, for the 2-D imaging system, the natural inhomogeneous magnetic field acts as the SEMs. An individual coil in a coil array has a specific sensitivity over the sample in a particular direction of the static magnet. The magnetic fields are changed by rotating the magnet and keeping the receive coils and sample stable. The coil has a specific sensitivity, which can be calculated by the Biot-Savart Law when it is located at a certain angle with the direction of the static field of the magnet. The static field in the field of view also has to be measured for image reconstruction.

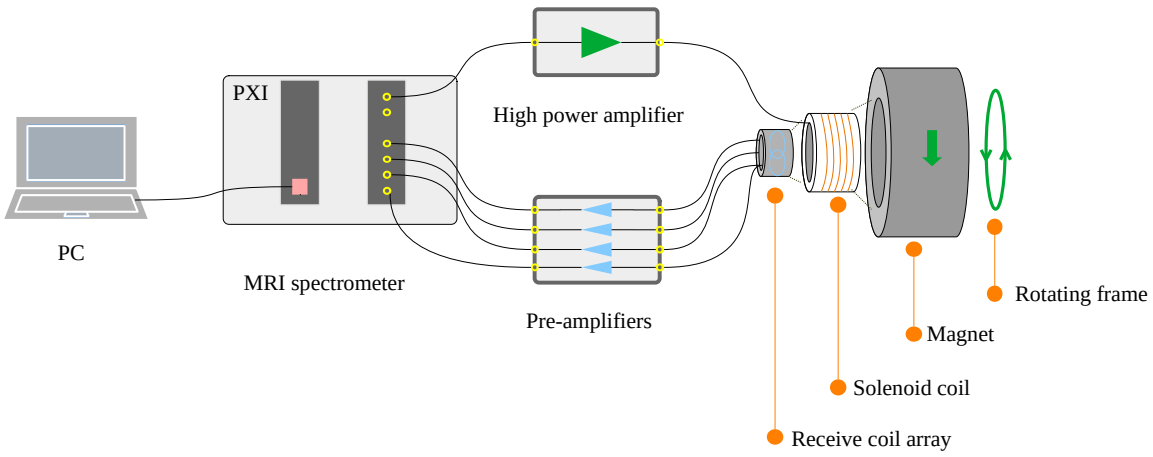


Fig. 5.1 Overview of the 2D imaging system.

Figure 5.1 shows the overview of the 2D imaging system. To complete that system, more work needs to be completed, such as measuring the static field map of the Halbach magnet, building the RF coil system, calculating the coil sensitivity of the receive coil.

5.2 Field Mapping

5.2.1 Methodology

The B_0 field created by the Halbach magnet has to be measured for image reconstruction. To measure the field map, each field of one point in the map is the frequency of nuclear spin precession because the frequency is proportional to the local magnetic field. This method is called NMR magnetometry, and it can be applied to both high-field and low-field [107–110]. NMR magnetometry adopts a small single NMR probe for both excitation and receiving to detect an FID, reflecting field information of the position where the probe is located. After collecting the FID data, one way to calculate the frequency is to use the FFT function in Python to transform the time domain data to a frequency one. However, in certain positions, the FID decays fast, and the effective data for the FFT function is limited, which leads to inaccurate results. Here, another method is used to calculate the frequency. The method is discussed in section 5.2.3.

The NMR probe should be as small as possible to get a better field resolution. A picture of the NMR probe is shown in Figure 5.2. Table 5.1 lists the parameters of the NMR probe. Copper sulfate water is used as the sample as it has a shorter T_1, which accelerates the field mapping. For the field mapping, a single pulse with a length of 12 us was used for excitation. At one point in the field-of-view, it scanned eight times for signal averaging to improve the SNR. The repetition time between each scan is 100 ms. The excitation frequency was 12.04 MHz.

Figure 5.3 shows the signals and the spectrum. The first row reveals the overlapped eight real part signals from the eight scans.The second row shows the averaged real and imaginary part of the FID signal. The third row is the spectrum of the averaged FID signal.

5.2.2 Frame for Rotating the Magnet

Figure 5.4 shows a picture of the frame for rotating the magnet, and Figure 5.5 lists the key components in the frame. There are two rings in the bearing, the outside and inner rings. By fastening the inner ring on the plate installed on the frame, a stepper motor can drive the

Fig. 5.2 The NMR probe for field mapping.

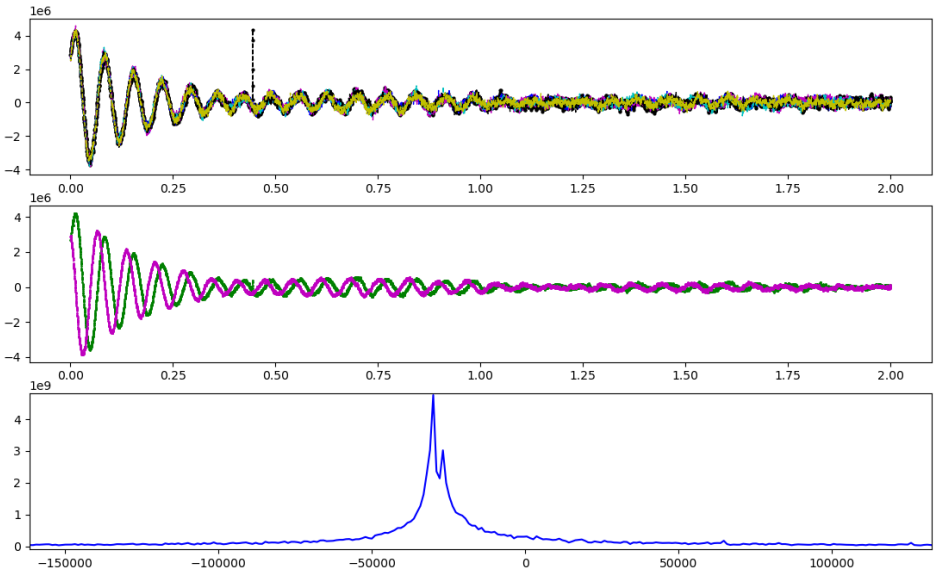


Fig. 5.3 FID signals and the spectrum. The timeline unit for the FID signals is millisecond; and the unit for the spectrum is Hz.

Table 5.1 NMR probe parameters. The inductance and resistance of the coil are measured at 12 MHz. The inductance and the resistance were measured by an E5061 Agilent Network Analyzer.

Items	**Values**
Copper diameter	0.5 mm
Turns	10
Tube diameter	3 mm
Inductance(L)	190 nH
Resistance(r)	0.3 Ω
Q-factor, $Q = \omega L/r$	47.7
Bandwidth, BW = f_c/Q	215.6 kHz

outside ring attached with gear through a belt. After mounting the magnet on the outside ring, the stepper can drive the magnet. The stepper was driven by a Sparkfun board [111] triggered by a TTL signal from the FPGA. The stepper moves after finishing acquiring data in one position.

Figure 5.6 illustrates the combination of rotating the magnet and moving the NMR probe to complete the field mapping. The resolution of the NMR probe movement is 2 mm. In each lap, 24 points were covered, i.e., 15 degrees in each rotation.

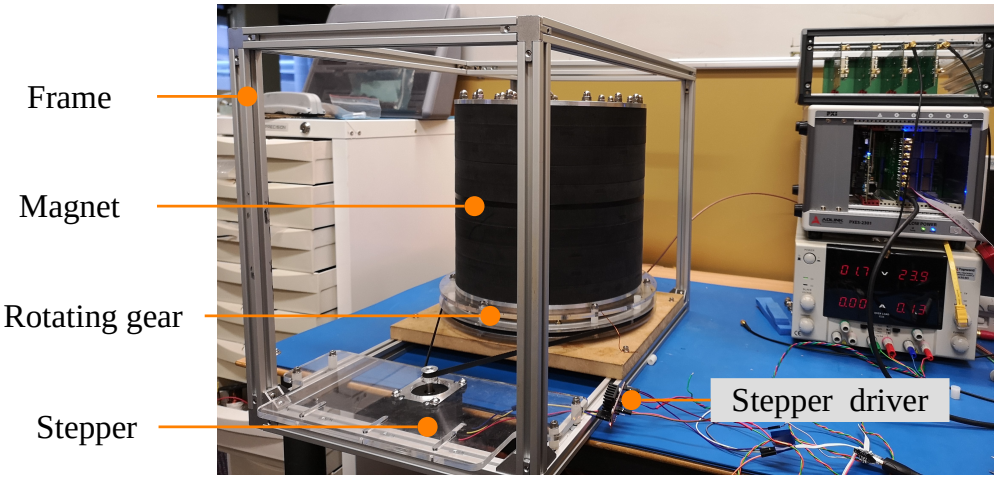


Fig. 5.4 The constructed frame for rotating magnet.

(a) (b) (c) (d)

Fig. 5.5 The components used for rotating the magnet. a. Rotating bearing plate. b. Belt. c. stepper. d. stepper driver.

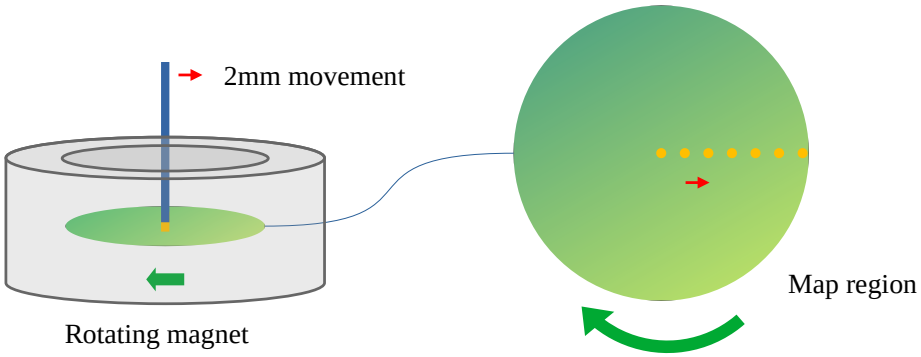


Fig. 5.6 Diagram of field map scan.

5.2.3 Frequency Calculation

Figure 5.7 shows the flow chart of frequency calculation. Firstly, the FID is acquired with typical parameters, such as the dwell time is 10 ns (100 MHz sampling rate), phase evolution calculation of the FID, and the frequency calculation based on the calculated phase. Figure 5.8 shows the FID signals and calculated phase evolution of the FID. The phase evolution can be expressed:

$$\phi(t) = \omega t + \phi_0 \tag{5.1}$$

where ω is related to the precession frequency, and ϕ_0 is the initial phase. In this equation, $\phi(t)$ can be obtained from the real and imaginary signal. Thus, ω can be calculated as it has a linear relationship with ϕt.

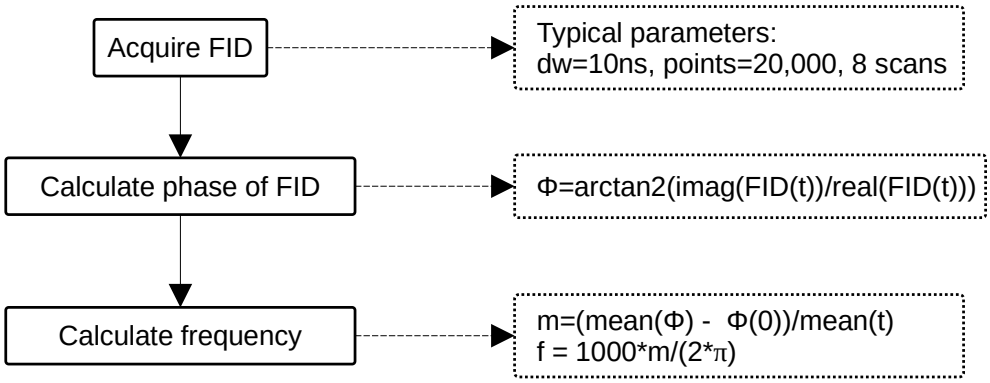
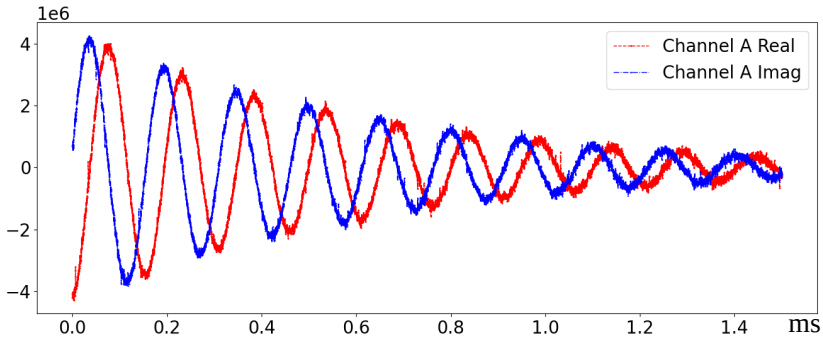


Fig. 5.7 Flow chart of frequency calculation.

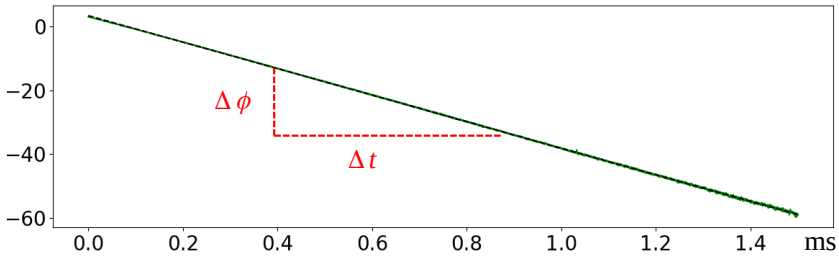
When the probe is located in a more inhomogeneous field, the FID signal will be shorter. But the frequency information of the FID still can be extracted by using the higher SNR signal, for example, the signal in 0-0.2 ms, as shown in Figure 5.9. Otherwise, the frequency calculation will not be accurate.

5.2.4 Temperature Drift

During field mapping, we found that the changing room temperature (about 2 degrees) significantly affects the field, i.e., the field strength drifts with temperature changes. Figure 5.10 shows the Larmor frequency offsets with a fixed excitation frequency (12.04 MHz) without



(a)



(b)

Fig. 5.8 FID phase calculation. (a). The real and imaginary FID signals. (b). Phase evolution plot.

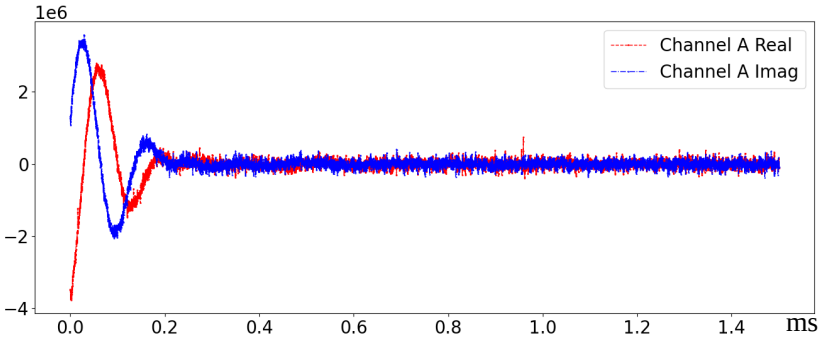


Fig. 5.9 Shorter FID.

the temperature controller unit. There are 24 points (the magnet rotated one lap) on each colored line, which means 24 points of fields are measured in a short time (in 2 minutes). After scanning one lap, it waited 4 hours to start another lap scan. In other words, the frequencies of the four points reveal the differences of the magnetic field in one particular position. The difference is about 4 kHz, which might lead to inaccurate magnetic fields for image reconstruction. Thus, pieces of foam were mounted on the surface of the frame, and a temperature controller (0.1 degree resolution) with a heater were added to make the magnet temperature stable. Once a target temperature is set, the controller will turn the heater on when the detected temperature is lower than the targeted temperature. Figure 5.11 shows the result from the temperature controlled environment. The offset frequency dropped from 4 kHz to about 1 kHz.

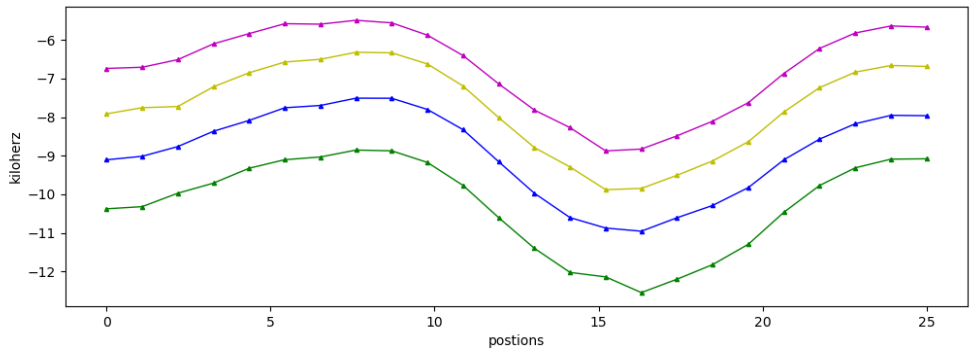


Fig. 5.10 Temperature drift of Larmor frequency without temperature controller. At the particular position, 24 points were measured by keeping the NMR probe stable rotating the magnet. Time gap is 4 hours between the adjacent lines.

5.2.5 Mapped Field of the Magnet

Figure 5.12 shows the plotted field map of the Halbach magnet.The constructed field map has the same center with the Halbach magnet. As shown in Figure 5.6, the probe was moved 2 mm away from the last position after one lap scan (24 points, angle, 15 degree). The map was created by the Python contour plot.

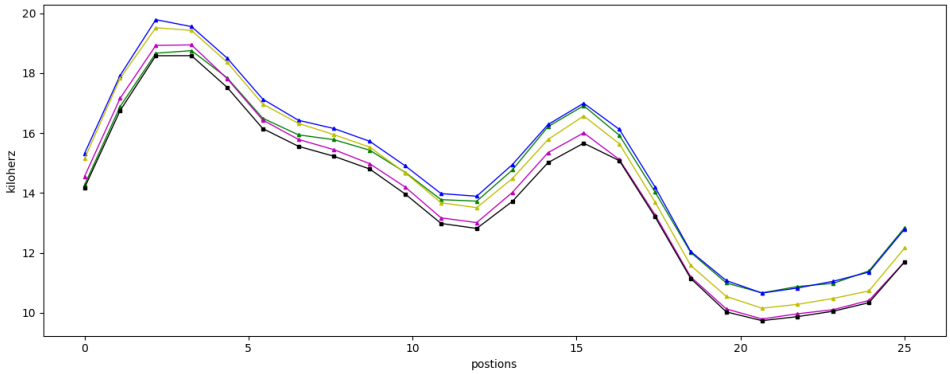


Fig. 5.11 Temperature drift with temperature controller. Measurement method is same to 5.10.

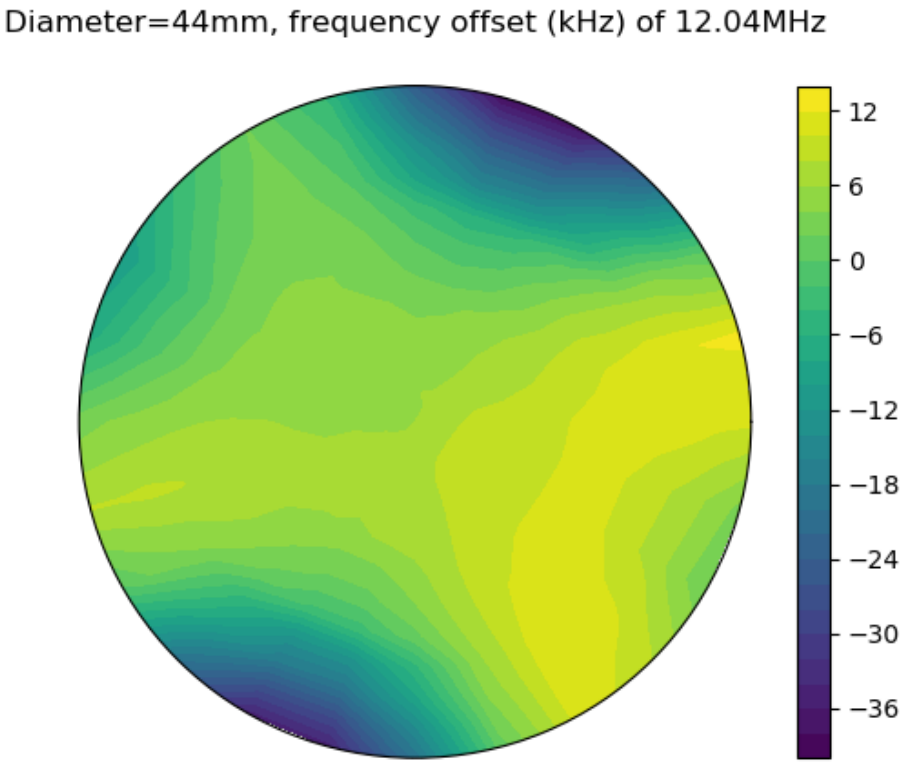


Fig. 5.12 Field mapping.

5.3 RF Coils

For this 2D imaging system, a single solenoid coil (outer layer) is used for transmitting and 4 surface coils (inside layer) were built for acquiring the signal simultaneously, as shown in Figure 5.1. The four-channel MR signal is amplified by four pre-amplifiers and digitized by a quad-channel ADC (AD9653). For that coil structure, coupling between the transmitter coil and receive coils and between receive coils is inevitable. To address the coupling problem, a strategy is to detune the receive coils to an off-resonant frequency during transmitting and detune the transmitter coil to an off-resonant frequency during receiving. To address the problem of coupling between receive coils, a geometrical decoupling method was adopted.

5.3.1 Transmitter Coil

Figure 5.13 illustrates a diagram of the transmitting coil. During transmitting, the high voltage can go across the parallel diodes. During receiving, the cross diodes are not conducting and break the resonance condition in the resonant circuit, which means that the transmitter part in not resonating at the Larmor frequency. That realizes detuning. During tuning the transmitter coil, the cross diodes are shorted manually. Figure 5.14 shows a picture of the transmitter coil. Table 5.2 shows the parameters of the transmitter coil:

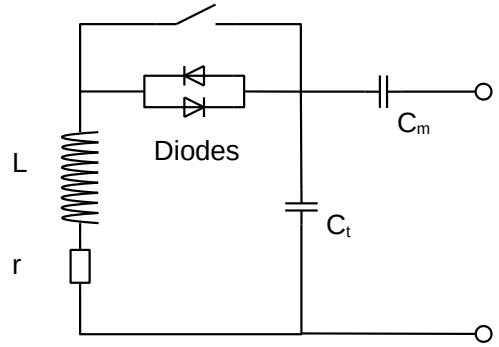


Fig. 5.13 Passive detuning in the transmitter coil [112].

Fig. 5.14 Transmitter coil.

5.3.2 Single Receive coil

Before making the receive coil array, a single receive coil was built for testing the decoupling effect in the coil system. Figure 5.15 shows the coil structure of the coil system; a single surface coil is attached to the surface of a receive coil frame.

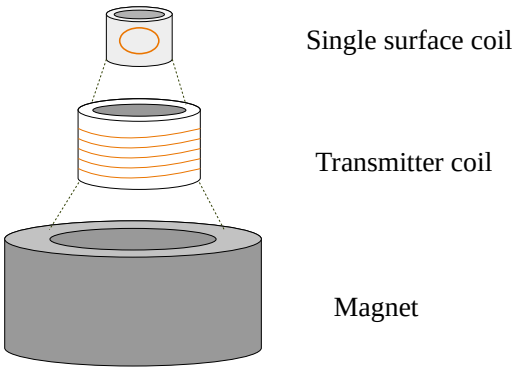


Fig. 5.15 The structure of the receive coil and transmitter coil.

Figure 5.16 shows the diagram of active detuning. During transmitting, a positive DC voltage is applied to turn on the PIN diode to make L_t and C_m as a parallel resonant circuit

Table 5.2 Transmitter coil parameters. The inductance and resistance of the coil are measured at 12 MHz. The inductance and the resistance were measured by an Agilent E4411B Network Analyzer.

Items	**Values**
Copper diameter	0.5 mm
Turns	8
Pitch	10 mm
Frame diameter	70 mm
Inductance	4 μH
Resistance	2.9 Ω
Q-factor, $Q = \omega L/r$	103.9
Bandwidth, BW = f_c/Q	115.5 KHz

which operates at Larmor frequency and has a high impedance that can protect the pre-amplifier from the inevitable coupled high voltage. The capacitor, C_b, is used to block the DC current from going to the pre-amplifier. During receiving, there is no DC voltage on PIN diode. Thus, L_t can be seen as a short circuit, and L, r, and serial capacitors, C_t and C_m, make the resonant circuit with which the whole resistor is divided to match 50 ohm.

