

UNIVERSITY OF OKLAHOMA
GRADUATE COLLEGE

DESIGN AND DEVELOPMENT OF A SYNTHETIC APERTURE RADAR
VOLUMETRIC IMAGING SYSTEM

A THESIS
SUBMITTED TO THE GRADUATE FACULTY
in partial fulfillment of the requirements for the
Degree of
MASTER OF SCIENCE

By
ANDREW GONZALES
Norman, Oklahoma
2024

DESIGN AND DEVELOPMENT OF A SYNTHETIC APERTURE RADAR
VOLUMETRIC IMAGING SYSTEM

A THESIS APPROVED FOR THE
SCHOOL OF ELECTRICAL AND COMPUTER ENGINEERING

BY THE COMMITTEE CONSISTING OF

Dr. Jay McDaniel, Chair

Dr. Hjalti Sigmarsson

Dr. Justin Metcalf

© Copyright by ANDREW GONZALES 2024

All Rights Reserved.

Acknowledgments

First and foremost, I want to thank my advisor, Dr. Jay McDaniel, for his continuous support throughout the three years we have spent working on this project. Performing research under his guidance has been an amazing and valuable experience for my growth as an aspiring engineer, and was undoubtedly the biggest highlight of my academic career. I also want to thank every student in the OU radar consortium, past and present, for their encouragement, teamwork, and the fun memories made from traveling to conferences. Furthermore, I want to thank the remaining members of my advisory committee, Dr. Hjalti Sigmarsson and Dr. Justin Metcalf, for making this thesis possible.

Next, I want to thank my family, especially my faithful parents, Sandra Gonzales and Anthony Gonzales. I am forever grateful for their love, support, and sacrifices, and for always believing in me. All of this has culminated in me completing this journey. They taught me the importance of education, hard work, and devout faith. Thus, I am proud to share this achievement with them. Finally, I also want to thank all my close friends I met here in Norman. I am forever grateful for their love and support throughout my time studying at the University of Oklahoma. This gratitude also extends to the wonderful community at St. Thomas More University Parish and Student Center.

Table of Contents

Acknowledgment	iv
Table of Contents	v
List of Tables	viii
List of Figures	ix
Abstract	xiii
1 Introduction	1
1.1 Thesis Overview	3
2 Background	5
2.1 Automotive Radar	5
2.2 Synthetic Aperture Radar	7
3 Hardware Setup	11
3.1 Motion Control System	11
3.2 Synthetic Aperture Design	14
3.3 Homing System Settings	19
3.4 Automotive Radar Module	21
3.5 Data Capture Adapter	24

3.6	SAR System Setup	27
4	Radar Software Configuration	32
4.1	Radar Connection and Installation	32
4.2	mmWave Studio	34
4.2.1	Startup and Connection	35
4.2.2	Static Configuration	37
4.2.3	Data Configuration	37
4.2.4	Sensor Configuration	38
4.2.5	PostProc Tool	41
5	Control System Design	44
5.1	System Overview	44
5.2	LabVIEW GUI	49
5.2.1	GUI Startup Section	50
5.2.2	GUI Input Parameters Section	51
5.2.3	GUI Synthetic Aperture Section	52
5.2.4	GUI SAR Image Section	53
5.2.5	GUI Sensor Configurations Section	54
5.2.6	GUI Toggles & Output Windows Section	56
5.3	SAR System Sequence	57
5.3.1	SAR Data Capture Sequence	60
5.3.2	Position Reset Sequence	63
5.3.3	SAR Data Post-Processing Sequence	66
5.3.4	Vertical Positioning Sequence	71
5.3.5	System Reset Sequence	73
5.4	Summary	74

6	SAR Post-Processing	75
6.1	Radar Baseband Data	75
6.2	Data Cube Processing	77
6.3	Backprojection	79
6.3.1	Image Formation Process	79
6.3.2	Backprojection Algorithm	80
6.4	SAR Imaging	81
7	Volumetric SAR Imaging	83
7.1	Target Scene: Four Scattered Targets	83
7.2	Target Scene: A Single Corner Reflector	89
7.3	Target Scene: Four Assorted Corner Reflectors	90
8	Conclusion and Future Work	94
	References	97

List of Tables

3.1	V-Slot® NEMA 23 linear actuator bundle (lead screw) parts list . . .	14
3.2	OpenBuilds hardware list.	18
3.3	Limit switch configuration.	19
3.4	Homing configuration settings.	20
3.5	Gcode commands.	21
6.1	The parameter values for performing post-processing.	76

List of Figures

2.1	Visual diagram of automotive radar sensors in automobiles [1]. . . .	7
2.2	Geometry of traditional SAR [2].	8
2.3	SAR image and optical photograph of Albuquerque Airport from a top-down perspective [3].	9
3.1	Schematic of TinyG board.	12
3.2	The TinyG board connected to a 24-V power supply.	13
3.3	NEMA 23 Stepper Motor.	13
3.4	Linear actuator with slide as advertised on OpenBuilds.	14
3.5	Motor plate for mounting the vertical actuator.	15
3.6	Vertical actuator mounted onto the horizontal actuator.	16
3.7	OpenBuilds limit switch.	17
3.8	Limit switch mounted onto the end of a linear actuator.	18
3.9	Front side of the AWR2243EVM board [4].	22
3.10	Back side of the AWR2243EVM board [4].	23
3.11	Simplified representation of AWR2243EVM device [5].	24
3.12	Front side of the DCA1000EVM [6].	26
3.13	Back side of the DCA1000EVM [6].	26
3.14	DCA1000EVM functional block diagram [6].	27

3.15	The AWR2243EVM and DCA1000EVM connected as a single radar module device [6].	28
3.16	The radar module mounted onto the SAR track system.	29
3.17	The complete SAR system setup with the radar module attached. . .	30
3.18	Block diagram of the SAR system hardware setup.	31
4.1	COM Ports for the SAR system.	33
4.2	IP address for Ethernet connection to DCA1000EVM.	34
4.3	Connection window in mmWave Studio upon startup completion. . .	36
4.4	DCA1000EVM configuration window.	36
4.5	Static Configuration window.	37
4.6	Data Configuration window.	38
4.7	Visualization of FMCW radar signal transmission [7].	40
4.8	Sensor Configuration window with user-defined parameters.	41
4.9	Post-processing GUI with four characterization plots.	43
5.1	Block diagram of how LabVIEW interfaces with the SAR system. . .	45
5.2	Default view of the LabVIEW GUI.	46
5.3	VISA instrumentation for interfacing with the TinyG board.	47
5.4	LabVIEW controlling mmWave Studio.	48
5.5	LabVIEW interfacing with mmWave Studio to perform a data capture. .	49
5.6	Closeup view of the LabVIEW GUI for user operations.	50
5.7	Launching mmWave Studio from LabVIEW.	51
5.8	Renaming captured dataset to filename defined by user.	52
5.9	Defining the length of the horizontal synthetic aperture distance. . .	53
5.10	Sequence for configuring the radar module from LabVIEW.	55
5.11	Block diagram of the SAR data capture sequence by case name. . .	58
5.12	Overview of the SAR data capture code and the “Initialize” case. . .	60

5.13	The “Init_Away” case.	61
5.14	The “Move_Away” case.	62
5.15	The “Check_Away_Status” case.	63
5.16	The “Init_Back” case.	64
5.17	The “Move_Back” case.	65
5.18	The “Check_Back_Status” case.	65
5.19	Block diagram of the LabVIEW sequence for performing post processing.	66
5.20	The “Start” case for the “Backprojection” case.	67
5.21	The “MATLAB_2D” case.	68
5.22	The “MATLAB_3D” case.	69
5.23	The “Image” case.	70
5.24	The “Finish” case.	71
5.25	The “Init_Up” case.	71
5.26	The “Move_Up” case.	72
5.27	The “Check_Up_Status” case.	72
5.28	The “Reset” case.	73
5.29	The “Terminate” case.	74
6.1	The DCA1000EVM captured data format [7].	77
6.2	The baseband data format in MATLAB.	77
6.3	Radar data cube format constructed from SAR data [8].	78
7.1	Target scene setup of four different targets.	84
7.2	Volumetric SAR image of the target scene without sidelobe suppression (top) and with sidelobe suppression (bottom).	85

7.3	Target scene and corresponding SAR image side-by-side for comparison both without sidelobe suppression (top) and with sidelobe suppression (bottom).	86
7.4	Alternate viewpoint of the target scene with each target labeled and its position measured.	87
7.5	The range-elevation plane (top) and range-azimuth plane (bottom) of the SAR image.	88
7.6	Target scene setup of a single corner reflector.	89
7.7	Volumetric SAR image of corner reflector with the increased pixel resolution.	90
7.8	Target scene setup of four corner reflectors in different positions. . .	91
7.9	Volumetric SAR images of the corner reflector target scene with a threshold of -10 db (top) and a threshold of -23 dB (bottom).	92
7.10	The range-elevation plane (top) and range-azimuth plane (bottom) of the SAR image.	93
8.1	The multi-IMU device mounted onto the SAR Demonstrator.	96

Abstract

Synthetic aperture radar (SAR) is a widely used technique for creating high-resolution images of scenes. While SAR is primarily used for 2-D imaging, there is growing interest in developing SAR imaging for 3-D reconstructions of objects and target scenes. This can be accomplished by adding a vertical motion dimension to the synthetic aperture setup, enabling elevation compression alongside azimuth compression. By employing two orthogonal linear actuators, high-resolution imaging in all three dimensions becomes possible. This thesis describes the design and development of a track-mounted SAR system leveraging a low-cost wideband automotive radar to form high-resolution volumetric images of a near-range target scene. The automotive radar industry has transitioned to operating in the 76 to 81 GHz band, which is ideal for applications requiring short-range target detection. Therefore, this system operates at 77 GHz, enabling precise object location in 3-D space without the drawbacks associated with large bandwidths over long distances. The setup incorporates an automotive radar module, an in-house signal processing toolkit, and a control system designed to autonomously and accurately position the radar evaluation module for sending and receiving radar pulses. These pulses and their corresponding positions are processed using the backprojection algorithm to generate a volumetric image of the target scene.

Chapter 1

Introduction

A radar is a system designed to detect the presence, direction, distance, and speed of objects by emitting pulses of high-frequency electromagnetic waves that reflect off objects and return to the source. One such application is automotive radar, which is typically used for short-range systems such as object detection and collision avoidance in autonomous vehicles [9]. These radars are low cost and have wide bandwidth, providing very good range resolution. Automotive radars operate at very high frequencies within the wideband of 76 to 81 GHz which has become the band of choice in the automotive radar industry [10][11]. This allows for detection systems employing this type of radar with the ability to locate objects with more precision without the negative effects of large bandwidths over long distances. The small size and light weight of these radar modules allow for ease of integration with other systems and applications.

SAR is an application of radar for creating images, typically for the 2-D reconstruction of objects. SAR utilizes the movement of the radar antenna over the target area to create a synthetic array [2]. The motion of the radar antenna over a target region to provide finer spatial resolution is the main advantage of SAR over

conventional stationary radars [3]. The distance the antenna travels while transmitting and receiving radar pulses forms a large synthetic aperture, which allows for the creation of high-resolution images [3]. A larger synthetic aperture results in higher-resolution images. Developing SAR imaging applications has become a prominent area of research in the industry, but usually for 2-D images. The use of SAR for creating 3-D reconstructions of objects remains an underdeveloped yet growing research topic with untapped application possibilities.

This thesis proposes a SAR system design for imaging 3-D reconstructions of objects in a volumetric space. A linear actuator with a slide is used to construct the synthetic array in the form of a track-mounted system. The purpose of this system design is to enable very precise position tracking during the radar data capture because of the very short wavelength. This produces accurate positioning of the radar over the pulsing interval for data captures. Using this design, the radar will be able to send out pulses while moving horizontally along the actuator. This positioning allows for high resolution in the horizontal plane of the target area but does not focus on the vertical dimension. While motion in one direction enables the formation of 2-D SAR images, motion in two dimensions can facilitate the creation of 3-D SAR images. This can be solved through the addition of a vertical motion dimension to a synthetic aperture setup. This enables elevation compression in addition to azimuth compression, which leads to 3-D radar imaging capabilities. To accomplish this, two actuators are positioned orthogonally to provide high resolution for all three dimensions. After the system receives pulses from the target scene, a post-processing algorithm that applies backprojection is used to create a volumetric image. The final SAR system design implements a 77 GHz automotive radar module and can

perform high-resolution volumetric SAR imaging of objects in near-range target scenes. The volumetric SAR system in its entirety will be referred to as the “SAR Demonstrator” throughout this thesis.

1.1 Thesis Overview

This thesis is separated into the following chapters. In Chapter 2, a background of both automotive radar and synthetic aperture radar is given. This briefly covers the history of both topics and their relevance in today’s society as well as the design of the SAR Demonstrator.

In Chapter 3, the hardware setup for the SAR Demonstrator is given. This includes the motion control system, the physical components of the track-mounted system, and its assembly. The radar module used in the design is a commercially-available automotive radar system from Texas Instruments. The linear actuators and motors are designed to autonomously provide motion to the radar module allowing for the creation of the synthetic array.

In Chapter 4, the radar software configuration is discussed. This chapter describes how mmWave Studio is connected to the radar module and the configuration sequence that follows. The steps for performing a data capture are also described.

In Chapter 5, the program for the control system is described. The program is designed to accurately position an automotive radar evaluation module to send and receive a series of continuous radar pulses. A GUI for the control system is developed in LabVIEW, which permits the automation and control that is required to move the actuators and appropriately time-align the entire configuration.

In Chapter 6, the post-processing sequence of captured datasets is discussed.

The datasets containing the pulses and corresponding positions are reformatted into a radar data cube. The data is then processed using the backprojection algorithm to create a high-fidelity volumetric SAR image of the target scene.

In Chapter 7, the resulting volumetric SAR images for three different target scene setups are shown and discussed. Every SAR image is constructed by plotting every pixel in the target scene that does not exceed a given threshold to display targets. Also, each target scene setup is designed to showcase a series of post-processing attributes for imaging.

In Chapter 8, the conclusion to the thesis and proposals for future work on the SAR Demonstrator are discussed. This chapter describes how the design of the SAR Demonstrator for imaging close-range, stationary objects can serve as a prototype for the development of future SAR volumetric imaging systems. Also, this chapter describes how the SAR Demonstrator can be used in the future to test state-of-the-art navigation equipment for SAR image formation at high frequencies.

Chapter 2

Background

2.1 Automotive Radar

In the early 1970s, the development of automotive radar was introduced, conceptualized for vehicles by offering the potential for automobile safety features [9]. High-end vehicles in the United States and Europe had access to these systems by the late 1990s. Automotive radar technology would later evolve to a wider variety of applications, ranging from military to law enforcement, and commercial use. The first generation of automotive radar sensors emerged by the year 2000, followed by a rapid development of new radar sensor generations [12]. Early systems operated on a range of frequencies and used various modulation schemes, but the majority centered around 24 GHz due to their perceived ease of design and fabrication as well as the wider availability of components [9].

Since then, the automotive radar industry has been driven to adopt higher frequency bands, recently jumping to the 77-81 GHz range with a wider bandwidth of 4 GHz [10][11]. This allows for automotive radar systems to utilize a series of frequency-modulated continuous-wave (FMCW) signals and remains an exten-

sively researched field in the industry [13]. The 77-81 GHz operating frequency range is well-suited for short-range automotive radar applications since the shorter distance improves the radar's ability to accurately detect objects, measure distances, and improve range resolution. Also, the antenna for a 77 GHz system is small in size, allowing for relatively simple integration with automotive systems. Having a wide bandwidth allows for more precise detection of target locations, making it easier to distinguish closely spaced targets [14]. Furthermore, wide-bandwidth automotive radar systems remain compliant with FCC Chapter 15 regulations [11].

Mainstream automotive radars have limitations in angular and/or range resolution [12] [15]. These are primarily used for basic object detection in assisted-driving features such as adaptive cruise control and emergency braking as shown in Figure 2.1. One solution to enhance imaging resolution is by adding more antennas/channels, but this comes with increased hardware costs, particularly in millimeter-wave (mmWave) systems [15]. However, advancements in system-on-chip technology have significantly improved the development of affordable FMCW mmWave radars [16]. These recent developments offer great potential for cost-effective imaging solutions across various applications [17]. A new line of commercial-off-the-shelf (COTS) radar sensors developed by Texas Instruments (TI) have been designed for commercial applications in the 77-81 GHz frequency band [10]. These sensors benefit from a compact size while operating within the automotive band, providing a range of sensor options along with different antenna setups for testing purposes.

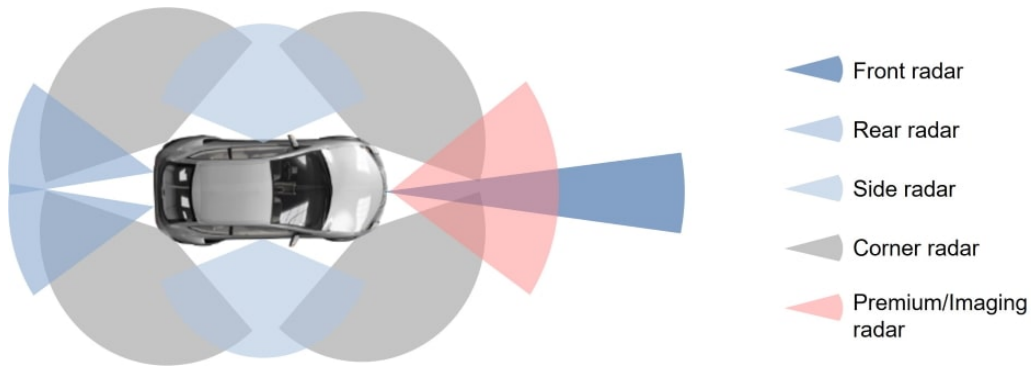


Figure 2.1: Visual diagram of automotive radar sensors in automobiles [1].

2.2 Synthetic Aperture Radar

SAR is a remote sensing technique consisting of a combination of radar hardware, signal processing, and relative motion for image formation [3]. This allows for the reconstruction of detectable objects and target scenes in both two dimensions and three dimensions. A traditional concept of SAR consists of forming a large synthetic antenna aperture from the motion of a radar module to achieve high-resolution images [2]. Since its introduction, SAR has been widely employed for a wide range of applications such as land mapping and identifying fixed targets [3]. The geometry of traditional air-borne SAR is shown in Figure 2.2. This technique is often employed for stationary targets and is limited to two-dimensional plots from a top-down perspective. An example of high-resolution 2-D SAR imaging in comparison to an optical photograph is shown in Figure 2.3.

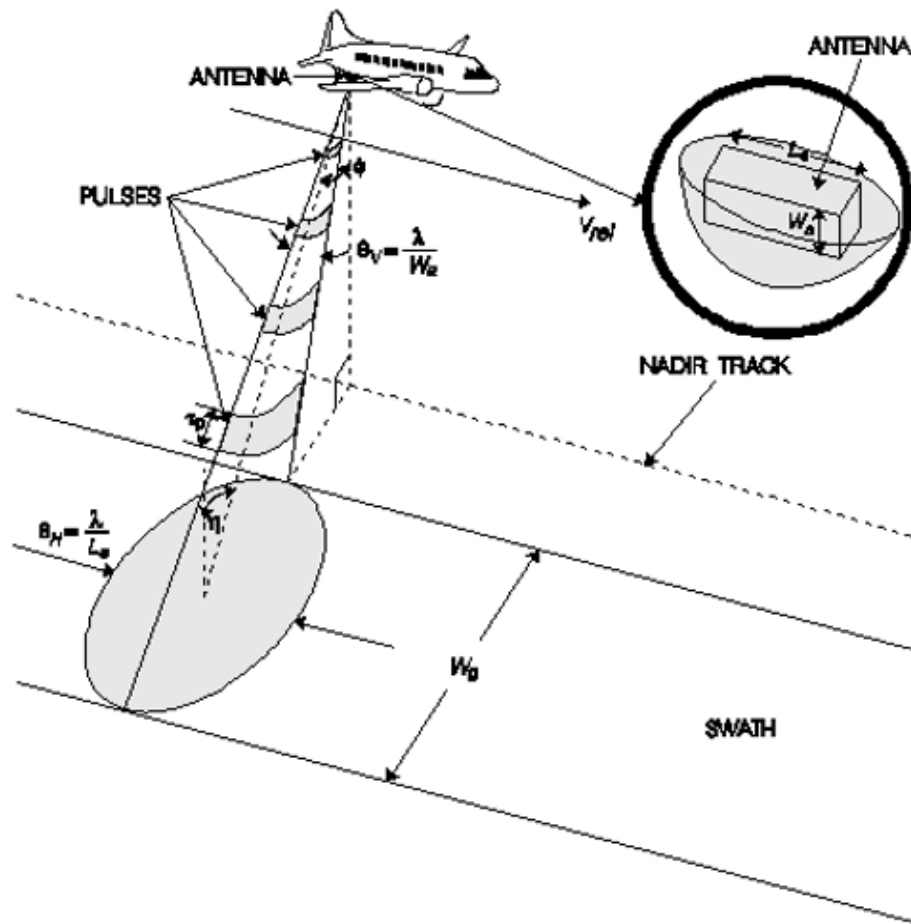


Figure 2.2: Geometry of traditional SAR [2].

While SAR has been utilized in space-borne and air-borne radar imaging for many years, its application in the automotive industry has been virtually nonexistent [18]. Standalone automotive radars are not typically designed with SAR processing capabilities. However, recent radar designs have been developed to allow for ease of integration with imaging applications, including SAR. The practicality of using an automotive radar for high-resolution imaging using SAR processing has been explored, resulting in successful 2-D images of targets [17].

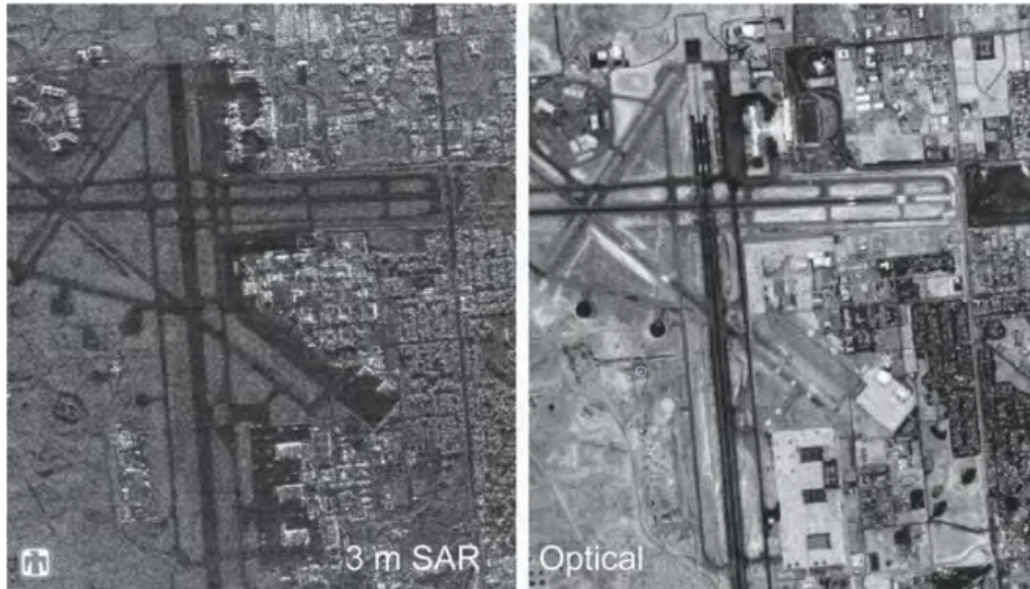


Figure 2.3: SAR image and optical photograph of Albuquerque Airport from a top-down perspective [3].

The previously mentioned COTS radar sensors by TI have been applied to the development of 2-D SAR imaging systems [17] [16]. One such system design includes a ground-based synthetic array positioner that performs both SAR imaging and target classification [14]. The system consists of a mechanical positioner and control system designed for target detection in a three-dimensional space. To achieve high-resolution SAR images, it is crucial to have precise knowledge of the radar's position as it travels across the synthetic aperture. Ideally, this position should be accurately known within a fraction of the carrier wavelength [15]. A frequency of 77 GHz puts the wavelength at only a few millimeters. SAR motion measurements need to be on the order of the wavelength to precisely locate the distance the pulses have to travel [14]. This justifies the use of a track-mounted system for its position-accurate motion, which allows for the creation of high-resolution

SAR images using a 77 GHz automotive radar.

For this thesis, the mechanical positioner and control system design has been selected as the framework for a track-mounted SAR system designed to perform volumetric imaging. The use of two orthogonal linear actuators enables high resolution in both cross-range dimensions. By using multiple actuators, resolution can be achieved in all three dimensions, enabling the creation of a volumetric image of the scene. Developed 3-D SAR imaging track-mounted systems employ multiple input and multiple output arrays for creating holographic reconstructions of objects in a 2-D space [19] [20]. This thesis proposes a SAR system designed for imaging 3-D reconstructions of objects in a volumetric space, consisting of the range, azimuth, and elevation planes. This also allows for measuring the locations of multiple targets in a single 3-D target scene. The overall setup in this thesis utilizes a 77 GHz automotive radar, a SAR imaging system setup, a signal processing toolkit, and the implementation of a control system. The first component covered in this thesis is the hardware and physical setup of the SAR Demonstrator, showcased in Chapter 3.

Chapter 3

Hardware Setup

In this chapter, the hardware setup for the SAR Demonstrator is covered in full detail. The motion control system setup is covered along with the design of the physical synthetic aperture. The hardware of the track setup is described to showcase the design of the vertical component for the synthetic aperture setup. Furthermore, the homing system design is revealed to showcase how the motion control system interfaces with the SAR Demonstrator. Detailed descriptions of the TI AWR2243EVM radar module and DCA1000EVM capture adapter are discussed within this chapter as well as how they are integrated into the SAR Demonstrator.

3.1 Motion Control System

The TinyG computer numerical control (CNC) board is a high-performance, USB-based 6-axis motion control system. The controller supports XYZ linear and ABC rotary axes with four motor outputs and implements the NIST RS274NGC dialect of Gcode. Additional features includes GPIO ports for limit and homing switches and real-time status output reports. This board is designed to be a complete embedded solution for small CNC applications that require highly controllable

motion control. A diagram of the TinyG board is shown in Figure 3.1.

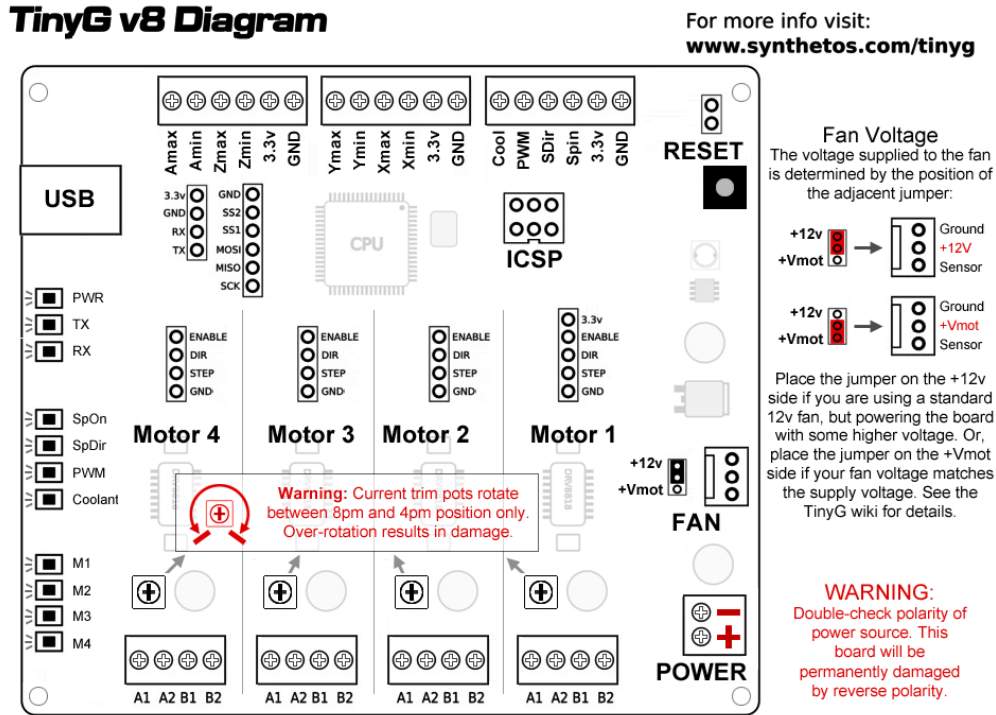


Figure 3.1: Schematic of TinyG board.

