2002-09

Performance analysis of a digital image synthesizer as a counter-measure against inverse synthetic aperture radar

Le Dantec, Fernando A.

Monterey California. Naval Postgraduate School

https://hdl.handle.net/10945/4955

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

PERFORMANCE ANALYSIS OF A DIGITAL IMAGE SYNTHESIZER AS A COUNTER-MEASURE AGAINST INVERSE SYNTHETIC APERTURE RADAR

by

Fernando A. Le Dantec

September 2002

Chairman of Committee and Supervisor:	Phillip E. Pace
Committee Members:	Douglas J. Fouts David C. Jenn

Approved for public release; distribution is unlimited.

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE | *Form Approved OMB No. 0704-0188*

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.

1. AGENCY USE ONLY ***(Leave blank)***	**2. REPORT DATE** September 2002	**3. REPORT TYPE AND DATES COVERED** Electrical Engineer's Thesis

4. TITLE AND SUBTITLE: Performance Analysis of a Digital Image Synthesizer as a Counter-Measure against Inverse Synthetic Aperture Radar

5. FUNDING NUMBERS

6. AUTHOR(S) Fernando A. Le Dantec

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Center for Joint Services Electronic Warfare
Naval Postgraduate School
Monterey, CA 93943-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Office of Naval Research
800 North Quincy St., Arlington VA 22217-5660

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT ***(maximum 200 words)***

This thesis is concerned with the development of a model to analyze a Digital Image Synthesizer (DIS) integrated circuit designed to create false target images to deceive Inverse Synthetic Aperture Radar (ISAR). The DIS is able to recreate the scattering effect of a moving target by using appropriate phase and gain modulations on an intercepted ISAR chirp signal before retransmitting it with the proper time delay. The DIS signal processing and the ISAR compression of the modulated return are modeled to examine the range-Doppler profile of a synthesized false target image. The image is representative of the image that would appear on an ISAR display. ISAR image quality is used to evaluate different DIS architectures and bit formats. Evaluation of the image quality is based on the deviation from an infinite resolution false target image. The results obtained from evaluating different DIS architectures indicate that the design is tolerant of significant quantization errors. The model is used to validate the architecture of the integrated circuit being fabricated. Finally, various different ISAR integration times and pulse repetition frequencies are used to confirm the integrity of the model.

14. SUBJECT TERMS
Digital Image Synthesizer, DIS, Inverse Synthetic Aperture Radar, ISAR, Synthetic Aperture Radar, SAR, False Target.

15. NUMBER OF PAGES
156

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UL

THIS PAGE INTENTIONALLY LEFT BLANK

PERFORMANCE ANALYSIS OF A DIGITAL IMAGE SYNTHESIZER AS A COUNTER-MEASURE AGAINST INVERSE SYNTHETIC APERTURE RADAR

Fernando A. Le Dantec
Lieutenant, Navy of Chile
B.S.S.E., United States Naval Academy, 1992
B.S.E.E., Naval Polytechnic Academy - Chile, 1995

Submitted in partial fulfillment of the
requirements for the degrees of

ELECTRICAL ENGINEER

and

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
September 2002

Author: Fernando A. Le Dantec

Approved by: Phillip E. Pace, Chairman of Committee and Supervisor

Douglas J. Fouts, Committee Member

David C. Jenn, Committee Member

John P. Powers, Chairman
Department of Electrical and Computer Engineering

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

This thesis is concerned with the development of a model to analyze a Digital Image Synthesizer (DIS) integrated circuit designed to create false target images to deceive Inverse Synthetic Aperture Radar (ISAR). The DIS is able to recreate the scattering effect of a moving target by using appropriate phase and gain modulations on an intercepted ISAR chirp signal before retransmitting it with the proper time delay. The DIS signal processing and the ISAR compression of the modulated return are modeled to examine the range-Doppler profile of a synthesized false target image. The image is representative of the image that would appear on an ISAR display. ISAR image quality is used to evaluate different DIS architectures and bit formats. Evaluation of the image quality is based on the deviation from an infinite resolution false target image. The results obtained from evaluating different DIS architectures indicate that the design is tolerant of significant quantization errors. The model is used to validate the architecture of the integrated circuit being fabricated. Finally, various different ISAR integration times and pulse repetition frequencies are used to confirm the integrity of the model.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