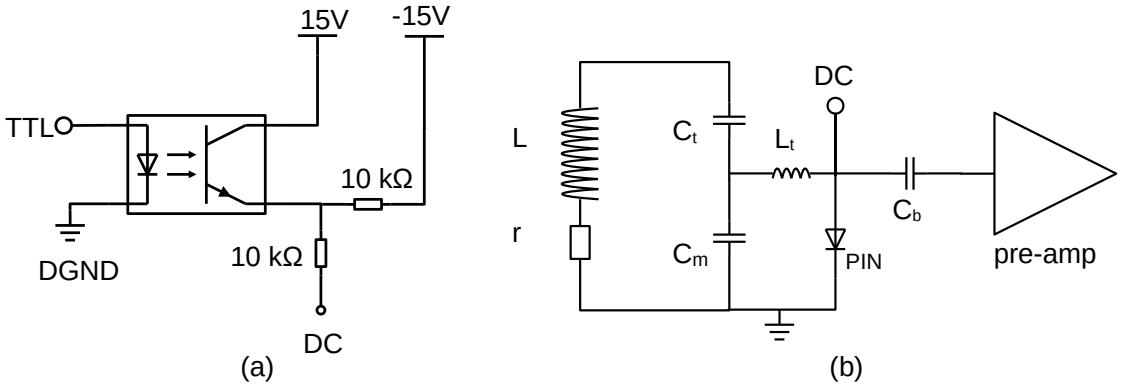


Fig. 5.16 Receiver coil detuning. (a) diagrams that an optocoupler is used to translate the TTL signal. Fig.(b) shows the passive detuning circuit between the coil and the pre-amplifier.

Figure 5.18 shows that the applied negative voltage significantly affected the result of the active detuning. As shown in Figure 5.18a, the high voltage lasts several milliseconds due

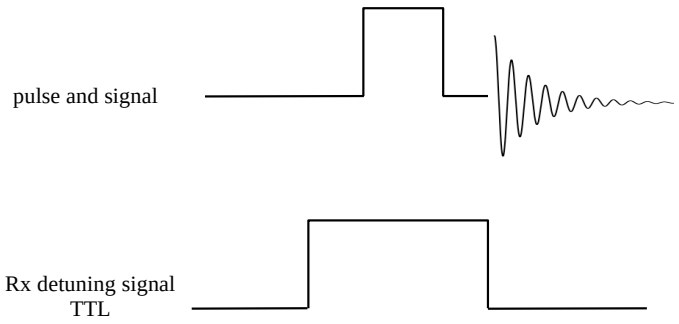
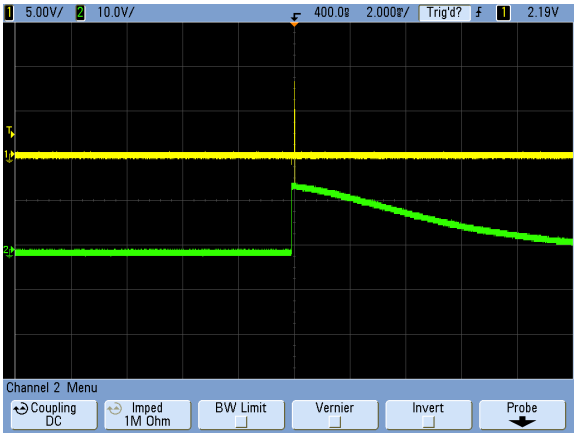
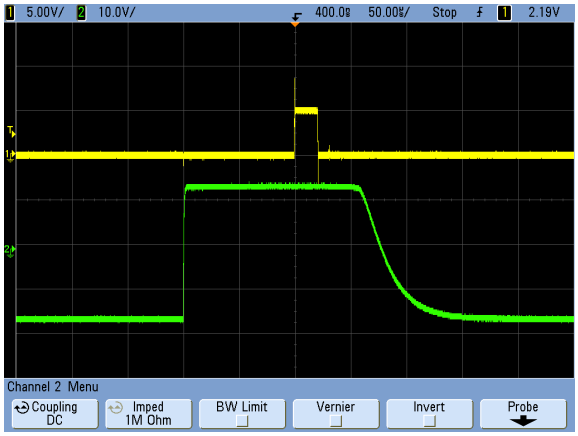


Fig. 5.17 Timeline for the pulse and receive detuning.



(a)



(b) When negative voltage is applied in the circuit in Figure 5.16

Fig. 5.18 Adopting a negative voltage to decrease the recovery time. (a) shows that when a negative voltage is not applied in the circuit in Figure 5.16. (b) shows when a negative voltage is applied in the circuit in Figure 5.16.

to the charging in the optocoupler, which could damage the FID signal acquisition. After applying the negative voltage, it discharged the optocoupler, which resulted in a fast falling edge (about 400 ns), as shown in Figure 5.18b.

5.3.3 Receive Coil Array

The coupling between the transmitter coil and single receive coil is addressed by shifting one coil's resonant frequency. But for the receive coil array, mutual couplings exist between adjacent coils and opposite coils.

One way to tackle that problem is to use geometric decoupling [113], i.e., to overlap the loop of the adjacent coils with a certain area. The adjacent coils are overlapped for canceling the mutual inductance of the adjacent coils. Another method is to use a transformer, which is a part of the inductor but does not contribute to the receive magnetic field, to cancel the mutual decoupling between the two coupling coils. The latter one could be more geometrically flexible than the overlapping one. However, four coils are needed to match the quad-channel receiver in this application. With a certain dimension of a coil array frame, the coil size will be too large compared with the sample size if we try to cancel the mutual coupling between the adjacent coils by overlapping. Also, the large coil dimension makes the coupling of the opposite coils considerable, making it difficult to tune the coils to the Larmor frequency.

Eventually, we decided to make a smaller rectangle array and use a transformer decoupling method between the two adjacent coils. Figure 5.19 shows the rectangle coil (left) and the transformer for decoupling. The windings in the transformer are extended from the two adjacent surface coils separately. Table 5.3 lists the parameters of one of the receive coils.

Figure 5.20 shows the coil tuning with the network analyzer (Agilent E4411B), which has two ports. During tuning, the unconnected coils are terminated with 50 ohm impedance. Figure 5.21 shows the S11, S22, and S12 of the two adjacent coils without proper decoupling. Figure 5.22 shows S11, S22, and S12 of the two adjacent coils with proper decoupling. In Figure 5.19, Figure 5.21, Figure 5.22, an Agilent E4411B Network Analyzer was used for the measurements.

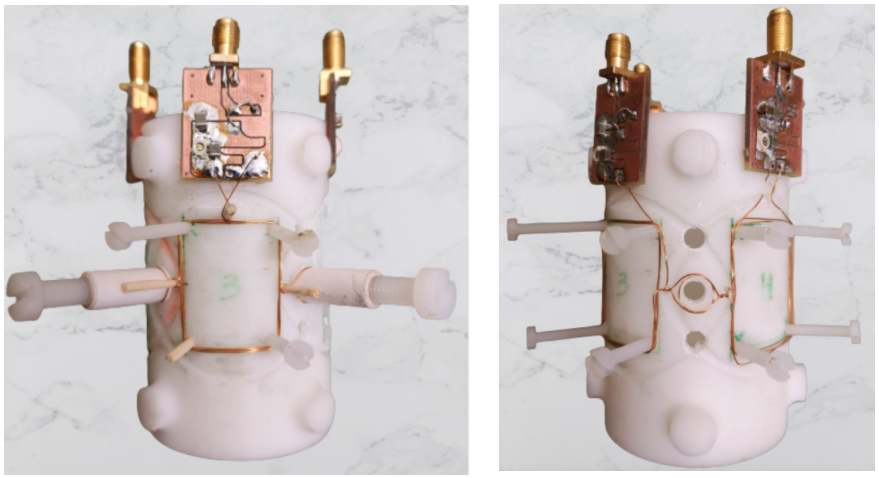


Fig. 5.19 Receive coil array.

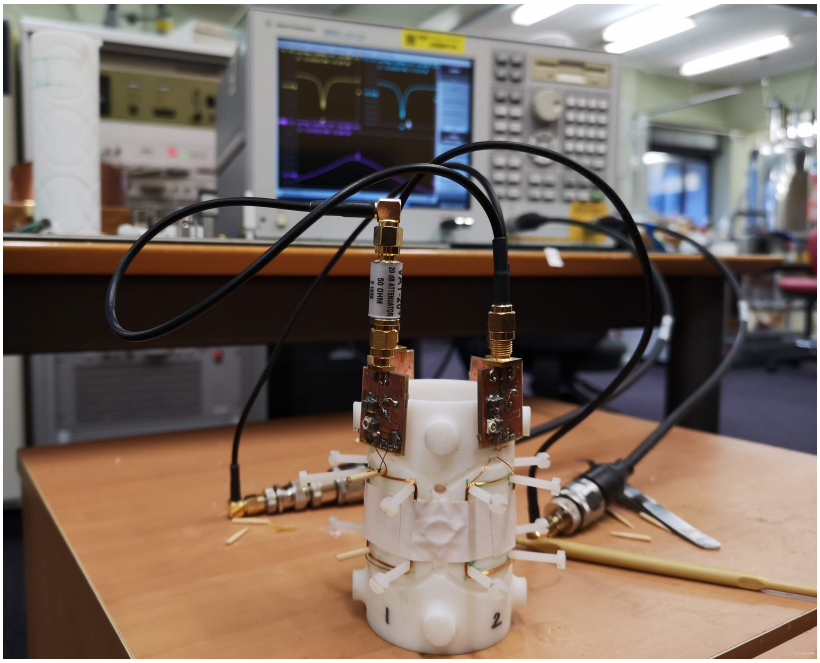


Fig. 5.20 Coil tuning with the network analyzer.

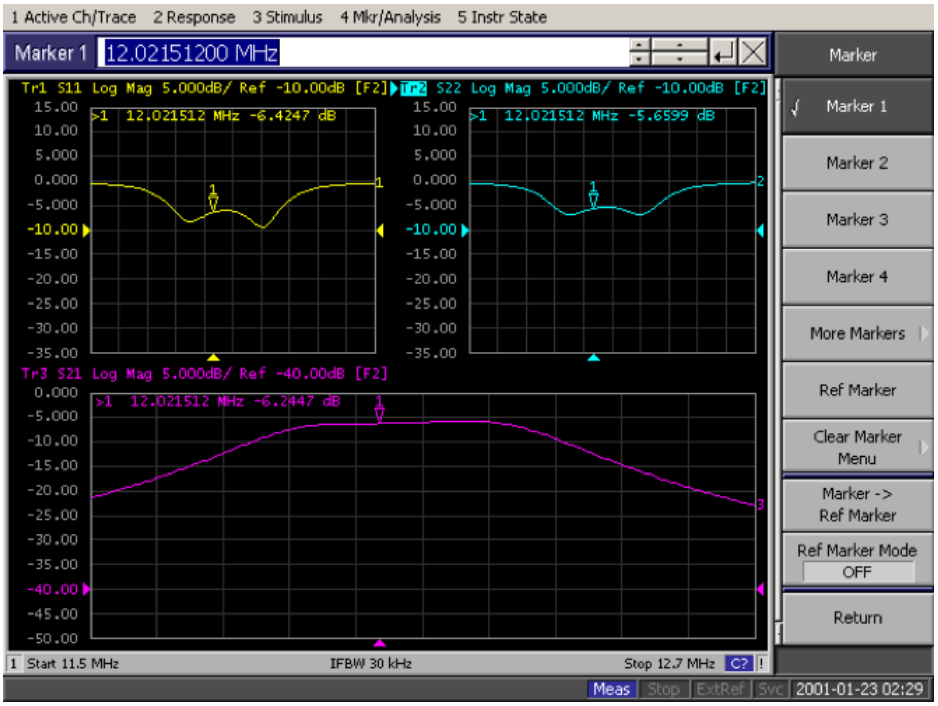


Fig. 5.21 Adjacent coils without proper decoupling.

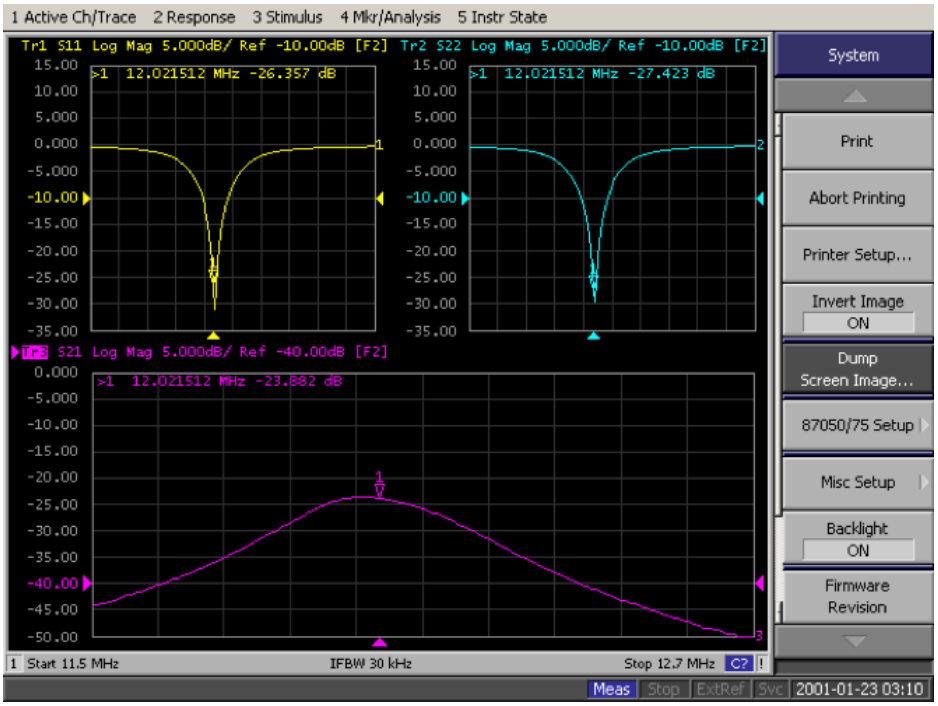


Fig. 5.22 Adjacent coils with proper decoupling.

Table 5.3 Surface receive coil parameters. The inductance and resistance of the coil are measured at 12 MHz. The inductance and the resistance were measured by an Agilent E4411B Network Analyzer.

Items	**Values**
Copper diameter	0.35 mm
Turns	3
Dimension	25 mm * 30 mm
Inductance	720 nH
Resistance	1 Ω
Q-factor, $Q = \omega L/r$	54.3
Bandwidth, BW = f_c/Q	222 kHz

For the coupling of the two opposite coils, the coupling is around -9 dB. However, the two independent coils can be tuned individually. i.e., tuning one coil does not change the RF response of the other one. Practically, we understand that one coil senses the magnetization and induces a current in its own coil, creating current on the opposite coil because of the mutual coupling. In this way, the current produced on one coil due to the mutual coupling creates a dummy signal. To make the receive coil array simple, we decided to deal with that dummy signal in the post signal processing rather than adding an extra coupling transformer, which could complicate the coil design.

An alternative way of decoupling receive array employs pre-amplifier decoupling [113]. That method works by reducing current flowing, which is the essential factor of inter-coil interaction, in the individual coil. To achieve that, a very low input impedance (1 ohm or less in real part) is used for the pre-amplifier while maintaining the 50 ohm (impedance for better noise figure) at the input to the preamplifier. However, compared with modifying preamplifier design, it is easier to implement geometrical decoupling.

5.3.4 S-parameter of the RF Coil System

Figure 5.23, 5.24, 5.25, 5.26, 5.27 show the S-parameters of the RF coil system after applying the detuning and decoupling strategies. S11 of the individual coils and the S12 of the transmit coil and receive coils were measured.

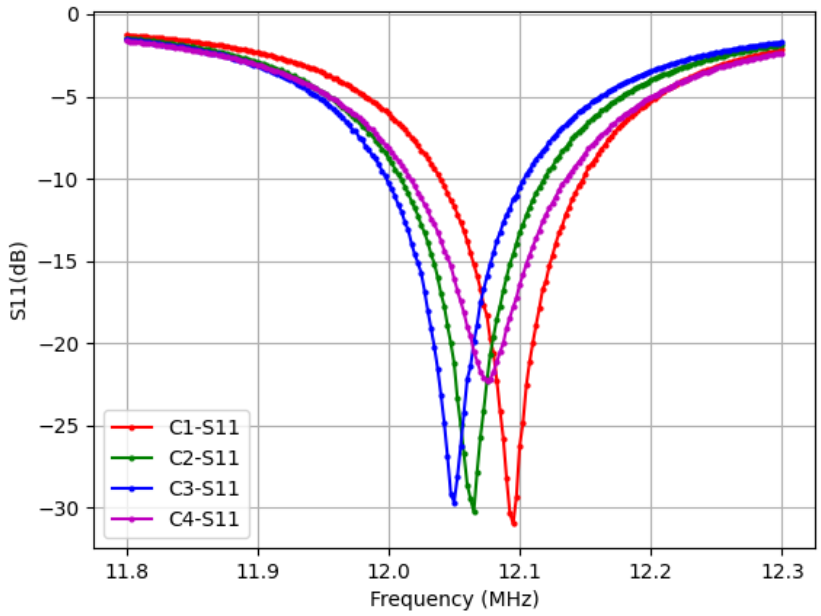


Fig. 5.23 S11 of the four surface coils in the coil array.

5.4 Coil Sensitivity Calculation

Coil sensitivity is the ability of the coil to sense the flux change which is produced by the change of magnetization during precessing. The sensitivity of the surface coil would change when the B_0 direction changes.

As shown in Figure 5.28, when the coil is in position 1, the coil is parallel to the direction of the magnetic field. Then the magnetization would precess around B_0. The projection of magnetization on the circular platform contributes to the signal in the coil. In other words, Bz, orthogonal to the direction of B_0, is the only field that contributes to the signal induced in the coil. When the coil moves its position from 1 to 2, the coil sensitivity changes due to

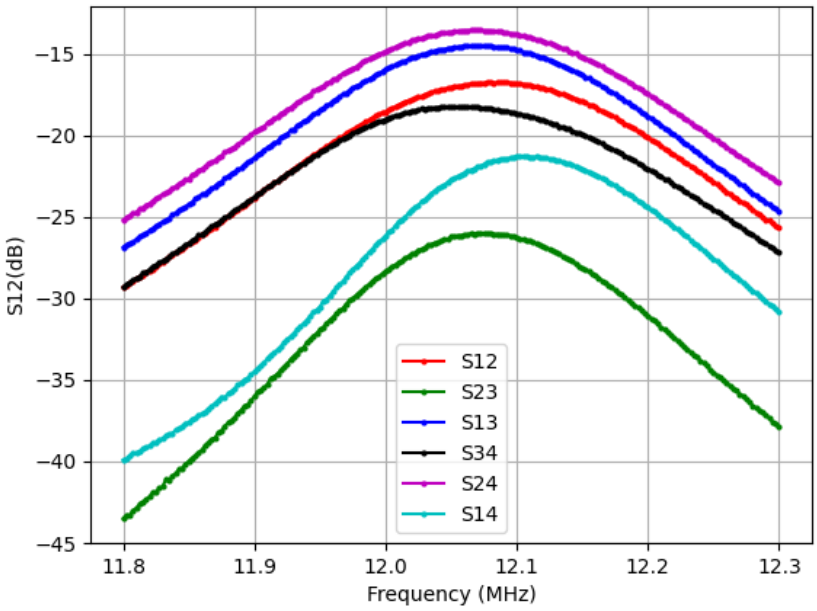


Fig. 5.24 S12 between two coils in the coil array. The two stronger couplings refer to the two pairs of the opposite coils.

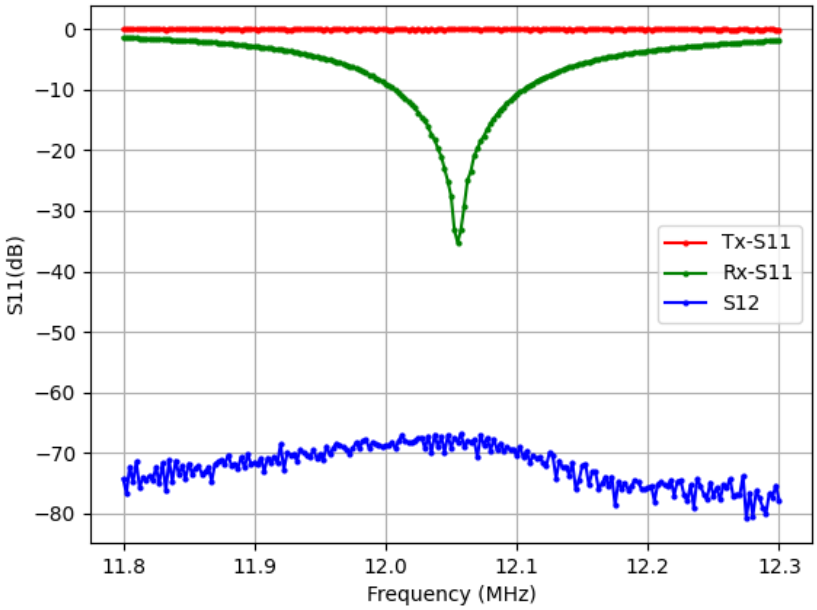


Fig. 5.25 In receive mode, the red line shows the S11 of the transmitter coil, the green curve is the S11 of the receive coil, and the blue curve shows the S12 of the two coils.

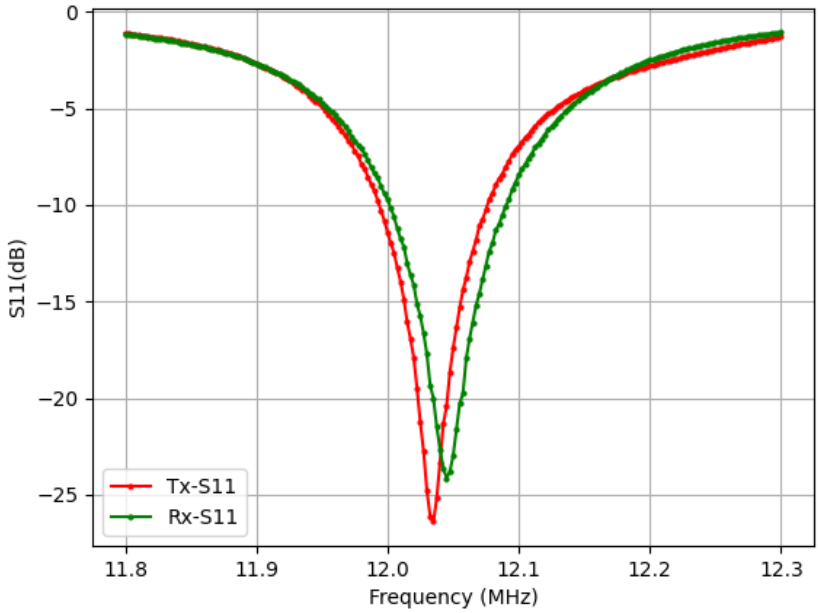


Fig. 5.26 S11 of the transmitter coil when receive coil is tuned or not.