The TinyG board requires a 24-V input voltage which is provided by a B&K Precision 9130 Triple Output Programmable DC Power Supply. Every ground pin on the TinyG board links is connected. The TinyG board setup while connected to the power supply is shown in Figure 3.2.

The SAR system consists of two linear actuators, each connected to a NEMA 23 stepper motor that is controlled by a single TinyG board. These motors receive instructions from the TinyG USB port, which establishes communication with the host computer. The NEMA 23 stepper motor is a hybrid stepper motor capable of employing precise control and position-accurate tracking over rotational motion. This motor has a positional accuracy of 0.091mm and is shown in Figure 3.3.

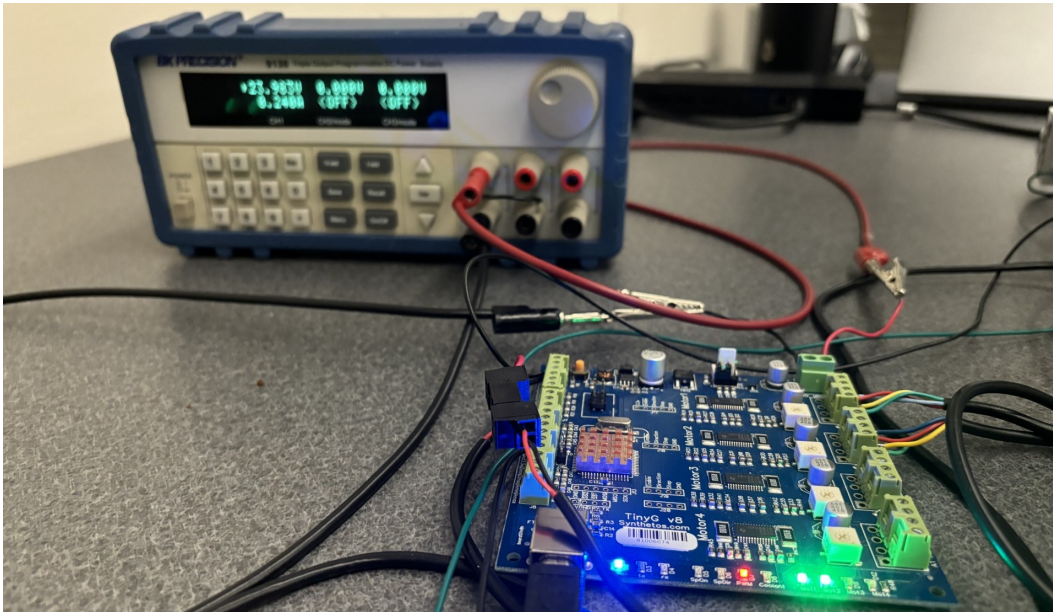


Figure 3.2: The TinyG board connected to a 24-V power supply.



Figure 3.3: NEMA 23 Stepper Motor.

3.2 Synthetic Aperture Design

To construct the horizontal synthetic aperture, a 1-meter linear actuator was selected for the radar module to travel across. The OpenBuilds V-Slot® NEMA 23 Linear Actuator Bundle was used since this bundle contains all the hardware necessary to construct the horizontal actuator of the SAR system. This bundle when assembled consists of a gantry cart and a 1-meter track for the cart to travel across. Also included is a NEMA 23 Stepper Motor that provides motion to the cart when both are connected to a threaded rod. The assembled actuator is shown in Figure 3.4. The complete list of hardware contained in this bundle is listed in Table 3.1.

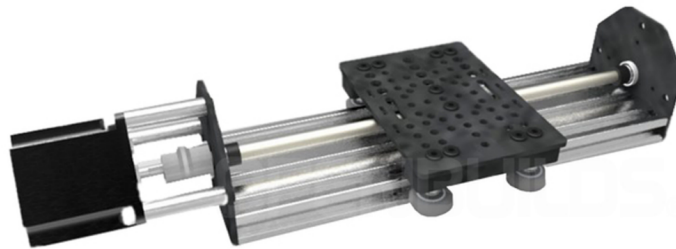


Figure 3.4: Linear actuator with slide as advertised on OpenBuilds.

Qty	Part Name
1	V-Slot Linear Rail
1	8mm Metric Acme Lead Screw
2	Threaded Rod Plate (NEMA 23)
1	NEMA 23 Stepper Motor
1	V-Slot Gantry Plate (Universal)
4	Xtreme Solid V Wheel
1	0.25in x 8mm Flexible Coupling
1	Anti-Backlash Nut Block
1	Assembly Hardware Kit

Table 3.1: V-Slot® NEMA 23 linear actuator bundle (lead screw) parts list

The same bundle but with a 0.25-meter track serves as the vertical actuator of the SAR system. This track length significantly improves stability and reduces vibration while in motion. The cart attached to the vertical actuator serves as the mount for the radar module. By extension, this linear actuator with the radar module attached is intended to be mounted onto the horizontal track. This indicates that the vertical actuator will always be in motion while the horizontal actuator is running.

There are two dimensions that each actuator represents for the system. The 1-meter horizontal actuator corresponds to the X-axis and the 0.25-meter vertical actuator corresponds to the Y-axis. To mount the vertical actuator onto the cart of the horizontal actuator, a custom-built plate was designed to hold the vertical actuator in place as shown in Figure 3.5. The plate is a simple aluminum piece cut from an aluminum sheet that is 3.50 mm thick. The dimensions of the aluminum plate are identical to the dimensions of the gantry cart plate. The center of the aluminum plate has a large hole cut out to fit the outer rim for the motor of the vertical actuator along with four small screw holes that connect the motor and plate.

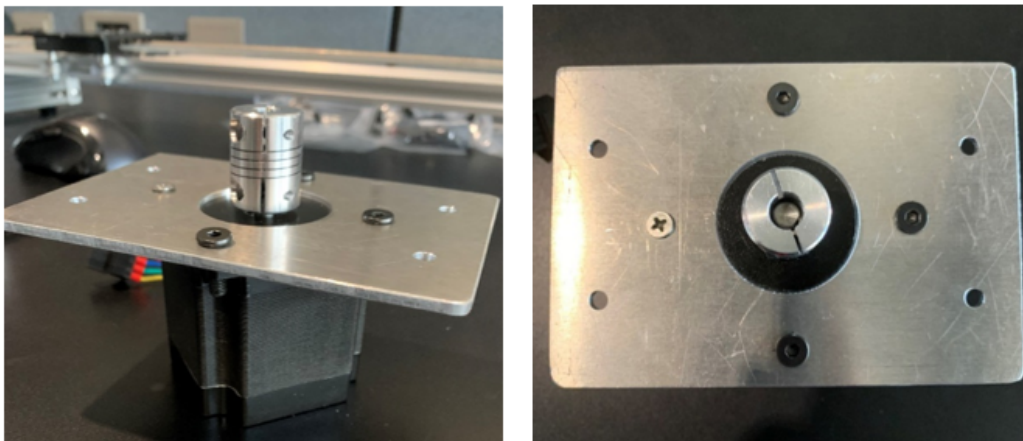


Figure 3.5: Motor plate for mounting the vertical actuator.

The motor is the heaviest component of the vertical actuator. If the radar module vibrates during data captures, image quality would be lost as a result. Therefore, the end of the linear actuator with the motor was positioned on the base of the gantry cart belonging to the horizontal actuator with the track mounted in a vertical orientation. This keeps the center of mass weighted on the cart and ensures the vertical actuator and radar module remain steady while in motion. To connect the motor and linear rail, the plate is designed to hold the motor on one side and the threaded rod plate on the other side to mount the vertical actuator in a fixed position. The plate also has four small screw holes cut out in each corner that connect to standoffs for mounting the plate onto the gantry cart of the horizontal actuator. The assembled mounting for the vertical actuator is shown in Figure 3.6.

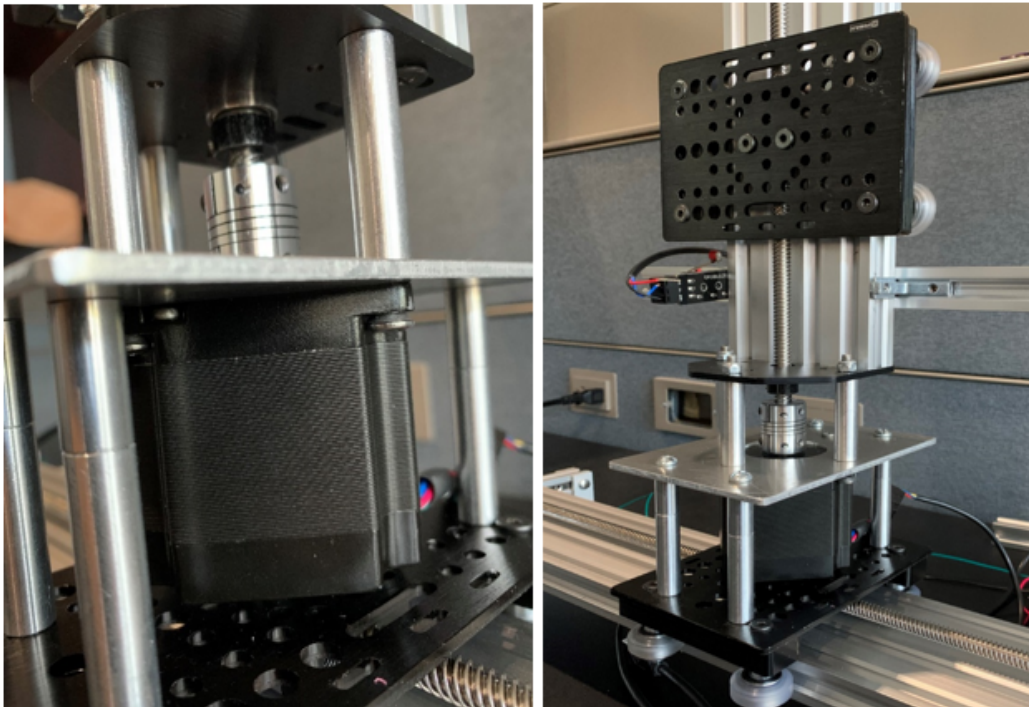


Figure 3.6: Vertical actuator mounted onto the horizontal actuator.

A controlled procedure is required to ensure the gantry carts follow an exact motion pattern while traveling along the actuators. Furthermore, fail-safe triggers for both actuators were necessary to ensure the gantry carts would not get pushed into the threaded rod plates at the end of their track in the event of a system malfunction. Therefore, a homing motion system approach was selected for the TinyG board to utilize.

OpenBuilds limit switches were selected to provide the feedback necessary for a homing motion system as shown in Figure 3.7. A total of four switches were used for the system. Two switches were installed at each end of the horizontal actuator to serve as the X-axis homing switches. Likewise, the remaining two switches were installed at each end of the vertical actuator to serve as the Y-axis homing switches. These limit switches are attached using 250 mm linear rails with corner brackets holding these pieces in place as shown in Figure 3.8.



Figure 3.7: OpenBuilds limit switch.

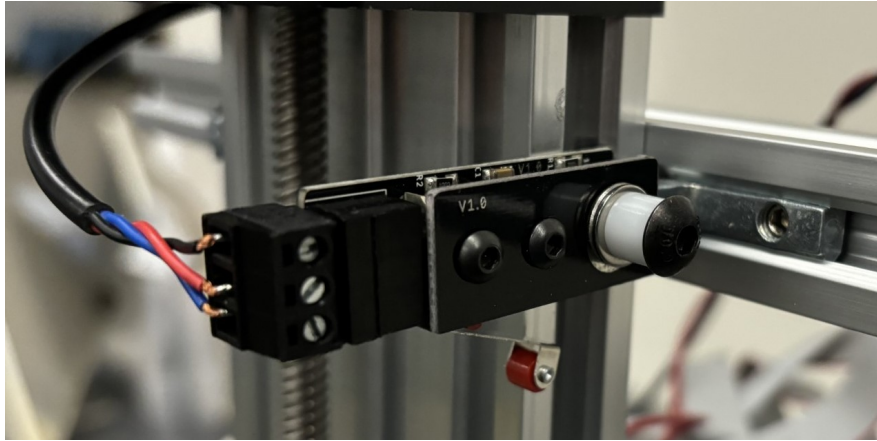


Figure 3.8: Limit switch mounted onto the end of a linear actuator.

This completes the hardware architecture of the SAR system. A complete list of hardware ordered from OpenBuilds that was used in the design and development of the SAR system is shown in Table 3.2.

Qty	Part Name	Length/Size
1	V-Slot® NEMA 23 Linear Actuator Bundle (Lead Screw)	250 mm
1	V-Slot® NEMA 23 Linear Actuator Bundle (Lead Screw)	1000 mm
6	V-Slot® 20x20 Linear Rail	250 mm
2	Xtension Wire Set - 4 Conductor	3 ft
1	Motor Mount Plate – NEMA 23 Stepper Motor	N/A
4	Xtension Limit Switch Kit	N/A
4	Xtension Wire Set - 2 Conductor	3 ft
8	Inside Hidden Corner Bracket	N/A
2	L Bracket	Single
24	Double Tee Nut?	N/A
1	Rubber Feet Set (4 Pack)	N/A
1	Aluminum Spacers (10 Pack)	20 mm
1	Aluminum Spacers (10 Pack)	40 mm

Table 3.2: OpenBuilds hardware list.

3.3 Homing System Settings

The limit switches are connected to the TinyG board, each with its configuration setting dependent on their placement. Serving as homing switches, the settings of the two limit switches for a linear actuator determines the direction the actuator's corresponding motor will rotate. For the horizontal actuator, the minimum and maximum switch mode settings of the X-axis must be configured to set the motor's direction of rotation. This also applies to the vertical actuator and the switch mode settings for the Y-axis. There are two switch mode settings used for this system: one for disabling the switch and one for activating the switch during homing. The values for these settings are 0 and 1 respectively. The configuration of the switch mode determines the linear direction of travel for a gantry cart corresponding to the dimension. The SAR system is designed for a gantry cart to move towards the switch in maximum switch mode. The settings used for the limit switch configuration are shown in Table 3.3.

Setting	Description	Values
\$ST	Switch Type	0
\$XSN	X Minimum Switch Mode	0,1
\$XSX	X Maximum Switch Mode	0,1
\$YSN	Y Minimum Switch Mode	0,1
\$YSX	Y Maximum Switch Mode	0,1
\$ZSX	Z Minimum Switch Mode	0
\$ZSN	Z Maximum Switch Mode	0
\$ASN	A Minimum Switch Mode	0
\$ASX	A Maximum Switch Mode	0

Table 3.3: Limit switch configuration.

A series of configuration settings are used to determine the distance and speed at which a gantry cart travels along its aperture during a homing sequence. The travel maximum value in the X-axis is set to match the distance of the horizontal aperture. Due to the size of the gantry cart and the mounted limit switches on each end, this value cannot exceed 0.9 meters. The homing search velocity determines the speed of a gantry cart while in motion. The homing search velocity in the X-axis essentially determines the speed the radar module travels while creating the synthetic aperture. This velocity is set to a low value to allow for precise positioning and alignment during data collection, leading to increased resolution of the target scene. The settings of each homing configuration setting are shown in Table 3.4.

Setting	Description	Value
\$XTN	Travel Minimum (X-axis)	0
\$XTM	Travel Maximum (X-axis)	123 ¹
\$YTN	Travel Minimum (Y-axis)	0
\$YTM	Travel Maximum (Y-axis)	0.203
\$XSV	Homing Search Velocity (X-axis)	150
\$YSV	Homing Search Velocity (Y-axis)	15
\$XLV	Homing Latch Velocity (X-axis)	10
\$YLV	Homing Latch Velocity (Y-axis)	10
\$XLB	Homing Latch Backoff (X-axis)	1
\$YLB	Homing Latch Backoff (Y-axis)	0
\$XZB	Homing Zero Backoff (X-axis)	1
\$YZB	Homing Zero Backoff (Y-axis)	3

Table 3.4: Homing configuration settings.

G-code commands are used to communicate with the TinyG board to control the motors. G-code consists of a series of commands containing instructions that tell the TinyG board which motor to trigger and the operation to perform. A list of each

Gcode command used in the system design is shown in Table 3.5. The G28.2 G-code is designed to home the motion to physical home switches. G28.2 locates the home switch of an axis and then establishes machine zero (the absolute coordinate system zero) for that axis, offset from the switch location [21]. The axis must be specified to determine which motor should be triggered for this command. Inputting G28.2 X0 essentially triggers the motion of the radar module across the horizontal actuator up to a predefined distance. G28.2 Y0 triggers the motion sequence of the radar module to travel up the vertical actuator, essentially repositioning the radar module for a new vertical data slice. G30 allows for the radar module to reset its position in both dimensions with the default position being 0,0 in each axis.

Gcode	Parameter	Command	Description
G28.2	X0	X-Axis Homing Sequence	Homes to X-axis
G28.2	Y0	Y-Axis Homing Sequence	Homes to Y-axis
G30		Go to G30.1 Position	Returns to reference position
G30.1		Set Position for G30	Set reference position

Table 3.5: Gcode commands.

3.4 Automotive Radar Module

This section describes the Texas Instruments AWR2243BOOST BoosterPack™ Evaluation Module (EVM). The TI AWR2243EVM is an automotive radar module that operates in the band of 76 to 81 GHz with 5 GHz available in bandwidth, which is ideal for short-range applications. The AWR2243EVM, designed for mmWave sensing, is provided as a convenient plug-in module and evaluation board. This

¹This value is the upper limit the radar module can travel across the horizontal actuator.

board features built-in emulation for programming and debugging, as well as on-board buttons and LEDs for the integration of a straightforward user interface. Moreover, the device is offered as an inclusive platform solution with TI providing many resources for the use of operations. This provides reference hardware design, software drivers, sample configurations, an API guide, and user documentation for ease of use. Images of the AWR2243EVM and its components labeled are shown in Figure 3.9 and Figure 3.10.

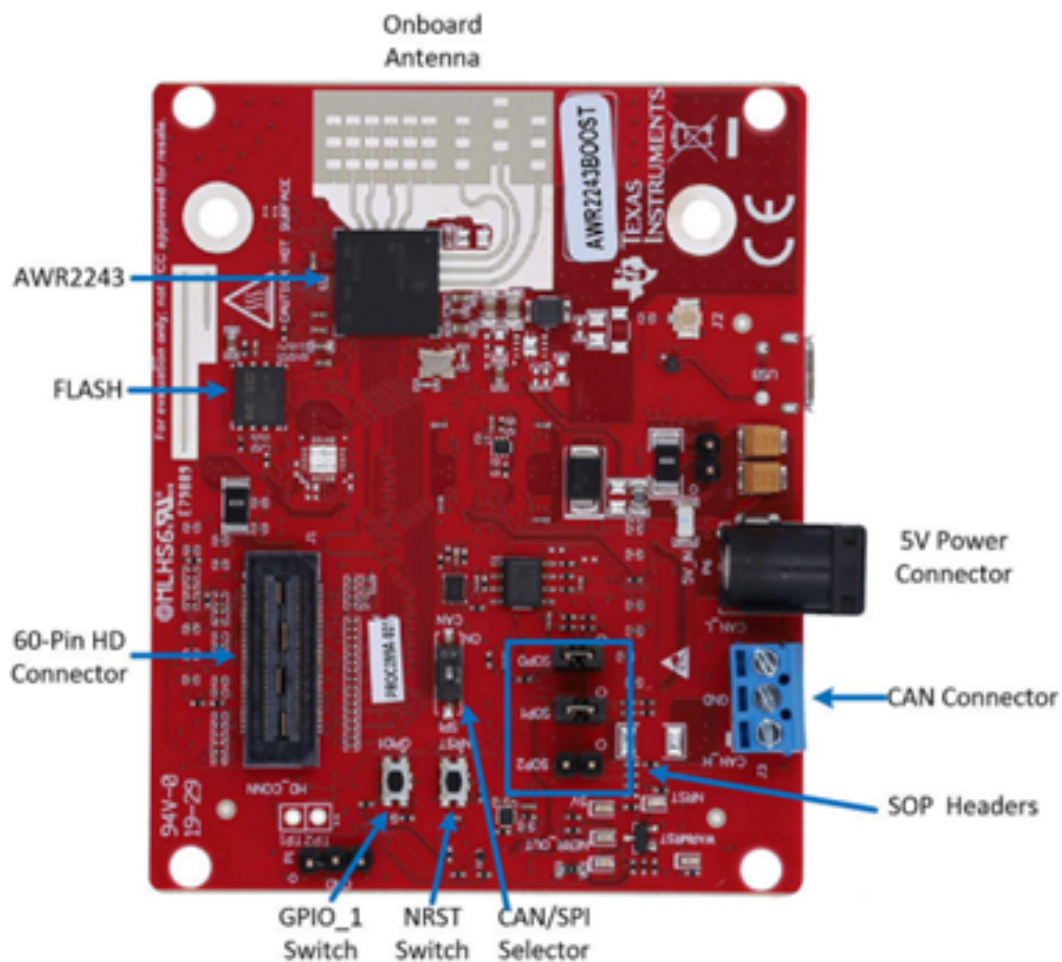


Figure 3.9: Front side of the AWR2243EVM board [4].

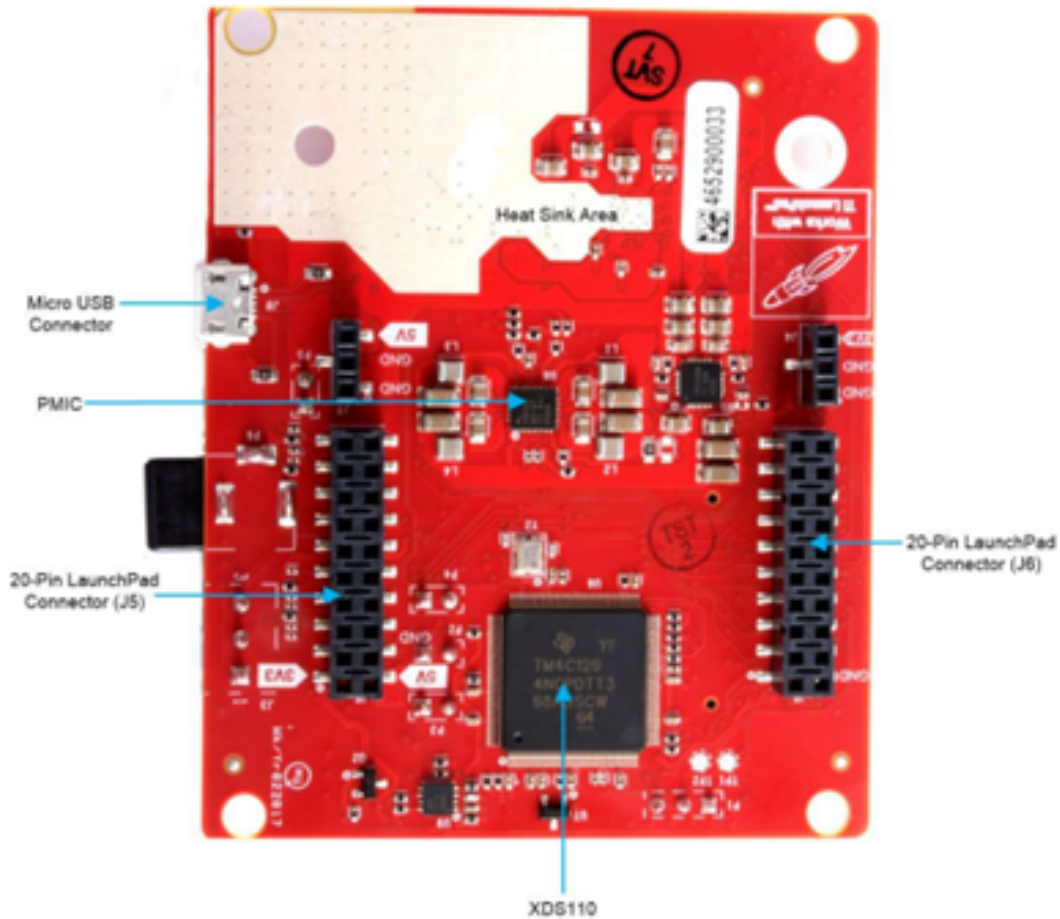


Figure 3.10: Back side of the AWR2243EVM board [4].

The board is powered with a five-volt DC adapter and controlled via a micro-USB connector that allows it to interface with a PC. This radar features an FMCW transceiver which is implemented in the form of a self-contained, single-chip solution that simplifies the implementation of the sensors in the frequency band. This transceiver is well-suited for automotive applications requiring low power consumption, self-monitoring capabilities, and ultra-accurate radar systems in the automotive industry. A block diagram of the FMCW transceiver is shown in Figure

3.11. Additionally, the AWR2243EVM includes an onboard etched PCB antenna which consists of three transmitter channels and four receiver channels, each with built-in PLL and ADC converters. This design allows for the estimation of both azimuth and elevation angles, which enables object detection in a 3-D plane.

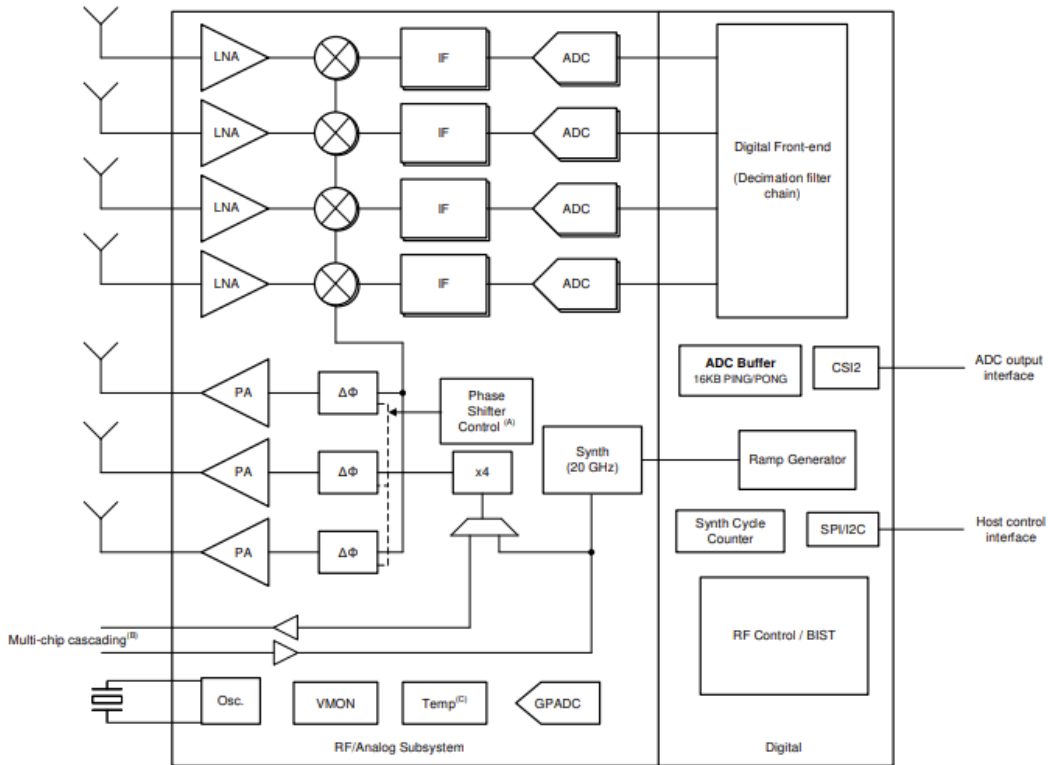


Figure 3.11: Simplified representation of AWR2243EVM device [5].

3.5 Data Capture Adapter

The TI DCA1000EVM serves as a real-time data capture adapter designed to interface with the AWR2243EVM. This device features two modes of data transfer: raw mode, designed for transferring raw low-voltage differential signaling (LVDS)

data, and data separated mode, enabling users to append specific headers to various data types. The raw mode is used for the entire SAR system in this thesis.

This evaluation module is powered by a 5-V supply voltage, an input voltage identical to the AWR2243EVM. The DCA1000EVM has two standard micro-USB connectors that connect to the host computer. Only the RADAR_FTDI connector is used to enable access to the AWR2243EVM via UART utilizing the FTDI chip, enabling communication with the TI RadarStudio application. This micro-USB cable linking the DCA1000EVM and host computer together permits for programming the FPGA which allows the AWR2243EVM to be configured and controlled. The signal interface between the capture card and the AWR2243EVM uses a 60-pin Samtec high-density connector. This connector provides the high-speed LVDS data, the UART and RESET control signals, and allows access to the AWR2243EVM through the FTDI chip. Images of the DCA1000EVM and its components labeled are shown in Figure 3.12 and Figure 3.13.