I would like to thank Professor Pace for his guidance, motivation and support throughout this thesis study. He has gone out of his way to help thesis students at expense of his free time. My best wishes to the success of the DIS design and future research efforts. I also appreciate the support from Professor Fouts for his guidance as academic advisor, his assistance in my BSEE equivalency approval upon my arrival to Monterey, and his extraordinary effort in reviewing and correcting this thesis document. I am also thankful to Professor Jenn for sponsoring my transfer from the Electronic Warfare Program to the Electrical Engineering Program, and assisting in reviewing this document. I am certainly proud of being able to work with such a distinguished group of individuals.

I am grateful to the Navy of Chile for giving me the privilege an opportunity to get a superb education at both the U.S. Naval Academy and the Naval Postgraduate School.

I am especially thankful to my beloved wife Ingrid, for her loving support and understanding throughout our stay in Monterey.

To almighty God.

THIS PAGE INTENTIONALLY LEFT BLANK

EXECUTIVE SUMMARY

This thesis is concerned with the development of a model to analyze the performance of a Digital Image Synthesizer (DIS) integrated circuit designed to create false target images to deceive an Inverse Synthetic Aperture Radar (ISAR). The DIS chip is being designed at the Naval Postgraduate School, Center of Joint Services Electronic Warfare. The circuit is able to recreate the scattering effect of a moving target by using appropriate phase and gain modulations on an intercepted ISAR chirp signal before retransmitting it with the proper time delays. The DIS is expected to be integrated with the Advance Multifunction RF Concept (AMRFC) test bed. The AMRFC is radio frequency system that will integrate radar, communications and electronic warfare equipment, and significantly increase the effectiveness of the shipboard electronic warfare (EW) system while reducing the number of topside antennas.

The model that is developed in this thesis is able to simulate the DIS operation by generating the required input signals and producing the chip signal processing to obtain the output-jamming signal. A second part of the model uses the DIS output to simulate the compression process that takes place in the ISAR receiver. The output of the ISAR range and azimuth compression is an image with the range-Doppler profile of the false target. This image is representative of the image that would appear on an ISAR display.

ISAR image quality is used to evaluate different DIS architectures and bit formats. Evaluation of the image quality is based on the deviation from an infinite resolution false target image. The results obtained from evaluating different DIS architectures indicate that the design is tolerant of significant quantization errors. The model is used to validate the architecture of the integrated circuit being fabricated.

Additional studies are performed to find the relation between the dynamic range of the DIS output and the image contrast at the ISAR display. Finally, various different ISAR integration times and pulse repetition frequencies are used in the model in order to study the image effects of different Doppler resolutions. These studies provide helpful results for the design effort, since they identify critical parameters and confirm the integrity of the model.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

A. DIGITAL IMAGE SYNTHESIZER OPERATIONAL CONCEPT

The Digital Image Synthesizer (DIS) is an application specific integrated circuit (ASIC) able to generate false target images to deceive an Inverse Synthetic Aperture Radar (ISAR). ISAR is a high-resolution radar technique capable of developing a two-dimensional image of a target based on the relative Doppler of the reflecting components. The DIS is able to generate a set of scattering effects that emulate the presence of multiple false targets, thus attempting to deceive the ISAR target recognition process.