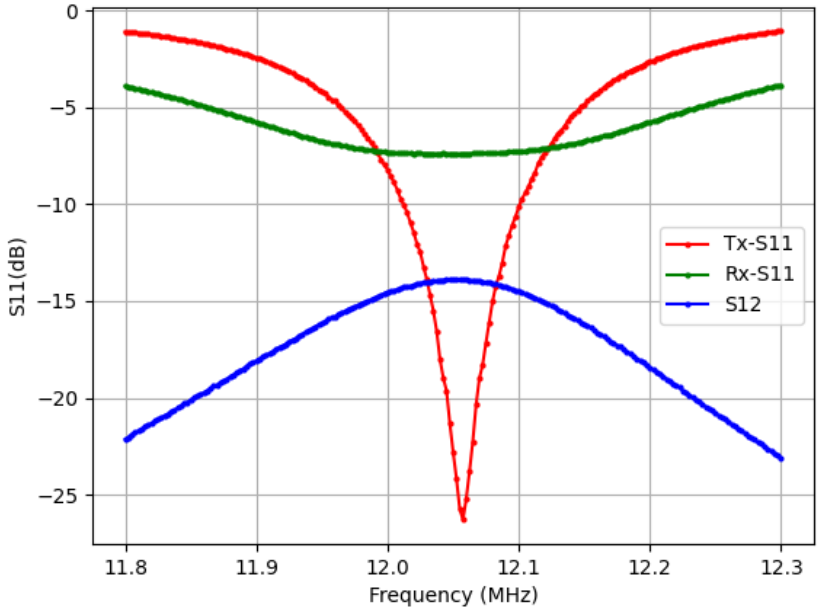


Fig. 5.27 In transmitting mode, the red curve shows the S11 of the transmitter coil, the green curve is the S11 of the receive coil, and the blue curve shows the S12 of the two coils.

the angle between the coil and the direction of B_0. As the figure illustrates, the field which contributes the signal in the coil becomes the sum of B'_z and B'_r, which are the projection of B_z and B_r. B_z and B_r can be calculated by the Biot-Sarvart law if the dimension of the coil is known. Then, when the angle is ϕ, the effective coil sensitivity becomes:

$$B_{effctive} = B'_z - B'_r = B_z cos\phi - B_r sin\phi \tag{5.2}$$

In the practical situation, the magnet rotates, which means the direction of B_0 changes. But the calculation result of the coil sensitivity is the same because the angle between the coil and direction of B_0 matters.

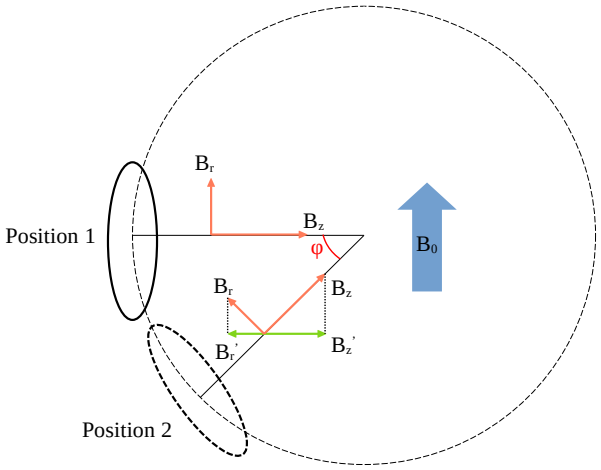


Fig. 5.28 Coil sensitivity changes due to the direction of B_0 change.

Figure 5.29 shows the contour plot of the coil sensitivity calculation. The differences in the contour plots in the six figures reveal that the coil sensitivity changes when the angle between the coil and the direction of the B_0 changes. When the coil is parallel to B_0, the coil sensitivity is maximum. When the coil is orthogonal to B_0, the coil sensitivity is minimum.

5.5 Signal Capture and Verification

Figure 5.30 shows a picture of the 2D imaging system. The left diagram in Figure 5.31 indicates the sample position and dimensions of the magnet and the transmitter/receive coil

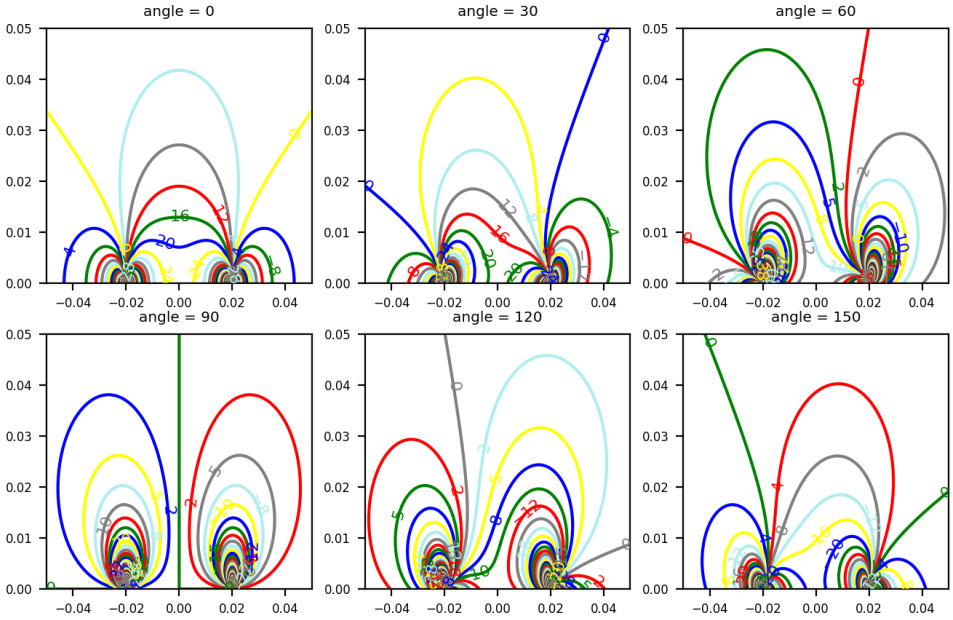


Fig. 5.29 Coil sensitivity contour plot. The six figures show that the coil sensitivity changes when the angle between coil and the direction of the B_0 changes. When the coil is parallel to B_0 , the coil sensitivity is maximum. When the coil is orthogonal to B0, the coil sensitivity is minimum.

frames; and the right picture shows the picture of the copper sulfate sample, of which the depth is 10 mm.

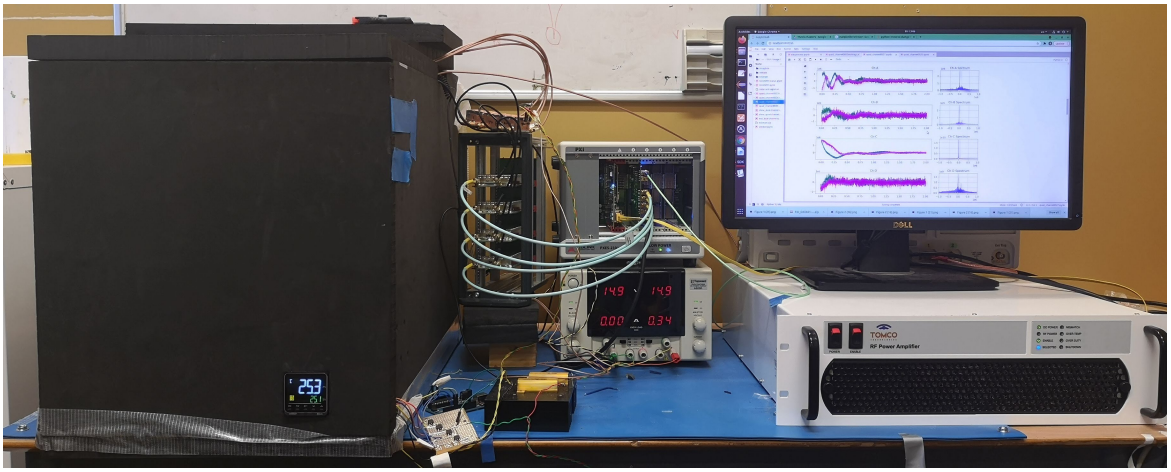


Fig. 5.30 Picture of the 2-D imaging system.

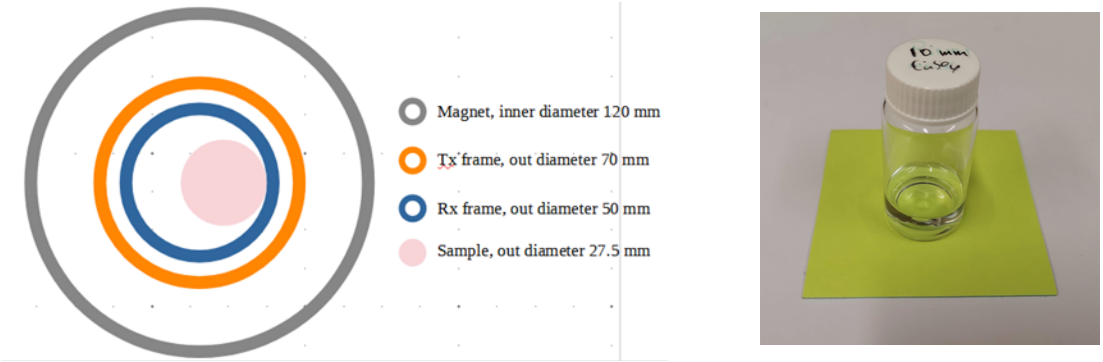


Fig. 5.31 Sample position and the copper sulfate sample.

The FID experiment was conducted for signal capture and verification of the 2D imaging system. The pulse width of the FID was 16 us, and the repetition time was 1 second. Similar to the field mapping experiment, at each point, eight scans were run for signal averaging. The magnet rotated 24 times to cover a lap, which resulted in 15 degrees for each rotation. Figure 5.32 shows the captured FID signals from four individual receive channels at one position. It can be seen that the amplitude of the FID signals from channel A and C are larger than the other two channels. The reason is that coil A and C are more sensitive to the

sample in the situation that more magnetization flux can go through these coils, as indicated in Figure 5.29. The right column in Figure 5.32 shows the spectra of the individual FID signal. The peaks reveal that spins in different locations contribute to the FID signal.

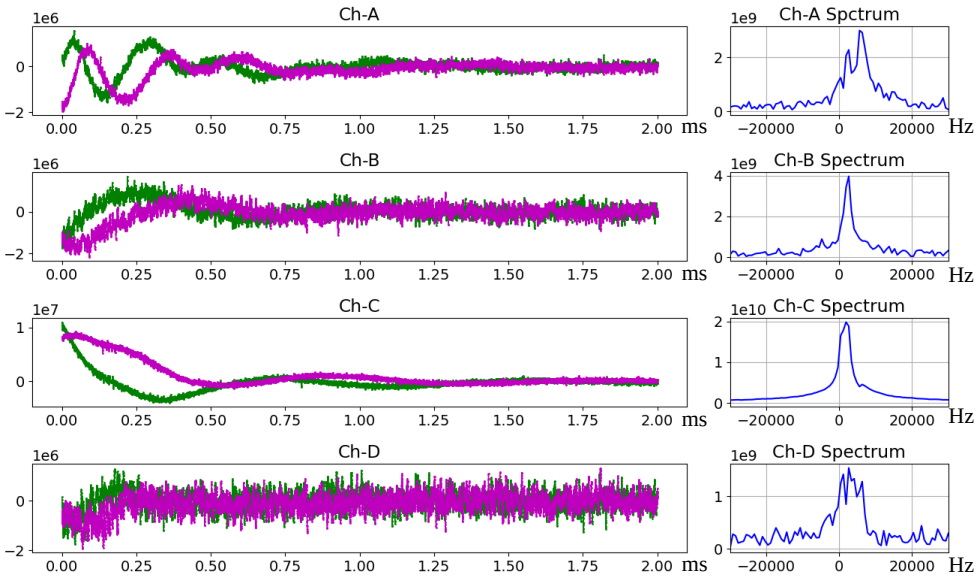


Fig. 5.32 FID and its spectra from the four receive channels. The spectra on the right column was calculated by Fourier Transform basing on the FID.

5.6 Image Reconstruction

The natural spatial encoding of a Halbach array produces a nonbijective mapping in the field-of-view, which leads to aliasing in the image. As proposed by Hennig et al [36], the ambiguity can be eliminated by using the profiles of the receive coil as it provides additional spatial encoding that localizes the signal within each source. According to the descriptions by Stockmann et al [114] and Cooley et al [3], the iterative matrix methods can be used for image reconstruction.

At a particular position and time, the discrete signal acquired by a particular coil can be described as

$$S_{q,r}(n) = \sum C_{q,r}(x) e^{-i2\pi k(r,x,n)} m(x) \tag{5.3}$$

In the equation, $C_{q,r}$ is the sensitivity of the coil q at position x, $2\pi k$ is the phase related to the non-linear gradient field at rotation r, position x and time n. m(x) is the magnetization of the object at position x to be resolved. The summation of x is the net magnetization of the sample. The exponential part and coil sensitivity are the known parameters, thus they can be grouped. The new expression can be

$$S_{q,r}(n) = \sum enc_{q,r}(x,n)m(x) \tag{5.4}$$

where the $enc_{q,r}(x,t)$ is grouped for phase term and coil sensitivity to form the encoding function.

The matrix form of that equation can be simplified as

$$\mathbf{S}_{q,r} = \mathbf{E}_{q,r}(x,n)\mathbf{m} \tag{5.5}$$

where **E** contains the predicted phase of each voxel in the image field for each time point in the acquisition. **m** is the magnetization to be solved.

With the acquired signal,$S_{q,r}$, and the encoding matrix, $E_{q,r}$, the **m** made up of all the image voxels can be solved.

Figure 5.33 shows the reconstructed the image. The highlight area relatively matches the sample position in Figure 5.31. The main reason for the "rough" image is that the magnetic field produced by the Halbach magnet which was not originally designed for image. With the current spectrometer and the front-end electronics, image can be improved by using a better magnet that is intended for MRI imaging.

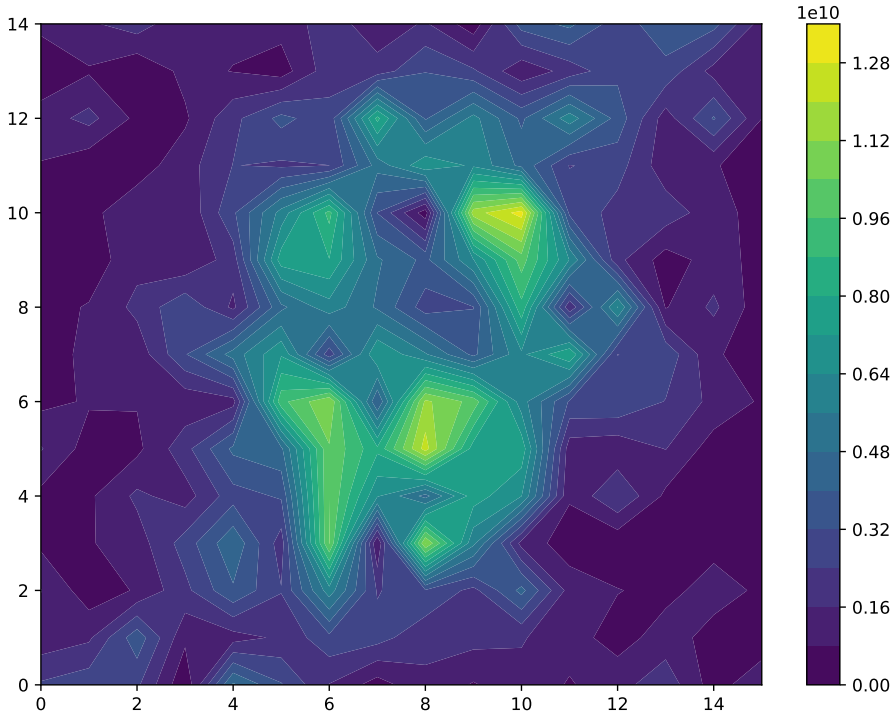


Fig. 5.33 Reconstructed Image.

Chapter 6

Conclusion and Future Work

This dissertation has presented the story of the engineering development of the 2D MRI system, starting with the multichannel transceiver manufacturing and its integration with the system controller, NMR testing, and MRI application. The last chapter will discuss the contribution, consider the unsolved problems, and propose the outlook in this field.

6.1 Contributions

Multichannel Transceiver

The multichannel transceiver board with the quad-channel receiver and dual-channel transmitter is the main contribution to the spectrometer. An eight-layer PCB board was designed, and all the components were assembled manually. After implementing the components, several FPGA projects were created to verify the individual implementations, such as the transmitting by the DAC, the receiving by the ADC, etc. A pulse program generator was developed to run the NMR related experiments, e.g., FID experiment. In the FPGA firmware implementation, the data movement unit for transferring the stream data to memory-mapped data was added. Synchronization of transmitting and receiving was verified by loop testing.

Spectrometer Integration

Spectrometer integration allows users to control the experiment and see the results on the user interface on the host computer. Spectrometer hardware integration was completed by implementing the PCIe-CDMA subsystem for performing data transfer between PCIe and AXI4 realms in the FPGA on the transceiver board. The software architecture was developed to address the problems from the hardware integrated spectrometer. A user interface on the host computer was developed to manage the NMR experiments, such as set up parameters, operate a specific pulse program, etc.

2D MRI System

Based on the spectrometer, a 2D MRI system was constructed. In addition to the spectrometer, other necessary parts were prepared or developed. The Halbach was designed in this NMR lab years ago. It was an early prototype for another project so was not designed specifically for this project. The high power amplifier is a commercial one. Four pre-amplifiers (with duplexers) and a backplane for mounting these pre-amplifiers were constructed. The pre-amplifiers controlled by TTL signals from the FPGA have variable gains. The RF coil system, including solenoid transmitter coil, receive coil array (four surface coils), and detuning and decoupling circuits, was constructed and verified. A rotating frame was constructed for rotating the magnet for field mapping and 2D imaging.

The center layer field of the Halbach magnet was mapped by using NMR magnetometry, which adopts a small NMR probe for excitation and acquisition. Based on the 2D imaging system, FID experiments were conducted and parallel acquisition with four receive channels was performed. The FID data was collected for image reconstruction. However, an image was unable to be reconstructed.

There are several concerns about the data accuracy of the field-mapping. Firstly, the NMR probe was moved manually without a rail, which might cause errors, i.e., the measured field was not matching the assumed position. Secondly, the diameter of the NMR probe is around 3 mm, which might not reflect the real field at the position. Therefore, a 3D printed rail attached to the rotating frame can be applied to move the NMR probe, and a smaller

NMR probe (diameter about 1 mm) could be created to improve the data accuracy of the field-mapping. Currently, the optocoupler circuit for supplying the DC voltage for receiver detuning was implemented on an additional circuit board. That circuit can be added to the pre-amplifier board to simplify the RF cable connection.

Third-Dimension Encoding

The dual-channel transmitter implemented in the transceiver board is designed to perform the third dimension encoding. At the beginning of this project, TRASE (transmitter array spatial encoding) [37, 38] was proposed to be applied in this study to achieve a 3D imaging system. The advantage of the two independent transmitters is that the phase and frequency are configurable in each transmitter.

Unfortunately, due to COVID, the plan had to be changed to build our own system rather than use the system in Singapore. This meant there was not enough time to implement TRASE encoding of the 3rd dimension.

6.1.1 Outlook

Low-field MRI stimulates interest to researchers and doctors because of the potential use of MRI. The presented spectrometer in this thesis will contribute to the community to empower other researchers without experience in spectrometer design to build their own MRI system.

Lastly, FPGA devices allows the development of sophisticated and compact solutions. Thanks to the performance improvement of the FPGA over time, such as more resources, more powerful computation ability, and the data throughput. It can act as a computing platform for image acceleration. Therefore, it would be a key technology to achieve low-cost and to develop more compact MRI applications in the future. Because that the FPGA module were used on the PXIe boards, it is easy to replace better FPGA modules on the boards without redesigning the whole system.

6.2 Future Work

The presented multichannel transceiver spectrometer showed its capability for low MRI system, however, a few aspects can be improved.

1. Currently, the decimation rate in the CIC Compiler is fixed, 10:1. So it can be optimized to be configurable.

2. The maximum data package of one transmission between the system controller and the transceiver is about 8 MB. In the future, the PCIe driver can be improved to increase the amount of single data transmission.

3. The user interface can be upgraded to be more user friendly.

For the system, a designated magnet for MRI can be used with this MRI spectrometer. In addition, gradient unit and shim unit board can be designed to be integrated to the current spectrometer.

References

[1] S. Geethanath and J. T. Vaughan Jr, "Accessible magnetic resonance imaging: A review," *Journal of Magnetic Resonance Imaging*, vol. 49, no. 7, pp. e65–e77, 2019.

[2] Hyperfine, *Portable MR imaging system*. Available at http://hyperfine.io/.

[3] C. Z. Cooley, J. P. Stockmann, B. D. Armstrong, M. Sarracanie, M. H. Lev, M. S. Rosen, and L. L. Wald, "Two-dimensional imaging in a lightweight portable MRI scanner without gradient coils," *Magnetic resonance in medicine*, vol. 73, no. 2, pp. 872–883, 2015.

[4] Z. H. Ren, L. Maréchal, W. Luo, J. Su, and S. Y. Huang, "Magnet array for a portable magnetic resonance imaging system," in *2015 IEEE MTT-S 2015 International Microwave Workshop Series on RF and Wireless Technologies for Biomedical and Healthcare Applications (IMWS-BIO)*, pp. 92–95, IEEE, 2015.

[5] T. O'Reilly, W. Teeuwisse, and A. Webb, "Three-dimensional MRI in a homogenous 27 cm diameter bore halbach array magnet," *Journal of Magnetic Resonance*, vol. 307, p. 106578, 2019.

[6] Y. Liu, A. T. Leong, Y. Zhao, L. Xiao, H. K. Mak, A. C. O. Tsang, G. K. Lau, G. K. Leung, and E. X. Wu, "A low-cost and shielding-free ultra-low-field brain MRI scanner," *Nature communications*, vol. 12, no. 1, pp. 1–14, 2021.

[7] B. Keil, J. N. Blau, S. Biber, P. Hoecht, V. Tountcheva, K. Setsompop, C. Triantafyllou, and L. L. Wald, "A 64-channel 3T array coil for accelerated brain MRI," *Magnetic resonance in medicine*, vol. 70, no. 1, pp. 248–258, 2013.

[8] S. Orzada, K. Solbach, M. Gratz, S. Brunheim, T. M. Fiedler, S. Johst, A. K. Bitz, S. Shooshtary, A. Abuelhaija, M. N. Voelker, *et al.*, "A 32-channel parallel transmit system add-on for 7T MRI," *Plos one*, vol. 14, no. 9, p. e0222452, 2019.

[9] A. Graessl, W. Renz, F. Hezel, M. A. Dieringer, L. Winter, C. Oezerdem, J. Rieger, P. Kellman, D. Santoro, T. D. Lindel, *et al.*, "Modular 32-channel transceiver coil array for cardiac MRI at 7.0 T," *Magnetic resonance in medicine*, vol. 72, no. 1, pp. 276–290, 2014.

[10] *Robin Dykstra*. https://people.wgtn.ac.nz/robin.dykstra.

[11] A. Ang, "Development of an open PXIe system based on FPGA modules," 2018.

[12] A. Ang and R. Dykstra, "An open source PXIe platform for instrumentation development," in *2019 IEEE International Instrumentation and Measurement Technology Conference (I2MTC)*, pp. 1–5, IEEE, 2019.

[13] *Paul Teal*. https://people.wgtn.ac.nz/paul.teal.

[14] P. T. Callaghan, *Principles of nuclear magnetic resonance microscopy*. Oxford University Press on Demand, 1993.

[15] R. W. Brown, Y.-C. N. Cheng, E. M. Haacke, M. R. Thompson, and R. Venkatesan, *Magnetic resonance imaging: physical principles and sequence design*. John Wiley & Sons, 2014.

[16] E. L. Hahn, "Spin echoes," *Physical review*, vol. 80, no. 4, p. 580, 1950.

[17] I. L. Pykett, J. H. Newhouse, F. S. Buonanno, T. J. Brady, M. R. Goldman, J. P. Kistler, and G. M. Pohost, "Principles of nuclear magnetic resonance imaging.," *Radiology*, vol. 143, no. 1, pp. 157–168, 1982.

[18] M. A. Brown and R. C. Semelka, *MRI: basic principles and applications*. John Wiley & Sons, 2011.

[19] K. P. Pruessmann, M. Weiger, M. B. Scheidegger, and P. Boesiger, "Sense: sensitivity encoding for fast MRI," *Magnetic Resonance in Medicine: An Official Journal of the International Society for Magnetic Resonance in Medicine*, vol. 42, no. 5, pp. 952–962, 1999.

[20] D. K. Sodickson and W. J. Manning, "Simultaneous acquisition of spatial harmonics (SMASH): fast imaging with radiofrequency coil arrays," *Magnetic resonance in medicine*, vol. 38, no. 4, pp. 591–603, 1997.

[21] M. A. Griswold, P. M. Jakob, R. M. Heidemann, M. Nittka, V. Jellus, J. Wang, B. Kiefer, and A. Haase, "Generalized autocalibrating partially parallel acquisitions (GRAPPA)," *Magnetic Resonance in Medicine: An Official Journal of the International Society for Magnetic Resonance in Medicine*, vol. 47, no. 6, pp. 1202–1210, 2002.