The DCA1000EVM is based on LATTICE's High-Performance FPGA LFE5UM-85F-8BG381I. In raw mode, this design utilizes DDR3L memory to stream ADC data over an Ethernet interface. The DCA1000EVM also facilitates a Gigabit Ethernet port for establishing a network connection to the host computer. This enables the transfer of captured data directly to an external application. Therefore, raw ADC data can be captured and immediately passed to a host computer for data processing and algorithm development. Operations of the DCA1000EVM are generally handled in mmWave Studio. Although there are physical switch configurations on the board, the default operations are used in the final system design. A block diagram of the DCA1000EVM is shown in Figure 3.14.

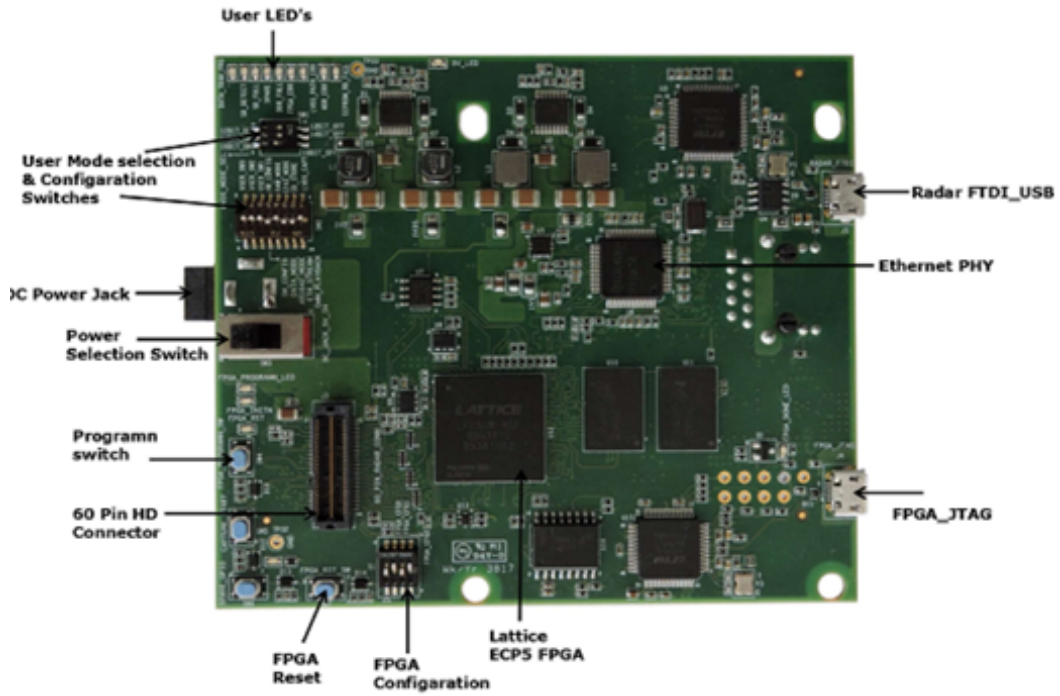


Figure 3.12: Front side of the DCA1000EVM [6].

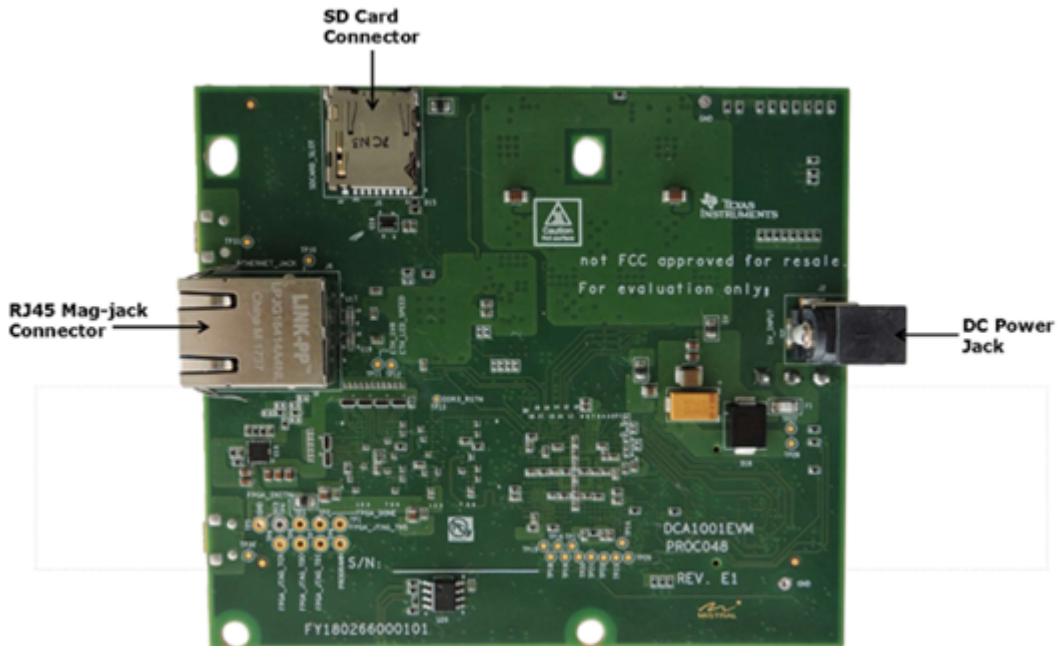


Figure 3.13: Back side of the DCA1000EVM [6].

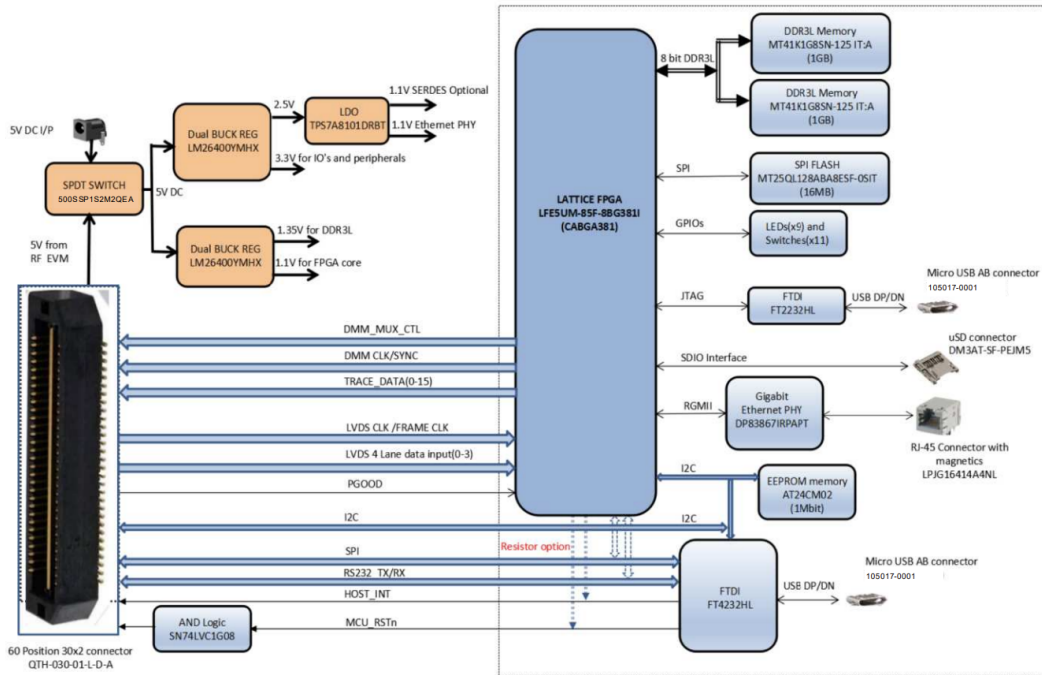


Figure 3.14: DCA1000EVM functional block diagram [6].

3.6 SAR System Setup

Together, the AWR2243EVM and DCA100EVM form a single radar sensing module commonly used in automotive applications. The AWR2243EVM provides the raw radar data by emitting and receiving electromagnetic waves toward target scenes, while the DCA1000EVM chip handles the data capture, digitization, and initial processing of the raw radar data. This integrated device enables real-time detection and analysis of objects, serving as an ideal radar module for designing a synthetic aperture radar system. The assembled radar module is showcased in Figure 3.15.



Figure 3.15: The AWR2243EVM and DCA1000EVM connected as a single radar module device [6].

From TI, the board kits provide L-brackets for vertically mounting the EVM boards. Two of these L-brackets have their long ends screwed onto the gantry cart of the vertical actuator while two smaller L-brackets purchased from OpenBuilds are connected to the back end of the DCA1000EVM. By securely locking these L-brackets together, a firm connection is established between the radar module and the gantry cart as shown in Figure 3.16. This ensures the radar module remains in a fixed and stable position while in motion.

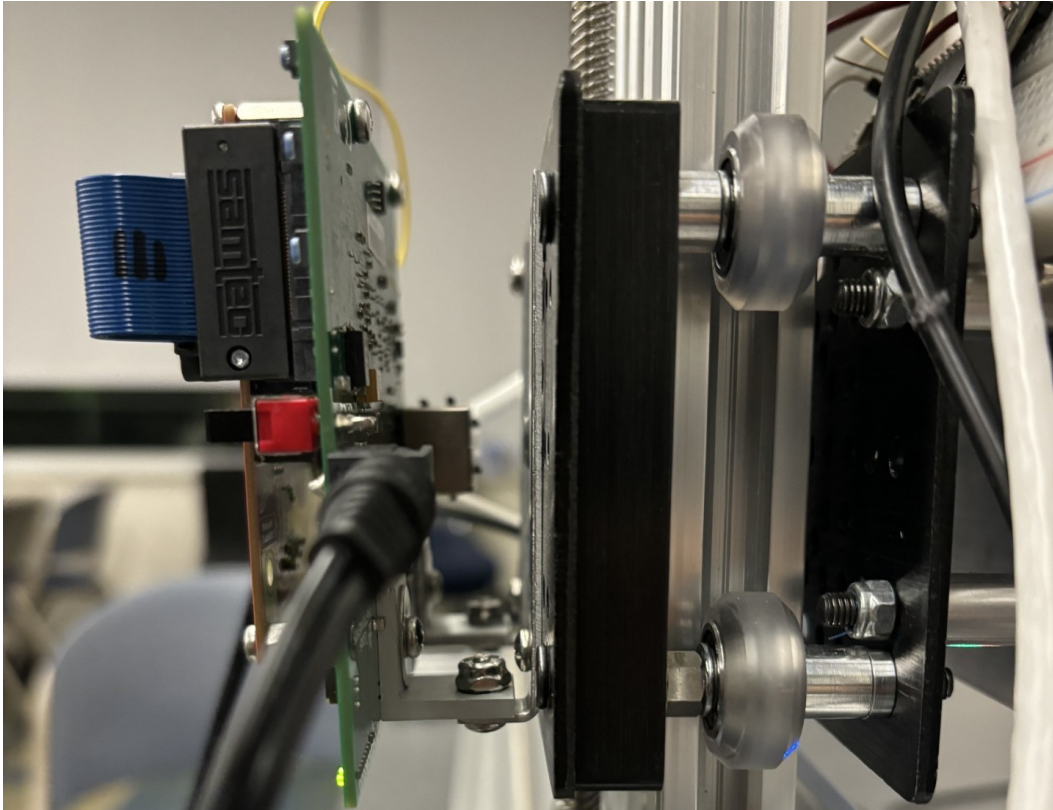


Figure 3.16: The radar module mounted onto the SAR track system.

The SAR system positions the AWR2243EVM in front of the DCA1000EVM, with the back end of DCA1000EVM mounted onto the gantry cart of the vertical actuator. The two boards are connected via a ribbon cable to establish a connection between them. The radar modules require connecting two USB cables and an Ethernet cable into the host computer and two 5-V DC power jacks into a power supply. The radar module attached to the SAR system is shown in Figure 3.17. Additionally, a block diagram of the hardware is shown in Figure 3.18. Chapter 4 discusses the connection and configuration process for the radar module, along with how to interface with it to perform a dataset capture.

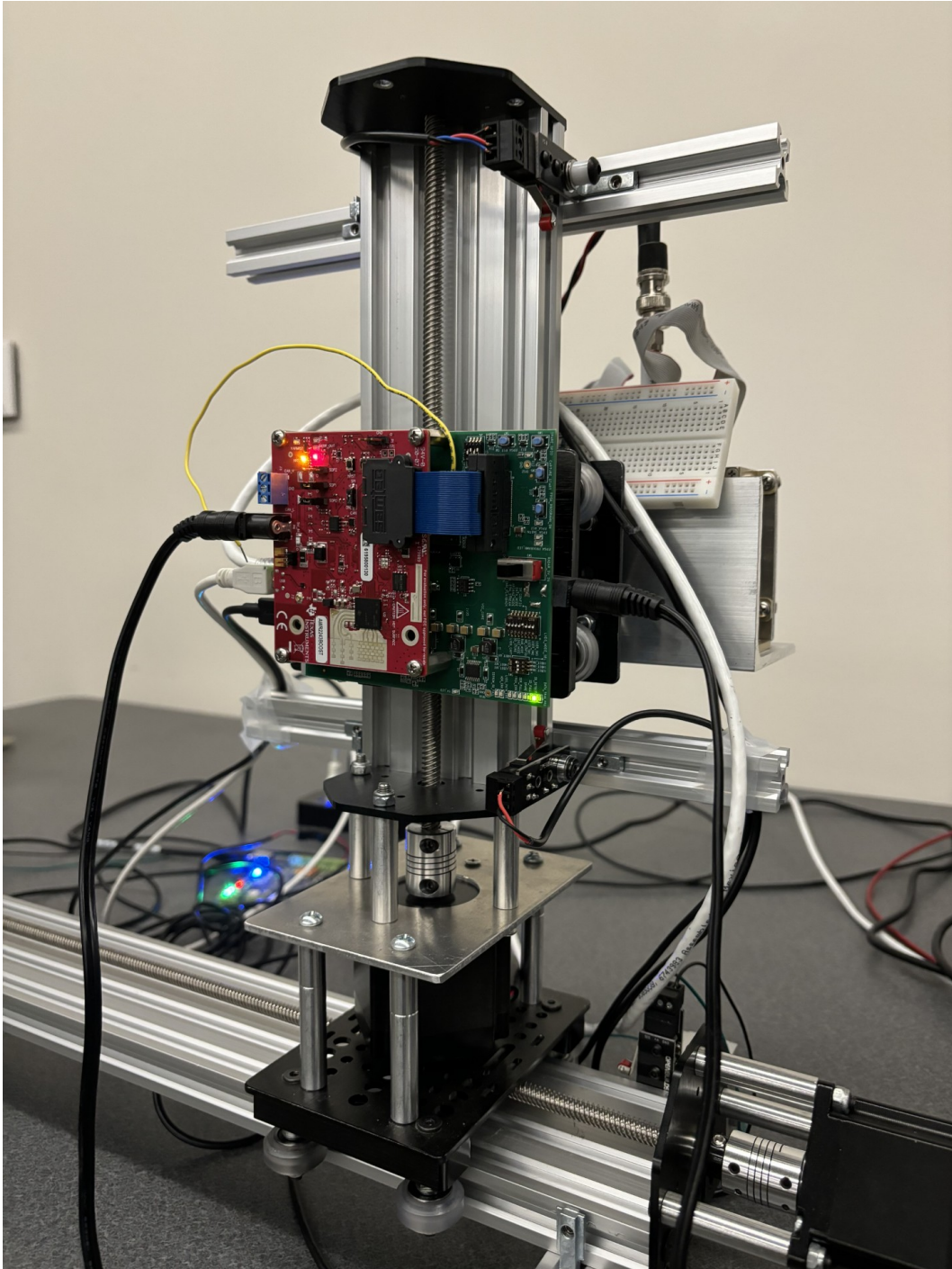


Figure 3.17: The complete SAR system setup with the radar module attached.

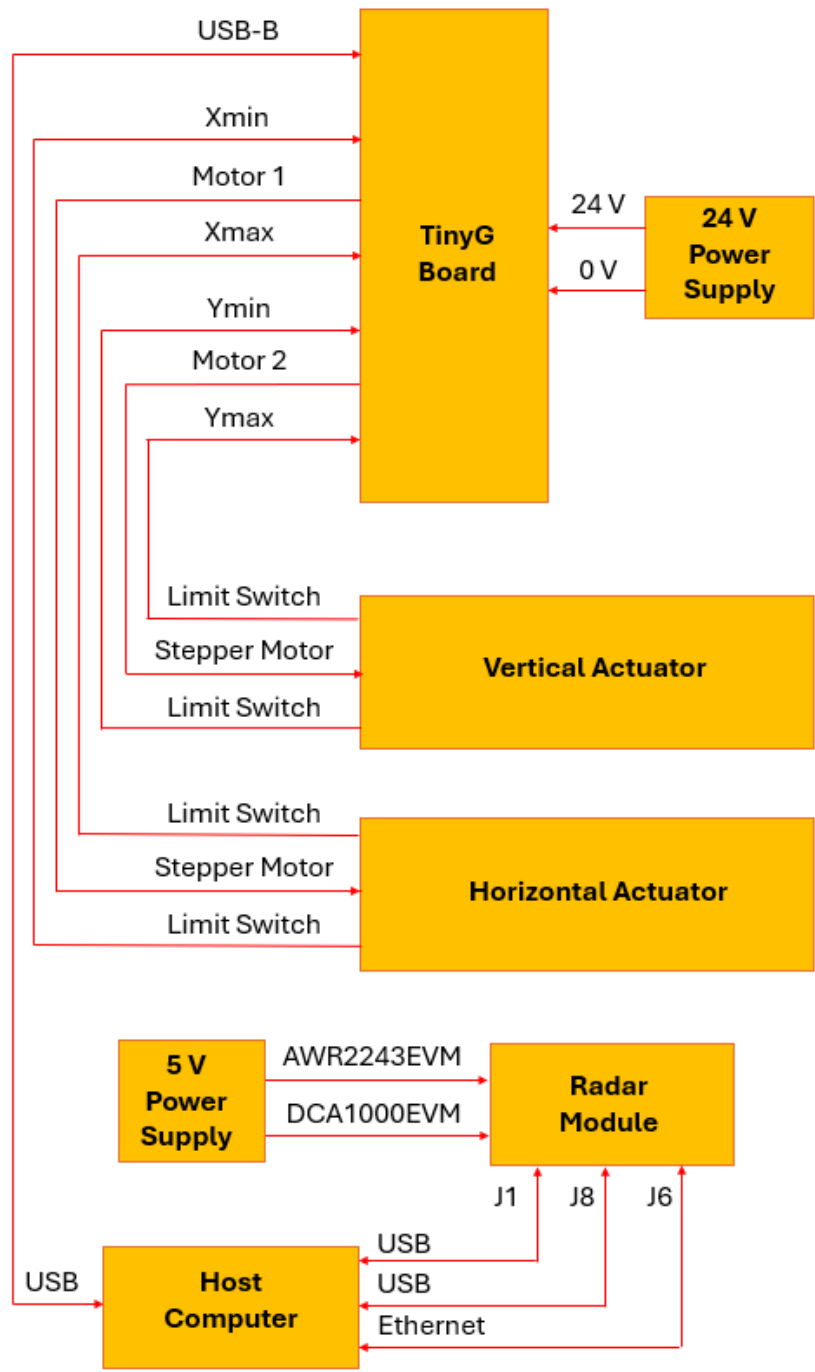


Figure 3.18: Block diagram of the SAR system hardware setup.

Chapter 4

Radar Software Configuration

In this chapter, the processes of interfacing with the AWR2243EVM and the DCA1000EVM as a single radar module using mmWave Studio are showcased. This starts with covering how the radar module is installed and connected to the program upon startup. Following this is the configuration sequence for the radar module using mmWave Studio. Furthermore, the steps for performing a data capture are revealed. This chapter also covers how to use mmWave Studio as a post-processing tool.

4.1 Radar Connection and Installation

Once the AWR2243EVM is connected to the DCA1000EVM, both are powered on by their power supply and connected to the host computer, each using a USB cable. When done for the first time, a series of FTDI USB drivers is installed on the host computer. After completion of the driver installation, the COM ports become available in the Windows Device Manager as shown in Figure 4.1. COM ports 7-8 correspond to the DCA1000EVM, COM ports 9-12 correspond to the AWR2243EVM, and COM13 corresponds to the TinyG board.

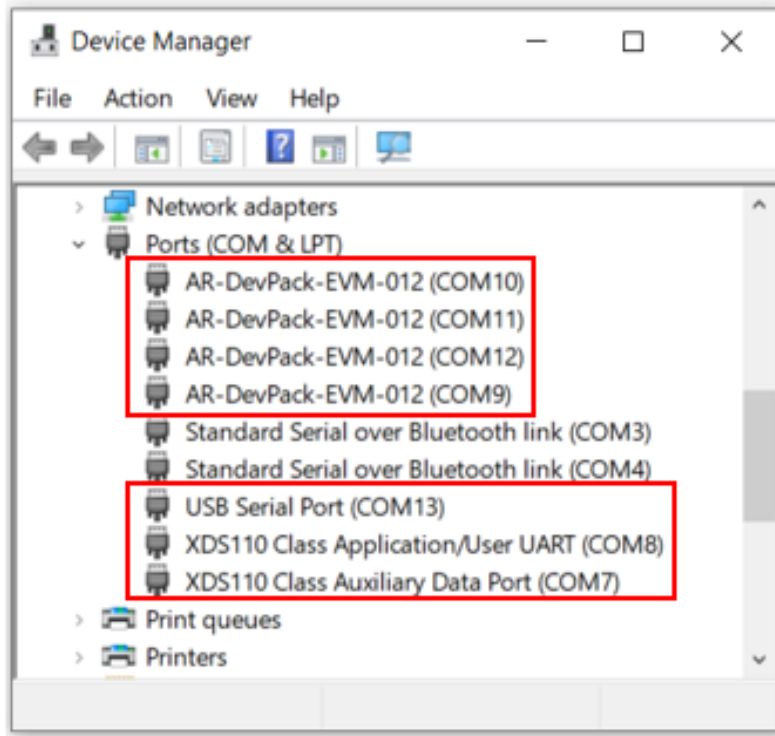


Figure 4.1: COM Ports for the SAR system.

The DCA1000EVM receives raw ADC data from the AWR2243EVM and transfers it to the host computer via the Ethernet interface. The default destination Ethernet port address configured in the DCA1000EVM is 192.168.33.30. The user must ensure that the host computer is set with this IP address to receive the raw ADC data from the DCA1000EVM. The configuration for setting this IP address is shown in Figure 4.2. Finally, the firewall for the host computer must be disabled to perform any operations with mmWave Studio.

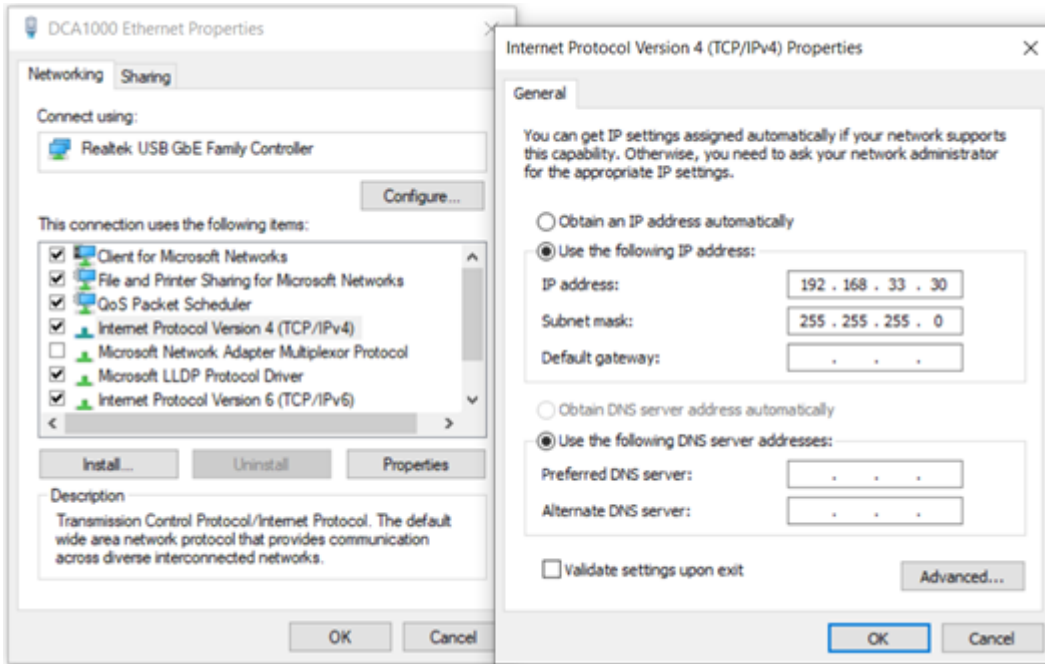


Figure 4.2: IP address for Ethernet connection to DCA1000EVM.

4.2 mmWave Studio

mmWave Studio is a Windows graphical user interface (GUI) designed for configuring and managing TI mmWave sensor modules. This program provides the ability to facilitate the collection of analog-to-digital (ADC) data for later offline analysis. This feature allows for evaluating and characterizing radio-frequency (RF) performance and developing radar signal processing algorithms. mmWave Studio offers additional functionalities such as basic post-processing and visualization of ADC data, along with post-processing examples using MATLAB. This GUI allows for board control, including SOP change and reset control, and establishing an RS-232 connection and firmware download. Moreover, mmWave Studio configures mmWave sensors using radar API commands and interfaces with

the DCA1000EVM for raw ADC data capture. The mmWave radar device communicates with the external host processor using the SPI interface. The mmWave device is configured and controlled from the external host processor by sending commands to the mmWave device over SPI. Finally, mmWave Studio allows for API commands to be executed using Lua scripts which can be used to test predefined sequences and automate functionality on the device.

4.2.1 Startup and Connection

The initial connection and startup of mmWave Studio is divided into six steps. First, the user should confirm that the SOP is set to mode 2 on the AWR2243 board and the USB connections to the radar module are detected. 77 GHz should be selected as the operating frequency and xWR22xx as the device variant. Next, the RS-232 operations controlling the radar module are established. This involves selecting the UART COM port and enabling the board to receive commands from mmWave Studio. Once the connection is established, both the BSS and MSS firmware files are downloaded to the radar module. After loading both firmware files, the SPI connection is established, and the RF system's powered-up sequence is initialized. The user is now able to send commands to the device over an SPI communication interface. The completed startup sequence in the GUI is shown in Figure 4.3. Additionally, the user needs to set up the DCA1000EVM configuration from the Connection tab as shown in Figure 4.4. As expected, the system IP address matches the IP address previously set for the DCA1000EVM Ethernet connection.

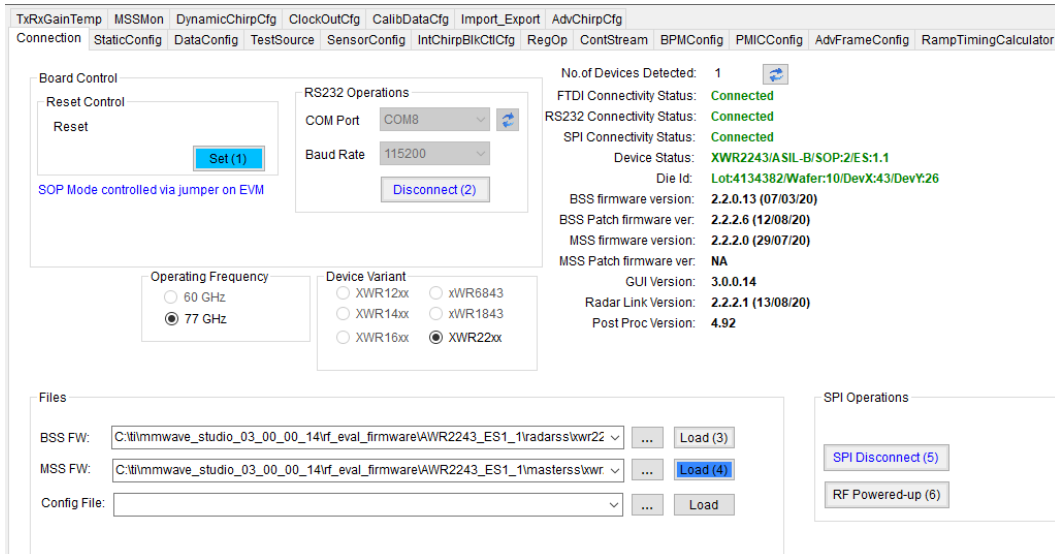


Figure 4.3: Connection window in mmWave Studio upon startup completion.