The DIS design takes advantage of readily available digital radio frequency memory (DRFM) technology to sample, store and replicate an intercepted ISAR signal. The DRFM is used to feed the DIS circuit with the intercepted ISAR signal in the form of digitized samples of the chirp pulses. The DIS circuit modulates the signal samples in amplitude and phase in order to recreate the scattering effect of a desired false target. The DIS output (modulated) signal is sent through a digital-to-analog converter to a radio frequency (RF) module for frequency up-conversion and retransmission. Note that a DRFM alone cannot perform the complex modulations required to synthesize large false targets.

The ISAR platform collects the returned modulated pulses and coherently processes them to form an image. The RF receiver first down-converts the signal frequency, samples the signal, and performs the pulse compression digitally. The signal is then sampled in both in-phase (I) and quadrature phase (Q), and sent to the ISAR signal processor. After range compression, the samples within each of the pulses are integrated coherently and resolved in Doppler (azimuth compression process) to display a two-dimensional target image. A block diagram of this process is presented in Figure 1.

Technical developments in naval warfare have improved the capabilities of anti-ship cruise missile (ASCM) strike systems, which are still considered the foremost threat to naval vessels. The range of these missiles has been significantly increased and capabilities such as supersonic speeds, high maneuverability, data-link control, GPS aided guidance, re-attacking options and active/passive seekers have been incorporated.

Launching platforms with their increased missile ranges have forced an increase in their sensor capabilities in order to properly evaluate potential hostile targets at greater distances. This problem has also resulted in maritime patrol and strike aircrafts, such as the U.S. Navy P-3C and S-3B, being fitted with ISAR in order to identify surface targets based on their profile before assigning a strike mission.

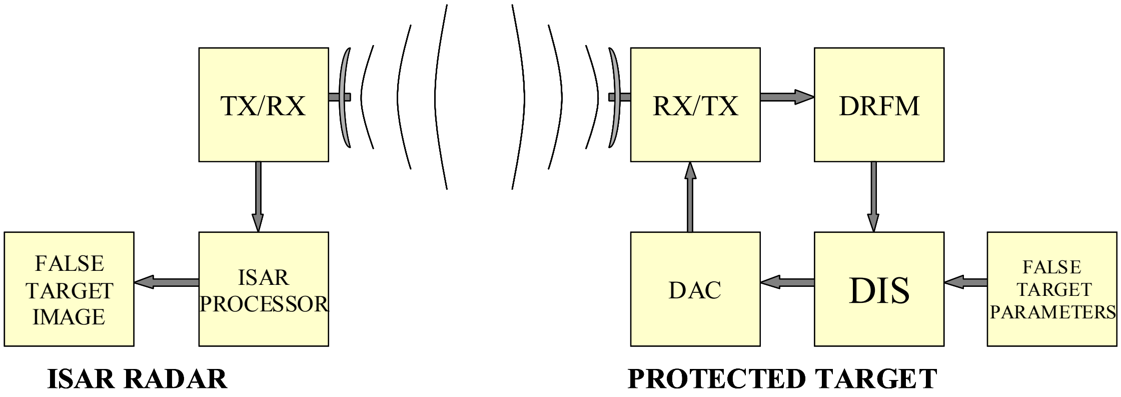


Figure 1 Block Diagram of the DIS Operational Concept.

The ASCM strike can be divided into six phases: surveillance, identification, targeting, missile launch, guidance and the terminal homing phase. Naval vessels attempt to counter this threat at all phases. Numerous tactics have been developed to counter each phase, but are not within the scope of this thesis. However, it must be mentioned that the DIS concept can be used for a number of situations including counter-identification, counter-targeting and counter-terminal (ISAR seekers).