[22] D. W. McRobbie, E. A. Moore, M. J. Graves, and M. R. Prince, *MRI from Picture to Proton*. Cambridge university press, 2017.

[23] D. Weishaupt, V. D. Köchli, B. Marincek, J. M. Froehlich, D. Nanz, and K. P. Pruessmann, *How does MRI work?: an introduction to the physics and function of magnetic resonance imaging*, vol. 2. Springer, 2006.

[24] C. Z. Cooley, P. C. McDaniel, J. P. Stockmann, S. A. Srinivas, S. F. Cauley, M. Śliwiak, C. R. Sappo, C. F. Vaughn, B. Guerin, M. S. Rosen, *et al.*, "A portable scanner for magnetic resonance imaging of the brain," *Nature Biomedical Engineering*, pp. 1–11, 2020.

[25] F. J. Mateen, C. Z. Cooley, J. P. Stockmann, D. R. Rice, A. C. Vogel, and L. L. Wald, "Low-field portable brain MRI in CNS demyelinating disease," *Multiple Sclerosis and Related Disorders*, vol. 51, p. 102903, 2021.

[26] S. S. Bhat, T. T. Fernandes, P. Poojar, M. da Silva Ferreira, P. C. Rao, M. C. Hanumantharaju, G. Ogbole, R. G. Nunes, and S. Geethanath, "Low-field MRIof stroke: Challenges and opportunities," *Journal of Magnetic Resonance Imaging*, p. e27324, 2020.

[27] J. P. Marques, F. F. Simonis, and A. G. Webb, "Low-field MRI: An MR physics perspective," *Journal of magnetic resonance imaging*, vol. 49, no. 6, pp. 1528–1542, 2019.

[28] L. L. Wald, P. C. McDaniel, T. Witzel, J. P. Stockmann, and C. Z. Cooley, "Low-cost and portable MRI," *Journal of Magnetic Resonance Imaging*, vol. 52, no. 3, pp. 686–696, 2020.

[29] K. Halbach, "Design of permanent multipole magnets with oriented rare earth cobalt material," *Nuclear instruments and methods*, vol. 169, no. 1, pp. 1–10, 1980.

[30] C. Z. Cooley, M. W. Haskell, S. F. Cauley, C. Sappo, C. D. Lapierre, C. G. Ha, J. P. Stockmann, and L. L. Wald, "Design of sparse halbach magnet arrays for portable MRI using a genetic algorithm," *IEEE transactions on magnetics*, vol. 54, no. 1, pp. 1–12, 2017.

[31] Z. H. Ren, W. C. Mu, and S. Y. Huang, "Design and optimization of a ring-pair permanent magnet array for head imaging in a low-field portable MRI system," *IEEE Transactions on Magnetics*, vol. 55, no. 1, pp. 1–8, 2018.

[32] Z. H. Ren, J. Gong, and S. Y. Huang, "An irregular-shaped inward-outward ring-pair magnet array with a monotonic field gradient for 2D head imaging in low-field portable MRI," *IEEE Access*, vol. 7, pp. 48715–48724, 2019.

[33] D. I. Hoult and R. Richards, "The signal-to-noise ratio of the nuclear magnetic resonance experiment," *Journal of Magnetic Resonance (1969)*, vol. 24, no. 1, pp. 71–85, 1976.

[34] J. T. Vaughan and J. R. Griffiths, *RF coils for MRI*. John Wiley & Sons, 2012.

[35] A. G. Webb, *Magnetic resonance technology: hardware and system component design*. Royal Society of Chemistry, 2016.

[36] J. Hennig, A. M. Welz, G. Schultz, J. Korvink, Z. Liu, O. Speck, and M. Zaitsev, "Parallel imaging in non-bijective, curvilinear magnetic field gradients: a concept study," *Magnetic Resonance Materials in Physics, Biology and Medicine*, vol. 21, no. 1-2, p. 5, 2008.

[37] Q. Deng, S. B. King, V. Volotovskyy, B. Tomanek, and J. C. Sharp, "B_1 transmit phase gradient coil for single-axis TRASE RF encoding," *Magnetic resonance imaging*, vol. 31, no. 6, pp. 891–899, 2013.

[38] J. C. Sharp, S. B. King, Q. Deng, V. Volotovskyy, and B. Tomanek, "High-resolution MRI encoding using radiofrequency phase gradients," *NMR in Biomedicine*, vol. 26, no. 11, pp. 1602–1607, 2013.

[39] P. Bohidar, H. Sun, G. E. Sarty, and J. C. Sharp, "TRASE 1D sequence performance in imperfect B_1 fields," *Journal of Magnetic Resonance*, vol. 305, pp. 77–88, 2019.

[40] H. Sun, S. Yong, and J. C. Sharp, "The twisted solenoid RF phase gradient transmit coil for TRASE imaging," *Journal of Magnetic Resonance*, vol. 299, pp. 135–150, 2019.

[41] H. Sun, A. AlZubaidi, A. Purchase, and J. C. Sharp, "A geometrically decoupled, twisted solenoid single-axis gradient coil set for TRASE," *Magnetic resonance in medicine*, vol. 83, no. 4, pp. 1484–1498, 2020.

[42] J. P. Stockmann, C. Z. Cooley, B. Guerin, M. S. Rosen, and L. L. Wald, "Transmit array spatial encoding (TRASE) using broadband wurst pulses for RF spatial encoding in inhomogeneous B_0 fields," *Journal of Magnetic Resonance*, vol. 268, pp. 36–48, 2016.

[43] E. Fukushima and S. B. Roeder, *Experimental pulse NMR: a nuts and bolts approach*. Addison-Wesley, 1981.

[44] C. A. Michal, K. Broughton, and E. Hansen, "A high performance digital receiver for home-built nuclear magnetic resonance spectrometers," *Review of scientific instruments*, vol. 73, no. 2, pp. 453–458, 2002.

[45] J. Zhen, C. Dykstra, G. Gouws, S. Obruchkov, and R. Dykstra, "Mobile low field magnetic resonance hardware development," *Journal of Magnetic Resonance*, vol. 322, p. 106852, 2021.

[46] K. Takeda, "A highly integrated FPGA-based nuclear magnetic resonance spectrometer," *Review of scientific instruments*, vol. 78, no. 3, p. 033103, 2007.

[47] K. Takeda, "Opencore NMR: Open-source core modules for implementing an integrated FPGA-based NMR spectrometer," *Journal of Magnetic Resonance*, vol. 192, no. 2, pp. 218–229, 2008.

[48] P. P. Stang, S. M. Conolly, J. M. Santos, J. M. Pauly, and G. C. Scott, "Medusa: a scalable MR console using USB," *IEEE transactions on medical imaging*, vol. 31, no. 2, pp. 370–379, 2011.

[49] L. L. W. Suma Anand1, Jason P. Stockmann and T. Witzel, "A low-cost (<$500 USD) FPGA-based console capable of real-time control," in *Proc Intl Soc Magn Reson Med, Paris*, p. 948, 2018.

[50] C. J. Hasselwander, Z. Cao, and W. A. Grissom, "gr-MRI: A software package for magnetic resonance imaging using software defined radios," *Journal of Magnetic Resonance*, vol. 270, pp. 47–55, 2016.

[51] Tecmag, "Tecmag MRI spectrometer." https://www.tecmag.com/bluestone/, 2008. [Online; accessed 19-July-2008].

[52] Magritek, *Kea2 Spectrometer*. Available at https://magritek.com/products/kea/.

[53] RS2D, "RS2D MRI Console." https://rs2d.com/en/oem/hardware/chameleon/, 2008. [Online; accessed 19-Feb-2021].

[54] National Instruments, *NI PXIe-1062Q User Manual*, 4 2012. 371843D-01.

[55] A. Wilen, J. Schade, and R. Thornburg, *Introduction to PCI Express*. Intel Press Santa Clara, 2003.

[56] P. Horowitz, W. Hill, and I. Robinson, *The art of electronics*, vol. 2. Cambridge university press Cambridge, 1989.

[57] D. RE, "Oversampled SAR ADC with PGA achieving greater than 125 dB dynamic range,"

[58] B. Brannon and A. Barlow, "Aperture uncertainty and ADC system performance," *Applications Note AN-501. Analog Devices, Inc.(September)*, 2000.

[59] W. Kester, "Aperture time, aperture jitter, aperture delay time-removing the confusion," *Analog Devices, MT-007 Tutorial*, 2008.

[60] W. Kester, "Which adc architecture is right for your application," in *EDA Tech Forum*, vol. 2, pp. 22–25, Citeseer, 2005.

[61] Analog Devices, *Quad, 16-Bit, 125 MSPS, Serial LVDS 1.8 V Analog-to-Digital Converter*, 5 2012. Rev. F.

[62] G. Diniz, "JESD204B vs. serial LVDS interface considerations for wideband data converter applications."

[63] Analog Devices, *Dual16-Bit, LVDS Interface, 500 MSPS DACs*, 11 2007. Rev. C.

[64] Xilinx Inc., *Spartan-6 Family Overview v2.0, DS160*. Available at https://www.xilinx.com, version v2.0.

[65] Xilinx Inc., *7 Series FPGAs Data Sheet Overview v2.61, DS180*. Available at https://www.xilinx.com, version v2.61.

[66] Xilinx Inc., *Zynq-7000 SoC Data Sheet: Overview v1.11.1, DS190*. Available at https://www.xilinx.com, version v1.11.1.

[67] D.-K. Electronics, *FPGA prices on Digi-Key Electronics*. Available at https://www.digikey.co.nz/.

[68] Xilinx Inc., *MicroBlaze Processor Reference Guide, UG984*. Available at https://www.xilinx.com, version 2017.3.

[69] TRENZ, *TE0712 Reference Manual*, 7 2017. Rev. 14.

[70] Silicon Labs, *Si5340, Low-Jitter, 10 or 4-Output, Any-Frequency, Any-Output Clock Generator*, 7 2016. Rev. D.

[71] Silicon Labs, *Timing - Clockbuilder Pro Software*. Available at https://www.skyworksinc.com/en/application-pages/clockbuilder-pro-software, version 2.45.

[72] Texas Instrument, *SN74LVC8T245 8-Bit Dual-Supply Bus Transceiver With Configurable Voltage Translation and 3-State Outputs*, 11 2014. N/A.

[73] *Altium Designer*. Available at https://www.altium.com/.

[74] *Analog Device*. Available at https://www.analog.com/en/index.html.

[75] Xilinx Inc., *AXI Reference Guide 4.0, UG1037*. Available at https://www.xilinx.com, version 4.0.

[76] Xilinx Inc., *AXI DMA v7.1, PG021*. Available at https://www.xilinx.com, version 7.1.

[77] Xilinx Inc., *AXI GPIO v2.0, PG114*. Available at https://www.xilinx.com, version 2.0.

[78] Xilinx Inc., *DDS Compiler v6.0, PG141*. Available at https://www.xilinx.com, version 6.0.

[79] M. Defossez, "Serial LVDS high-speed ADC interface," *XAPP 524 (v1. 1)*, 2012.

[80] Xilinx Inc., *7 Series FPGAs SelectIO Resources*. Available at https://www.xilinx.com, version 1.10.

[81] Xilinx Inc., *Integrated Logic Analyzer v6.1, PG172*. Available at https://www.xilinx.com, version 6.1.

[82] Xilinx Inc., *AXI Quad SPI v3.2, PG153*. Available at https://www.xilinx.com, version 3.2.

[83] Xilinx Inc., *Multiplier v12.0, PG108*. Available at https://www.xilinx.com, version 12.0.

[84] Xilinx Inc., *CIC Compiler v4.0, PG140*. Available at https://www.xilinx.com, version 4.0.

[85] Xilinx Inc., *LogiCORE IP Concat (v2.1), PB041*. Available at https://www.xilinx.com, version 2.1.

[86] Xilinx Inc., *AXI4-Stream Infrastructure IP Suite v3.0, PG085*. Available at https://www.xilinx.com, version 3.0.

[87] Xilinx Inc., *Zynq-7000 SoC and 7 Series Devices Memory Interface Solutions v4.2, PG586*. Available at https://www.xilinx.com, version 4.2.

[88] Xilinx Inc., *MicroBlaze Debug Module v3.2, PG115*. Available at https://www.xilinx.com, version 2.8.

[89] Xilinx Inc., *AXI Timer v2.0, PG079*. Available at https://www.xilinx.com, version 2.0.

[90] Xilinx Inc., *MicroBlaze Micro Controller System v3.0, PG116*. Available at https://www.xilinx.com, version 3.0.

[91] Avnet, *PicoZed 7Z015 / 7Z030 SOM (System-On-Module) Hardware User Guide*. Available at https://www.avnet.com/opasdata/d120001/medias/docus/203/$v2/5279-UG-PicoZed-7015-7030-V2_1.pdf, version 2.1.

[92] National Instrument, *Introduction to the PXI architecture*. Available at https://www.ni.com/en-nz/innovations/white-papers/14/introduction-to-the-pxi-architecture.html, version 1.6.0.

[93] Xilinx, *Vivado Design Suite*. Available at https://www.xilinx.com/products/design-tools/vivado.html, version 2019.1.

[94] Xilinx Inc., *AXI Memory Mapped to PCI Express (PCIe) v2.8, PG055*. Available at https://www.xilinx.com, version 2.8.

[95] Xilinx, *XSDK (Xilinx Software Development Kit)*. Available at https://www.xilinx.com/products/design-tools/embedded-software/sdk.html, version 2019.1.

[96] Xilinx, *PetaLinux Tool*. Available at https://www.xilinx.com/products/design-tools/embedded-software/petalinux-sdk.html#licensing, version 2019.1.

[97] Project Jupyter, *JupyterLab*. Available at https://jupyter.org/, version 3.0.12.

[98] J. Corbet, A. Rubini, and G. Kroah-Hartman, *Linux device drivers*. " O'Reilly Media, Inc.", 2005.

[99] B. Martin, *PCI Express Endpoint-DMA Initiator Subsystem v1.0, XAPP1171*. Xilinx Inc. Available at https://www.xilinx.com/support/documentation/application_notes/xapp1171-pcie-central-dma-subsystem.pdf, version 1.0.

[100] Paramiko, *Paramiko for SSHv2*. Available at https://www.paramiko.org/, version 2.8.0.

[101] S. Lamikhov-Center, *ELFIO - C++ library for reading and generating ELF files*. Available at http://elfio.sourceforge.net/.

[102] ADLINK, *PXIe chassis, PXES-2301*. Available at https://www.adlinktech.com/Products/PXI_PXIe_platform/PXIChassis/PXES-2301?lang=en.

[103] E. Andrew and K. Jurga, "NMR probe with short recovery time," *Journal of Magnetic Resonance (1969)*, vol. 73, no. 2, pp. 268–276, 1987.

[104] I. Lowe and C. Tarr, "A fast recovery probe and receiver for pulsed nuclear magnetic resonance spectroscopy," *Journal of Physics E: Scientific Instruments*, vol. 1, no. 3, p. 320, 1968.

[105] D. Griffin, R. Kleinberg, and M. Fukuhara, "Low-frequency NMR spectrometer," *Measurement Science and Technology*, vol. 4, no. 9, p. 968, 1993.

[106] G. Yang, S. Obruchkov, and R. Dykstra, "An open-source multichannel MRI console based on PXIe standards.," in *In Proceedings of the 29th Annual Meeting of ISMRM, Virtual Meeting*, p. 1264, 2020.

[107] N. De Zanche, C. Barmet, J. A. Nordmeyer-Massner, and K. P. Pruessmann, "NMR probes for measuring magnetic fields and field dynamics in MR systems," *Magnetic Resonance in Medicine: An Official Journal of the International Society for Magnetic Resonance in Medicine*, vol. 60, no. 1, pp. 176–186, 2008.

[108] C. Barmet, N. De Zanche, B. J. Wilm, and K. P. Pruessmann, "A transmit/receive system for magnetic field monitoring of in vivo MRI," *Magnetic Resonance in Medicine: An Official Journal of the International Society for Magnetic Resonance in Medicine*, vol. 62, no. 1, pp. 269–276, 2009.

[109] B. E. Dietrich, D. O. Brunner, B. J. Wilm, C. Barmet, S. Gross, L. Kasper, M. Haeberlin, T. Schmid, S. J. Vannesjo, and K. P. Pruessmann, "A field camera for MR sequence monitoring and system analysis," *Magnetic resonance in medicine*, vol. 75, no. 4, pp. 1831–1840, 2016.

[110] S. Gross, C. Barmet, B. E. Dietrich, D. O. Brunner, T. Schmid, and K. P. Pruessmann, "Dynamic nuclear magnetic resonance field sensing with part-per-trillion resolution," *Nature communications*, vol. 7, no. 1, pp. 1–7, 2016.

[111] Sparkfun, *Stepper Motor Driver*. Available at https://www.sparkfun.com/products/12779.

[112] E. A. Barberi, J. S. Gati, B. K. Rutt, and R. S. Menon, "A transmit-only/receive-only (TORO) RF system for high-field MRI/MRS applications," *Magnetic Resonance in Medicine: An Official Journal of the International Society for Magnetic Resonance in Medicine*, vol. 43, no. 2, pp. 284–289, 2000.

[113] P. B. Roemer, W. A. Edelstein, C. E. Hayes, S. P. Souza, and O. M. Mueller, "The NMR phased array," *Magnetic resonance in medicine*, vol. 16, no. 2, pp. 192–225, 1990.

[114] J. P. Stockmann, P. A. Ciris, G. Galiana, L. Tam, and R. T. Constable, "O-space imaging: highly efficient parallel imaging using second-order nonlinear fields as encoding gradients with no phase encoding," *Magnetic resonance in medicine*, vol. 64, no. 2, pp. 447–456, 2010.

Appendix A

Electromotive Force Calculation

The magnetic field associated with the magnetization of a sample arising from the effective current density can be expressed:

$$\vec{J}_M(\vec{r},t) = \vec{\nabla} \times \vec{M}(\vec{r},t) \tag{A.1}$$

where the $\vec{J}$ implies $|\vec{J}|$ charge per unit time per unit area in the direction of J. The curl operation computes the net circulation of the magnetization.

The vector at position $\vec{r}$ stemming from a source current is

$$\vec{A}(\vec{r}) = \frac{\mu_0}{4\pi} \int d^3 r' \frac{\vec{J}(\vec{r}\,')}{|\vec{r} - \vec{r}\,'|} \tag{A.2}$$

where the time dependence has been expressed. Due to the ignorance of the time delay between the source and the measurement, the magnetic field is calculated from

$$\vec{B} = \vec{\nabla} \times \vec{A} \tag{A.3}$$

Then the flux (2.5) through a coil can be written as

$$\Phi = \int_{area} \vec{B} \cdot d\vec{S} = \int (\vec{\nabla} \times \vec{A}) \cdot d\vec{S} = \oint d\vec{l} \cdot \vec{A} \tag{A.4}$$

It demonstrates that the flux Φ_M through a coil due to a magnetization source can be related to a flux due to the coil that goes through the magnetization. Combining A.2, A.3, A.4, and the vector identity, $\vec{A}\cdot(\vec{B}\times\vec{C}) = -(\vec{A}\times\vec{C})\cdot\vec{B}$, the flux can be expressed

$$\begin{aligned}\Phi_M &= \oint d\vec{l}\cdot\left[\frac{\mu_0}{4\pi}\int d^3r'\frac{\vec{\nabla}'\times\vec{M}(\vec{r}\,')}{|\vec{r}-\vec{r}\,'|}\right] \\ &= \frac{\mu_0}{4\pi}\int d^3r'\vec{M}(\vec{r}\,')\cdot\left[\vec{\nabla}'\times(\oint\frac{d\vec{l}}{|\vec{r}-\vec{r}\,'|})\right]\end{aligned} \tag{A.5}$$

Now Equation A.4 can be evaluated at position $\vec{r}'$

$$\vec{A}(\vec{r}\,') = \frac{\mu_0}{4\pi}\oint\frac{Id\vec{l}}{|\vec{r}-\vec{r}\,'|} \tag{A.6}$$

shows that the curl of the line integral over the current path in equation A.5 is $\vec{B}^{receive}$, then the magnetic field produced by the coil at $\vec{r}$ is

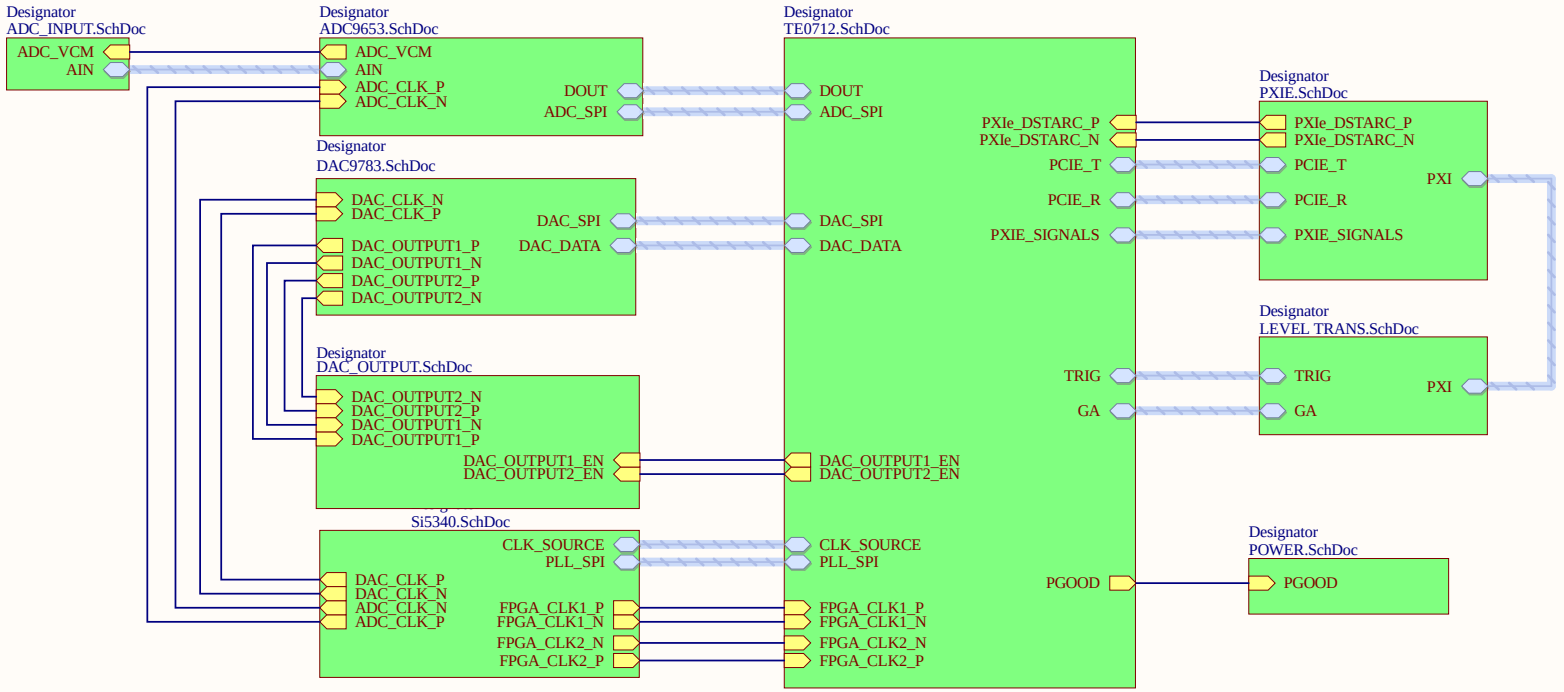
$$\vec{B}^{receive}(\vec{r}\,') = \vec{B}(\vec{r}\,')/I = \vec{\nabla}'\times(\frac{\mu_0}{4\pi}\oint\frac{Id\vec{l}}{|\vec{r}-\vec{r}\,'|}) \tag{A.7}$$