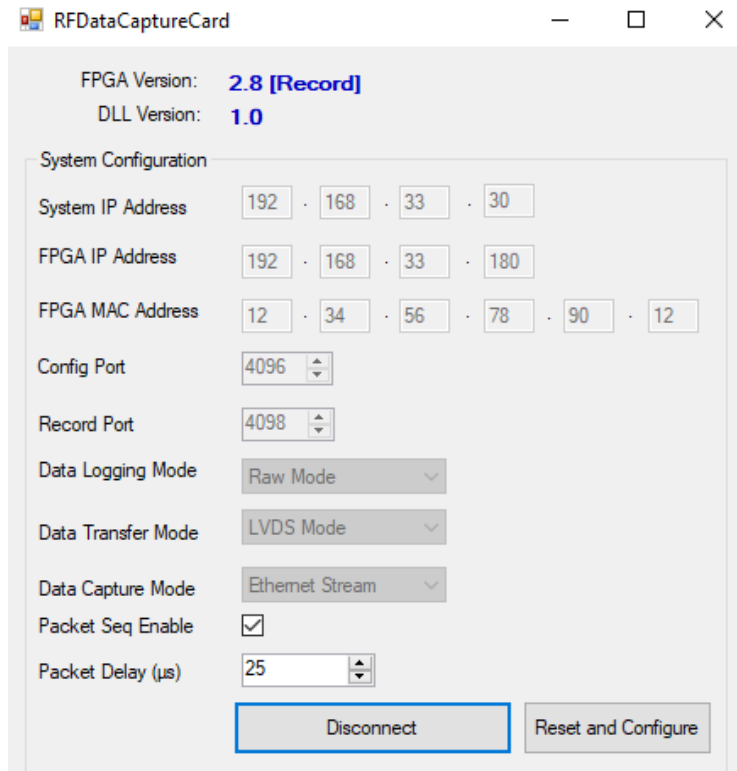


Figure 4.4: DCA1000EVM configuration window.

4.2.2 Static Configuration

The “Static Configuration” tab allows for the user to set static configurations for the device. This includes selecting the RX and TX channel(s) necessary for device functionality, configuring the data format of the ADC output, setting the RF LDO bypass, enabling frequency operation limits, and setting trigger basic calibrations and RF initializations. The default configurations were kept for this thesis. Following a successful SPI connection and RF power-up, the user can click the Set buttons in the specified order outlined in Figure 4.5.

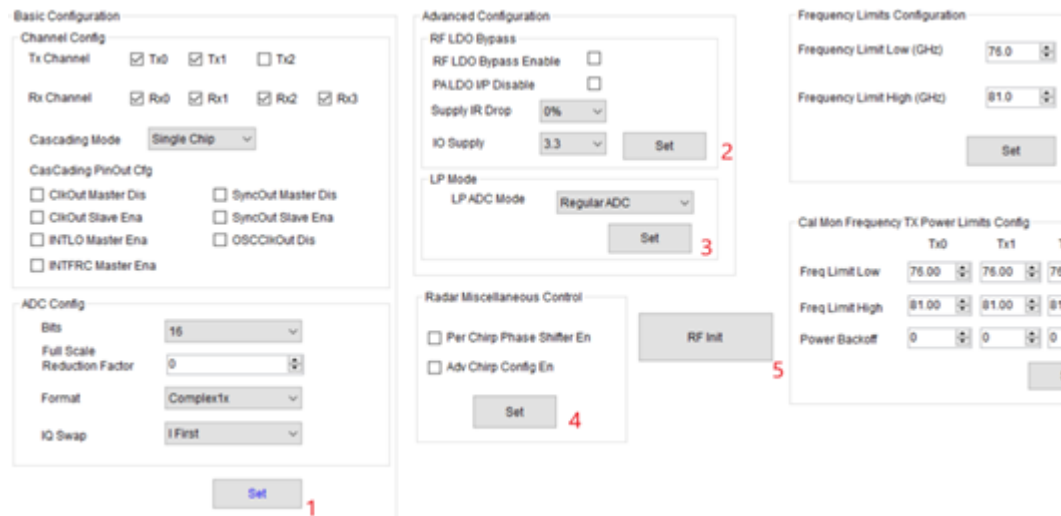


Figure 4.5: Static Configuration window.

4.2.3 Data Configuration

The “Data Configuration” tab allows the user to configure the Data Path. Users can set the data path format to transfer the captured ADC samples received over the receive chain out to an external host. Additionally, users can adjust the Lane

Configuration of the LVDS path for the transferring of radar data to an external host and configure the Clock configurations of the LVDS lanes. The default configurations were kept for this thesis. After completing the Data Configuration, the user can configure the Data Path by clicking the Set buttons as outlined in 4.6.

The screenshot shows the 'Data Configuration' window with the following settings:

- Data Path Configuration:**
 - Data Path: LVDS
 - Virtual Channel No: 0
 - CQ Cfg: 16 Bit
 - Packet 0: ADC_ONLY, CQ0TransSize (16bit): 132
 - Packet 1: Suppress Pacl, CQ1TransSize (16bit): 132
 - CQ2TransSize (16bit): 72
 - Set button (1)
- Clock Configuration:**
 - Lane Clock: DDR Clock
 - Data Rate: 600 Mbps
 - Set button (2)
- LVDS Lane Configuration:**
 - Lane Format: Format 0
 - Lane Config: Lane1, Lane2, Lane3, Lane4 (all checked)
 - MSB First: checked, CRC: unchecked
 - Packet End Pulse: unchecked
 - Discontinuous Clock Mode: unchecked
- CSI2 Lane Configuration:**
 - Lane0 Position: 1, Lane0 Polarity: +/- Pin Order (unchecked)
 - Lane1 Position: 2, Lane1 Polarity: +/- Pin Order (unchecked)
 - Lane2 Position: 4, Lane2 Polarity: +/- Pin Order (unchecked)
 - Lane3 Position: 5, Lane3 Polarity: +/- Pin Order (unchecked)
 - Clock Position: 3, Clock Polarity: +/- Pin Order (unchecked)
 - Disable Line Start/End: unchecked
 - Set button (3)

Figure 4.6: Data Configuration window.

4.2.4 Sensor Configuration

The “Sensor Configuration” tab is where the user can define the final set of parameters for the radar module and execute the commands for performing a data

capture. This tab allows the user to configure various aspects such as the chirp, profile parameters, associating defined profiles with chirp indices, frame configuration, and other RF parameters. The “Profile” section is dedicated to setting FMCW radar chirp profiles or properties like FMCW slope, chirp duration, and TX power. Multiple profiles are supported, each defined within this section. Internal RF and analog calibrations can be triggered upon receiving this sub-block, with an async event response sent upon completion. For this thesis, the frequency slope is set to 49.97 MHz/ μ s, the idle time is set to 1920 μ s, and the ramp end time is set to 80 μ s. This results in a bandwidth of 3997.6 MHz, the maximum possible value mmWave Studio can compute. The bandwidth is calculated using the following equation:

$$\textit{Bandwidth} = \textit{Frequency Slope} * \textit{Ramp End Time} \quad (4.1)$$

In the “Chirp” section, the user can define variations between chirps based on the profiles set in the Profile API. This includes determining which profile is used for each chirp within a frame, along with minor adjustments such as small dithers in FMCW start frequency and idle time for each chirp. The default values are kept for this section. Figure 4.7 offers a visual for how these parameters create the FMCW radar signal transmission.

Finally, the “Frame” section allows the user to define a frame, comprising a sequence of chirps to be transmitted sequentially. Parameters such as the number of frames, frame periodicity, and triggering methods are set within this section. A single frame consists of the defined number of pulses or chirp loops, usually 128 pulses. During a SAR data capture, the radar module must continuously trigger frames while in motion across the horizontal actuator. The user should keep the

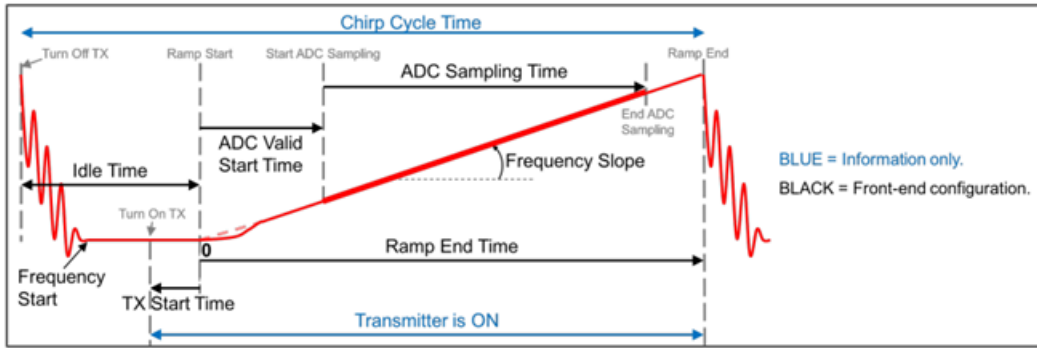


Figure 4.7: Visualization of FMCW radar signal transmission [7].

duty cycle close to 100% by adjusting the periodicity. A higher duty cycle means that the radar is transmitting pulses for a longer duration relative to the total cycle time [3]. This results in higher average power, which is crucial for improving the radar’s ability to detect distant or small objects. The duty cycle is calculated using the following equation:

$$Duty\ Cycle = \frac{Idle\ Time + Ramp\ End\ Time}{Periodicity} * No.\ of\ Chirp\ Loops \quad (4.2)$$

mmWave Studio only allows for a maximum duty cycle value of 99.7%. Therefore, the periodicity is set to 256.7 ms to ensure the duty cycle is 99.7% based on the previously defined parameters. The user can set the configurations by clicking the Set buttons as outlined in Figure 4.8.

Once every configuration has been set in mmWave Studio, the radar module is ready to perform data captures. The “Capture and Post Processing” section offers the tools necessary to interface with the radar module to perform a data capture and, upon completion, follow it up with post-processing. The text input field allows the user to choose a filename under which captured ADC data is to be stored. The default data filename is adc_data.bin. After clicking “DCA1000 ARM” and

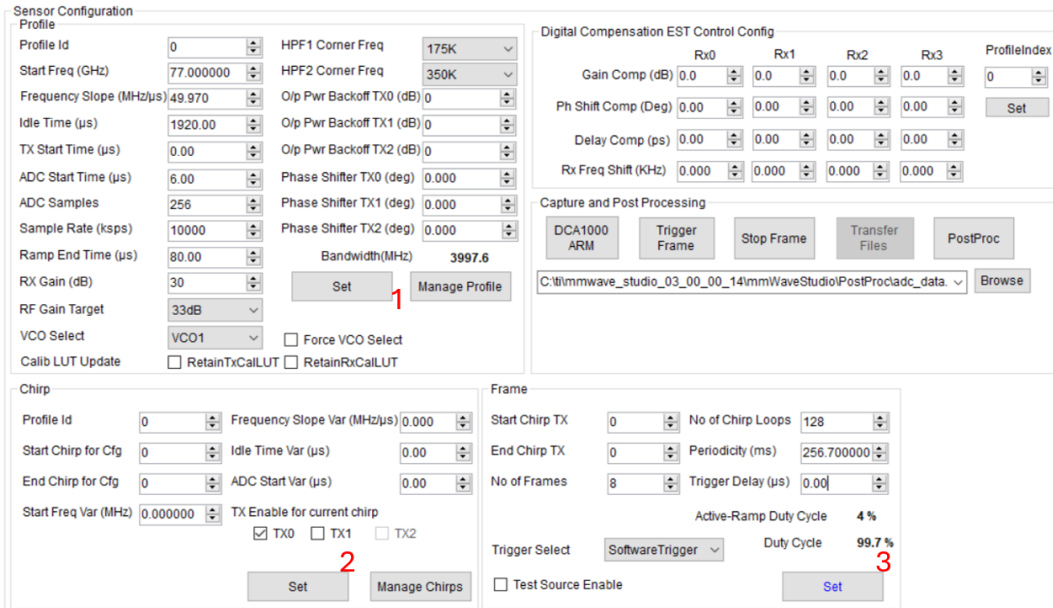


Figure 4.8: Sensor Configuration window with user-defined parameters.

then “Trigger Frame,” a raw ADC data file named `adc_data_RAW_0.bin` is initially saved. A successful data capture is indicated in the mmWave Studio output terminal. The data file is stored in the PostProc folder within the installation path. Data captured using mmWave Studio can be processed with the GUI, but for custom post-processing, users must create a custom script.

4.2.5 PostProc Tool

mmWave Studio also includes a MATLAB post-processing tool named PostProc that performs radar post-processing on raw ADC data followed by radar imaging which provides the user visualization of the processed data. This tool operates with two inputs: the initial raw LVDS capture (saved as a `.bin` file) and the sequence of APIs used to program the radar device (saved as a `.log` file). Due to the Ethernet

protocol, the DCA1000EVM-received file may not have UDP packets in the correct order and could miss data. Running the packet reorder and zero-fill utility scans and reorders packets, filling gaps with zeros. This process removes metadata, leaving raw ADC data as is from the AWR2243EVM. Clicking PostProc initiates packet reorder and zero-fill on the data, saving the file as `adc_data.bin`. By utilizing these two data sources, PostProc interprets the radar device's collected data and generates insightful plots. The mmWave Studio user guide contains details of these plots and applications [7].

Using the PostProc tool opens a mmWave Studio radar post-processing application titled RadarStudio PostProcessing. This program reads the raw data file, processes it, and finally displays a new GUI window that generates a series of useful plots for radar signal analysis. Some of these plots include FFT amplitude position plots, range-angle plots, angle estimation results, and time domain plots. The user can choose to have four plots shown that can be independently selected. Each plot can be customized to display different channels and types of data representations. Certain plots are applicable to all channels, serving as a universal measurement, while others are specific to individual channels. Additionally, some plots represent data on a per-chirp basis, while others display data per frame. Figure 4.9 showcases the GUI with some plots corresponding to a raw data file.

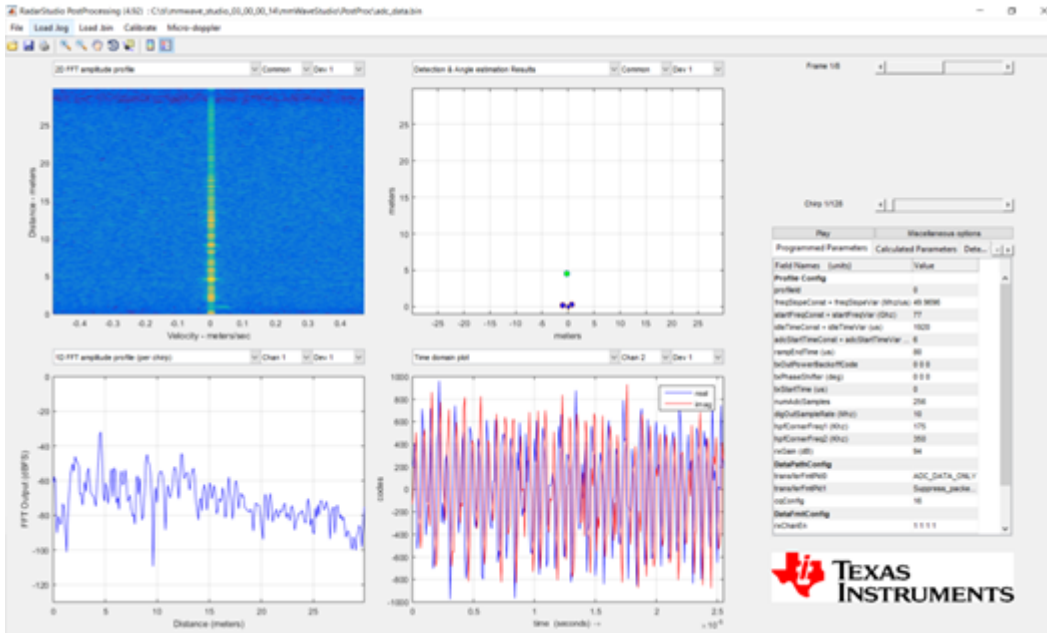


Figure 4.9: Post-processing GUI with four characterization plots.

This tool helps check the calculated parameters of captured data, analyzing the results of detected targets, and demonstrating radar functionality. The program can also open pre-existing saved .bin files for viewing previously captured data. In this thesis, only the program’s function to generate the raw data file is used in separate post-processing applications designed for the SAR Demonstrator.

Having described the radar software configuration, a control system for performing SAR data captures can be designed. To avoid manually inputting the commands for performing each individual SAR data capture, an autonomous system design was desired. The conceptualized and realized control system design is described in Chapter 5.

Chapter 5

Control System Design

In this chapter, the LabVIEW program designed for running the SAR Demonstrator is described in detail. The software for how the control system operates is showcased and discussed. This includes the integration and configuration of both the TinyG board and mmWave Studio and how the two systems are synchronized. The GUI for the user operating the system is showcased with each section explained individually. Finally, the complete sequence of the radar module capturing SAR datasets is described case-by-case from start to finish.

5.1 System Overview

Before running the LabVIEW program, the power supply for the TinyG board should be turned on and both EVMs for the radar module should already be plugged into their power supply. Every serial port should also be connected to the host computer. The system should already be positioned with the radar module facing the target scene with every target visible and within the range of the radar. Once the physical setup of the hardware and the target scene is ready, the user can execute the LabVIEW program to allow the system to perform an SAR data capture.

The LabVIEW program is designed to operate as the master control system over the SAR Demonstrator. A GUI was designed to grant the user full control over the system. The program executes an algorithm designed to perform a complete series of SAR data captures. The program is designed to interface with multiple modules to operate as a single functioning system. A sequence of one-way integrations with other programming languages allows for commands in mmWave Studio to be executed from LabVIEW. The program also utilizes built-in VISA instruments to interface with the TinyG board. Synchronous communication with mmWave Studio and the TinyG board allows for an autonomous system design. A block diagram showcasing how LabVIEW interfaces with the SAR system is shown in Figure 5.1.

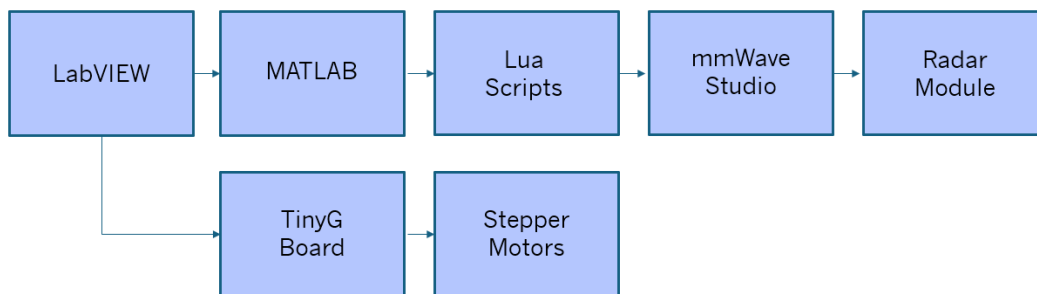


Figure 5.1: Block diagram of how LabVIEW interfaces with the SAR system.

LabVIEW serving as the software for the SAR Demonstrator allows for the system to run autonomously via communication with both the TinyG board and mmWave Studio simultaneously. Both the TinyG board and mmWave Studio are synchronized such that the radar module can perform data captures while in motion on the horizontal actuator. The autonomous design of the program ensures that the SAR Demonstrator can capture multiple datasets in a single SAR data capture session with LabVIEW as the control system.

The LabVIEW program is designed to communicate with mmWave Studio using a MATLAB function and Lua scripts. The program can launch mmWave Studio and programmatically connect the radar module to the software. Sensor configurations for the radar module can be tuned in LabVIEW to specify the desired RF parameters in mmWave Studio. Additionally, the length of the horizontal synthetic aperture, the total number of datasets to capture, and the filename for these datasets can be specified by the user. All these parameters can be defined in the GUI.

After an SAR data capture has finished, the program can also immediately call a MATLAB function designed to perform post-processing if desired. The MATLAB function applies the backprojection algorithm to the captured dataset and performs post-processing followed by imaging. This results in the creation of a 3-D SAR image displayed on the GUI when multiple datasets have been captured. The 3-D image reveals the reflectivity of the target and its exact location from the radar system in a volumetric space. Overall, the LabVIEW program is designed to function as a standalone system for the SAR Demonstrator. The GUI for the SAR Demonstrator is shown in Figure 5.2.

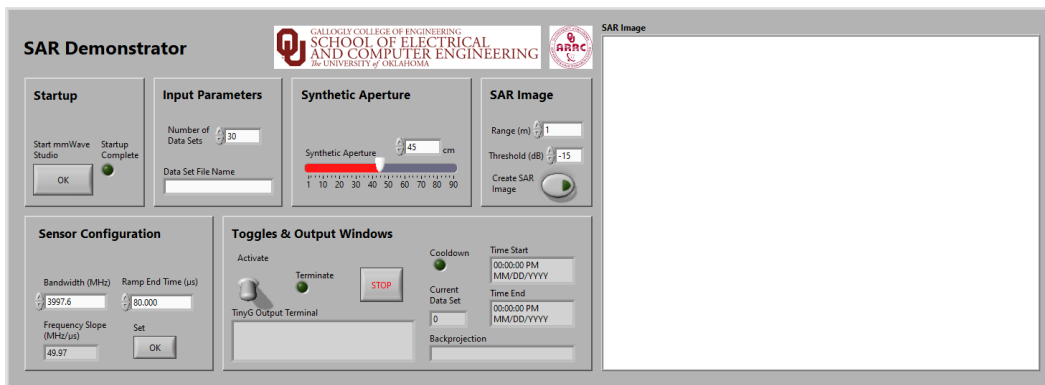


Figure 5.2: Default view of the LabVIEW GUI.

To connect LabVIEW to the TinyG board, a series of VISA functions are used to establish a connection for configuring settings and enabling commands to be written to the device. The VISA configure serial port instrument initializes the serial port corresponding to the TinyG board and configures it with the specified input settings. The VISA functions for transferring data from the program to the TinyG board and managing the device session include the write, read, clear, and close functions. The write function allows for commands to be written to the TinyG board and the read function allows for data to be read from the TinyG output terminal. The code architecture for the VISA instrumentation is shown in Figure 5.3.

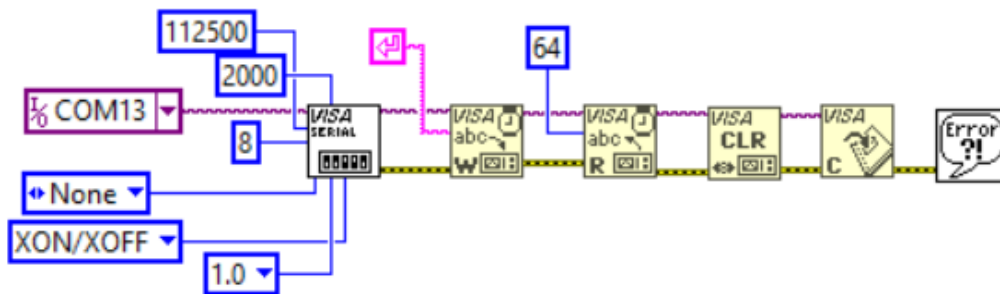


Figure 5.3: VISA instrumentation for interfacing with the TinyG board.

The LabVIEW program is designed to call and execute any MATLAB function from a MATLAB script node for performing miscellaneous operations. The most used operation is to control mmWave Studio from using LabVIEW. This is done by running a MATLAB function to execute a Lua script containing the commands for performing desired operations in mmWave Studio. The name of the Lua script is defined in the MATLAB script node which becomes the input parameter for the LabVIEW function that calls a MATLAB function. A MATLAB script for executing Lua scripts which has been written by TI is provided in the mmWave Studio

User Guide [7]. This establishes a connection with the mmWave Studio software. An example of this setup in LabVIEW being used for the startup of mmWave Studio is shown in Figure 5.4.

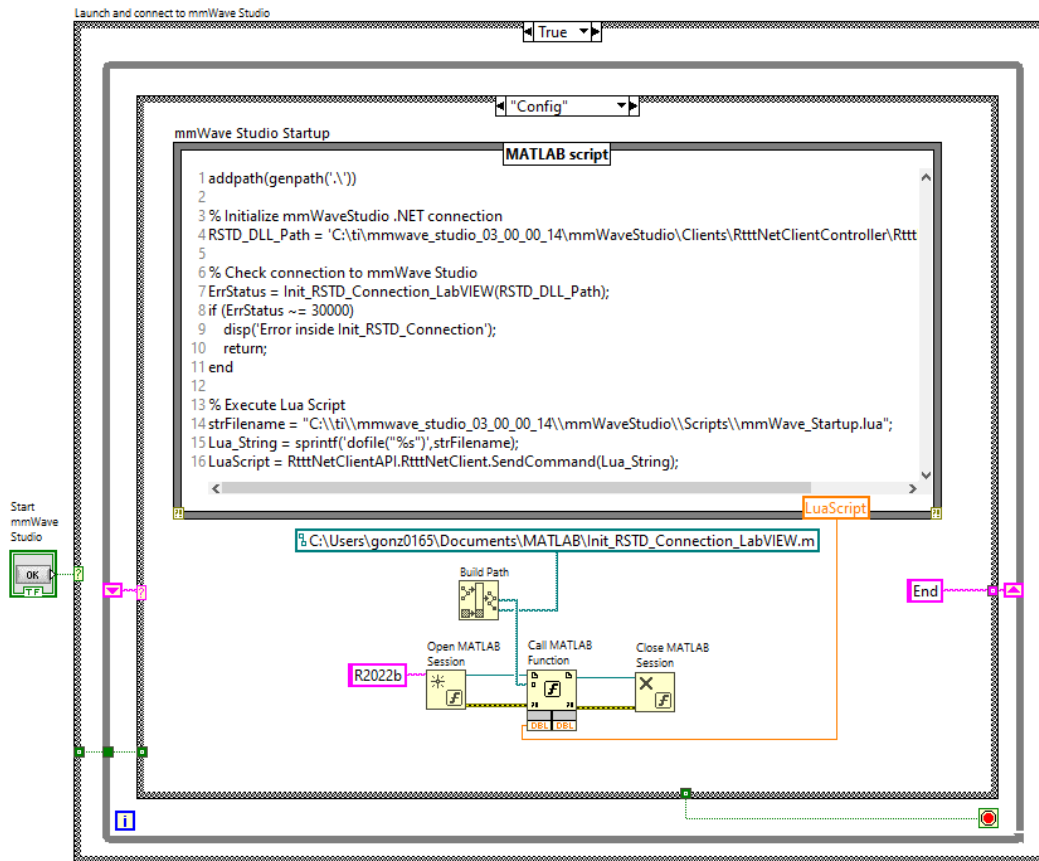
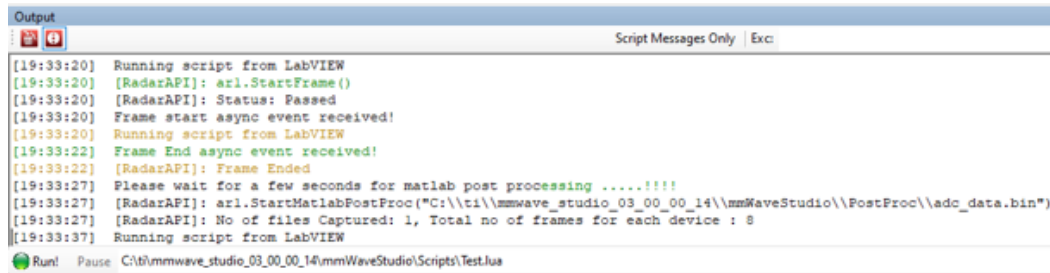


Figure 5.4: LabVIEW controlling mmWave Studio.