A common counter-identification scenario to illustrate the DIS operational concept would be a destroyer being detected by a maritime patrol aircraft (MPA) at a range beyond 100 nautical miles. The aircraft knows the ship's position and has tracked its course and speed. The aircraft's next step is to attempt to evaluate the unknown target. Electronic support (ES) could provide radar interceptions suitable for identification, but the ship, once aware of the presence of an MPA, will have restricted its emissions. The range is too great for infrared (IR) imaging and is aggravated by drizzle that severely affects IR propagation. The MPA decides to switch the radar to ISAR mode in order to identify the silhouette of the ship's hull. To obtain the best image, the MPA positions itself 45 degrees off the ship's bow. The ship intercepts the ISAR chirp signal and activates the DIS countermeasure. Based on the aircraft's relative location obtained from the ES, the DIS is then able to generate a ship silhouette similar to a merchant vessel that was supposed to have transited the area that same day. In addition, the ship activates a commercial navigation radar to make the presence of a merchant ship more credible. The ISAR display in the MPA shows the silhouette of a merchant ship, in agreement with what was expected to transit that area, which is also supported by the emission of a commercial radar. The MPA classifies the target as neutral and leaves the scene.

A second scenario for employing a DIS is on an aircraft countering Non-Cooperative Target Recognition (NCTR) techniques based on scattering characteristics. Radar NCTR is the classification or identification of a type of aircraft using radar based techniques, including range profile, 2D-scattering centers, time-frequency analysis, natural frequencies and feature extraction. A properly programmed DIS mounted on a threat aircraft would be able to replicate the scattering characteristics of a friendly aircraft.

Further, DIS applications can also include the generation of false land targets to protect an area against a reconnaissance aircraft mapping with a Synthetic Aperture Radar (SAR) or any defensive effort against platforms using high resolution Doppler processing radars.

The DIS is currently being designed at the Center for Joint Services Electronic Warfare at the Naval Postgraduate School and is expected to be integrated with the Advance Multifunction RF Concept (AMRFC) test bed. The AMRFC is an RF system that will integrate radar, communications and electronic warfare equipment, and will significantly increase the effectiveness of the shipboard electronic warfare (EW) system while reducing the number of topside antennas.

B. PRINCIPAL CONTRIBUTIONS

The objective of this thesis is to study the DIS design and develop an updated version of the model. The principal contributions in this thesis can be identified as follows:

- Decrease the DIS simulation processing time by increasing the efficiency of the mathematical models and reducing their complexity

- Incorporate a model to simulate, measure and manage the overflow effect within the DIS output adders
- Develop a model to simulate a DIS with infinite bit resolution
- Create a model to compare and correlate the ISAR image with the infinite resolution image
- Change the DIS and ISAR models to enable the evaluation of chip processing without the preceding constant phase sequence that is currently used
- Evaluate the impact of the DIS architecture and bit formats on the output image quality
- Evaluate the impact of the different ISAR parameters and sampling criteria on the output image quality
- Incorporate the capability to model a 512-range-bin target
- Produce input and output vectors to support the very high-speed integrated circuit hardware description language (VHDL) simulations. (VHDL is the hardware description language used for documentation, verification and synthesis of large digital designs currently used in the DIS project.)
- Create a common software version usable for different types of targets and radar signal parameters

In summary, this thesis provides the DIS design team with an efficient modeling tool to support the DIS chip design efforts, by being able to accurately predict the performance of various DIS architecture configurations.

C. THESIS OUTLINE

This thesis first introduces the concepts of the ISAR and DIS. A description of the DIS software model and the performance studies obtained with its simulation follows. Chapter II presents the SAR concept and its angular resolution enhancement. Next, ISAR is introduced with a description of its resolution, the output target image and the basics of its compression signal processing. This analysis provides a background for understanding the math models used to simulate ISAR reception and image generation. Chapter III mathematically describes the DIS process from the generation of the target modulation parameters to the complex pulse modulation. It also includes a mathematical description of the ISAR range and azimuth compression processes used in the model to generate the expected target image. Chapter IV describes the DIS model developed in MATLAB. The highlights of the software are explained using selected code lines, plots with signal characteristics and tables with values obtained from simulation runs. Finally, this chapter describes the modifications required to obtain an ideal model with infinite bit resolution. Chapter V studies the performance of the DIS design using the simulation model presented in Chapter IV. The image obtained from the current DIS design is compared to an infinite bit resolution image to quantify quantization and resolution errors. The effects from individual stages of the DIS process are isolated to numerically evaluate their contribution to the image quality. Chapter VI examines the effectiveness of the DIS against an ISAR using different pulse repetition frequencies (PRFs) and integration times. The relation between ISAR signal parameters and image Doppler resolution is investigated in theory and demonstrated with simulation outputs. Chapter VII studies the dynamic range of the DIS signal and its effect on ISAR target image contrast. Theoretical dynamic range values are developed and compared with results obtained from simulations. Chapter VIII presents conclusions to this thesis study and recommendations for future efforts in designing and testing the DIS.