Finally, the flux can be written as

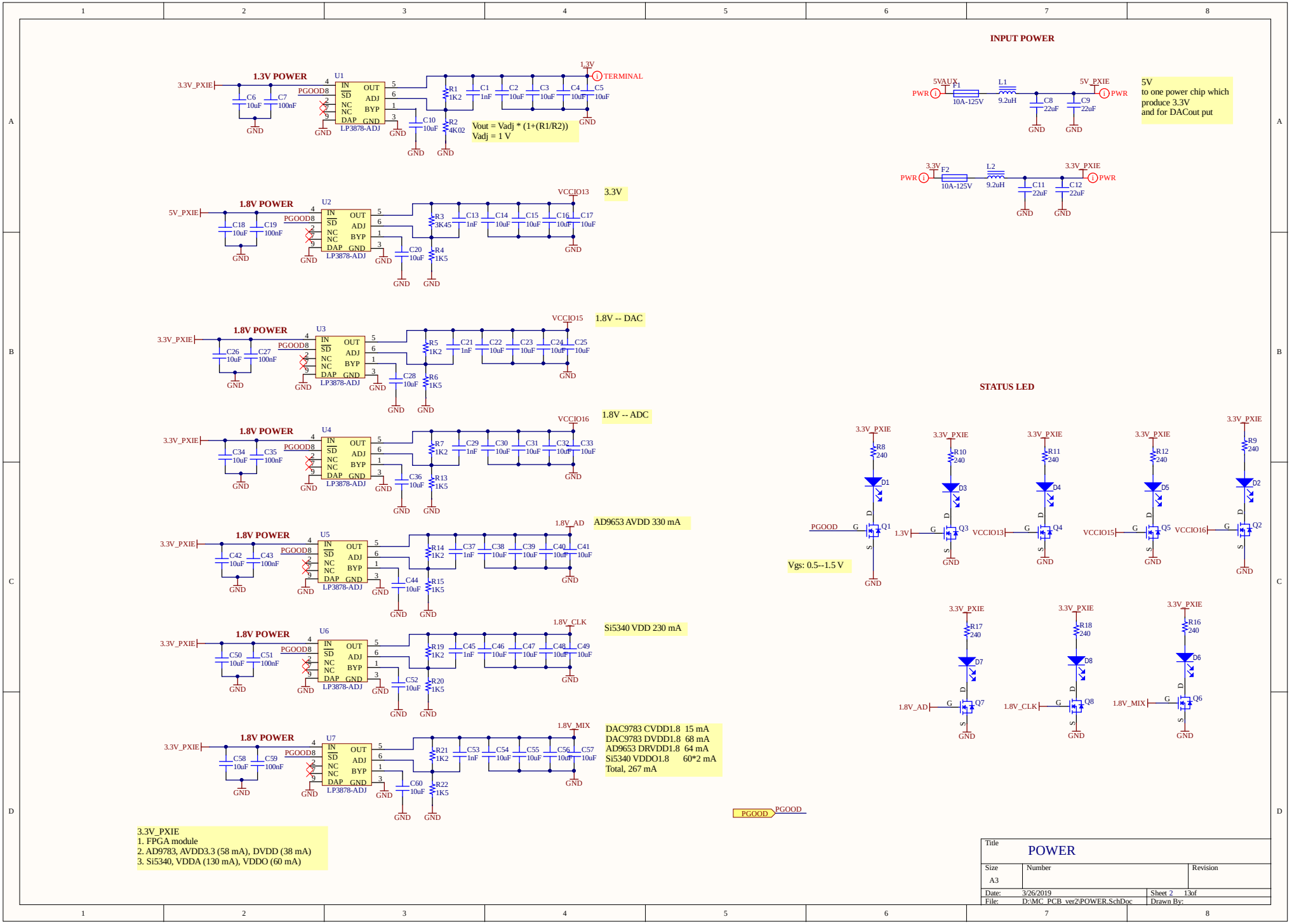
$$\Phi_M(t) = \int_{sample} d^3r\vec{B}^{receive}(\vec{r})\cdot\vec{M}(\vec{r},t) \tag{A.8}$$

Appendix B

Schematics of PCB Designs

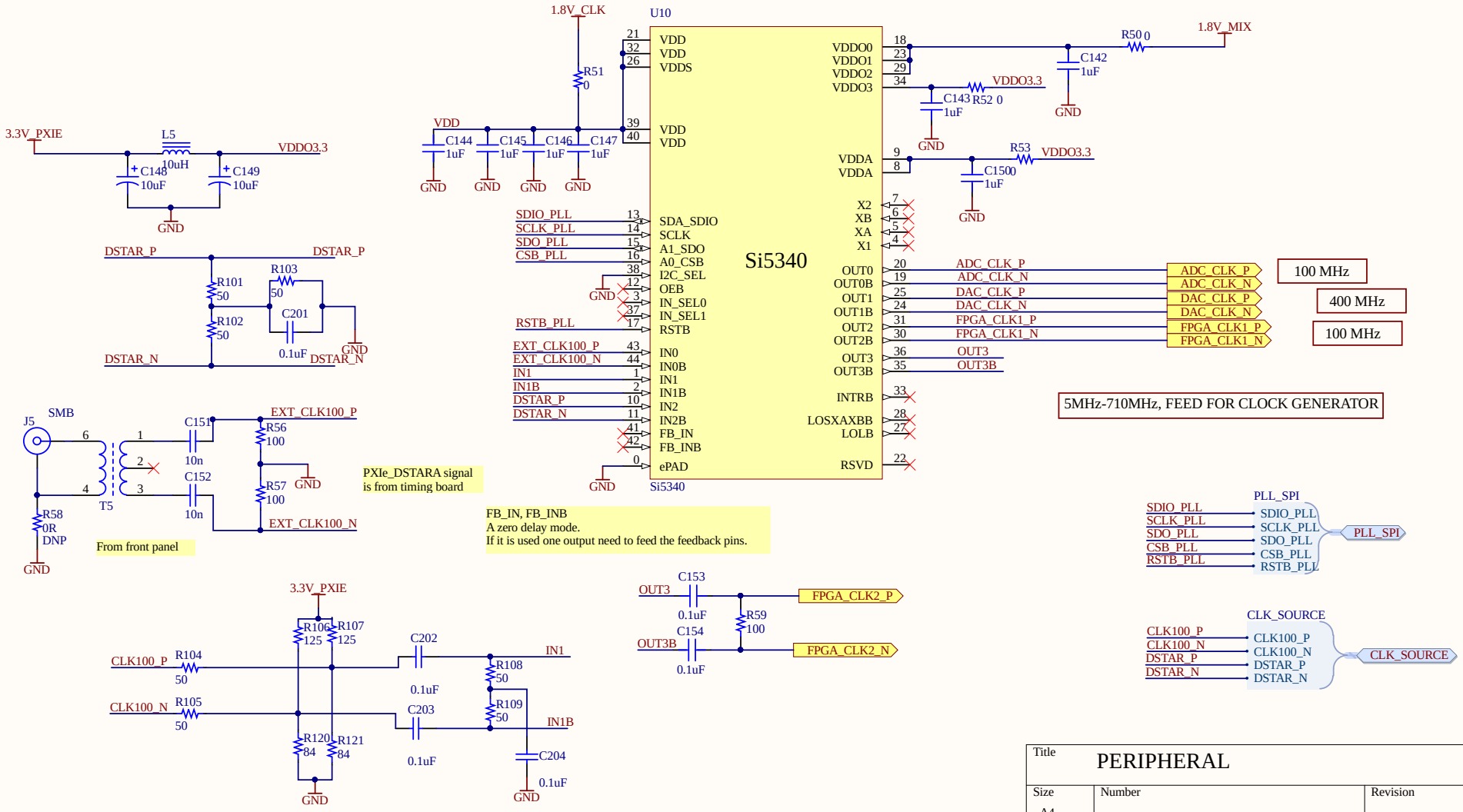


Title	TOP_DESIGN	
Size: A4	Number:	Revision:
Date: 3/26/2019	Sheet 1 of 1	
File: D:\MC_PCB_ver2\MC_top.SchDoc	Drawn By: GUANG	



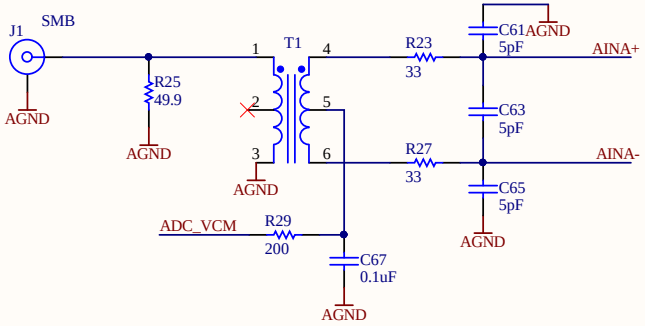
Title	POWER	
Size: A3	Number	Revision
Date: 3/26/2019	Sheet 2 13of	
File: D:\MC_PCB_ver2\POWER.SchDoc	Drawn By:	

CLOCK DISTRIBUTOR AND PLL

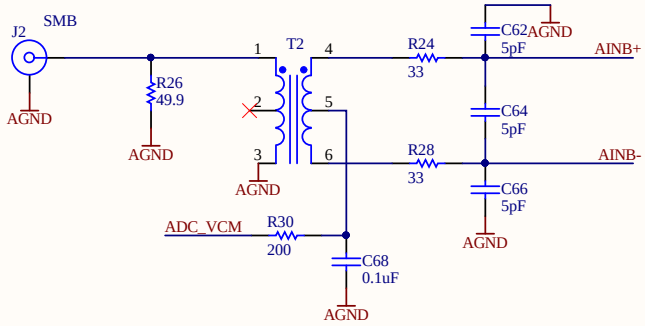


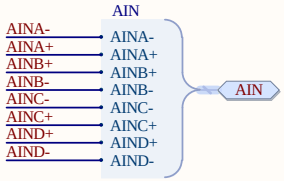
Title	PERIPHERAL	
Size A4	Number	Revision
Date: 3/26/2019	Sheet 3 of 13	
File: D:\MC_PCB_ver2\Si5340.SchDoc	Drawn By: GUANG	

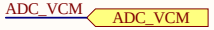
CHANNEL A



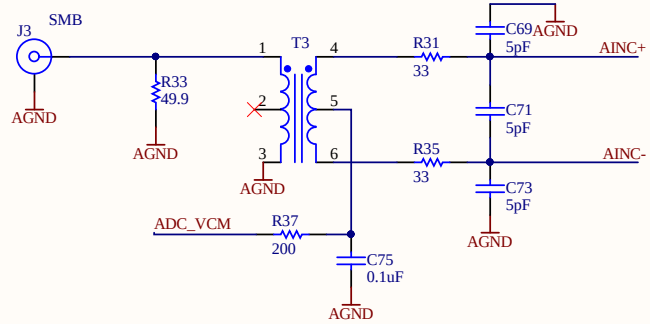
CHANNEL B



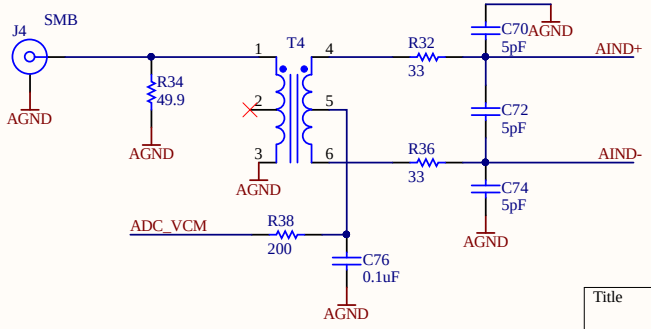




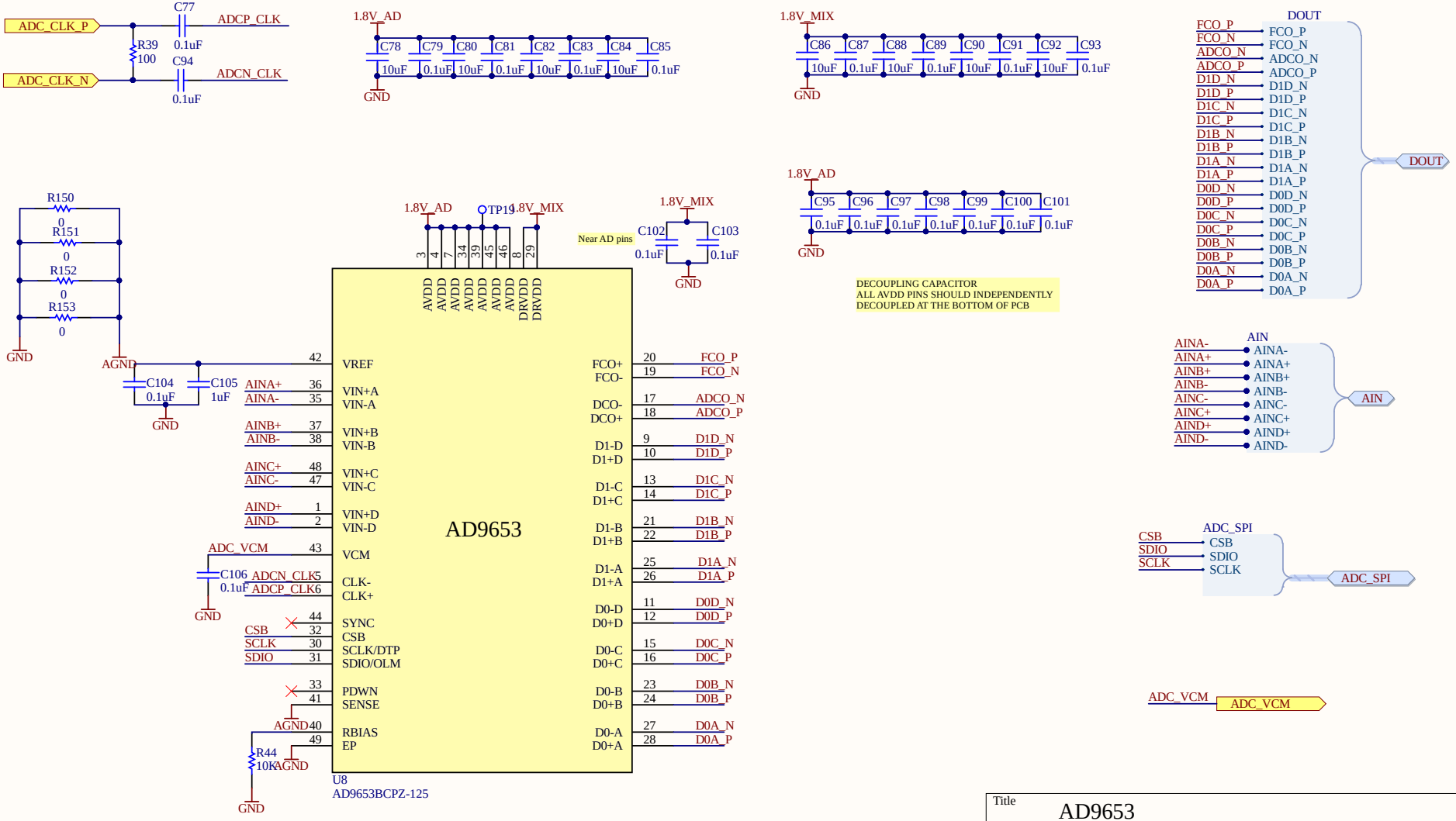
CHANNEL C



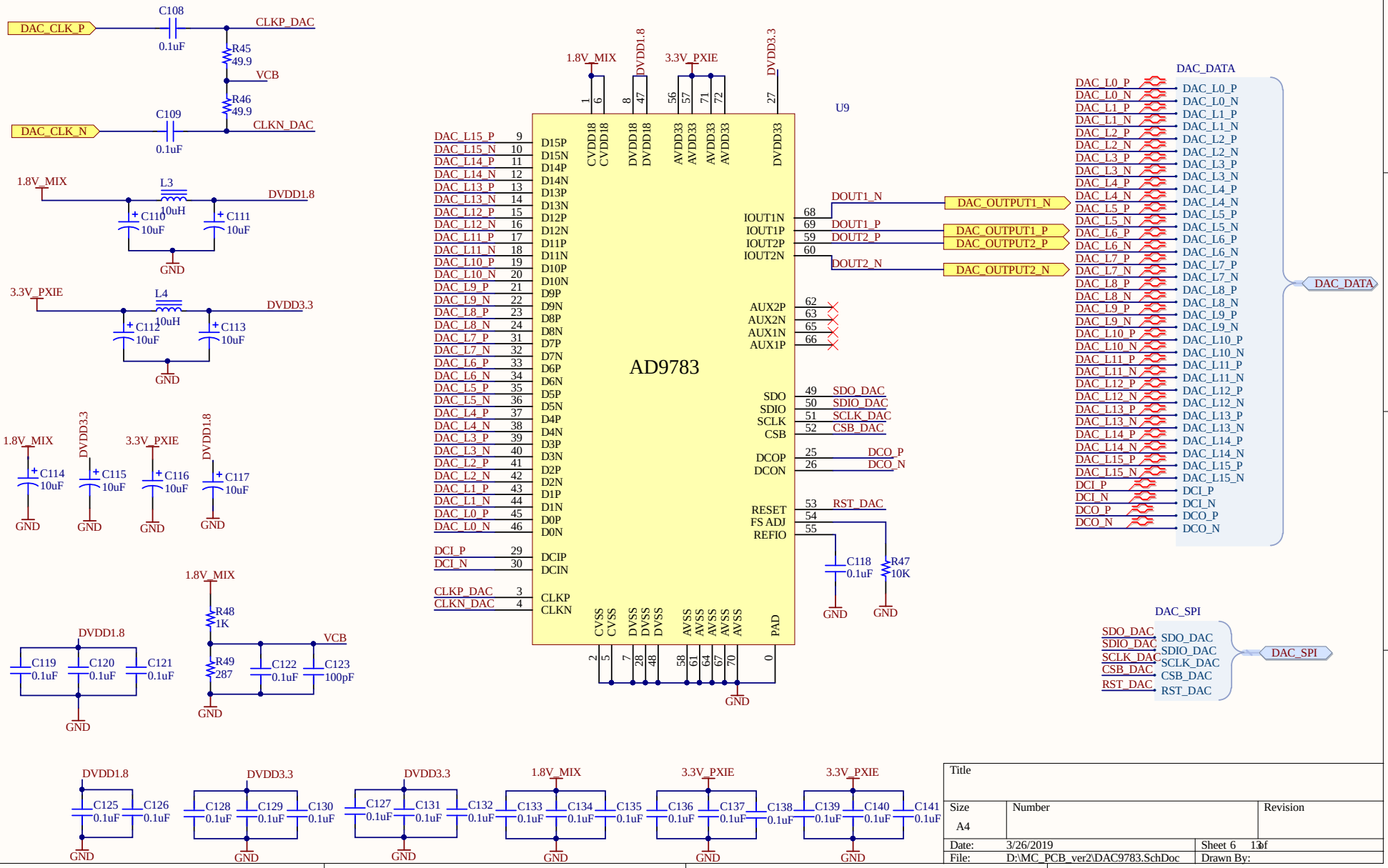
CHANNEL D

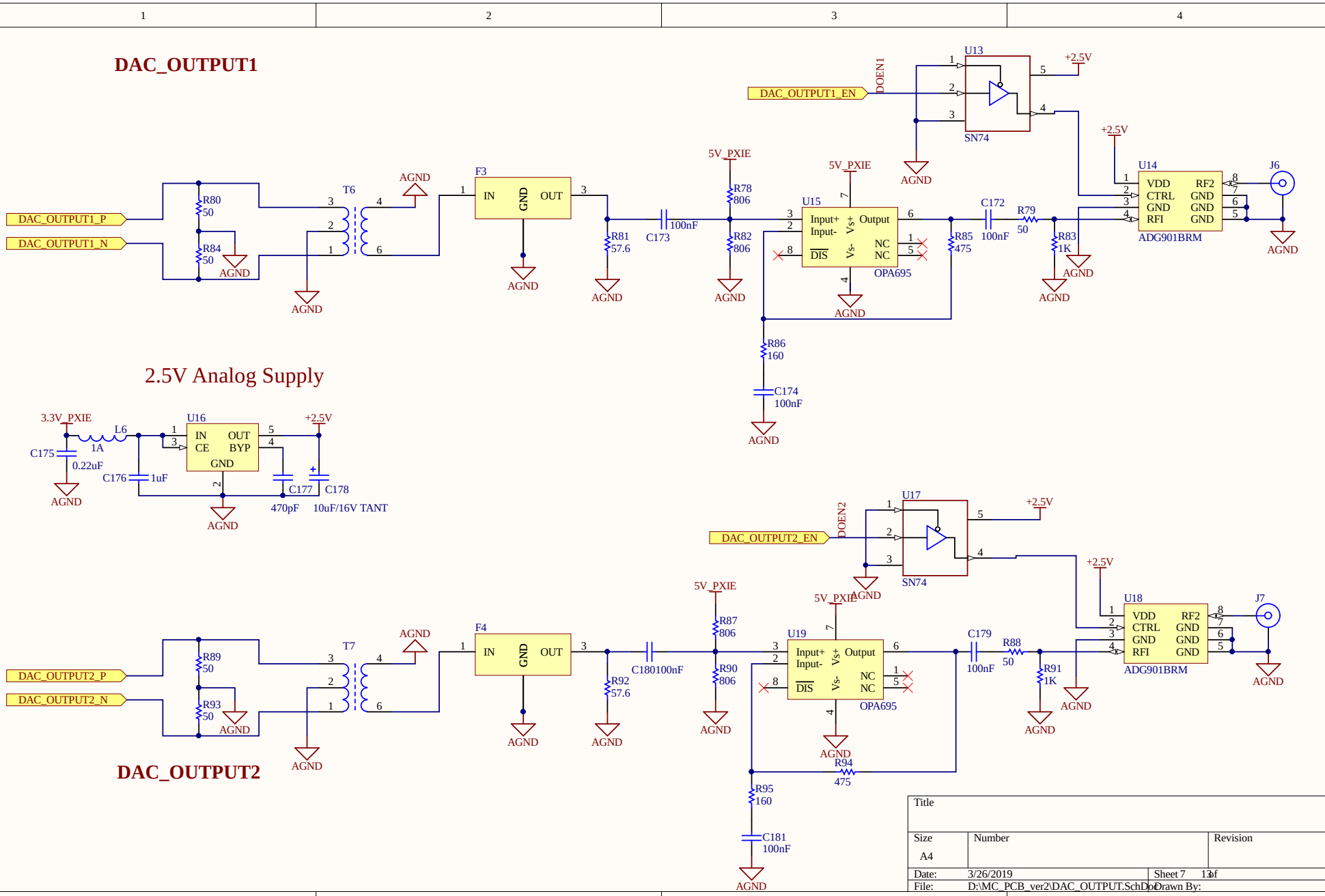


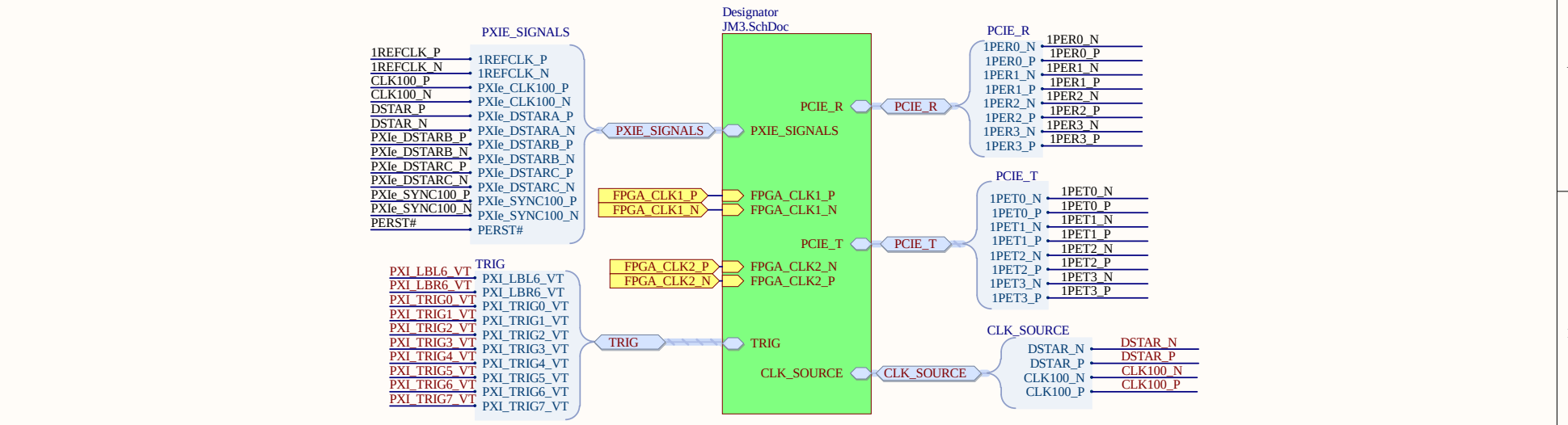
Title: ANALOG INPUT		
Size: A4	Number:	Revision:
Date: 3/26/2019	Sheet 4 of 8	
File: D:\MC_PCB_ver2\adc\ADC_INPUT.SchDoc	Drawn By: Guang	