The same MATLAB script node design is used throughout the program with the LuaScript variable changing depending on the Lua script being used. Implementing this function allows for the system to autonomously execute the commands for capturing multiple datasets from mmWave Studio in a single execution of the LabVIEW program. The LabVIEW system controlling mmWave Studio to perform an

SAR data capture is shown in Figure 5.5.



```
Output
Script Messages Only | Exc
[19:33:20] Running script from LabVIEW
[19:33:20] [RadarAPI]: ar1.StartFrame()
[19:33:20] [RadarAPI]: Status: Passed
[19:33:20] Frame start async event received!
[19:33:20] Running script from LabVIEW
[19:33:22] Frame End async event received!
[19:33:22] [RadarAPI]: Frame Ended
[19:33:27] Please wait for a few seconds for matlab post processing .....!!!!
[19:33:27] [RadarAPI]: ar1.StartMatlabPostProc("C:\\ti\\mmwave_studio_03_00_00_14\\mmWaveStudio\\PostProc\\adc_data.bin")
[19:33:27] [RadarAPI]: No of files Captured: 1, Total no of frames for each device : 8
[19:33:37] Running script from LabVIEW
Run! Pause C:\ti\mmwave_studio_03_00_00_14\mmWaveStudio\Scripts\Test.lua
```

Figure 5.5: LabVIEW interfacing with mmWave Studio to perform a data capture.

5.2 LabVIEW GUI

The LabVIEW GUI for the SAR Demonstrator consists of six sections for setting up, configuring, and monitoring the radar system. These sections provide the user access to the system's controls, inputs, toggles, and outputs. If the user desires to only capture datasets, there is an optional setting for performing post-processing via backprojection and creating an SAR image. The order of operations the user should follow for using this GUI is from left to right and top to bottom. This section of the chapter also details the GUI in this exact order. A closeup of the GUI section for user operations is shown in Figure 5.6.

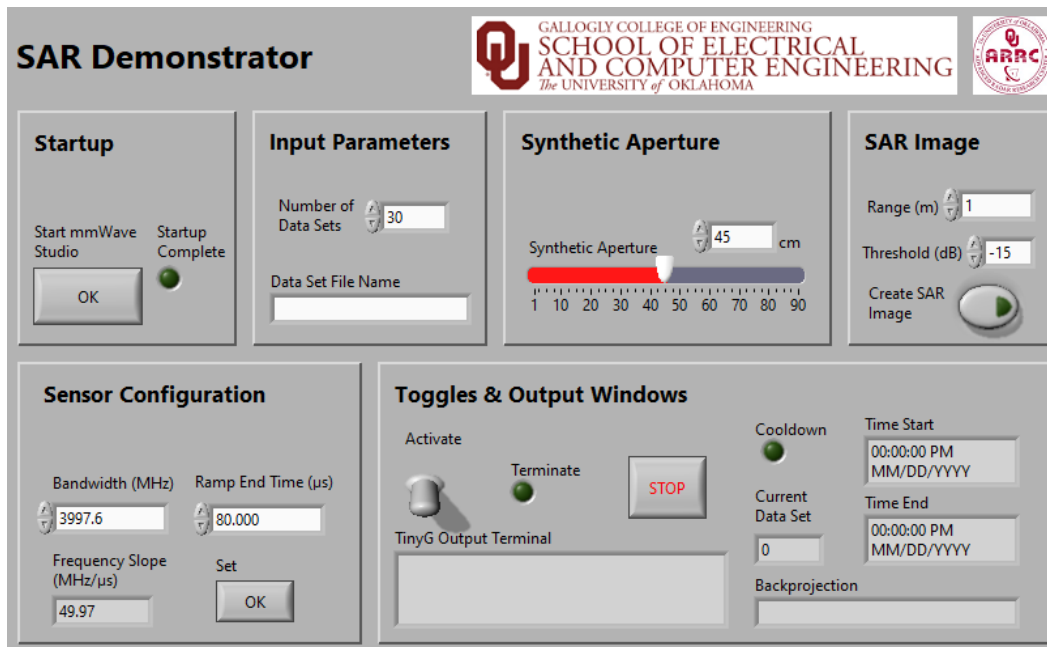


Figure 5.6: Closeup view of the LabVIEW GUI for user operations.

5.2.1 GUI Startup Section

The Startup section features one button and one indicator. The “Start mmWave Studio” button, when pressed, launches the mmWave Studio GUI and automatically performs every operation to set up the software. This is done by having a MATLAB script node run a MATLAB function to execute a custom-made Lua script that calls every command necessary for connecting and configuring the radar module to mmWave Studio. This Lua script calls each command for setting up mmWave Studio in the same startup sequence as listed in the mmWave Studio User Guide up until the radar sensor configuration. Once the startup is complete, the indicator will enable. This startup feature ensures the user will not have to manually input anything into mmWave Studio which keeps all user activity on the LabVIEW GUI

for running the SAR Demonstrator. This feature consistently works upon booting up the host computer. If the user starts up mmWave Studio using LabVIEW and later closes the program, it is advised to reboot the computer before running the LabVIEW program and starting up mmWave Studio again. The code for launching mmWave Studio is shown in Figure 5.7.

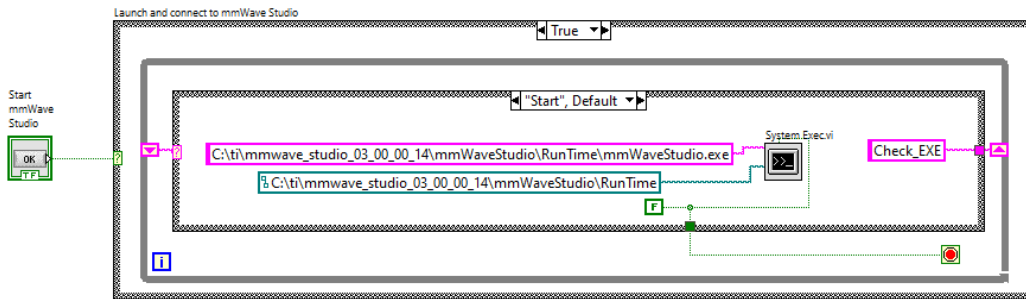


Figure 5.7: Launching mmWave Studio from LabVIEW.

5.2.2 GUI Input Parameters Section

The Input Parameters section is used for choosing the filename for every dataset and setting the total number of datasets for the system to capture. If the user leaves the dataset filename input as blank, the system will keep the default filename of “adc_data.bin” as given by mmWave Studio. Every dataset will have its corresponding dataset number added to the end of the filename upon renaming. For example, the first dataset captured will be renamed to “adc_data_1.bin” to avoid overwriting every new dataset. For the total number of datasets input, the default value is 30 datasets. Due to the length of the vertical actuator, the total number of datasets maxes out at 70 captures. The code for renaming the most recently captured dataset to the filename specified by the user is shown in Figure 5.8.

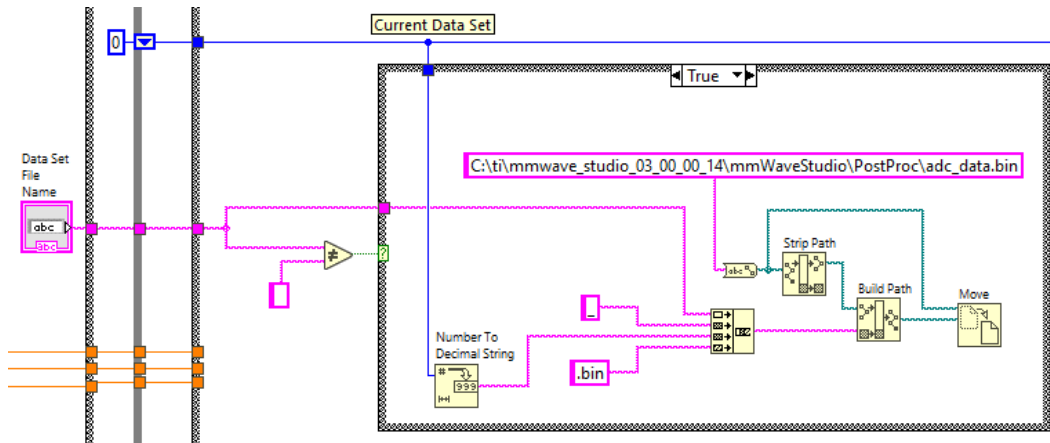


Figure 5.8: Renaming captured dataset to filename defined by user.

5.2.3 GUI Synthetic Aperture Section

The Synthetic Aperture section is used for setting the distance the radar module will travel across the horizontal actuator. The slide has a range between 1 cm and 90 cm for setting the distance. The commands for setting this distance are sent to the VISA write function for the TinyG board to read and execute. The value set for the synthetic aperture distance requires a conversion to a number equivalent to the TinyG travel maximum value using the motor's rate of rotation. A real-world 90 cm travel distance is equivalent to a 160 cm maximum travel for the TinyG board. The conversion and wiring used for setting the synthetic aperture distance is shown in Figure 5.9. The distance of the synthetic aperture also determines the number of frames the radar will trigger during a data capture. Although this value is not immediately configured in this section when the aperture distance is set. Therefore, the user should set the synthetic aperture distance before setting the values in the Sensor Configuration section, where the number of frames to be triggered is defined in mmWave Studio.

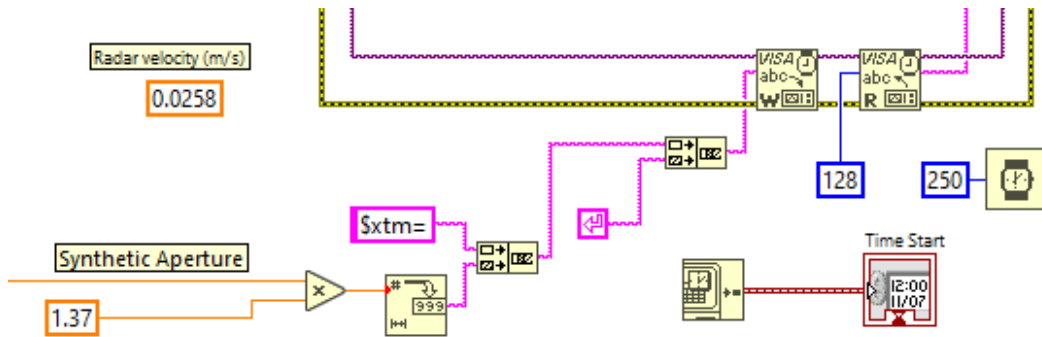


Figure 5.9: Defining the length of the horizontal synthetic aperture distance.

5.2.4 GUI SAR Image Section

Under the SAR Image section is a single button and a couple of numeric controls. If the user enables the “Create SAR Image” button, the program will automatically perform post-processing on recently captured SAR data for imaging. This operation is done for each dataset with the corresponding SAR image displayed in the window on the right side of the GUI. If only a single dataset is captured or the system captures the first dataset for a collection of data, the program will generate a 2-D range profile SAR image. Any subsequently captured datasets will be added to the pre-existing baseband data to produce volumetric SAR images. This allows the user to visualize the creation of the SAR image with each new data capture enhancing the elevation of targets. If the button is left disabled, the system will skip the post-processing and imaging altogether and only perform SAR data-capturing. The user can also define the range of the target scene in the azimuth dimension for the SAR image up to 10 meters using the control function. Finally, the user can define the threshold value for each pixel to be plotted. The details on how this threshold operates are explained in Chapter 6.

5.2.5 GUI Sensor Configurations Section

The Sensor Configuration section is used for defining the RF parameters in mmWave Studio. The user is normally unable to explicitly define the bandwidth in mmWave Studio. The ramp end time and frequency slope must be defined first for the bandwidth to be calculated using these values. However, the LabVIEW program allows for the user to define the desired bandwidth and the ramp end time. Since mmWave Studio can only set the bandwidth to a max value of 3997.6 MHz, this value is kept as the default bandwidth upon startup. A ramp end time of 80 μs is kept as the default value which results in a frequency slope of 49.97 MHz/ μs . Using these values, the frequency slope is calculated for mmWave Studio to define. If the user changes the ramp end time, a new periodicity is calculated to keep the duty cycle at 99.7%. Additionally, the number of frames the radar will trigger during a data capture is defined. This value is calculated using the distance of the horizontal synthetic aperture set by the user. A mathematical conversion is used to determine this value based on how many frames the radar can trigger while in motion with its constant velocity before stopping. Clicking the “Set” button allows for these parameters to be defined in mmWave Studio. This is done through the creation of a Lua script which contains the commands to make the configurations.

For every new sensor configuration, its corresponding Lua script must be edited to replace the old values with the new ones. In LabVIEW, each command is defined as a string value and concatenated together along with the RF parameter values in the correct position. Manually creating a new concatenated string allows the user to edit the Lua script with any desired changes. Replacing the old file, this

string is then saved as a .txt file and converted into a .lua file to be executed in MATLAB and subsequently passed into mmWave Studio. The sequence of code for this configuration is shown in Figure 5.10.

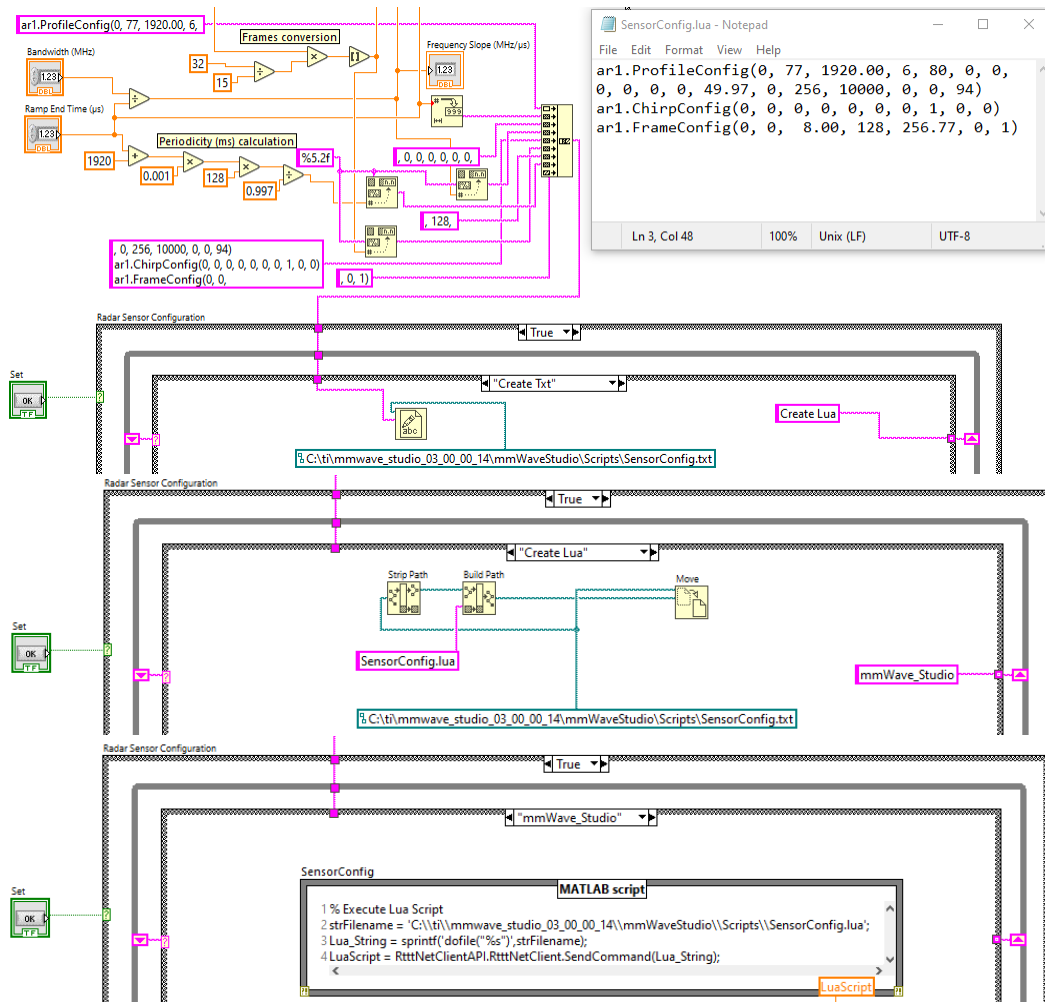


Figure 5.10: Sequence for configuring the radar module from LabVIEW.

5.2.6 GUI Toggles & Output Windows Section

The Toggles & Output Windows section is used for activating and shutting down the SAR Demonstrator as well as monitoring the progress of the system while it is capturing SAR datasets. There are two toggles, each for activating the system and shutting down the program. The Activate switch triggers the system to begin the SAR data capture process. The switch defaults to the “Initialize” case in the SAR data capture sequence of events, all of which will be covered later in this chapter. The Stop button disconnects mmWave Studio from the radar module and shuts down the LabVIEW program.

The output windows showcase different parameters for monitoring the status of the system. The TinyG Output Terminal functions as the command window for the serial port terminal application running the TinyG board. This terminal showcases the command lines of the TinyG board from given input commands for configuring settings and running Gcodes. The Terminate indicator will turn on when the system has finished collecting all the datasets and the gantry carts have returned to their original starting positions. The Current Data Set indicator displays which dataset the system is currently capturing. This reveals the progress of the SAR data-capturing when compared to the total number of datasets to be captured. The Backprojection indicator reveals whether the current dataset is undergoing post-processing and reveals when the backprojection code has finished running. The Time Start and Time End indicators reveal the time and date for when the system is activated and when the system resets after the SAR data-capturing is done. Finally, the Cooldown indicator turns on when homing commands are being sent to

the TinyG board to change the rotational direction of the next stepper motor to run in the SAR data-capturing sequence. There is often a short period of idleness from the system during cooldowns to allow the TinyG board to fully process all the given commands before activating a motor.

5.3 SAR System Sequence

Now that LabVIEW GUI has been explained in detail, the design pattern of the SAR data-capturing software will be covered with every step showcased. This sequence can be modeled as a state machine with a while loop containing a case structure with each case representing a step in the SAR data-capturing software. There are a total of 13 cases in the sequence for the program to execute. This sequence is designed to have the system autonomously perform multiple dataset captures without the user having to manually reconfigure and trigger the radar module for each SAR data capture. A block diagram showcasing the sequence of events for each case in the SAR data-capturing software is shown in Figure 5.11.

The basic layout of the code for the SAR data-capturing software consists of two case structures and a while loop. A case structure is used to contain every event in the SAR data-capturing sequence. This case structure is contained within a while loop which holds some of the indicators for the GUI and two pairs of shift registers for storing variables. The variable for storing the current dataset being captured is initialized to zero upon startup and stored in a shift register. Every case contains at least one possible string value with the name of the next case to be executed in the sequence which is stored in a shift register. The string shift registers allow the program to determine which case will be executed on the next iteration of the loop

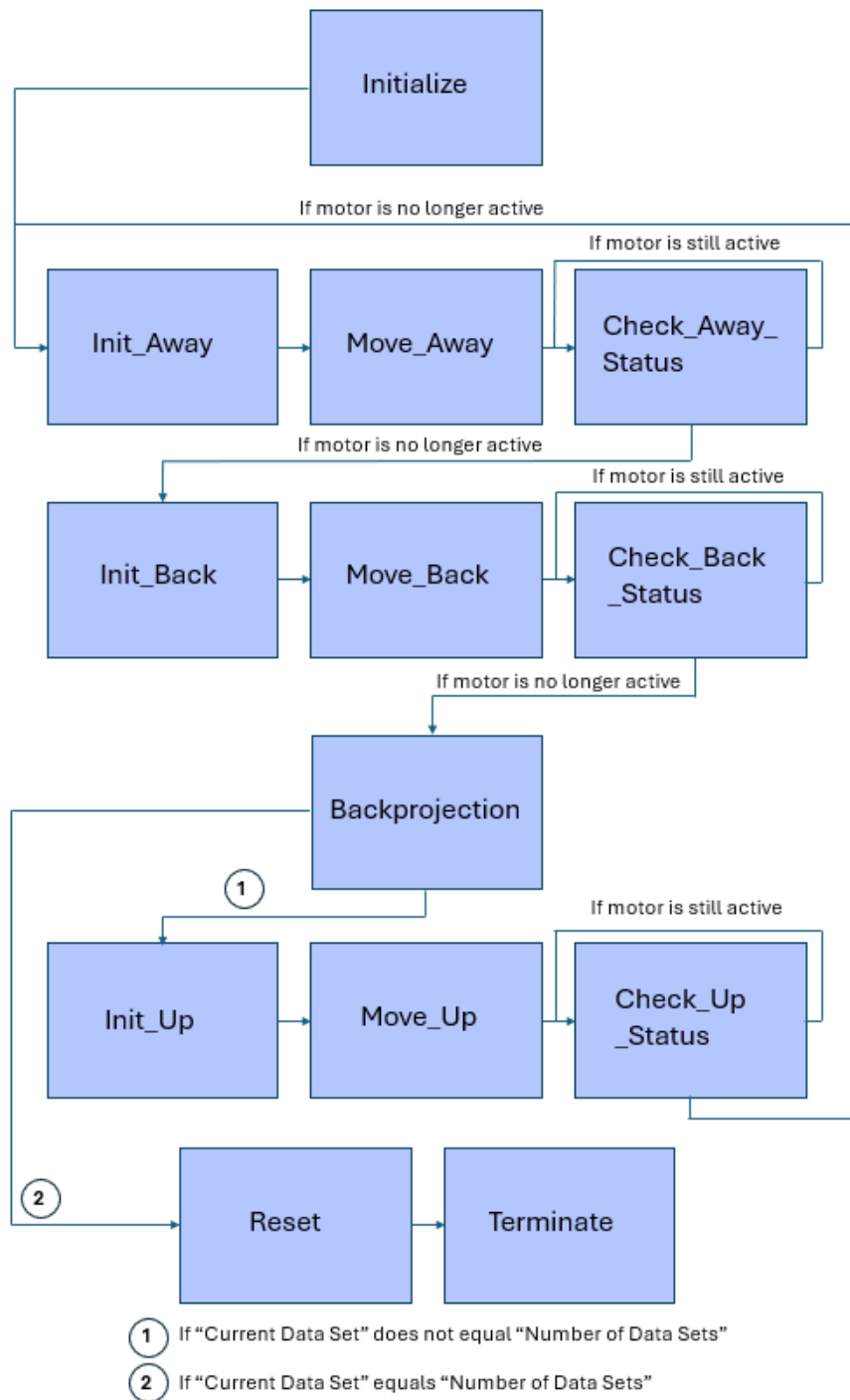


Figure 5.11: Block diagram of the SAR data capture sequence by case name.

by connecting the left shift register to the selector of the case structure. The while loop is contained within a true/false case structure with the selector connected to the activation switch on the GUI.

The “Initialize” case serves as the default case for starting up the sequence. This case consists of the code for connecting the LabVIEW program to mmWave Studio, configuring the distance of the horizontal synthetic aperture, and recording the start time of the sequence. The MATLAB script node contains the code for executing the RSTD_Interface_Example.m program. The command for setting the maximum travel distance is written to the TinyG board using the value chosen in the synthetic aperture section of the GUI. Having code for running the TinyG board and configuring mmWave Studio in the same case allows for both operations to be executed together at the same time. This general design allows for the synchronization of both subsystems to create a single functioning autonomous SAR system. Both the overall architecture for the SAR data-capturing software along with the “Initialize” case is shown in Figure 5.12.

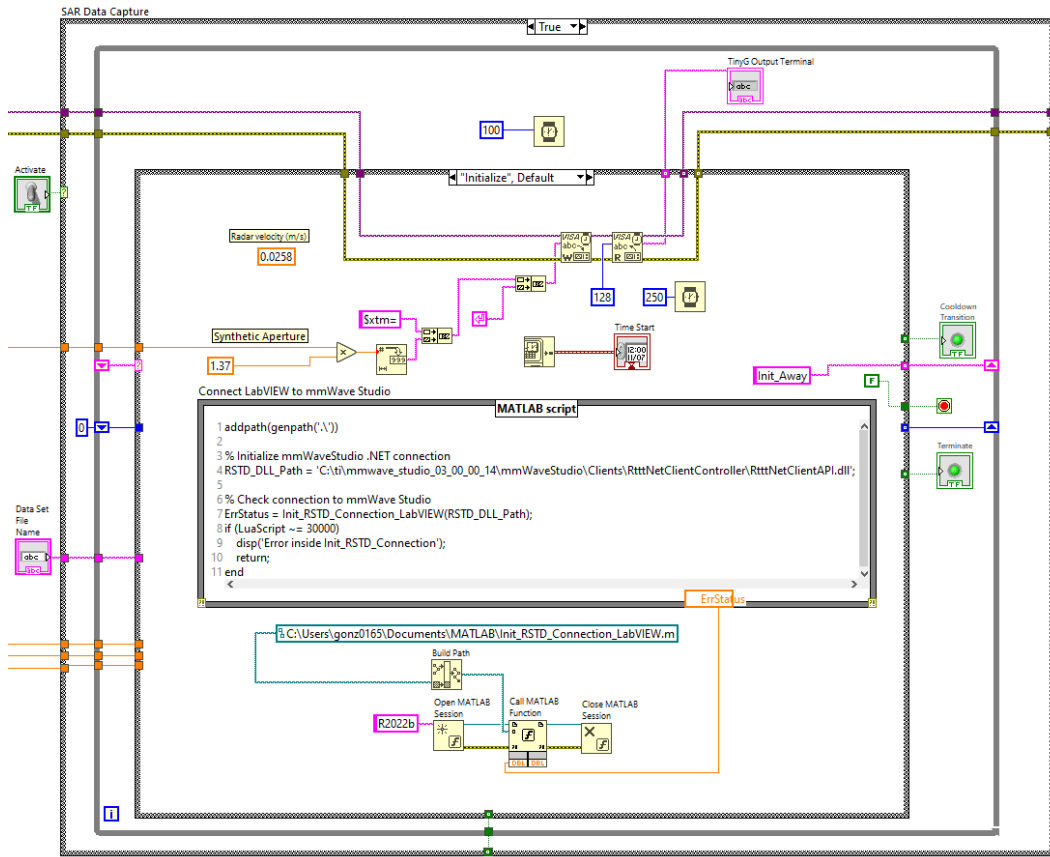


Figure 5.12: Overview of the SAR data capture code and the “Initialize” case.

5.3.1 SAR Data Capture Sequence

The next case is the “Init_Away” case which refers to initializing the sequence for when the radar module travels away from its starting position to perform an SAR data capture while in motion. In this case, the commands for homing the system to the Xmax limit switch and setting the homing search velocity are written to the TinyG board. Additionally, the MATLAB script node executes the Lua script for configuring the DCA1000EVM in mmWave Studio which prepares the radar module to store a captured dataset. Finally, the variable for storing the current

dataset being captured in the sequence is incremented by one and stored in a shift register. The code contained in this case is shown in Figure 5.13. Once these operations are complete, the radar module is ready to capture a dataset.

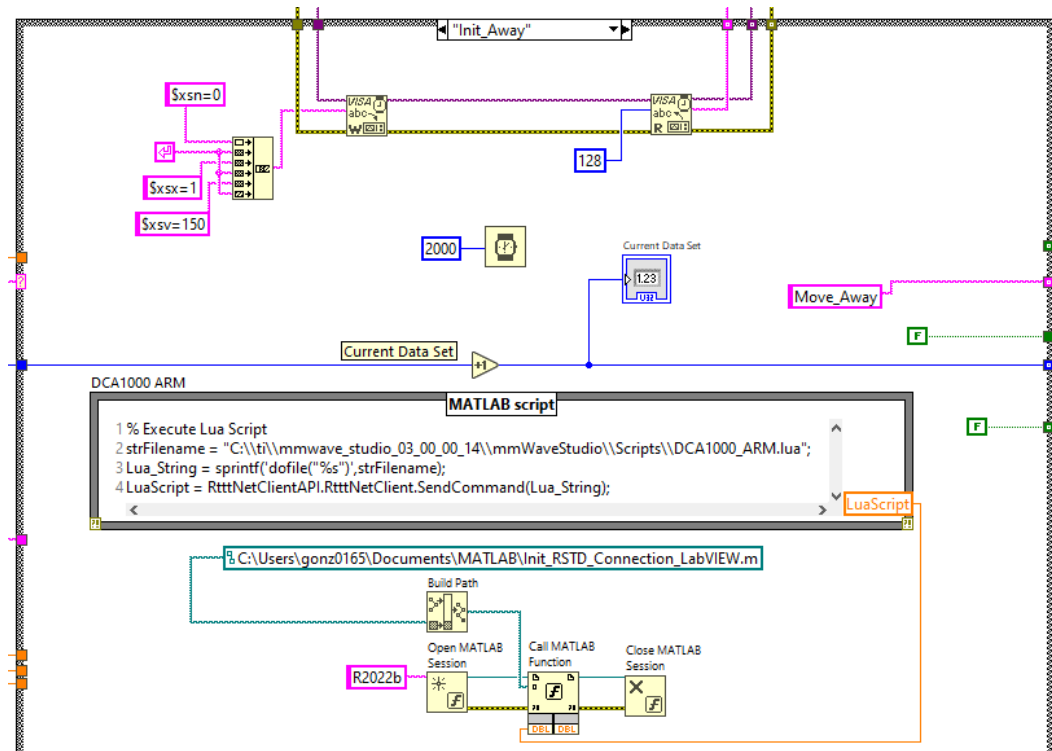


Figure 5.13: The “Init_Away” case.

The next case is the “Move_Away” case which performs the dataset capture. In this case, the MATLAB script node executes the Lua script for running the Trigger Frame command in mmWave Studio which immediately begins the dataset capture. The Gcode for activating the homing sequence in the X-axis is also written on the TinyG board. The radar module moves with a realized velocity of 0.0258 meters per second. The system synchronizes the TinyG board and mmWave Studio to execute at the same time such that the Trigger Frame activates the moment the radar

module begins sliding across the horizontal track. This is accomplished by having initialized both the TinyG board and executing the mmWave Studio DCA1000 ARM command in the previous case. A Lua script was written to execute the Trigger Frame command in mmWave Studio. Executing this Lua script and the TinyG homing command synchronizes the radar module to begin capturing data the moment it begins traveling across the horizontal track. This is followed by the radar module triggering every frame right before the motion ends. The code contained in this case is shown in Figure 5.14.

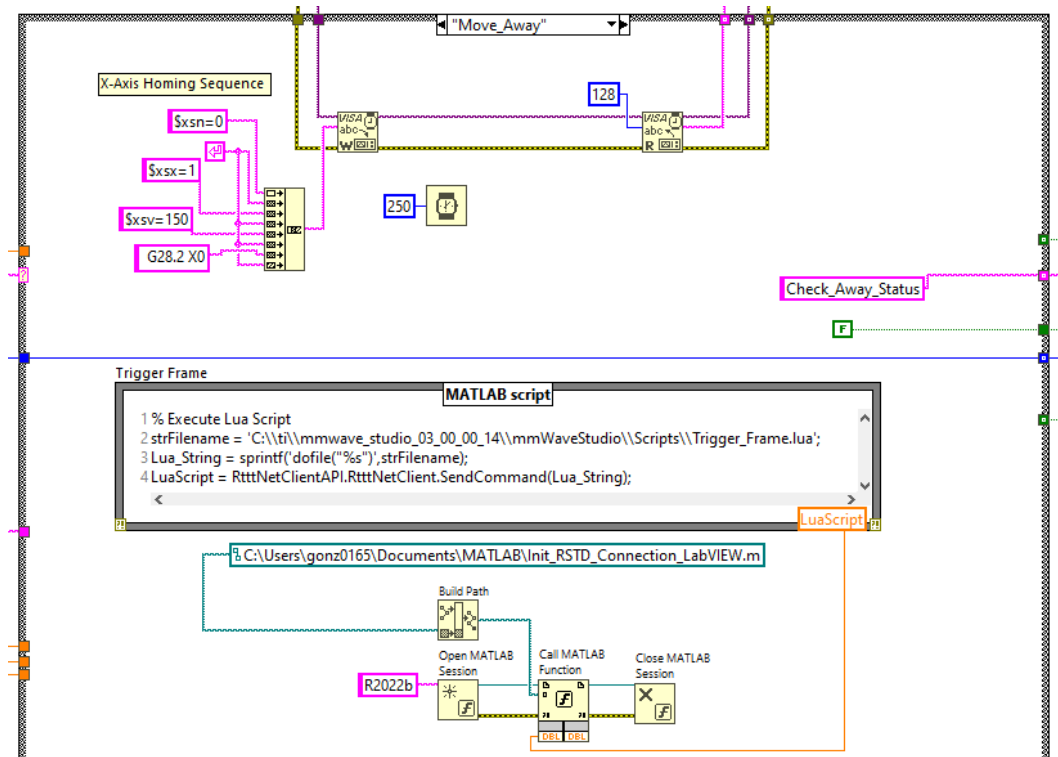


Figure 5.14: The “Move_Away” case.

The next case is the “Check_Away_Status” case which continuously checks on the status of the TinyG to determine when the radar module is no longer in motion.

In this case, the LabVIEW read function checks the output terminal of the TinyG board for an empty output. If the TinyG board is still tracking the rotational position of the motor, the case will send a string containing its name to the shift register to run the code again. This process is done repeatedly until the TinyG board is no longer tracking the position of the radar module, resulting in an empty output. The program will then move on to the next case in the sequence. When this condition is true, the cooldown indicator on the GUI will turn on. The code contained in this case is shown in Figure 5.15.

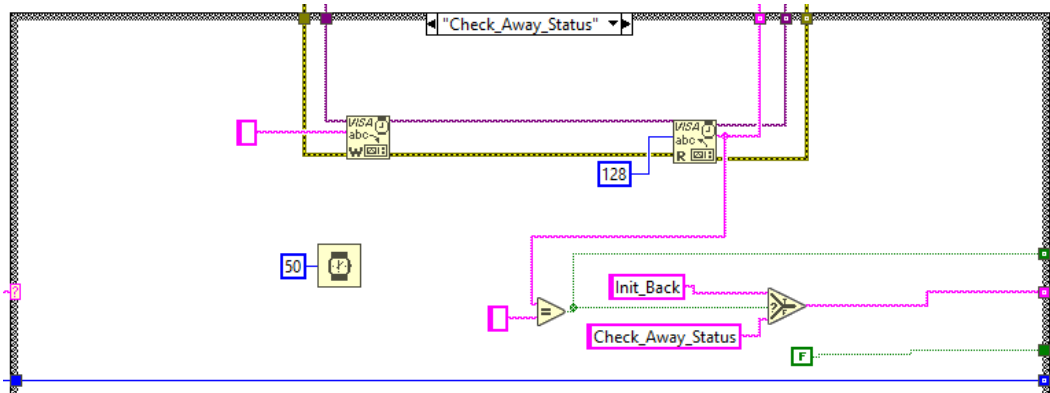


Figure 5.15: The “Check_Away_Status” case.

5.3.2 Position Reset Sequence

The next case is the “Init_Back” case which refers to initializing the sequence for moving the radar back to its starting position post-capture. In this case, the MATLAB script node executes the Lua script for running the PostProc button in mmWave Studio. This ends the data capture and performs post-processing on the captured raw ADC data to create the corresponding adc_data.bin file. Furthermore, the commands for homing the system to the Xmin limit switch and setting the hom-

ing search velocity are written to the TinyG board. The homing search velocity is increased to speed up the motion sequence since this velocity does not affect the recently captured dataset. The code contained in this case is shown in Figure 5.16.

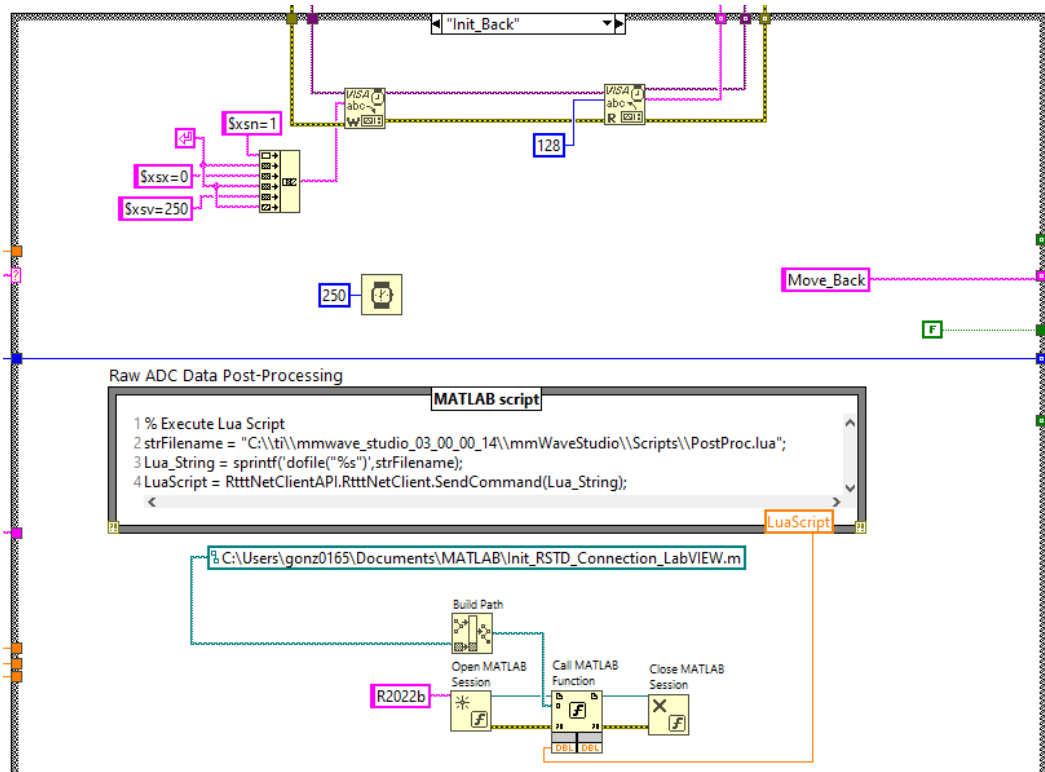


Figure 5.16: The “Init_Back” case.

The next case is the “Move_Back” case which returns the radar module back to its starting position and renames the .bin file of the recently captured data. The Gcode for activating the homing sequence in the X-axis is written to the TinyG board. A series of LabVIEW functions take the recently created adc_data.bin file and rename it as the dataset filename as described in the input parameters section of the GUI. Furthermore, the dataset number is added onto the end of the filename. The code contained in this case is shown in Figure 5.17.

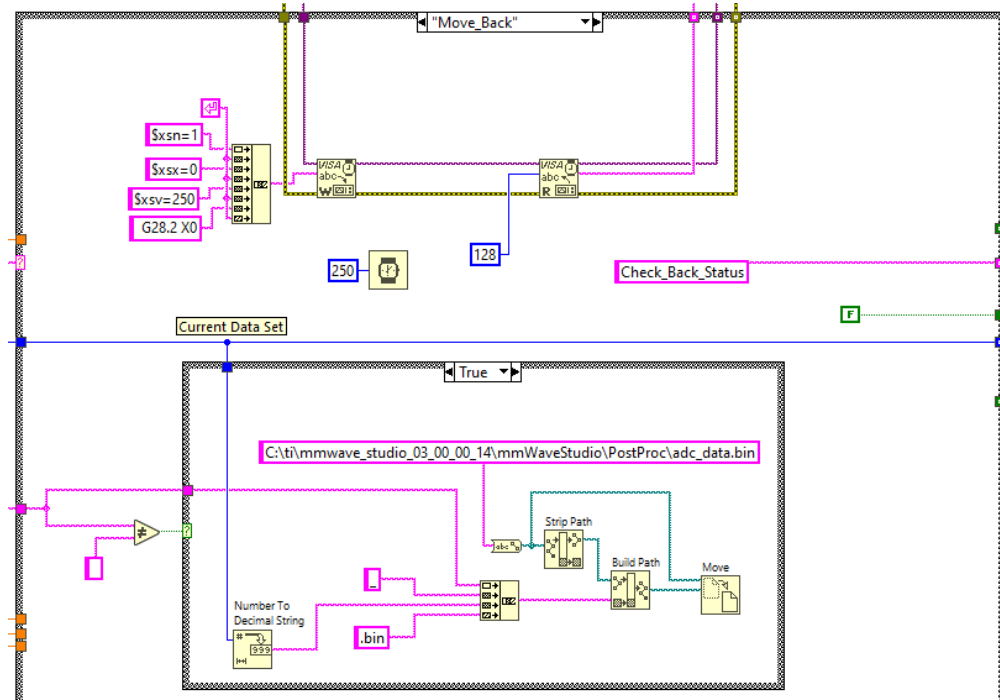


Figure 5.17: The “Move_Back” case.

The next case is the “Check_Back_Status” case which functions almost identically to the previously described “Check_Away_Status” case. The only difference is the name of the case next in the sequence when the select function condition is true. The code contained in this case is shown in 5.18.

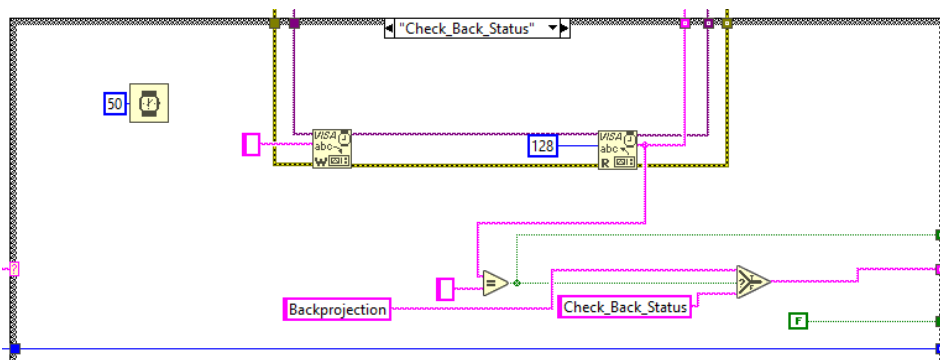


Figure 5.18: The “Check_Back_Status” case.

5.3.3 SAR Data Post-Processing Sequence

The next case is the “Backprojection” case which performs post-processing on the recently captured dataset and displays the corresponding SAR image. If the “Create SAR Image” button on the GUI is pressed, the case will perform all the post-processing and imaging operations. This case contains an additional case structure consisting of a new set of cases for performing the SAR data post-processing sequence. A block diagram of the Backprojection sequence of events is shown in Figure 5.19.

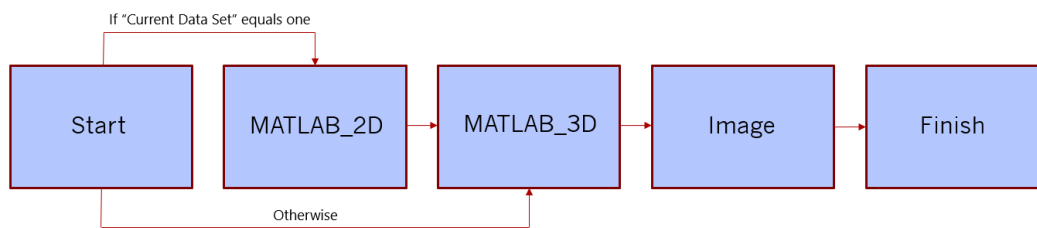


Figure 5.19: Block diagram of the LabVIEW sequence for performing post processing.

The first case in Backprojection sequence is the Start case which determines the MATLAB program to be used for performing post-processing on the most recently captured dataset. If this dataset is the first out of a collection, the program for creating a 2-D SAR image will be executed. Otherwise, the program for creating a volumetric SAR image will be executed instead. Starting with this case, the Backprojection output window will read “Backprojection in progress...” in the GUI. If the “Create SAR Image” button is disabled, this case will pass the name of the next case in the SAR data capture sequence to the shift register. The design of this case is identical to the “Finish” case in the Backprojection sequence which is showcased

frames triggered calculated based on the length of the horizontal synthetic aperture. If the dataset being post-processed is the only one being captured, the program will move on to the “Image” case. Otherwise, the program will move on to the case for performing post-processing to create a volumetric image. The process for applying the backprojection algorithm to SAR datasets will be covered in Chapter 6. The code contained in this case is shown in Figure 5.21.

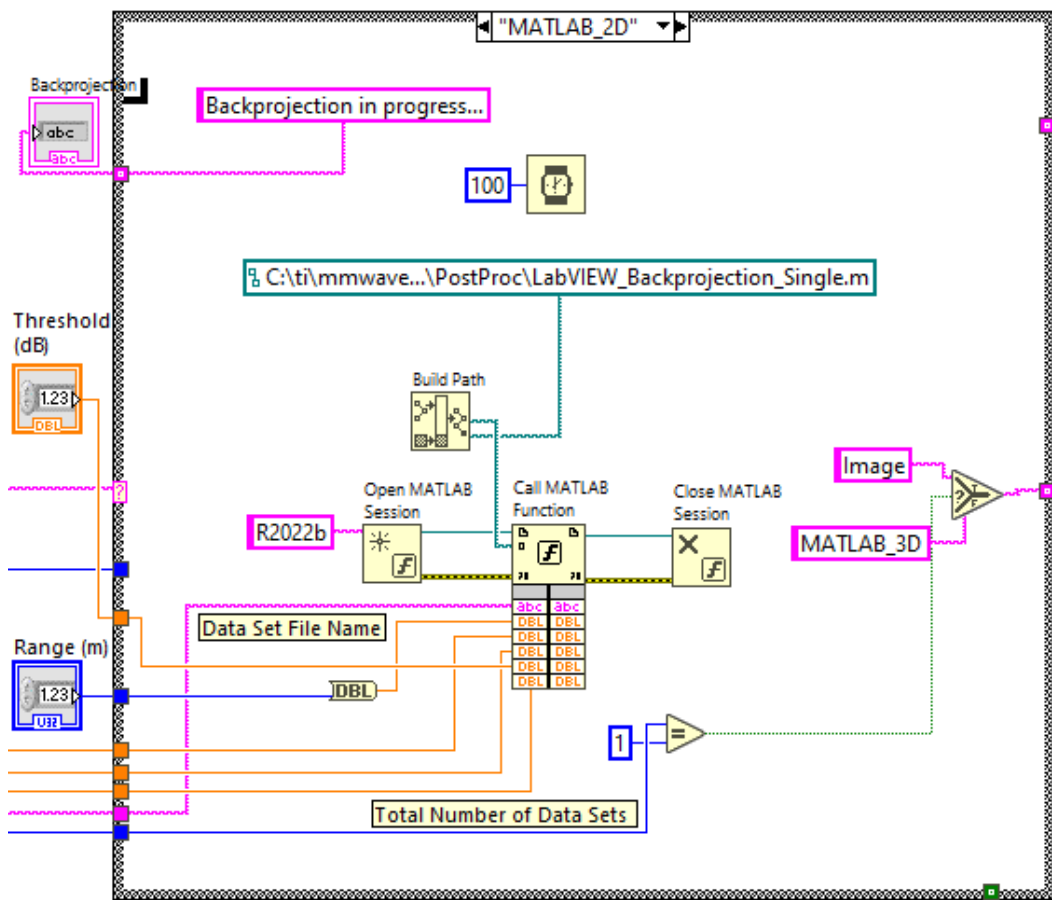


Figure 5.21: The “MATLAB_2D” case.

The “MATLAB_3D” case executes the program for performing post-processing on a single dataset to create a volumetric SAR image. If the dataset being processed

is the first to be captured, the program will initialize every parameter and create the baseband dataset to hold all the subsequently captured data. These parameters are saved onto the host computer and loaded back into the program for the next dataset. Any additional datasets will be parsed into the baseband dataset and individually processed before being added to the volumetric image. The program reads seven inputs: the same six inputs read for the “MATLAB_2D” case as well as the total number of datasets to be captured. The code contained in this case is shown in Figure 5.22.

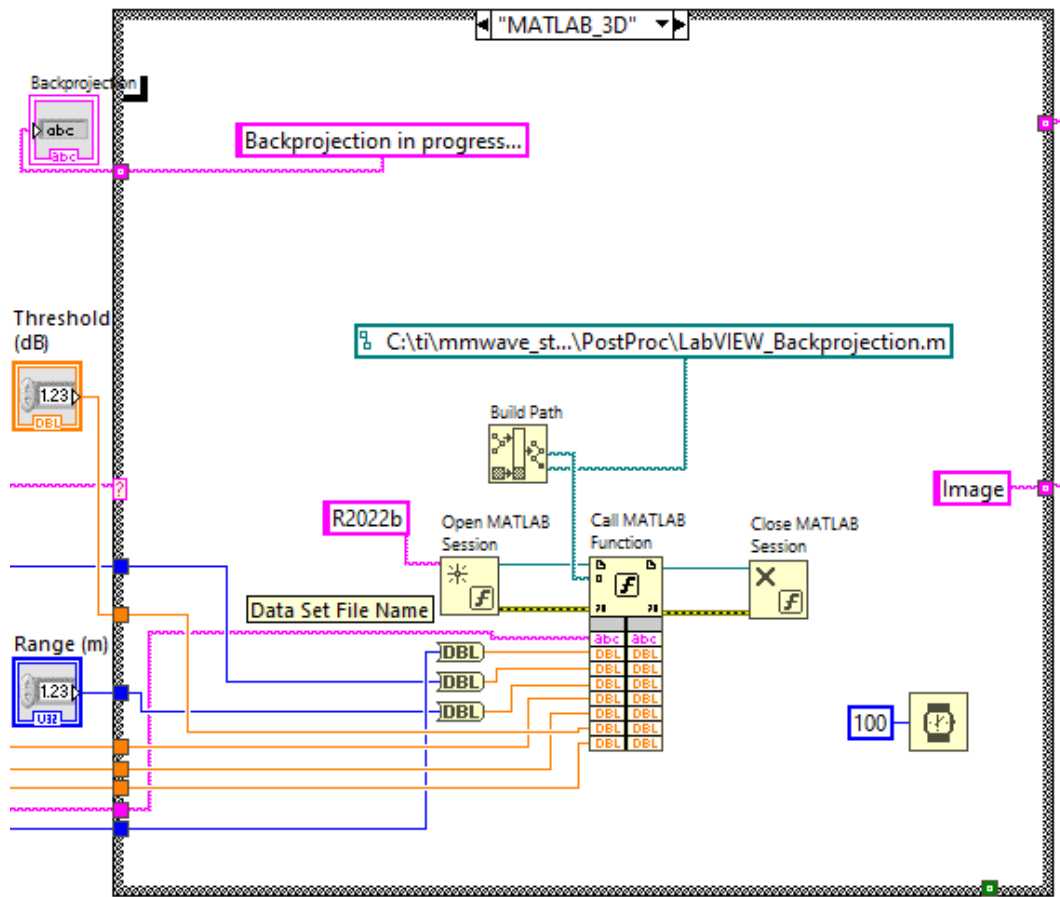


Figure 5.22: The “MATLAB_3D” case.

The “Image” case displays the SAR image created from the previously mentioned MATLAB post-processing programs. The fully realized SAR image file gets saved onto the host computer after the post-processing program has finished running. Using the “Read PNG File” LabVIEW function, the image file is read into LabVIEW and rescaled before being displayed in the SAR image window in the GUI. The code contained in this case is shown in Figure 5.23.

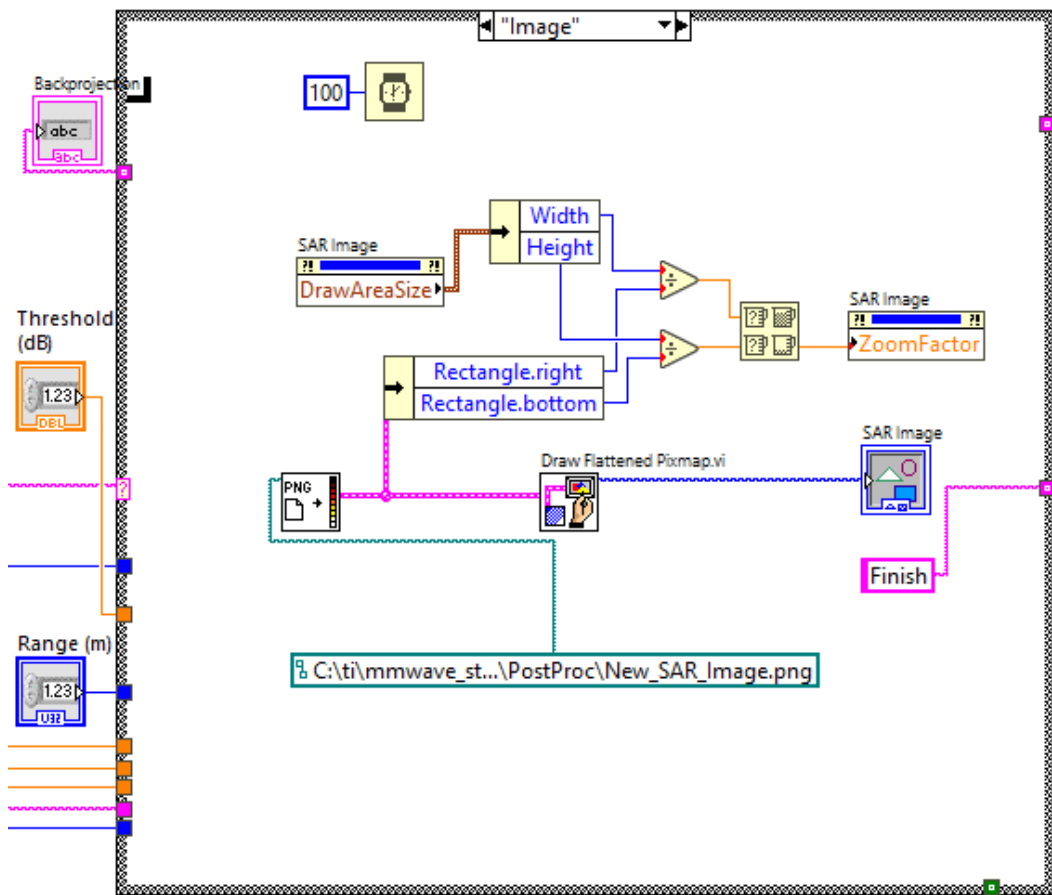


Figure 5.23: The “Image” case.

The last case in the Backprojection sequence is the “Finish” case which determines the next case in the SAR system sequence to be executed. If the most recently

captured dataset was the final dataset to be captured, the program will move on to the “Reset” case. Otherwise, the program will move on to the “Init_Up” case to repeat the process of an SAR data capture for a new dataset. The Backprojection output window will also now read “Backprojection complete” on the GUI. The code contained in this case is shown in Figure 5.24.

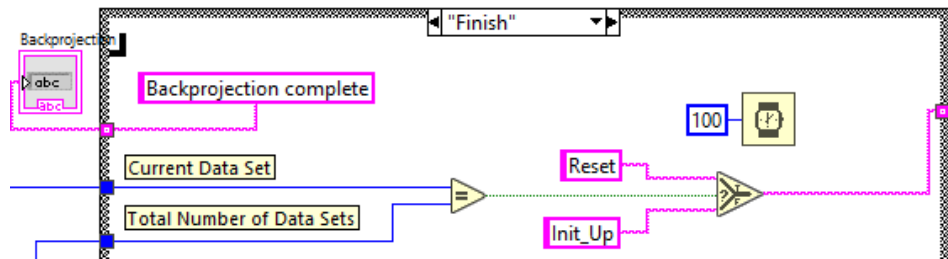


Figure 5.24: The “Finish” case.

5.3.4 Vertical Positioning Sequence

The “Init_Up” initializes the TinyG board to control the motion sequence for the vertical actuator. In this case, the commands for homing the system to the Ymax limit switch are written to the TinyG board. The code contained in this case is shown in Figure 5.25.

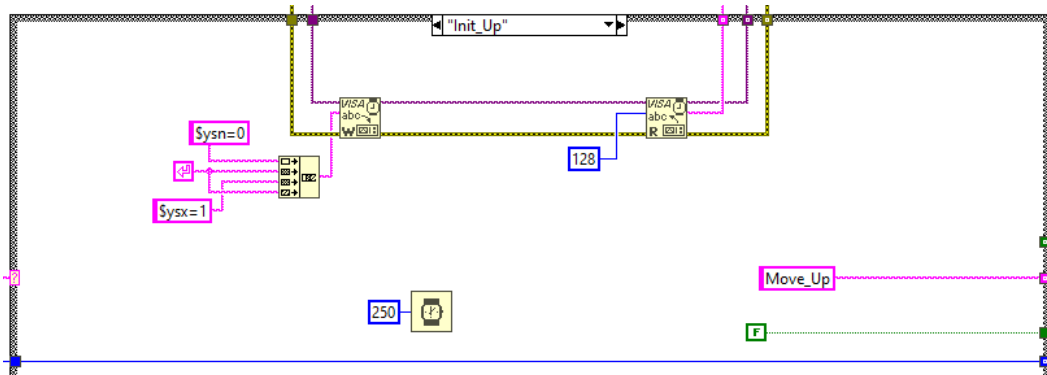


Figure 5.25: The “Init_Up” case.