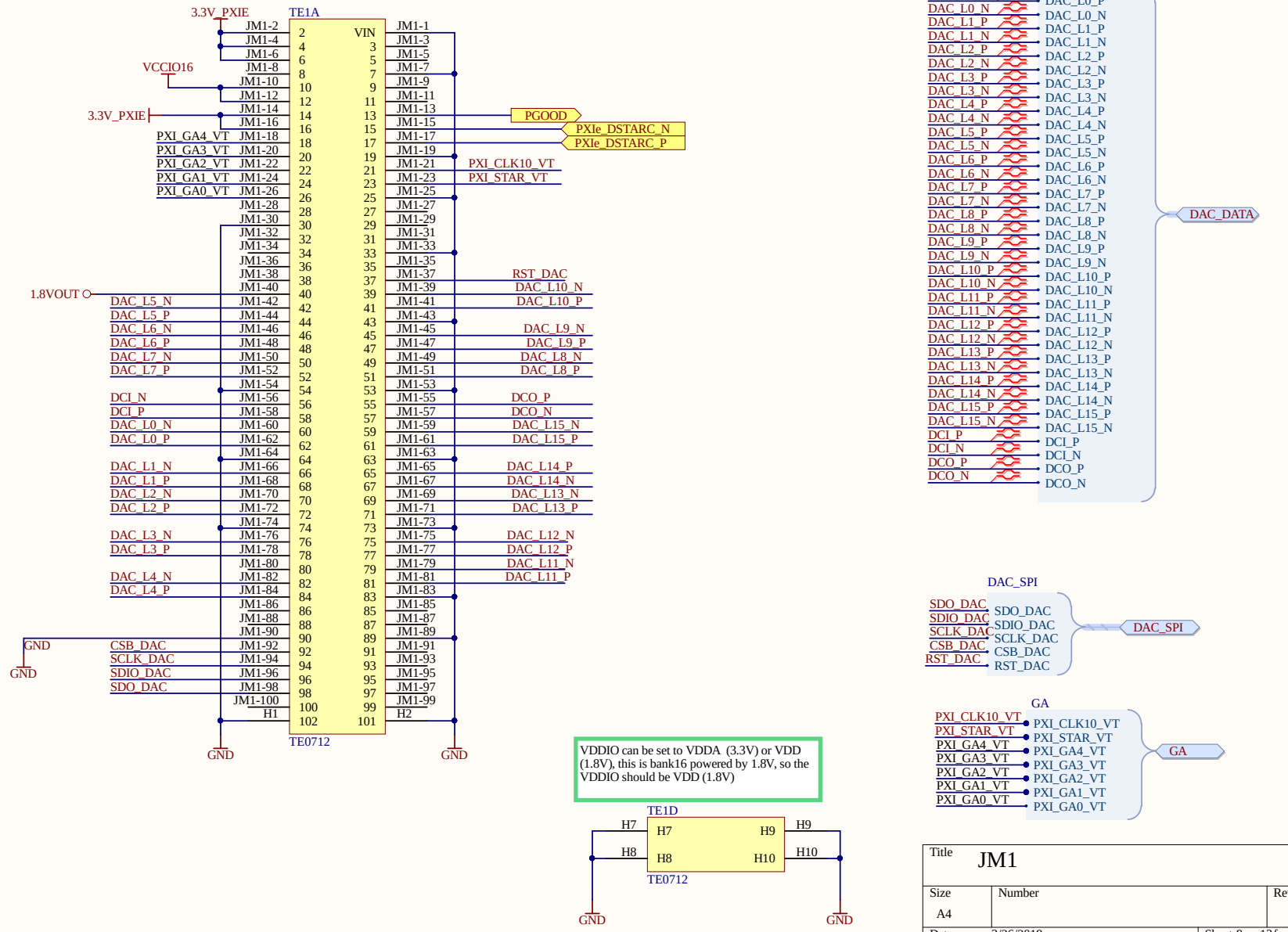


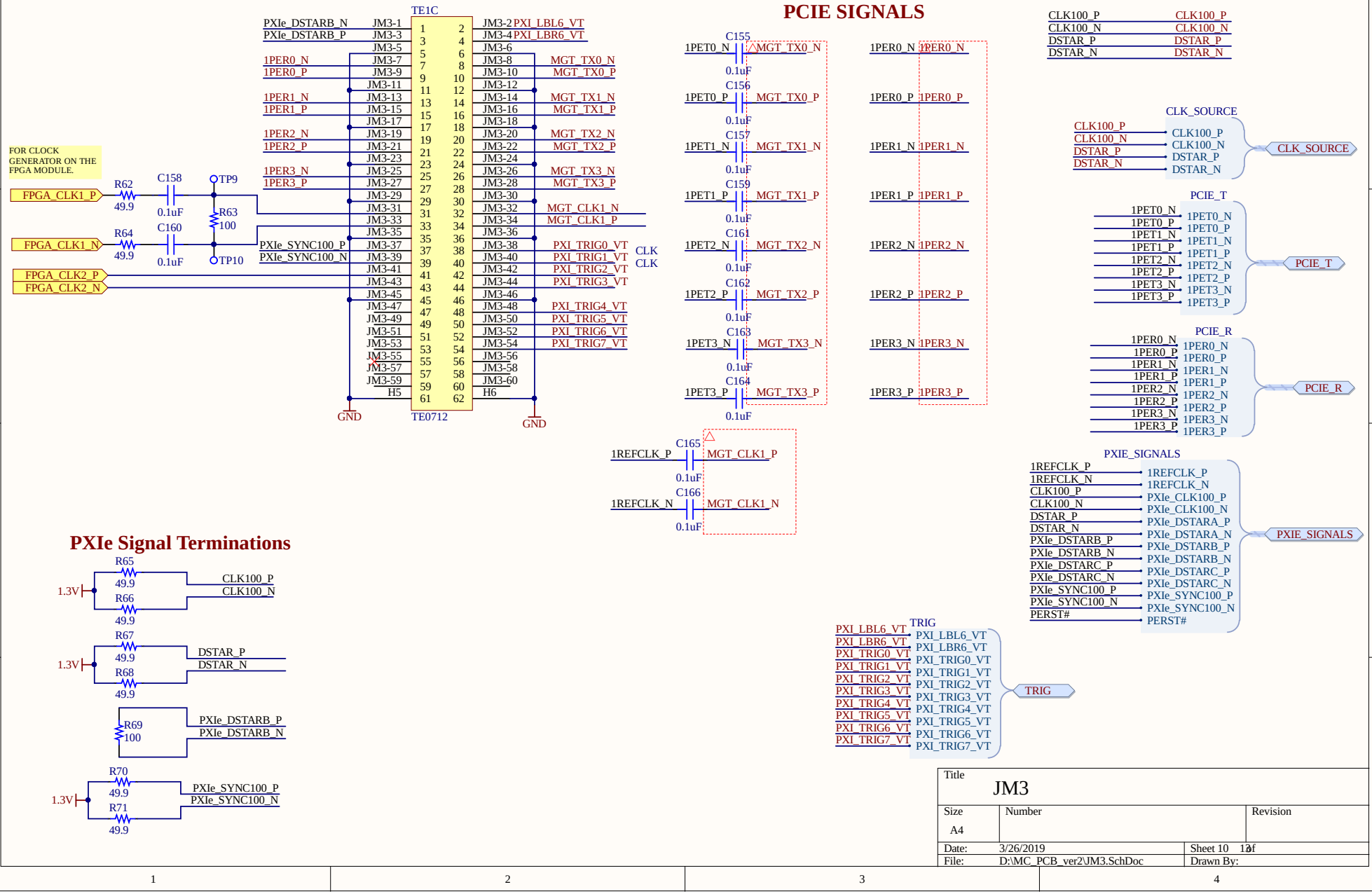
Title	AD9653	
Size: A4	Number:	Revision: 3.0
Date: 3/26/2019	Sheet 5 of 13	
File: D:\MC_PCB_ver2\adc\ADC9653.SchDoc	Drawn By: Guang	

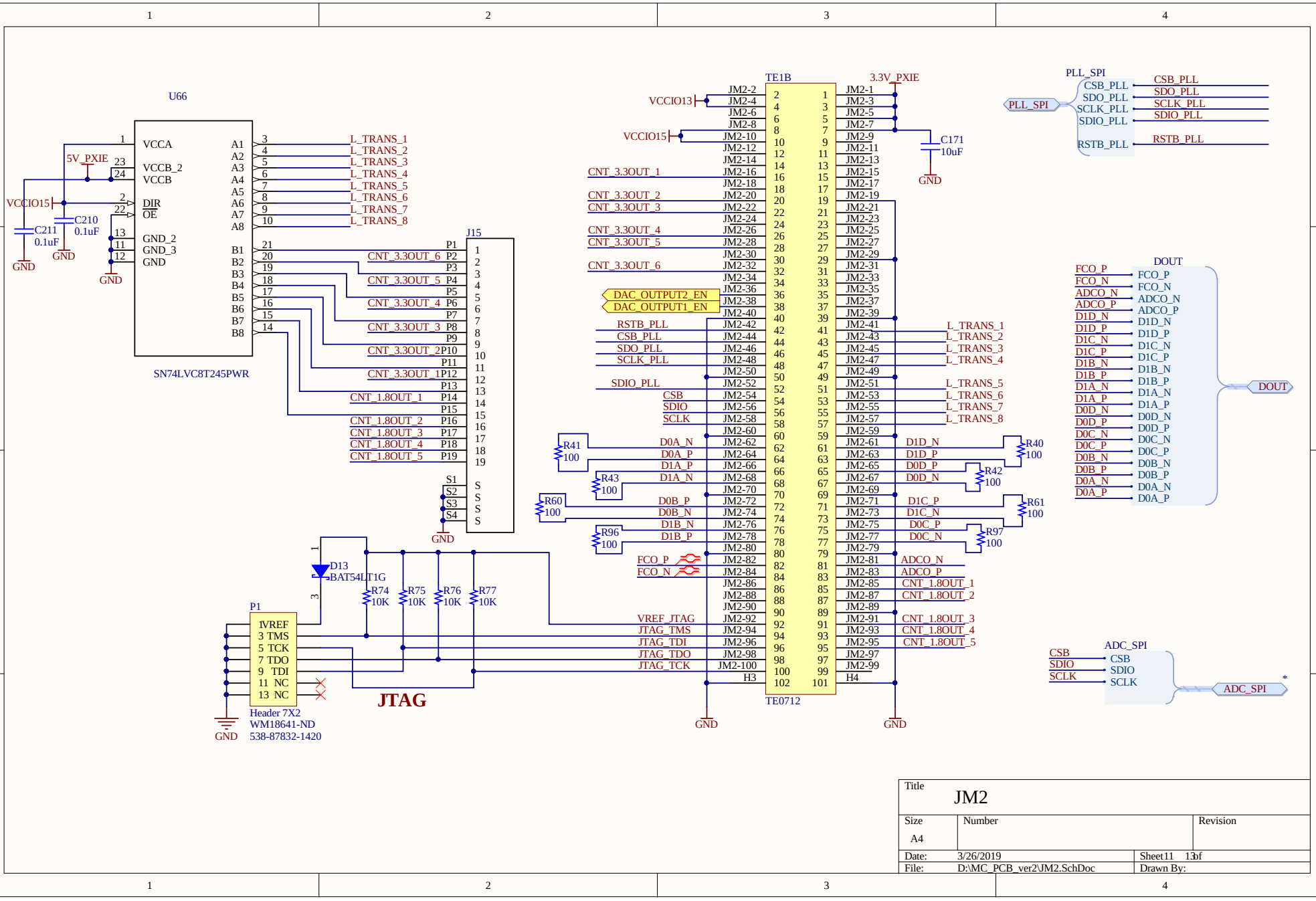

DAC_CLK_P
C108
0.1uF
CLKP_DAC
R45
49.9
VCB
R46
49.9
DAC_CLK_N
C109
0.1uF
CLKN_DAC
1.8V_MIX
L3
10uH
C110
10uF
C111
10uF
DVDD1.8
GND
3.3V_PXIE
L4
10uH
C112
10uF
C113
10uF
DVDD3.3
GND
1.8V_MIX
C114
10uF
DVDD3.3
C115
10uF
3.3V_PXIE
C116
10uF
DVDD1.8
C117
10uF
GND
1.8V_MIX
R48
1K
VCB
DVDD1.8
C119
0.1uF
C120
0.1uF
C121
0.1uF
R49
287
C122
0.1uF
C123
100pF
GND
U9
AD9783
1.8V_MIX
DVDD1.8
3.3V_PXIE
DVDD3.3
CVDD18
DVDD18
AVDD33
DVDD33
DAC_L15_P
DAC_L15_N
DAC_L14_P
DAC_L14_N
DAC_L13_P
DAC_L13_N
DAC_L12_P
DAC_L12_N
DAC_L11_P
DAC_L11_N
DAC_L10_P
DAC_L10_N
DAC_L9_P
DAC_L9_N
DAC_L8_P
DAC_L8_N
DAC_L7_P
DAC_L7_N
DAC_L6_P
DAC_L6_N
DAC_L5_P
DAC_L5_N
DAC_L4_P
DAC_L4_N
DAC_L3_P
DAC_L3_N
DAC_L2_P
DAC_L2_N
DAC_L1_P
DAC_L1_N
DAC_L0_P
DAC_L0_N
DCI_P
DCI_N
CLKP_DAC
CLKN_DAC
CVSS
DVSS
AVSS
PAD
IOUT1N
IOUT1P
IOUT2P
IOUT2N
DOUT1_N
DOUT1_P
DOUT2_P
DOUT2_N
DAC_OUTPUT1_N
DAC_OUTPUT1_P
DAC_OUTPUT2_P
DAC_OUTPUT2_N
AUX2P
AUX2N
AUX1N
AUX1P
SDO
SDIO
SCLK
CSB
SDO_DAC
SDIO_DAC
SCLK_DAC
CSB_DAC
DCOP
DCON
DCO_P
DCO_N
RESET
FS ADJ
REFIO
RST_DAC
C118
0.1uF
R47
10K
GND
DAC_DATA
DAC_SPI
DVDD1.8
C125
0.1uF
C126
0.1uF
DVDD3.3
C128
0.1uF
C129
0.1uF
C130
0.1uF
DVDD3.3
C127
0.1uF
C131
0.1uF
C132
0.1uF
1.8V_MIX
C133
0.1uF
C134
0.1uF
C135
0.1uF
3.3V_PXIE
C136
0.1uF
C137
0.1uF
C138
0.1uF
3.3V_PXIE
C139
0.1uF
C140
0.1uF
C141
0.1uF
GND
Title
Size
A4
Number
Revision
Date:
3/26/2019
Sheet 6 of 13
File:
D:\MC_PCB_ver2\DAC9783.SchDoc
Drawn By:


DAC_OUTPUT1
DAC_OUTPUT1_P
DAC_OUTPUT1_N
DAC_OUTPUT1_EN
DOEN1
U13
SN74
+2.5V
5V_PXIE
R80 50
R84 50
T6
F3
R78 806
R82 806
R81 57.6
C173 100nF
U15
Input+
Input-
Vs+
Vs-
Output
NC
DIS
OPA695
C172 100nF
R79 50
R83 1K
R85 475
R86 160
C174 100nF
U14
ADG901BRM
VDD
CTRL
GND
RFI
RF2
J6
AGND
2.5V Analog Supply
3.3V_PXIE
L6 1A
C175 0.22uF
C176 1uF
U16
IN
CE
OUT
BYP
C177 470pF
C178 10uF/16V TANT
DAC_OUTPUT2_P
DAC_OUTPUT2_N
DAC_OUTPUT2_EN
DOEN2
U17
T7
F4
R87 806
R90 806
R92 57.6
C180 100nF
U19
R88 50
R91 1K
R94 475
R95 160
C179 100nF
C181 100nF
U18
J7
DAC_OUTPUT2
Title
Size
A4
Number
Revision
Date: 3/26/2019
Sheet 7 of 13
File: D:\MC_PCB_ver2\DAC_OUTPUT.SchDoc
Drawn By:


PXIE_SIGNALS
1REFCLK_P
1REFCLK_N
CLK100_P
CLK100_N
DSTAR_P
DSTAR_N
PXIe_DSTARB_P
PXIe_DSTARB_N
PXIe_DSTARC_P
PXIe_DSTARC_N
PXIe_SYNC100_P
PXIe_SYNC100_N
PERST#
Designator
JM3.SchDoc
PXIE_SIGNALS
FPGA_CLK1_P
FPGA_CLK1_N
FPGA_CLK2_P
FPGA_CLK2_N
TRIG
PCIE_R
PCIE_T
CLK_SOURCE
PCIE_R
1PER0_N
1PER0_P
1PER1_N
1PER1_P
1PER2_N
1PER2_P
1PER3_N
1PER3_P
PCIE_T
1PET0_N
1PET0_P
1PET1_N
1PET1_P
1PET2_N
1PET2_P
1PET3_N
1PET3_P
CLK_SOURCE
DSTAR_N
DSTAR_P
CLK100_N
CLK100_P
TRIG
PXI_LBL6_VT
PXI_LBR6_VT
PXI_TRIG0_VT
PXI_TRIG1_VT
PXI_TRIG2_VT
PXI_TRIG3_VT
PXI_TRIG4_VT
PXI_TRIG5_VT
PXI_TRIG6_VT
PXI_TRIG7_VT
DAC_DATA
DAC_L0_P
DAC_L0_N
DAC_L1_P
DAC_L1_N
DAC_L2_P
DAC_L2_N
DAC_L3_P
DAC_L3_N
DAC_L4_P
DAC_L4_N
DAC_L5_P
DAC_L5_N
DAC_L6_P
DAC_L6_N
DAC_L7_P
DAC_L7_N
DAC_L8_P
DAC_L8_N
DAC_L9_P
DAC_L9_N
DAC_L10_P
DAC_L10_N
DAC_L11_P
DAC_L11_N
DAC_L12_P
DAC_L12_N
DAC_L13_P
DAC_L13_N
DAC_L14_P
DAC_L14_N
DAC_L15_P
DAC_L15_N
DCI_P
DCI_N
DCO_P
DCO_N
Designator
JM1.SchDoc
DAC_DATA
PXIe_DSTARC_N
PXIe_DSTARC_P
DAC_SPI
PGOOD
GA
PXI_GA0_VT
PXI_GA1_VT
PXI_GA2_VT
PXI_GA3_VT
PXI_GA4_VT
PXI_STAR_VT
PXI_CLK10_VT
DAC_SPI
CSB_DAC
SCLK_DAC
SDIO_DAC
SDO_DAC
RST_DAC
Designator
JM2.SchDoc
ADC_SPI
DOUT
DAC_OUTPUT1_EN
DAC_OUTPUT2_EN
PLL_SPI
CSB
SDIO
SCLK
CSB_PLL
SDO_PLL
SCLK_PLL
SDIO_PLL
RSTB_PLL
DOUT
D0A_P
D0A_N
D0B_P
D0B_N
D0C_P
D0C_N
D0D_P
D0D_N
D1A_P
D1A_N
D1B_P
D1B_N
D1C_P
D1C_N
D1D_P
D1D_N
ADCO_N
ADCO_P
FCO_N
FCO_P
Title
TE0712
Size
A4
Number
Revision
Date: 3/26/2019
Sheet 8 of 13
File: D:\MC_PCB_ver2\TE0712.SchDoc
Drawn By:

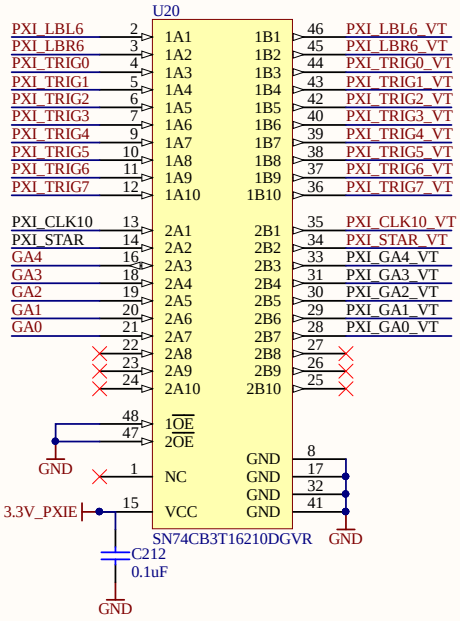

3.3V_PXIE
TE1A
JM1-2
JM1-4
JM1-6
JM1-8
VCCIO16
JM1-10
JM1-12
JM1-14
3.3V_PXIE
JM1-16
PXI_GA4_VT JM1-18
PXI_GA3_VT JM1-20
PXI_GA2_VT JM1-22
PXI_GA1_VT JM1-24
PXI_GA0_VT JM1-26
JM1-28
JM1-30
JM1-32
JM1-34
JM1-36
JM1-38
JM1-40
1.8VOUT
DAC_L5_N JM1-42
DAC_L5_P JM1-44
DAC_L6_N JM1-46
DAC_L6_P JM1-48
DAC_L7_N JM1-50
DAC_L7_P JM1-52
JM1-54
DCI_N JM1-56
DCI_P JM1-58
DAC_L0_N JM1-60
DAC_L0_P JM1-62
JM1-64
DAC_L1_N JM1-66
DAC_L1_P JM1-68
DAC_L2_N JM1-70
DAC_L2_P JM1-72
JM1-74
DAC_L3_N JM1-76
DAC_L3_P JM1-78
JM1-80
DAC_L4_N JM1-82
DAC_L4_P JM1-84
JM1-86
JM1-88
JM1-90
GND
CSB_DAC JM1-92
SCLK_DAC JM1-94
SDIO_DAC JM1-96
SDO_DAC JM1-98
JM1-100
H1
GND
GND
2 VIN JM1-1
4 3 JM1-3
6 5 JM1-5
8 7 JM1-7
10 9 JM1-9
12 11 JM1-11
14 13 JM1-13
16 15 JM1-15
18 17 JM1-17
20 19 JM1-19
22 21 JM1-21
24 23 JM1-23
26 25 JM1-25
28 27 JM1-27
30 29 JM1-29
32 31 JM1-31
34 33 JM1-33
36 35 JM1-35
38 37 JM1-37
40 39 JM1-39
42 41 JM1-41
44 43 JM1-43
46 45 JM1-45
48 47 JM1-47
50 49 JM1-49
52 51 JM1-51
54 53 JM1-53
56 55 JM1-55
58 57 JM1-57
60 59 JM1-59
62 61 JM1-61
64 63 JM1-63
66 65 JM1-65
68 67 JM1-67
70 69 JM1-69
72 71 JM1-71
74 73 JM1-73
76 75 JM1-75
78 77 JM1-77
80 79 JM1-79
82 81 JM1-81
84 83 JM1-83
86 85 JM1-85
88 87 JM1-87
90 89 JM1-89
92 91 JM1-91
94 93 JM1-93
96 95 JM1-95
98 97 JM1-97
100 99 JM1-99
102 101
TE0712
JM1-1
JM1-3
JM1-5
JM1-7
JM1-9
JM1-11
JM1-13
PGOOD
JM1-15
PXIe_DSTARC_N
JM1-17
PXIe_DSTARC_P
JM1-19
JM1-21 PXI_CLK10_VT
JM1-23 PXI_STAR_VT
JM1-25
JM1-27
JM1-29
JM1-31
JM1-33
JM1-35
JM1-37 RST_DAC
JM1-39 DAC_L10_N
JM1-41 DAC_L10_P
JM1-43
JM1-45 DAC_L9_N
JM1-47 DAC_L9_P
JM1-49 DAC_L8_N
JM1-51 DAC_L8_P
JM1-53
JM1-55 DCO_P
JM1-57 DCO_N
JM1-59 DAC_L15_N
JM1-61 DAC_L15_P
JM1-63
JM1-65 DAC_L14_P
JM1-67 DAC_L14_N
JM1-69 DAC_L13_N
JM1-71 DAC_L13_P
JM1-73
JM1-75 DAC_L12_N
JM1-77 DAC_L12_P
JM1-79 DAC_L11_N
JM1-81 DAC_L11_P
JM1-83
JM1-85
JM1-87
JM1-89
JM1-91
JM1-93
JM1-95
JM1-97
JM1-99
H2
GND
VDDIO can be set to VDDA (3.3V) or VDD (1.8V), this is bank16 powered by 1.8V, so the VDDIO should be VDD (1.8V)
TE1D
H7
H8
H7 H9
H8 H10
TE0712
H9
H10
GND
GND
DAC_DATA
DAC_L0_P
DAC_L0_N
DAC_L1_P
DAC_L1_N
DAC_L2_P
DAC_L2_N
DAC_L3_P
DAC_L3_N
DAC_L4_P
DAC_L4_N
DAC_L5_P
DAC_L5_N
DAC_L6_P
DAC_L6_N
DAC_L7_P
DAC_L7_N
DAC_L8_P
DAC_L8_N
DAC_L9_P
DAC_L9_N
DAC_L10_P
DAC_L10_N
DAC_L11_P
DAC_L11_N
DAC_L12_P
DAC_L12_N
DAC_L13_P
DAC_L13_N
DAC_L14_P
DAC_L14_N
DAC_L15_P
DAC_L15_N
DCI_P
DCI_N
DCO_P
DCO_N
DAC_L0_P
DAC_L0_N
DAC_L1_P
DAC_L1_N
DAC_L2_P
DAC_L2_N
DAC_L3_P
DAC_L3_N
DAC_L4_P
DAC_L4_N
DAC_L5_P
DAC_L5_N
DAC_L6_P
DAC_L6_N
DAC_L7_P
DAC_L7_N
DAC_L8_P
DAC_L8_N
DAC_L9_P
DAC_L9_N
DAC_L10_P
DAC_L10_N
DAC_L11_P
DAC_L11_N
DAC_L12_P
DAC_L12_N
DAC_L13_P
DAC_L13_N
DAC_L14_P
DAC_L14_N
DAC_L15_P
DAC_L15_N
DCI_P
DCI_N
DCO_P
DCO_N
DAC_DATA
DAC_SPI
SDO_DAC
SDIO_DAC
SCLK_DAC
CSB_DAC
RST_DAC
SDO_DAC
SDIO_DAC
SCLK_DAC
CSB_DAC
RST_DAC
DAC_SPI
GA
PXI_CLK10_VT
PXI_STAR_VT
PXI_GA4_VT
PXI_GA3_VT
PXI_GA2_VT
PXI_GA1_VT
PXI_GA0_VT
PXI_CLK10_VT
PXI_STAR_VT
PXI_GA4_VT
PXI_GA3_VT
PXI_GA2_VT
PXI_GA1_VT
PXI_GA0_VT
GA
Title
JM1
Size
A4
Number
Revision
Date: 3/26/2019
Sheet 9 of 13
File: D:\MC_PCB_ver2\JM1.SchDoc
Drawn By:

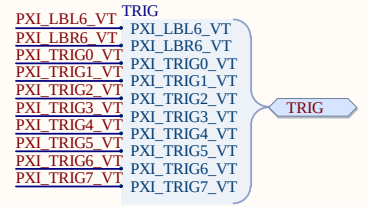

PCIE SIGNALS
TE1C
TE0712
FOR CLOCK GENERATOR ON THE FPGA MODULE.
FPGA_CLK1_P
FPGA_CLK1_N
FPGA_CLK2_P
FPGA_CLK2_N
R62
R64
49.9
C158
C160
0.1uF
TP9
TP10
R63
100
PXIe_SYNC100_P
PXIe_SYNC100_N
PXIe_DSTARB_N
PXIe_DSTARB_P
1PER0_N
1PER0_P
1PER1_N
1PER1_P
1PER2_N
1PER2_P
1PER3_N
1PER3_P
MGT_TX0_N
MGT_TX0_P
MGT_TX1_N
MGT_TX1_P
MGT_TX2_N
MGT_TX2_P
MGT_TX3_N
MGT_TX3_P
MGT_CLK1_N
MGT_CLK1_P
PXI_TRIG0_VT
PXI_TRIG1_VT
PXI_TRIG2_VT
PXI_TRIG3_VT
PXI_TRIG4_VT
PXI_TRIG5_VT
PXI_TRIG6_VT
PXI_TRIG7_VT
PXI_LBL6_VT
PXI_LBR6_VT
CLK
CLK
GND
GND
C155
C156
C157
C159
C161
C162
C163
C164
C165
C166
1PET0_N
1PET0_P
1PET1_N
1PET1_P
1PET2_N
1PET2_P
1PET3_N
1PET3_P
1REFCLK_P
1REFCLK_N
0.1uF
CLK100_P
CLK100_N
DSTAR_P
DSTAR_N
CLK_SOURCE
PCIE_T
PCIE_R
PXIE_SIGNALS
TRIG
PXIe Signal Terminations
1.3V
R65
R66
R67
R68
R69
R70
R71
100
CLK100_P
CLK100_N
DSTAR_P
DSTAR_N
PXIe_DSTARB_P
PXIe_DSTARB_N
PXIe_SYNC100_P
PXIe_SYNC100_N
Title
JM3
Size
A4
Number
Revision
Date:
3/26/2019
Sheet 10 of 13
File:
D:\MC_PCB_ver2\JM3.SchDoc
Drawn By:

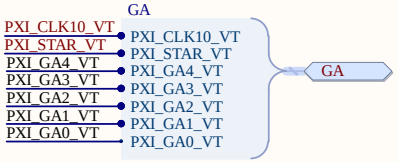


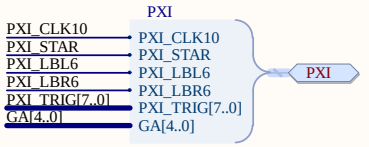
Title	JM2	
Size: A4	Number	Revision
Date: 3/26/2019	Sheet11 13of	
File: D:\MC_PCB_ver2\JM2.SchDoc	Drawn By:	

LEVEL TRANSLATOR

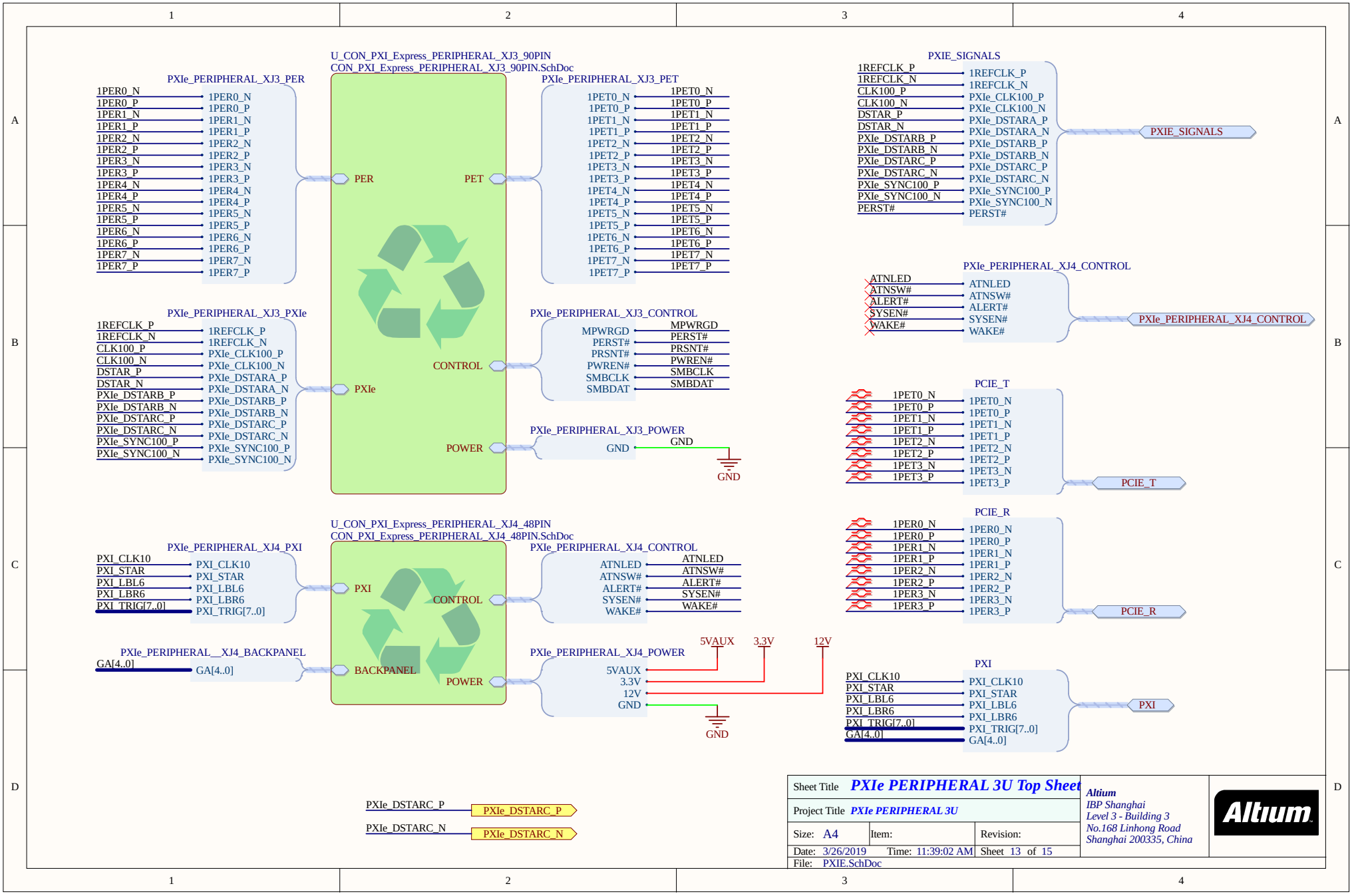






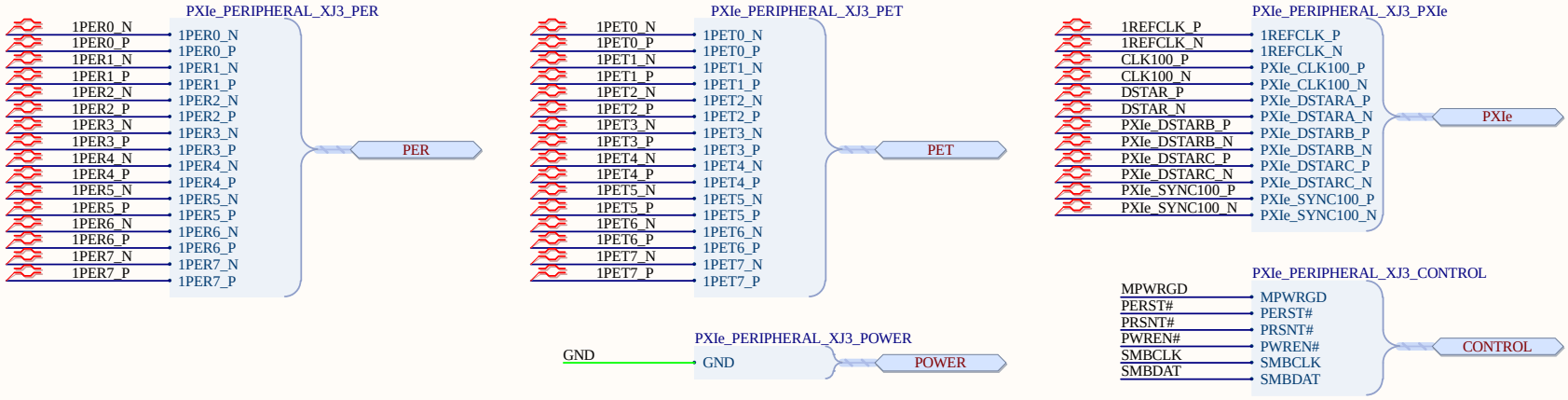


Title		
Size A4	Number	Revision
Date: 3/26/2019	Sheet of	
File: D:\MC_PCB_ver2\LEVEL TRANS.SchDoc	Drawn By:	



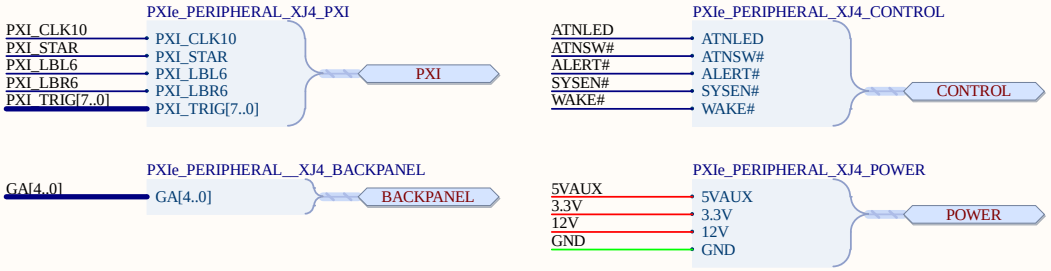
Sheet Title	PXIe PERIPHERAL 3U Top Sheet
Project Title	PXIe PERIPHERAL 3U
Size: A4	Item: / Revision:
Date: 3/26/2019 Time: 11:39:02 AM	Sheet 13 of 15
File: PXIE.SchDoc	

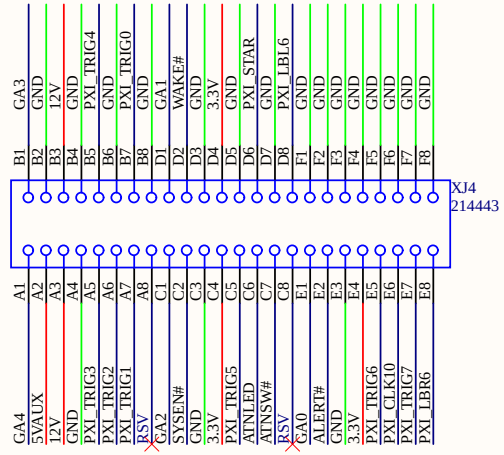
Altium
IBP Shanghai
Level 3 - Building 3
No.168 Linhong Road
Shanghai 200335, China



Sheet Title	***PXIe PERIPHERAL XJ3 90Pin***	
Project Title	***PXIe PERIPHERAL 3U***	
Size: A4	Item:	Revision:
Date: 3/26/2019	Time: 11:39:02 AM	Sheet * of 15
File: CON_PXI_Express_PERIPHERAL_XJ3_90PIN.SchDoc		

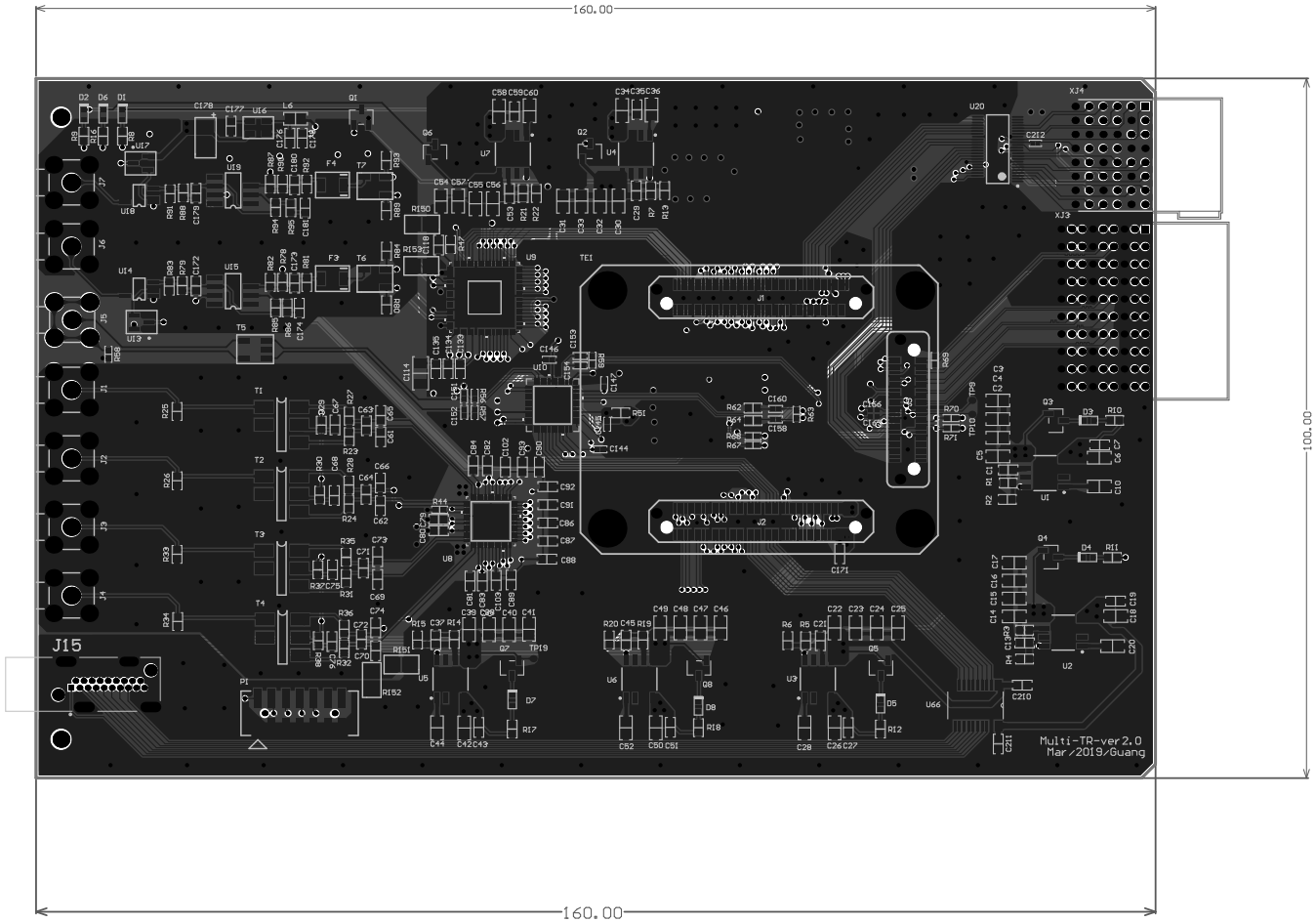
Altium
IBP Shanghai
Level 3 - Building 3
No.168 Linhong Road
Shanghai 200335, China





Sheet Title	***PXIe PERIPHERAL XJ4 48Pin***		
Project Title	***PXIe PERIPHERAL 3U***		
Size: A4	Item:	Revision:	
Date: 3/26/2019	Time: 11:39:02 AM	Sheet * of 15	
File: CON_PXI_Express_PERIPHERAL_XJ4_48PIN.SchDoc			

Altium
IBP Shanghai
Level 3 - Building 3
No.168 Linhong Road
Shanghai 200335, China


160.00
100.00
160.00
J15
Multi-TR-ver2.0
Mar/2019/Guang

Appendix C

Schematics of FPGA Designs

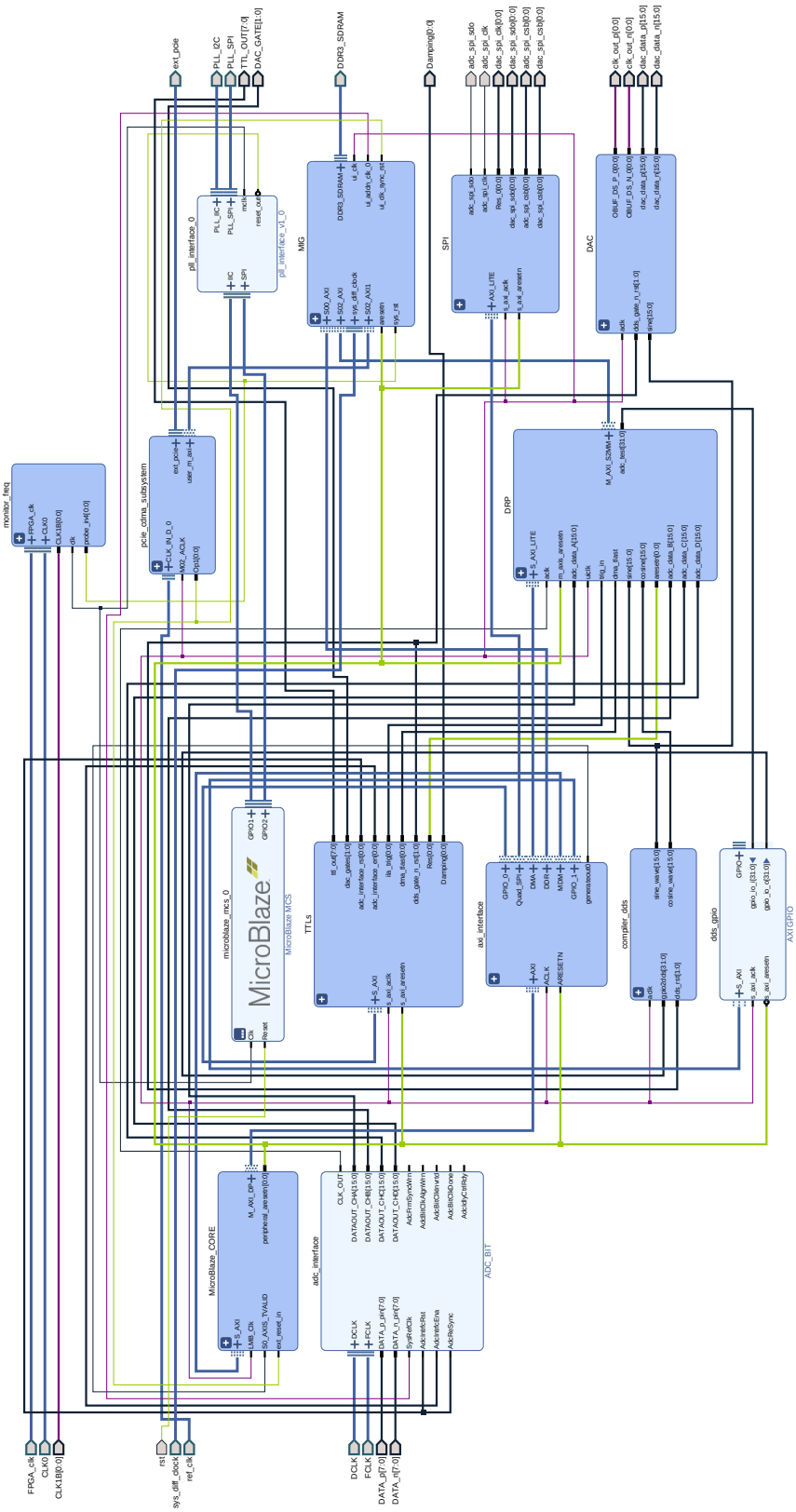


Fig. C.1 Big Picture of FPGA design.

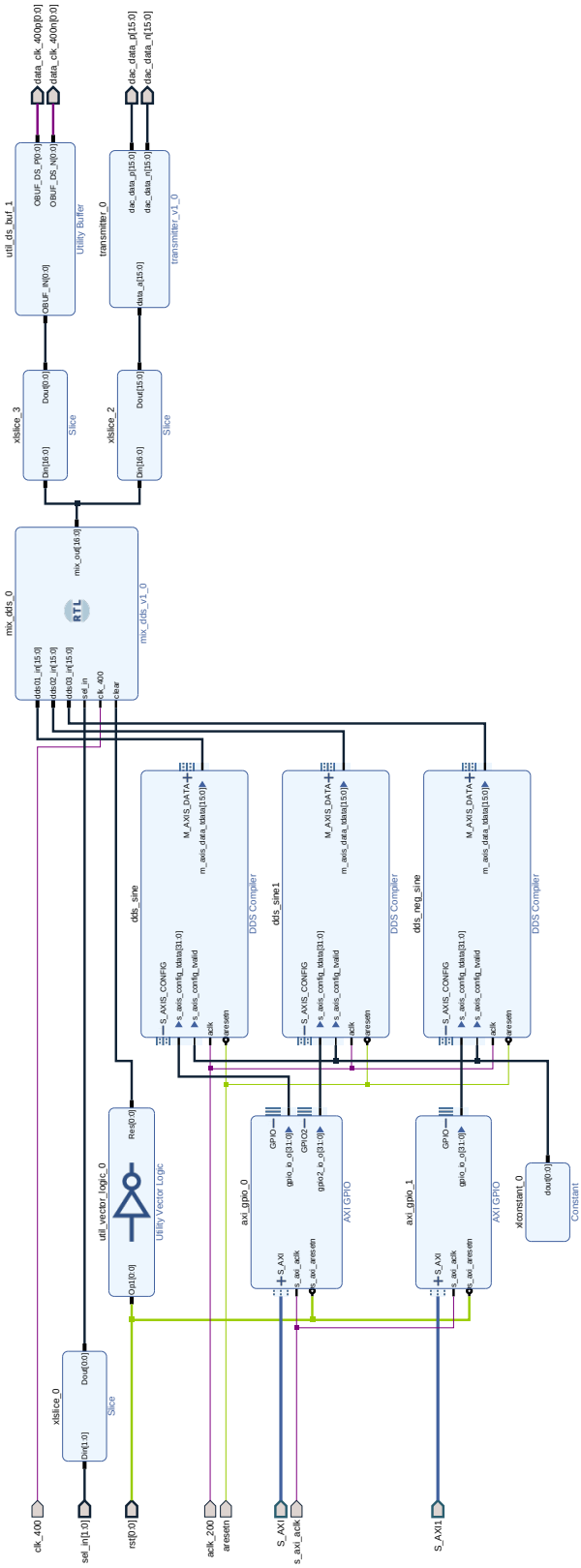


Fig. C.2 DAC FPGA driver.