The next case is the “Move_Up” case which raises the vertical position of the radar module by 1.25 mm for the next SAR data capture. In this case, the Gcode for activating the homing sequence in the Y-axis is written to the TinyG board. The code contained in this case is shown in Figure 5.26.

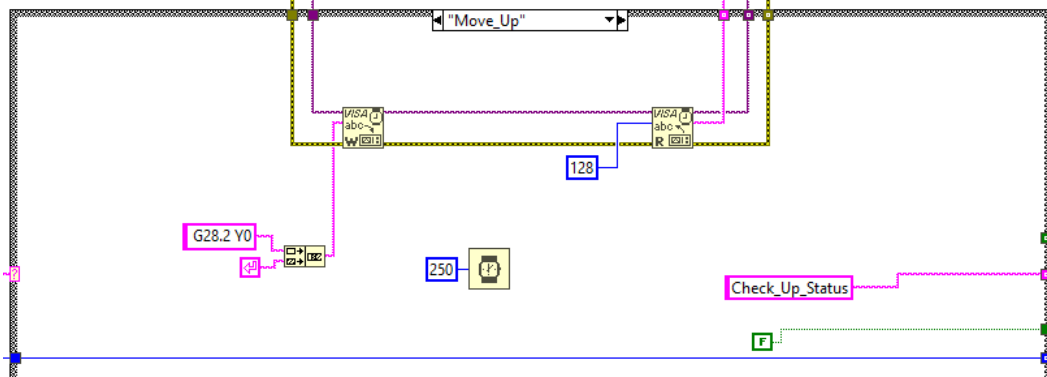


Figure 5.26: The “Move_Up” case.

The next case is the “Check_Up_Status” case which once again functions almost identically to the previously described cases that continuously checks the status of the TinyG board. The next case in the sequence is the previously used “Init_Away” case to reinitialize the TinyG board and radar module for repeating the SAR data capture process. The code contained in this case is shown in Figure 5.27.

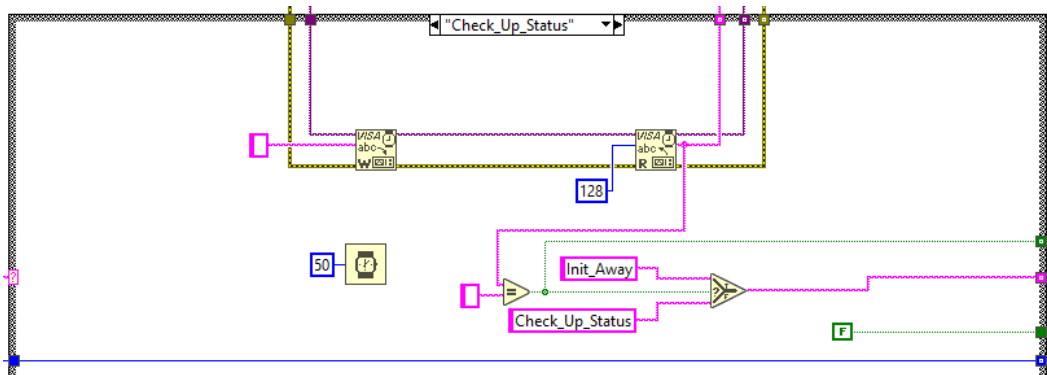


Figure 5.27: The “Check_Up_Status” case.

5.3.5 System Reset Sequence

The next case is the “Reset” case which returns the gantry carts to their initial starting position from the start of the sequence. The Gcode for changing the saved positions of the stepper motors to their previously saved starting positions is written to the TinyG board. The gantry cart for the horizontal actuator will have already returned to its starting position, resulting in no changes. The gantry cart for the vertical actuator will travel down back to its starting position if multiple SAR datasets have been captured. The code contained in this case is shown in Figure 5.28.

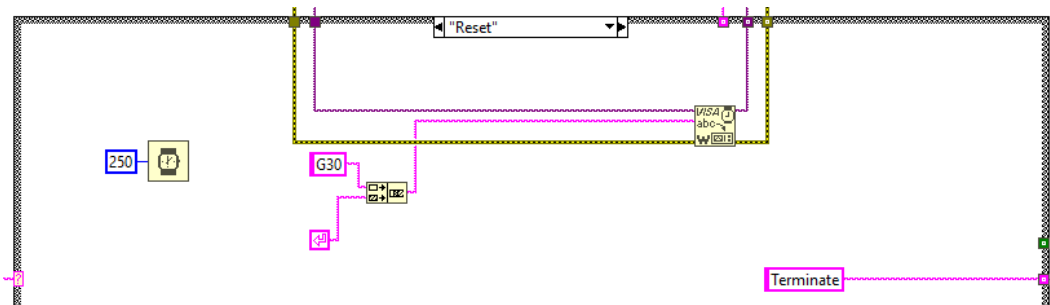


Figure 5.28: The “Reset” case.

The final case in the SAR system sequence is the “Terminate” case which ends the sequence. The current time for this case is recorded and displayed on the GUI below the start time. The Terminate indicator on the GUI will turn on in this case, informing the user that the system has concluded the SAR data capture. The conditional terminal for both while loops in the program are now connected to a true constant, ending any current operations for the system. The code contained in this case is shown in Figure 5.29.

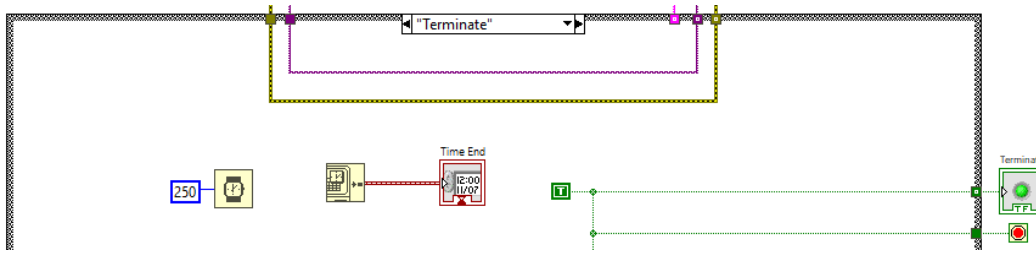


Figure 5.29: The “Terminate” case.

5.4 Summary

In this chapter, the LabVIEW program for executing the SAR Demonstrator was discussed and showcased. The program was designed to serve as the master controller for autonomously managing both the hardware’s motion control system and the radar module’s interface. Individual states for the automation were designed to arrange the motion sequence for the radar module when performing an SAR data capture. The realization of this program has allowed for the SAR Demonstrator to function as a fully autonomous and complete radar system for capturing multiple SAR datasets. The steps for performing post-processing on datasets captured using this system are detailed in Chapter 6.

Chapter 6

SAR Post-Processing

In this chapter, the post-processing sequence of captured datasets is showcased in detail. This starts with how the raw data is reformatted to match the structure of a baseband data cube. Next is a sequence of radar signal processing techniques to form range profiles. Finally, the implementation of the backprojection algorithm is revealed and discussed, followed by how the resulting SAR image is formatted.

6.1 Radar Baseband Data

For post-processing, the following parameters are defined in Table 6.1. These variables correspond to the sensor configuration profile used for the radar. The radar velocity and pulse repetition time are used to determine the position of the radar at each pulse. The default values for chirp rate, idle time, and ramp end time established in the LabVIEW GUI are only being considered. Furthermore, every dataset collected for this thesis was captured using 8 frames, resulting in 1024 pulses per dataset.

The raw ADC data is captured over four LVDS lanes and is stored in a binary file in an interleaved format with each lane corresponding to a receiver [7]. The

Parameter	Parameter Value
Carrier Frequency	77 GHz
Chirp Rate	49.97 MHz/ μ s
Idle Time	1920 μ s
Ramp End Time	80 μ s
Radar Velocity	0.0258 m/s
Z-axis Increment Distance	1.25 mm
ADC Samples	256
Number of Pulses	1024
Sampling Frequency	10 MHz

Table 6.1: The parameter values for performing post-processing.

file size of a dataset is directly proportional to the number of pulses transmitted and received per frame. A single frame of data with four receive channels results in a file size of 0.5 MB. LVDS over Ethernet streaming can be used in two modes of operation: raw mode and data separated mode. The DCA1000EVM is set to raw mode, which results in the FPGA recording all the data appearing on the LVDS lines as is. The raw data sent over the LVDS lanes is controlled by the configuration from mmWave Studio. The data must be formatted in such a way that allows for it to be programmatically interpreted correctly for post-processing followed by backprojection. An example of the initial format for the binary files is shown in Figure 6.1.

The raw data is initially separated into real and imaginary components per sample. This is due to each sample being captured for each LVDS lane per receiver. These components are combined and then programmatically reshaped to form the complex baseband data. The formatted data results in a matrix with all the received data stored and organized into four channels for each receiver [22]. Each row contains a number of columns equal to the number of ADC samples per chirp

Chirp 1	Rx0I0	Rx1I0	Rx2I0	Rx3I0	Rx0Q0	Rx1Q0	Rx2Q0	Rx3Q0
	Rx0I1	Rx1I1	Rx2I1	Rx3I1	Rx0Q1	Rx1Q1	Rx2Q1	Rx3Q1

	Rx0IN-1	Rx1IN-1	Rx2IN-1	Rx3IN-1	Rx0QN-1	Rx1QN-1	Rx2QN-1	Rx3QN-1
Chirp 2	Rx0I0	Rx1I0	Rx2I0	Rx3I0	Rx0Q0	Rx1Q0	Rx2Q0	Rx3Q0
	Rx0I1	Rx1I1	Rx2I1	Rx3I1	Rx0Q1	Rx1Q1	Rx2Q1	Rx3Q1

	Rx0IN-1	Rx1IN-1	Rx2IN-1	Rx3IN-1	Rx0QN-1	Rx1QN-1	Rx2QN-1	Rx3QN-1

Figure 6.1: The DCA1000EVM captured data format [7].

multiplied by the total number of chirps. Since there are 256 ADC samples and 1024 pulses or chirps, each row will contain 262144 columns. Therefore, the first 256 columns correspond to the first chirp, the next 256 columns to the second chirp, and so on. The resulting matrix with all the channel data stored in rows is shown in Figure 6.2. The data is now ready to be processed as desired. Moving forward, only the data for the first channel is processed.

	1	2	3	4	5	6	7	8
1	-2.9000e+0...	-4.0300e+0...	1.8400e+0...	6.0100e+0...	1.7600e+0...	9.5000e+0...	5.5900e+0...	6.2900e+0...
2	4.1900e+0...	5.4500e+0...	9.4600e+0...	1.1780e+0...	6.1500e+0...	1.4300e+0...	1.0600e+0...	-1.6400e+0...
3	5.6700e+0...	6.8900e+0...	8.8400e+0...	7.9100e+0...	5.7500e+0...	7.5000e+0...	-3.1200e+0...	-5.4700e+0...
4	7.0600e+0...	5.4000e+0...	4.2300e+0...	1.0800e+0...	-2.1400e+0...	-4.9500e+0...	-6.2100e+0...	-7.5000e+0...

Figure 6.2: The baseband data format in MATLAB.

6.2 Data Cube Processing

Reshaping all the baseband data into a data cube organizes and prepares the raw radar data into a structured format suitable for processing. In this format, the dimensions correspond to fast-time, slow-time, and receiver channels. Fast-time

represents the number of range samples, while slow-time represents the number of pulses per range sample. Together, these dimensions form the data structure of a pulsed radar system [8]. Since receiver channels are also considered, the overall data structure for the pulsed radar data becomes three-dimensional and forms a data cube. For the collected SAR data, the cube's axes correspond to range bins (ADC samples), pulse repetitions (number of pulses), and vertical samples (number of datasets). This allows for the data to be formatted as a multi-channel data matrix as shown in Figure 6.3.

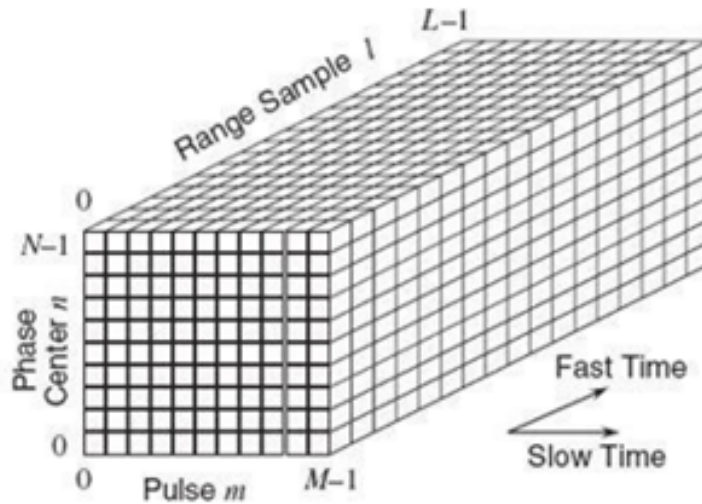


Figure 6.3: Radar data cube format constructed from SAR data [8].

Next, the data cube needs to undergo a windowing operation. The Hanning window was selected since it is designed to reduce spectral leakage in Fourier analysis [23]. This allows for preserving signal energy in the main lobe while attenuating energy in the sidelobes. This improves the accuracy of frequency domain analysis before the data undergoes a Fourier Transform. Using the built-in MATLAB *hann* function, a Hanning window is applied to the data cube for each range sample fol-

lowed by windowing along the azimuth and elevation dimensions. Afterward, the windowed data cube undergoes a Fast Fourier Transform with respect to the range axis, converting the time-domain data into the frequency domain. Next, the range profile data corresponding to the data cube is interpolated using the MATLAB *interpft* function. This is followed by pulse compression using the MATLAB *squeeze* function. This new interpolated data cube is now ready for backprojection.

6.3 Backprojection

The backprojection algorithm efficiently implements matched filtering and is adaptable to any range/velocity motion profile. This algorithm offers the advantage of generating SAR images incrementally as phase history accumulates pulse by pulse, integrating newly obtained data into the SAR image as it becomes available [24].

6.3.1 Image Formation Process

First, the fields *AntX*, *AntY*, and *AntZ* are initialized in a local Cartesian coordinate system with the origin at scene center. These represent the initial positions of the radar antenna in the X, Y, and Z directions, respectively. The X-axis represents the horizontal synthetic aperture, the Y-axis represents the azimuth dimension, and the Z-axis represents the elevation dimension. The linear positions of the antenna in the X-axis and Y-axis are calculated with respect to the radar's velocity on the track and pulse repetition time. The linear position of the antenna in the Z-axis is calculated based on the number of vertical samples and the incremental change in the

Z direction. The pixel resolution is also defined to determine the spacing between pixels. This value is kept at 0.05 meters by default. To construct the range profile of the target scene, the image needs to be formed onto a regular 2-D or 3-D grid. In this case, the MATLAB function *meshgrid* is used to build the pixel location matrices.

A range axis is defined with respect to the frequency axis corresponding to the baseband data. A new range corresponding to the interpolated data points is computed, ensuring that the interpolated data aligns correctly with the original range axis and maintains the correct range setup. Following this, the MATLAB *ndgrid* function is used to create the rectangular grid for the x, y, and z coordinates of the SAR image. These coordinates can then be compared with the location of the target to compute the pixel range on each pulse during backprojection. Using this grid allows for the initialization of the SAR image.

6.3.2 Backprojection Algorithm

To reconstruct the SAR image, the backprojection algorithm with imaging equations is implemented. First, the differential range (R_m) must be computed for every pixel corresponding to each pulse per vertical slice. This range measures the distance from the antenna phase center to any targets using the previously defined pixel coordinates [24]. Every pixel coordinate in the image is represented by x , y , and z in the X, Y, and Z direction respectively. Furthermore, $AntX$, $AntY$, and $AntZ$ correspond to the antenna location for every pulse in the X, Y, and Z direction respectively. The equation for computing the differential range on the current pulse

is the following:

$$R_m = \sqrt{(x - AntX)^2 + (y - AntY)^2 + (z - AntZ)^2} \quad (6.1)$$

Next, the range profile for the pixel on the current pulse must be linearly interpolated. The MATLAB *interp1* function is used to implement linear interpolation. Afterward, the phase correction (Φ) is applied to the interpolated value for each pixel and then that pixel's value is accumulated [24]. The phase correction is calculated using the carrier frequency of the signal (F_0), the differential range of a target, and the speed of light (c). The phase correction value has the following form:

$$\Phi = e^{j4\pi F_0 R_m / c} \quad (6.2)$$

Following the phase correction, the updated pixels are added to the backprojection image. This process repeats using the same number of pulses for every vertical sample. The final image response is the summation of the interpolated pixel values for every pulse. The pixels in the resulting matrix are converted to decibel values.

6.4 SAR Imaging

Finally, the threshold for the pixel intensity values is defined to determine which pixels should be plotted in the SAR image. The peak signal return strength of the SAR image is normalized to 0 dB. The threshold range value is set to -15 dB by default, creating a range of 0 dB to -15 dB for visualizing the pixels. The threshold helps define the intensity range for the detected targets in the SAR image.

This value can be adjusted to accommodate for targets with weaker signal return strengths or further filter out surrounding noise. Using the 3-D scatter plot function, the selected pixels are plotted resulting in the final SAR image. Examples of these SAR images are showcased in Chapter 7.

Chapter 7

Volumetric SAR Imaging

In this chapter, volumetric SAR images for three different target scene setups are created and discussed. Each target scene setup also serves as an experiment to showcase the following post-processing attributes for imaging. In these images, the positions of targets are measured with respect to the starting position of the radar. Performing volumetric imaging also allows for multiple viewpoints of the target scene from the radar's perspective. Furthermore, the threshold of the final volume image is selected to pick out every pixel over the threshold and plot them in a scatter plot so that only prominent objects are displayed. Finally, the finite frequency giving sidelobes in the range dimension and the finite spatial frequency (finite motion span) giving sidelobes in azimuth and elevation are revealed.

7.1 Target Scene: Four Scattered Targets

In this section, there are four targets of interest: two corner reflectors positioned at different heights, a sphere, and a cube placed at an oriented angle. The target scene is shown in Figure 7.1. A collection of 30 datasets, 1024 pulses per vertical cut, were captured for this experiment and post-processed with a range of 1 to 3

meters in azimuth and -0.5 to 1.5 meters in elevation. This setup showcases how the shape and reflectivity of a target determine its resolution in a volumetric SAR image. A threshold value of -23 dB was the largest value that allowed for every target of interest to be visible in the SAR image. This increased threshold value allows for unwanted noise to appear in the volumetric image, such as sidelobes and other detectable objects in the target scene. The resulting volumetric SAR images, with and without the Hanning window applied, are shown in Figure 7.2. Every target of interest has pixels visible in these SAR images. The lower corner reflector yields the strongest signal return strength whereas the sphere yields the weakest. Adding the sidelobe suppression helps consolidate the pixels for the lower corner reflector into its measured position and improves the resolution of each target.



Figure 7.1: Target scene setup of four different targets.

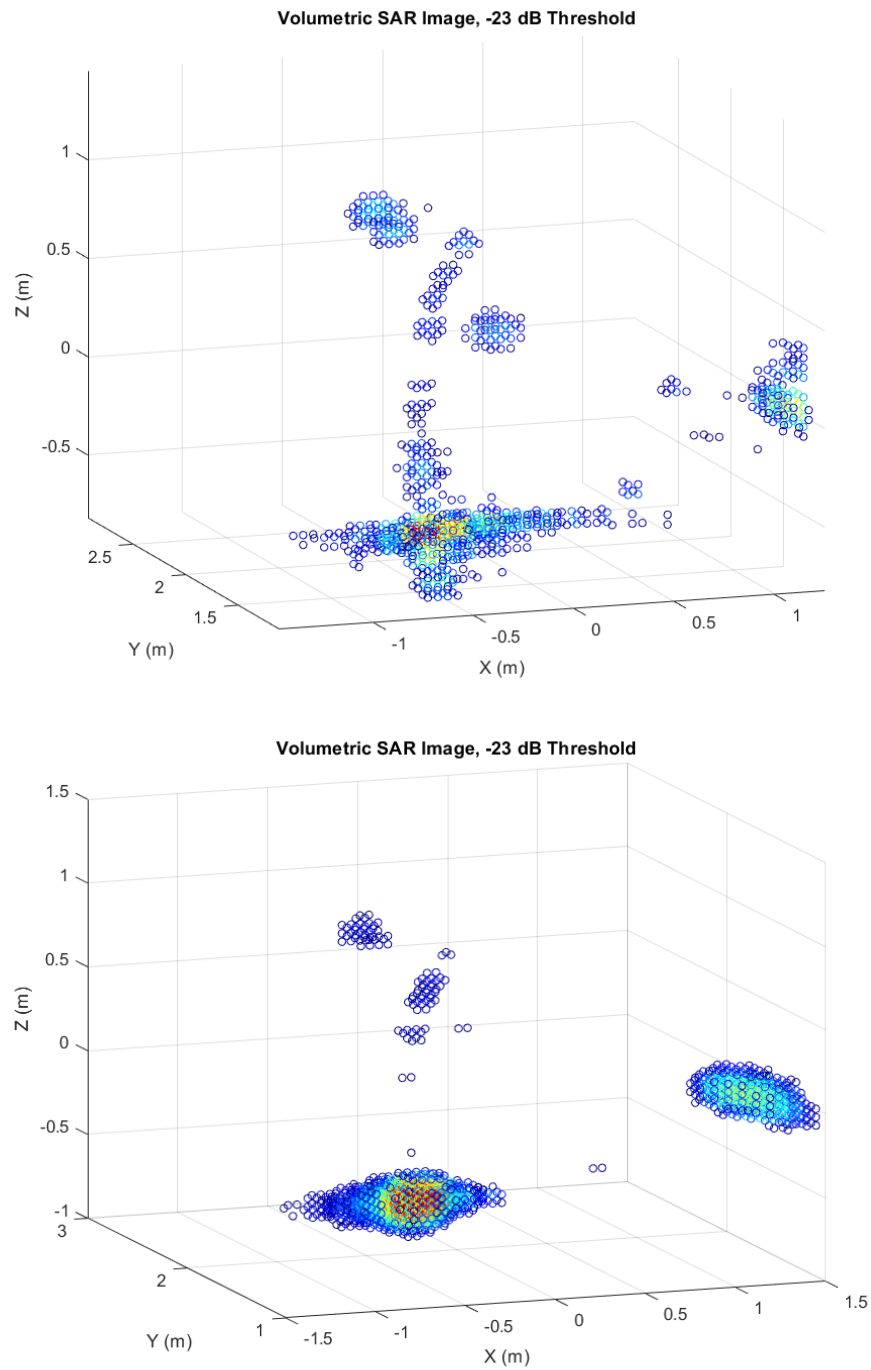


Figure 7.2: Volumetric SAR image of the target scene without sidelobe suppression (top) and with sidelobe suppression (bottom).

The SAR Demonstrator post-processing is designed to measure the radar signal return strength by normalizing the peak value in the backprojection image to 0 dB. The selected threshold creates a range of pixels from 0 dB to the threshold value to be plotted in the volumetric SAR image. This allows the SAR Demonstrator to consistently generate images with respect to the reflectivity of targets in different target scene setups. If a target yields a strong return strength, there will be a greater focus on the resolution, and unwanted noise will be eliminated. A side-by-side comparison of both SAR images and the target scene is shown in Figure 7.3.

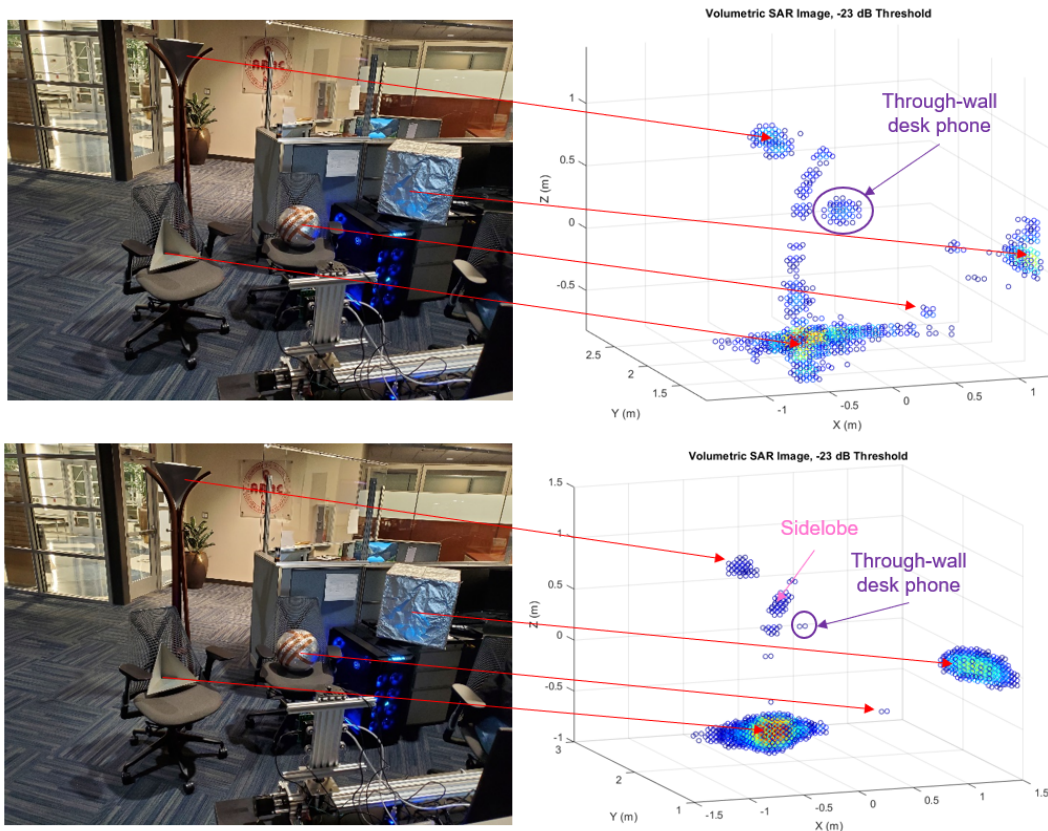


Figure 7.3: Target scene and corresponding SAR image side-by-side for comparison both without sidelobe suppression (top) and with sidelobe suppression (bottom).

The MATLAB-generated volumetric SAR image can be freely rotated, allowing the user to view the target scene at any perspective. An example of this is shown in Figure 7.4. Utilizing this feature, the positions of each target can be verified in both elevation and azimuth. Reorienting the SAR image into a two-dimensional viewpoint allows for the target scene to be viewed from the perspective of the SAR Demonstrator and a bird's eye point of view. These perspectives of the target scene are shown in Figure 7.5.

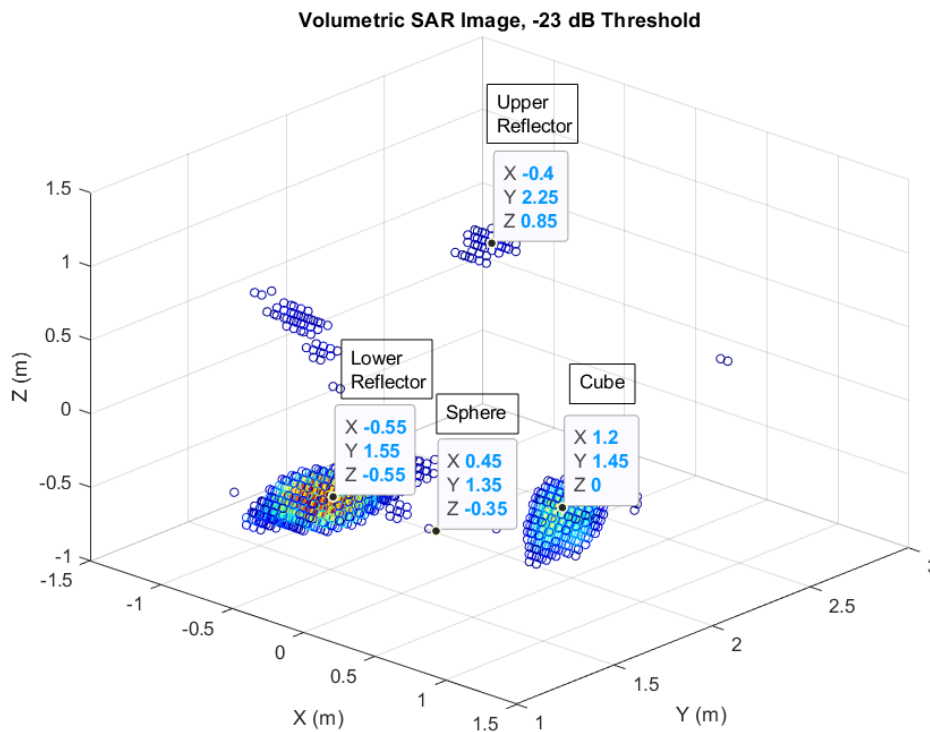


Figure 7.4: Alternate viewpoint of the target scene with each target labeled and its position measured.

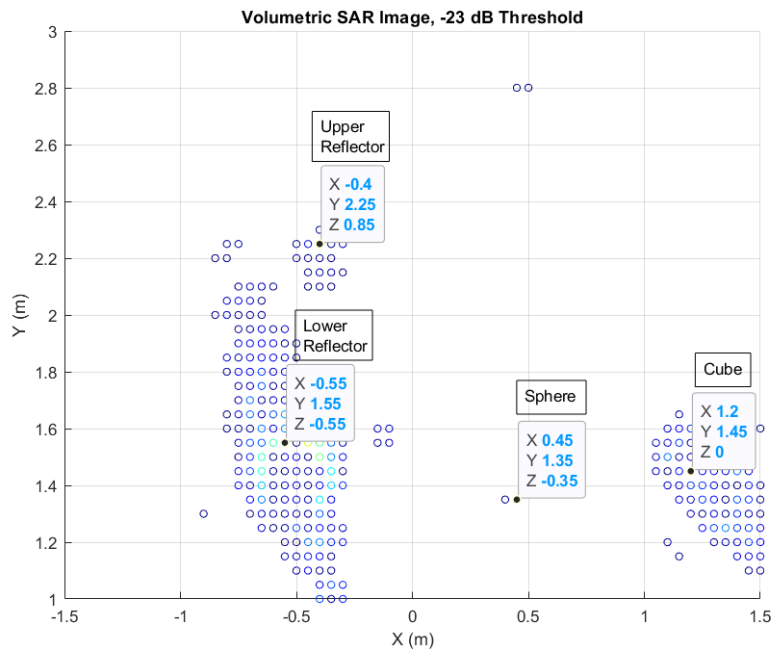
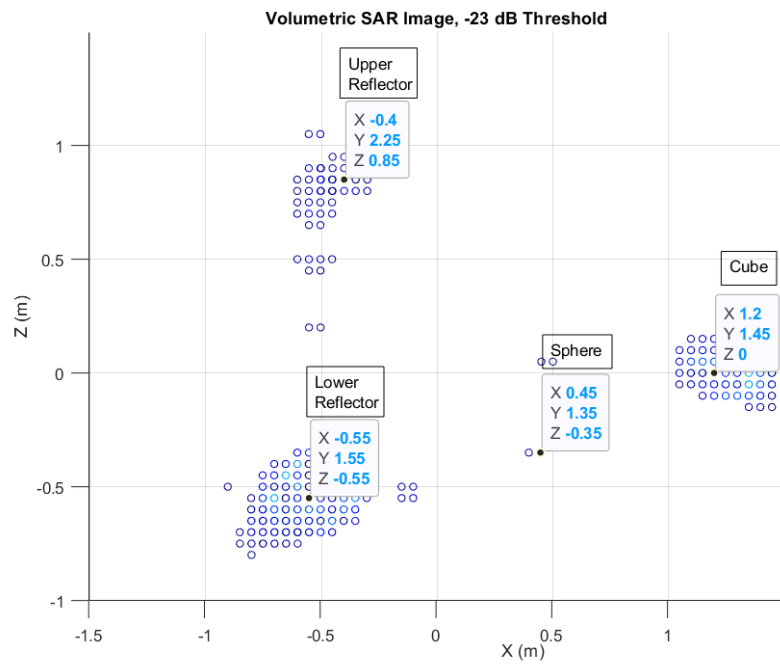


Figure 7.5: The range-elevation plane (top) and range-azimuth plane (bottom) of the SAR image.

7.2 Target Scene: A Single Corner Reflector

In this section, the target scene consists of only a single corner reflector placed directly in front of the SAR Demonstrator at approximately 2 meters. The setup for this target scene is shown in Figure 7.6. A collection of 30 datasets, 1024 pulses per vertical cut, were captured for this experiment. In post-processing, the pixel resolution was increased to better showcase how the energy reflects off the corner reflector and create a 3-D shape of the target. The finite frequency observation window causes the energy to spread out in the form of sidelobes. The threshold value was kept at -23 dB to further reveal these sidelobes to create a 3-D point spread response. The resulting volumetric SAR image is shown in Figure 7.7.

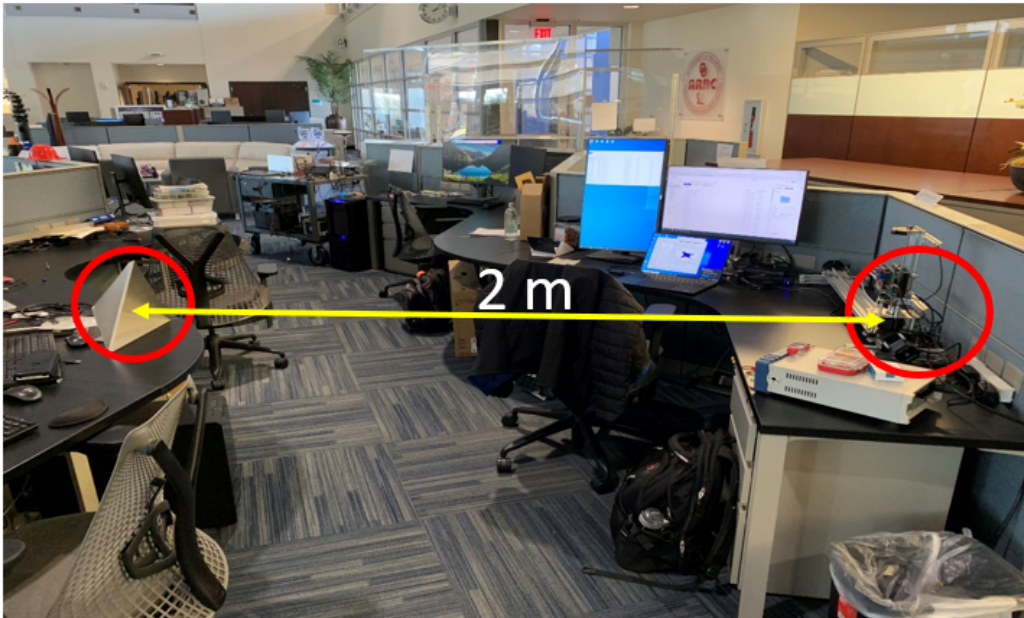


Figure 7.6: Target scene setup of a single corner reflector.

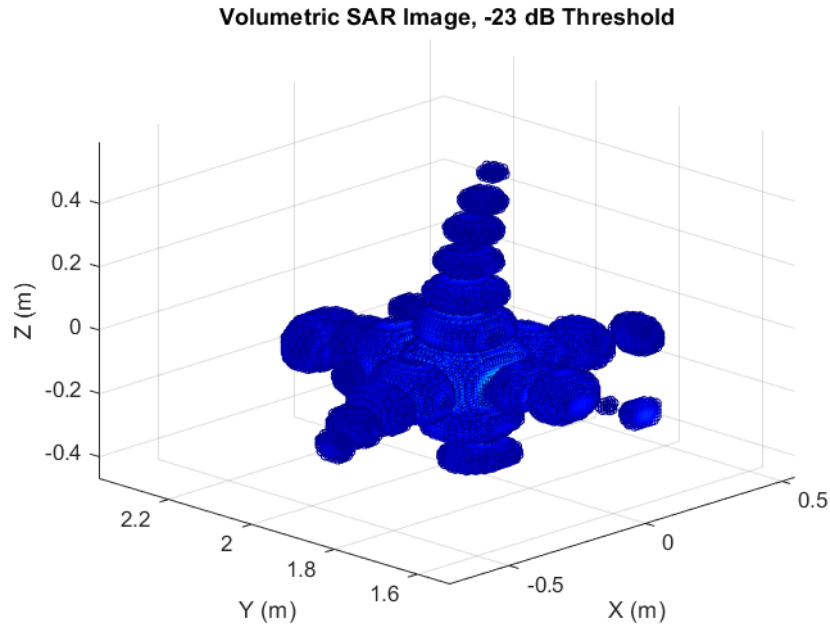


Figure 7.7: Volumetric SAR image of corner reflector with the increased pixel resolution.

7.3 Target Scene: Four Assorted Corner Reflectors

In this section, the target scene consists of four corner reflectors serving as targets. The corner reflector closest to the radar has a 3.2-inch edge length and was 3-D printed and wrapped in aluminum foil. The same applies to the second-closest target but with a 6.4-inch edge length. The last two corner reflectors are identical with 13-inch edge lengths and being fully aluminum. This experiment was conducted to compare the signal return strengths of corner reflectors differing in range, size, and material. The setup for this target scene is shown in Figure 7.8. A collection of 50 datasets, 1024 pulses per vertical cut, were captured and post-processed with a range of 0 to 6 meters in azimuth and -0.5 to 1.5 meters in elevation.



Figure 7.8: Target scene setup of four corner reflectors in different positions.

In post-processing, the pixel resolution was no longer decreased from the previous section. Two volumetric SAR images were generated, each with different threshold values. The first image keeps the threshold value at -23 dB while the second image has the threshold value reduced to -10 dB to minimize unwanted noise and focus on the resolution of each target. The resulting volumetric SAR images comparing their different threshold values are shown in Figure 7.9. Each corner reflector was positioned to ensure they each yielded similar results in signal return strength. The physical size of each corner reflector is also directly proportional to the size of their corresponding pixel reconstruction in the SAR image. Finally, using the volumetric SAR image with a -10 dB threshold, the position of every corner reflector measured in both elevation and azimuth is shown in Figure 7.10.

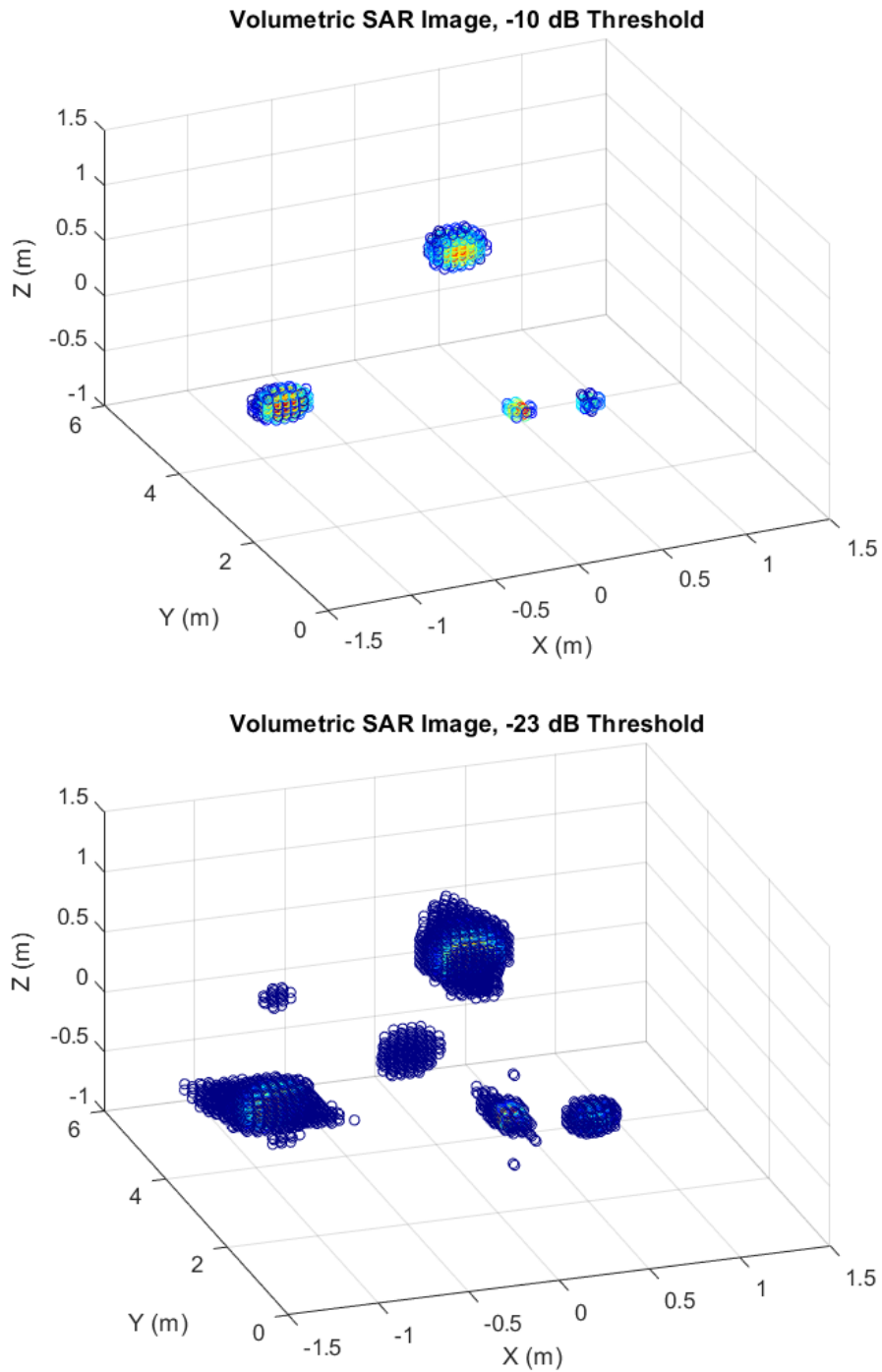


Figure 7.9: Volumetric SAR images of the corner reflector target scene with a threshold of -10 db (top) and a threshold of -23 dB (bottom).

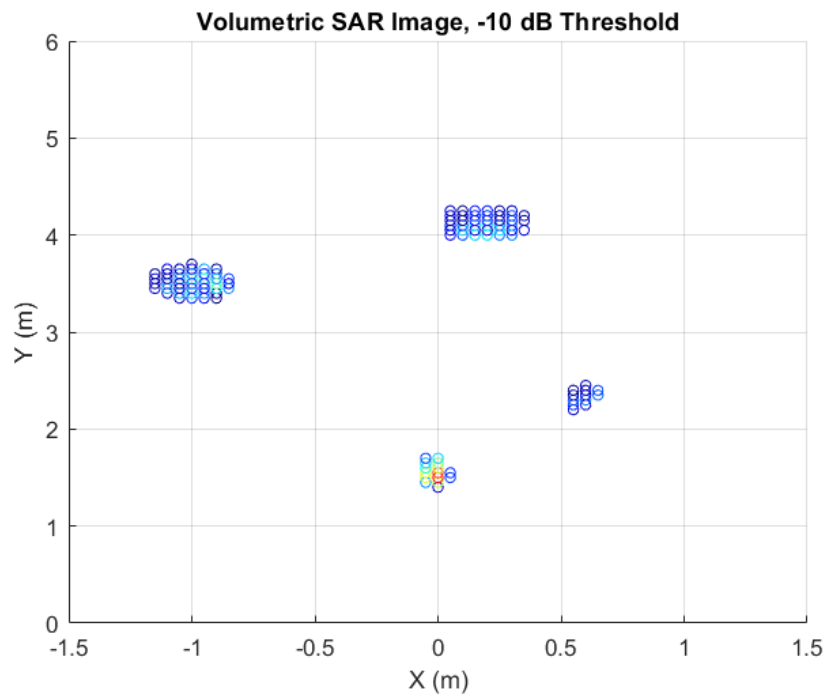
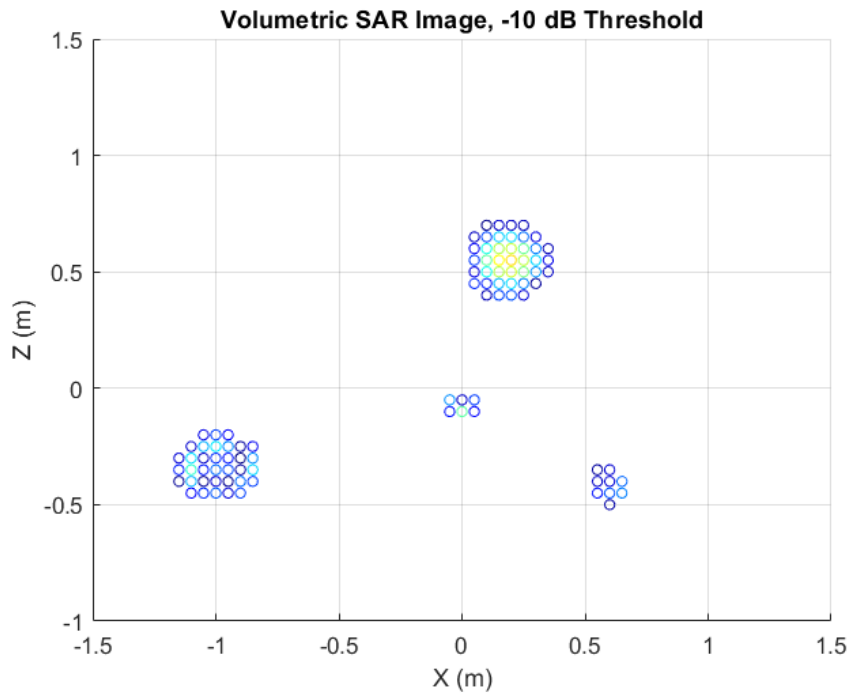


Figure 7.10: The range-elevation plane (top) and range-azimuth plane (bottom) of the SAR image.

Chapter 8

Conclusion and Future Work

This thesis showcases how a track-mounted SAR system can leverage a 77 GHz automotive radar to form high-resolution volumetric images of a near-range target scene. The SAR Demonstrator features a control system capable of autonomously capturing multiple datasets and immediately performing post-processing to generate a volumetric SAR image. The SAR Demonstrator also features a LabVIEW GUI that can fully launch, configure, and communicate with mmWave Studio, allowing for the system to interface with the radar device and linear actuators. Thus, the GUI has every desired feature that completes the system as a standalone SAR imaging demonstrator. Overall, the SAR Demonstrator proved to be an effective system for creating 3-D images of short-range target scenes.

Moving forward, the SAR imaging system will serve as a tool for demonstrating the core fundamentals of SAR at the Advanced Radar Research Center (ARRC). Beyond its design for educational purposes, this advanced system has the potential to further advance the development of cutting-edge navigation equipment for SAR image formation at high frequencies. There is a desire to develop SAR systems with low cost, weight, size, and power (C-SWaP) for deployment on lightweight

unmanned aerial vehicles (UAVs). At the ARRC, future project interests include the construction of low C-SWaP SAR systems and implementing novel, low C-SWaP navigation systems to enable high-fidelity SAR imaging onboard lightweight UAVs. Therefore, the SAR Demonstrator can serve as a system of interest for reference and even integration to realize these potential projects.

One critical component for precise SAR navigation is the inertial measurement unit (IMU) which is essential for accurate navigation within SAR systems. Current IMU technology scales in C-SWaP as performance improves, with the most accurate systems costing over \$100,000 while being too large and heavy for integration with other systems. Therefore, the ARRC has developed a particle fusion-based technique for combining multiple small and low-cost IMUs as a single device [25]. This multi-IMU surpasses the performance of standard averaging methods and can match the performance of a larger and higher-quality IMU [26].

To explore these SAR imaging capabilities, an ongoing project is to integrate the multi-IMU instrumentation with the SAR Demonstrator for data collecting to form highly focused SAR images. This is accomplished by tracking the radar's position during the data capture period using the multi-IMU system. This will allow for the performance of track-mounted SAR image captures when the SAR Demonstrator and multi-IMU are synchronized as a single system. A mounting kit to physically attach the multi-IMU system to the SAR Demonstrator has already been developed as shown in Figure 8.1. There has been early testing with synchronizing these two systems to produce a simple 2-D SAR image of a target scene but further development has since been paused. If this project were to continue, it would culminate in integrating the SAR system onboard a lightweight UAV along with using the multi-

IMU fusion technique. This would demonstrate the capability of high-frequency SAR imaging on previously prohibitively small platforms, resulting in the formation of ground-based and airborne SAR images. This integration can serve as the next step toward further exploring this area of SAR imaging research.

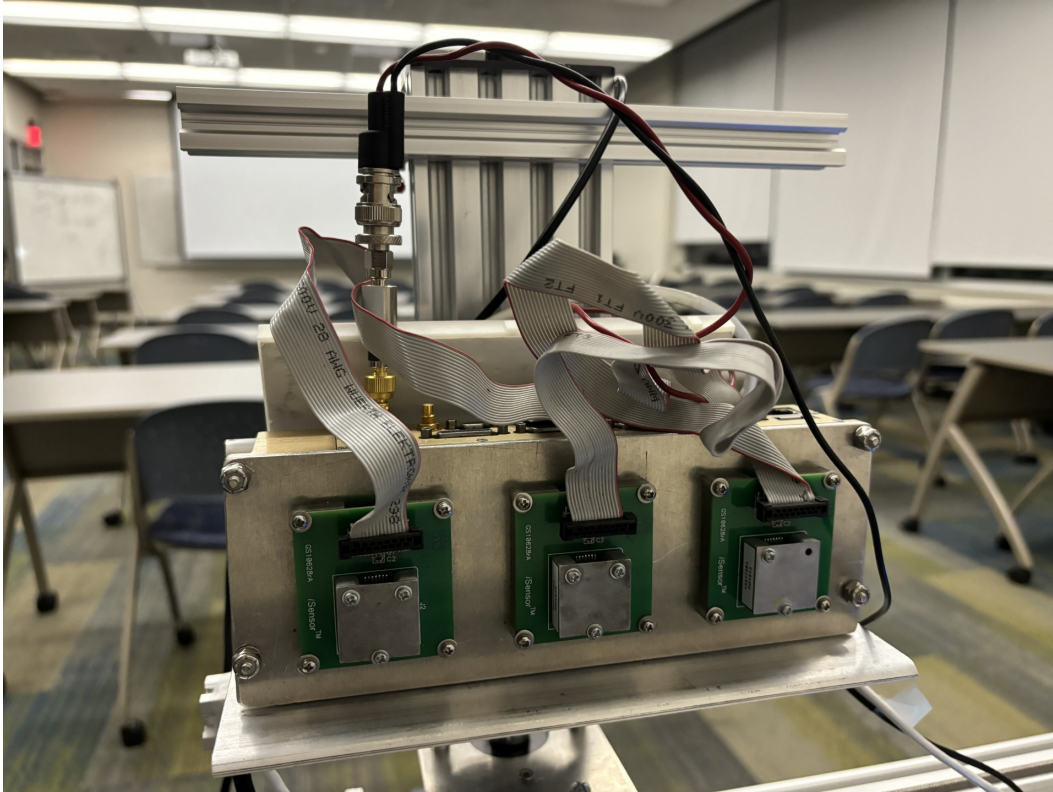


Figure 8.1: The multi-IMU device mounted onto the SAR Demonstrator.

Finally, there is also interest in investigating autofocus techniques relevant to side-looking SAR image formation when deployed on a UAV platform. This comprehensive approach aims to push the boundaries of SAR imaging technology, making high-performance SAR systems more accessible and practical for a wide range of applications.

References

- [1] M. M. Vazquez, “Radar for Automotive: Why Do We Need Radar?” *Semiconductor Engineering*, Mar. 2022. [Online]. Available: <https://semiengineering.com/radar-for-automotive-why-do-we-need-radar/>
- [2] J. J. Kovaly, *Synthetic Aperture Radar*. Artech House, 1976.
- [3] M. Richards, J. Scheer, and W. Holm, *Principles of Modern Radar: Basic Principles*, Jan. 2010.
- [4] *AWR2243 Evaluation Module (AWR2243BOOST) mmWave Sensing Solution User’s Guide*, Texas Instruments, Feb. 2021, Rev. D.
- [5] *AWR2243 Single-Chip 76- to 81-GHz FMCW Transceiver*, Texas Instruments, Feb. 2024, Rev. D.
- [6] *DCA1000EVM Data Capture Card User’s Guide*, Texas Instruments, May. 2019, Rev. A.
- [7] *mmWave Studio GUI*, Texas Instruments, 2020.
- [8] M. A. Richards, *Fundamentals of Radar Signal Processing*, M. McCabe, Ed. McGraw-Hill Education, 2005, vol. 2.
- [9] M. Walden, “Automotive Radar – From Early Developments to Self-Driving Cars,” *ARMMS RF and Microwave Society*. [Online]. Available: <https://www.armms.org/media/uploads/p05---mark-walden---automotive-radar---from-early-.pdf>
- [10] K. Ramasubramanian, K. Ramaiah, and A. Aginskiy, “Moving from legacy 24 GHz to state-of-the-art 77 GHz radar,” Texas Instruments, Tech. Rep., 2017.
- [11] Federal Communications Commission, “Radar Services in the 76-81 GHz Band,” 2017. [Online]. Available: <https://docs.fcc.gov/public/attachments/DOC-345476A1.pdf>

- [12] C. Waldschmidt, J. Hasch, and W. Menzel, "Automotive Radar-From First Efforts to Future Systems," *IEEE Journal of Microwaves*, 2021.
- [13] N. Petrović, M. Petrović, and V. Milovanović, "Radar Signal Processing Architecture for Early Detection of Automotive Obstacles," *Electronics*, vol. 12, no. 8, 2023. [Online]. Available: <https://www.mdpi.com/2079-9292/12/8/1826>
- [14] S. A. Flandermeyer, R. H. Kenney, H. Kim, K. Konyalioglu, A. Pham, and C. Schone, "An All-COTS Automotive Radar for SAR Imaging and Target Classification," 2020. [Online]. Available: <https://api.semanticscholar.org/CorpusID:230084094>
- [15] G. Ciaramitaro, P. Morri, D. Tagliaferri, M. Nicoli, U. Spagnolini, I. Russo, and D. Badini, "On the Minimum Set of Navigation Sensors to Enable High-Resolution Automotive SAR Imaging," in *2023 IEEE Radar Conference (RadarConf23)*, 2023, pp. 1–6.
- [16] M. Yanik and M. Torlak, "Millimeter-Wave Near-Field Imaging with Two-Dimensional SAR Data," Sept. 2018.
- [17] M. E. Yanik and M. Torlak, "Near-Field 2-D SAR Imaging by Millimeter-Wave Radar for Concealed Item Detection," in *2019 IEEE Radio and Wireless Symposium (RWS)*, 2019.
- [18] K. Ouchi, "Recent Trend and Advance of Synthetic Aperture Radar with Selected Topics," *Remote Sensing*, vol. 5, 2013. [Online]. Available: <https://www.mdpi.com/2072-4292/5/2/716>
- [19] J. Gao, B. Deng, Y.-l. Qin, H. Wang, and X. Li, "Fast Super-resolution 3D SAR Imaging Using an Unfolded Deep Network," 2018.
- [20] M. E. Yanik, D. Wang, and M. Torlak, "Development and Demonstration of MIMO-SAR mmWave Imaging Testbeds," 2020.
- [21] "Homing and Limits Description and Operation," GitHub: synthetos/TinyG, 2021. [Online]. Available: <https://github.com/synthetos/TinyG/wiki/Homing-and-Limits-Description-and-Operation>
- [22] *Mmwave Radar Device ADC Raw Data Capture*, Texas Instruments, Oct. 2018, Rev. B.

- [23] “Understanding FFTs and Windowing,” National Instruments, Apr. 2024. [Online]. Available: <https://download.ni.com/evaluation/pxi/Understanding%20FFTs%20and%20Windowing.pdf>
- [24] L. A. Gorham and L. J. Moore, “SAR image formation toolbox for MATLAB,” in *Algorithms for Synthetic Aperture Radar Imagery XVII*, vol. 7699, International Society for Optics and Photonics. SPIE, 2010. [Online]. Available: <https://doi.org/10.1117/12.855375>
- [25] B. M. Sun, “FUSION OF MULTIPLE INERTIAL MEASUREMENTS UNITS AND ITS APPLICATION IN REDUCED COST, SIZE, WEIGHT, AND POWER SYNTHETIC APERTURE RADARS,” Ph.D. dissertation, University of Oklahoma, 2022.
- [26] B. M. Sun, R. H. Kenney, M. B. Yeary, H. H. Sigmarsson, and J. W. McDaniel, “Reduced Navigation Error Using a Multi-Sensor Fusion Technique and Its Application in Synthetic Aperture Radar,” *IEEE Journal of Microwaves*, vol. 4, 2024.