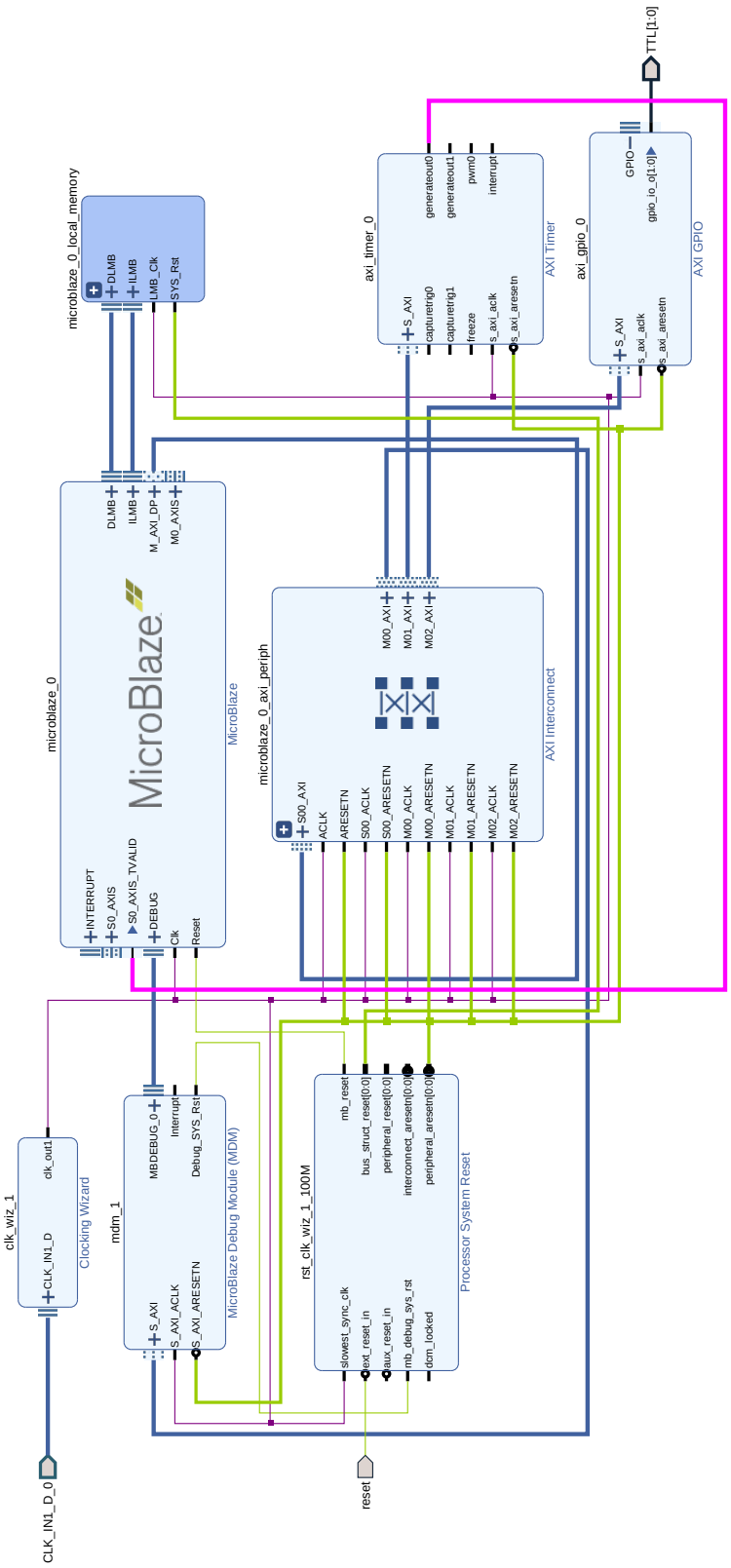


Fig. C.3 FPGA timer.

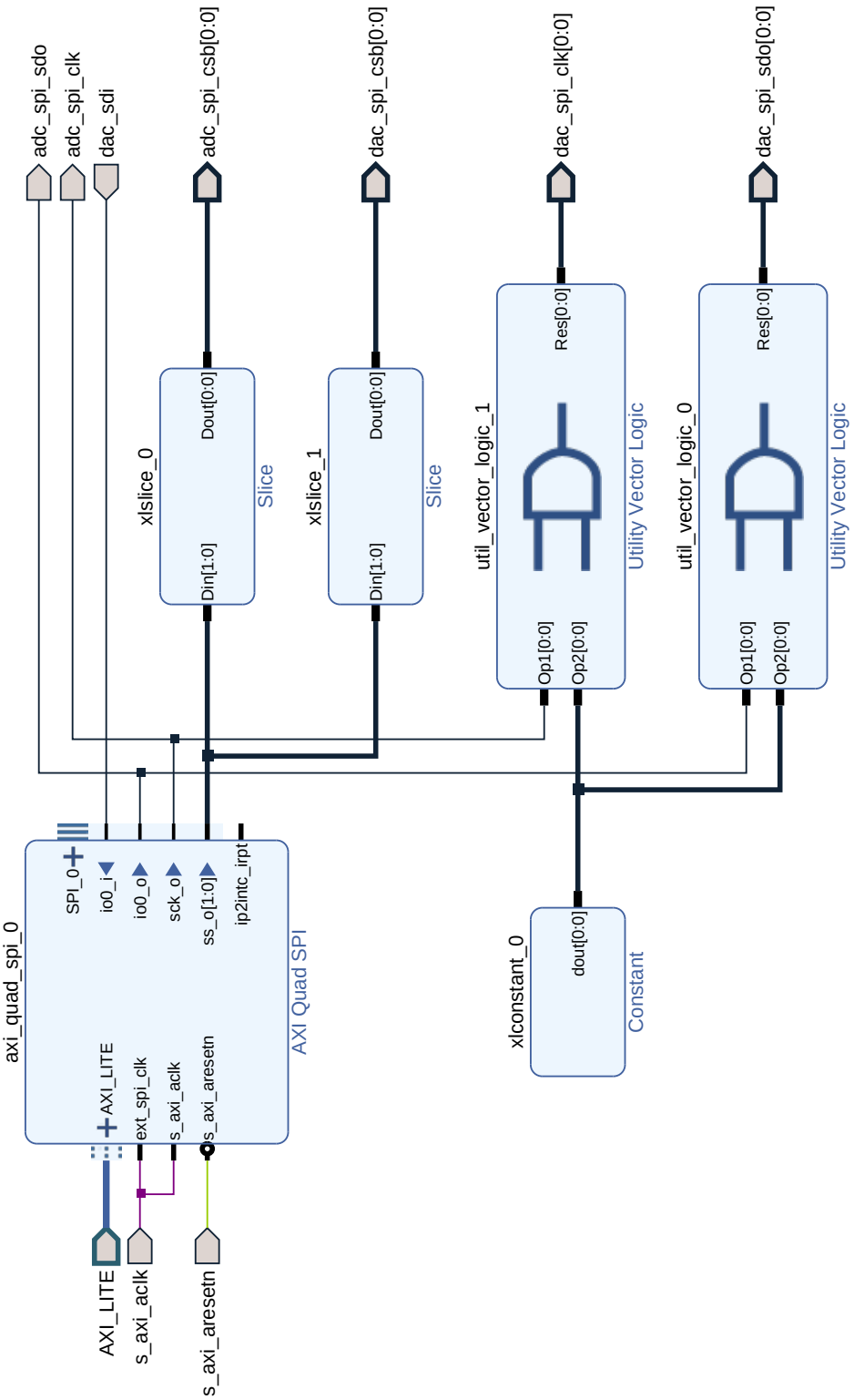


Fig. C.4 SPI FPGA.

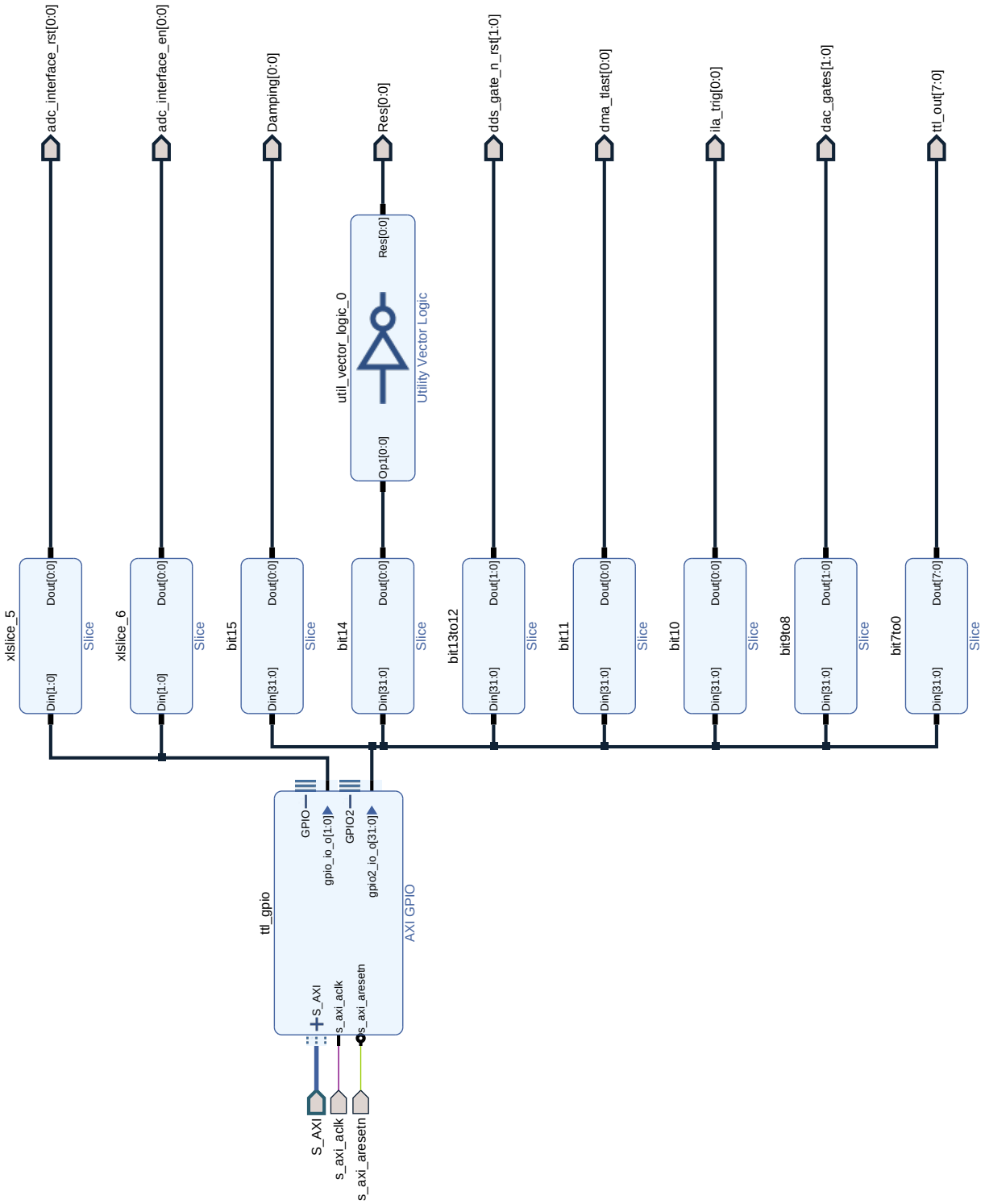


Fig. C.5 TTL FPGA.

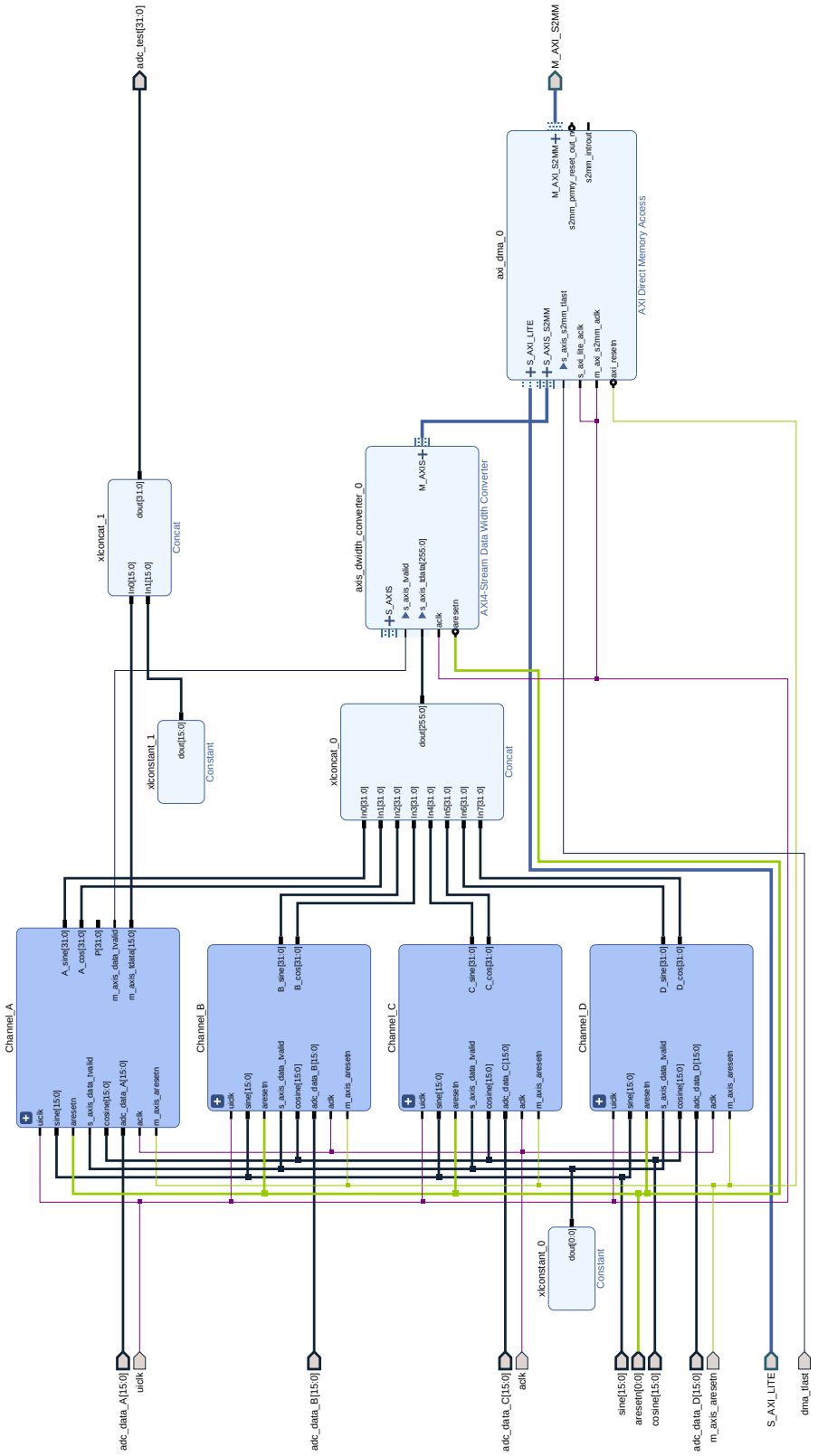


Fig. C.6 DRP data flow.

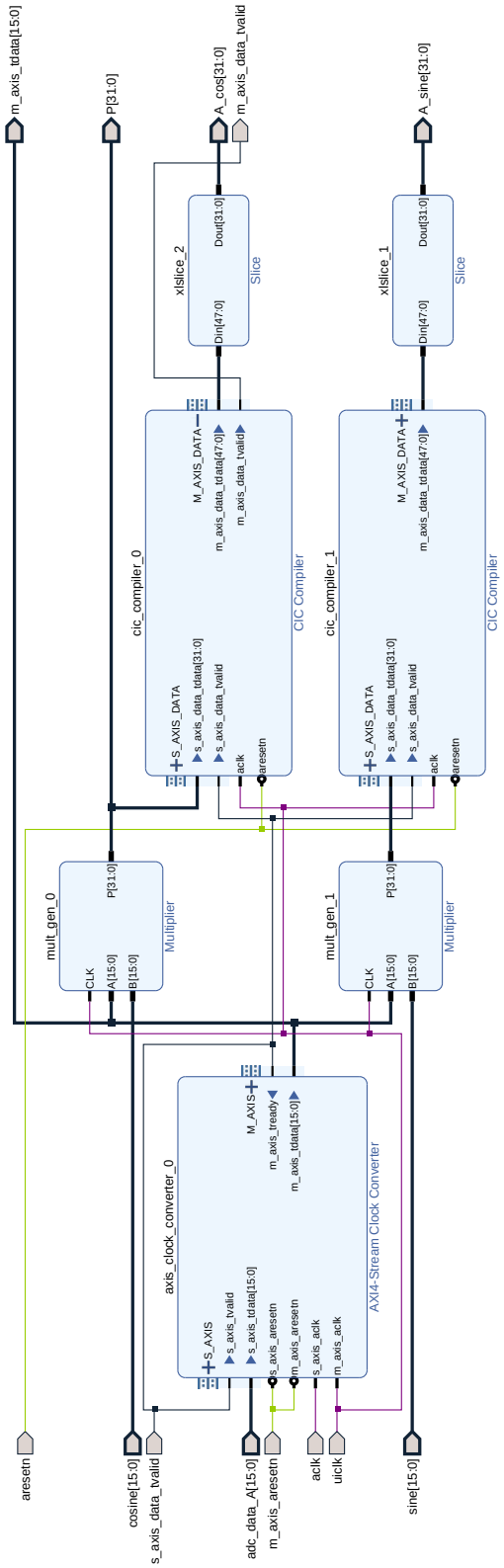


Fig. C.7 DRP FPGA design.

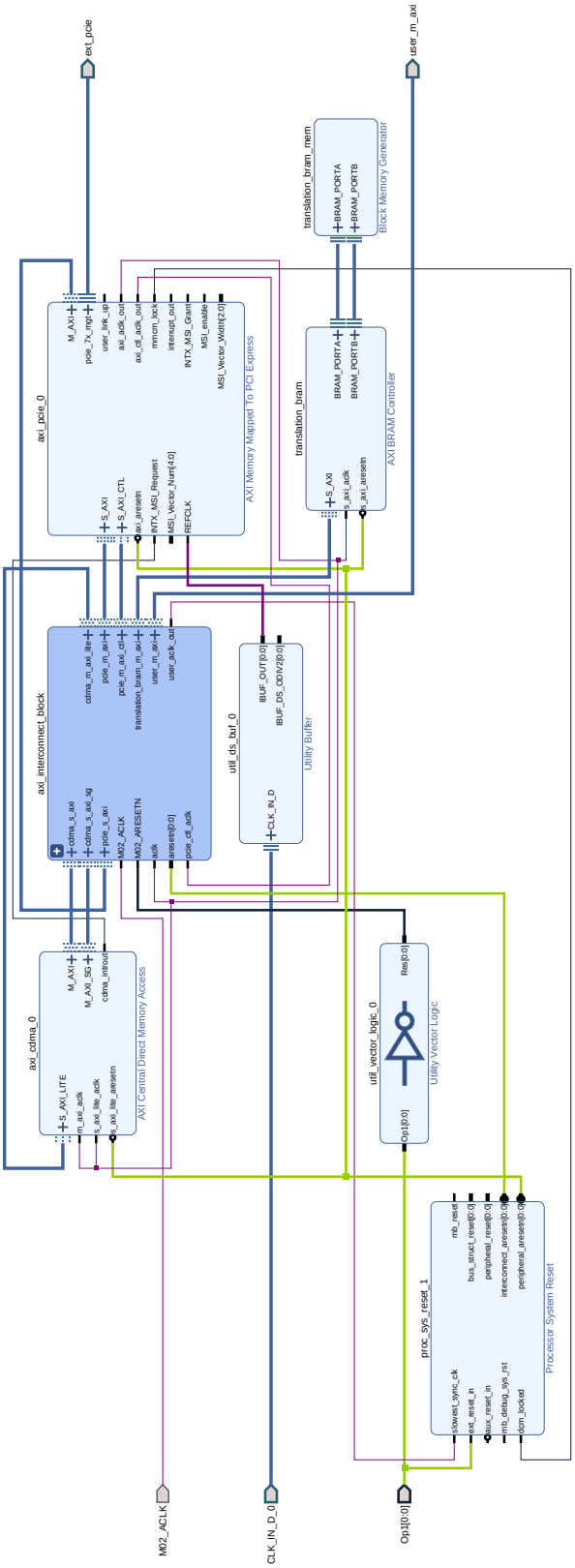


Fig. C.8 CDMA PCIe subsystem.

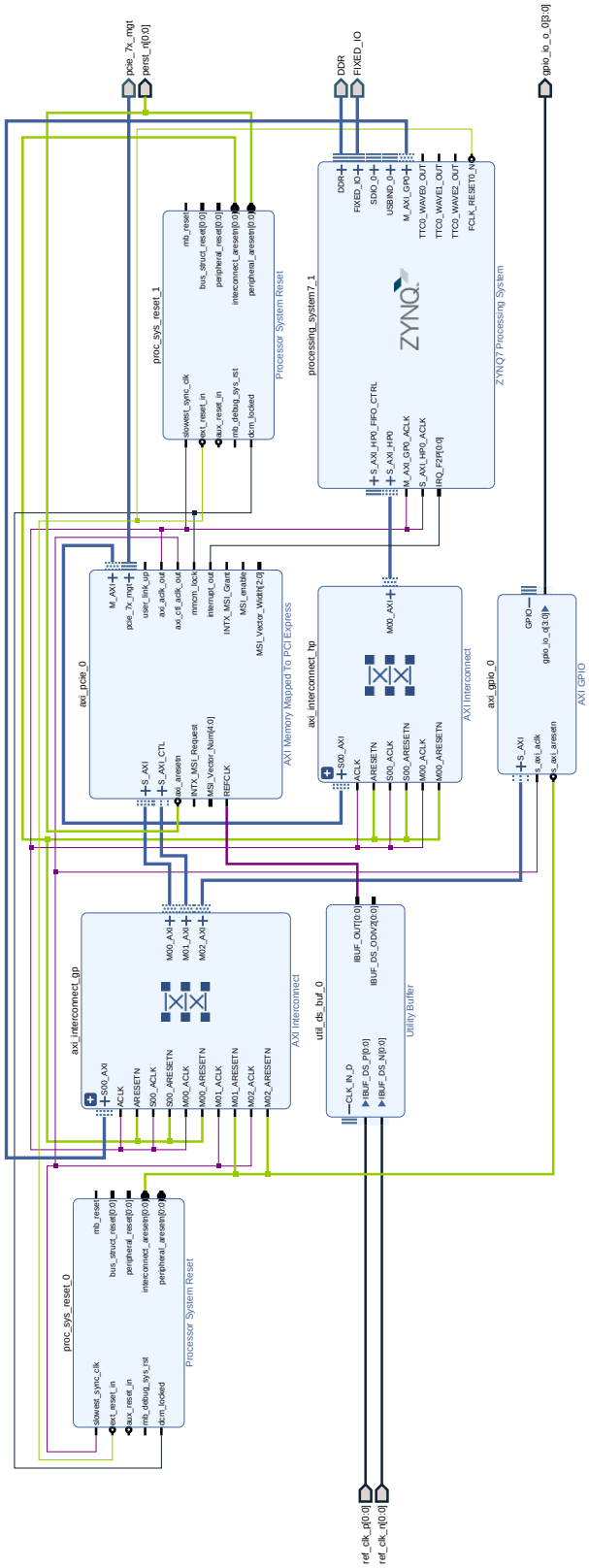


Fig. C.9 FPGA design of system controller.