This thesis also includes three appendices to support the information presented. Appendix A contains the MATLAB m-files that comprise the DIS simulation model. Appendix B contains a brief description of the Digital RF Memory (DRFM), which is an integral part of the DIS concept. Finally, Appendix C contains a brief description of the AN/APS-137, an ISAR radar operated by the U.S. Navy onboard the P3-C and the S-3B aircrafts.

THIS PAGE INTENTIONALLY LEFT BLANK

II. SAR/ISAR FUNDAMENTALS

ISAR is a high-resolution radar usually operated from airborne platforms. The high resolution obtained by this radar in both range and azimuth makes it possible to obtain the image of a moving target in two dimensions. Among the most popular operational applications of ISAR is that of a maritime patrol aircraft tracking a surface ship. In this application, ISAR exploits the ship's movement to obtain a two-dimensional image of its length and height. This broadside aspect is very suitable for target recognition, but as will be demonstrated in later chapters, the use of electronic attack (EA) makes it vulnerable to deception. This chapter describes the basic principles of SAR and, more specifically, those of ISAR. This background information helps the reader understand the fundamentals of the DIS EA in the chapters to follow.

A. RADAR AZIMUTH RESOLUTION

For a radar antenna, the angular resolution is approximately equivalent to the angular width of the antenna beam. Using the definition of arc-length, the radar azimuth resolution can be approximated as:

$$d_{a\ conv} = R\,\theta_{3dB} \tag{2.1}$$

where R is the range to the target and θ_{3dB} is the half-power beamwidth in radians. A uniformly illuminated sin(x)/x linear aperture antenna has a half power beamwidth given by:

$$\theta_{3dB} = 0.88\ \lambda / D \tag{2.2}$$

where λ is the wavelength of the carrier frequency and D is the length of the physical antenna [1]. The beamwidth expression in (2.2) can be simplified using a 4dB width equal to λ / D. The resulting azimuth resolution for the conventional antenna becomes:

$$d_{a\ conv} = R\,\lambda / D. \tag{2.3}$$

From Equation (2.3), it can be concluded that an extremely narrow beam is required to obtain a fine azimuth resolution at a long range. Such a beam can be obtained either with a big antenna or a very small wavelength. Big antennas are impractical when operated from an aircraft because of size, weight and aerodynamic considerations. On the other hand, a radar signal with a very small wavelength is severely affected by the strong atmospheric attenuation occurring at very high frequencies.

B. SAR CONCEPT

SAR takes advantage of the motion of an aircraft to simulate a large antenna in order to obtain a fine azimuth resolution. An aircraft flying a straight path with a coherent side looking pulsed radar is used to create an extremely large synthetic aperture. The equivalent aperture size of this antenna is the flight path length *L*.

SAR pulses are transmitted at equally spaced locations along the flight path. The complex samples for each pulse of the returning signal, amplitude and phase, are stored in memory and added coherently. This summation results in a similar effect as a physical linear array antenna of aperture length *L*.

The main difference between the synthetic aperture and a physical array antenna, is that the SAR beam and, thus its resolution is finer by a factor of two [2]. This distinction occurs because physical array antennas resolve angle selectivity using the phase difference at the reception process (one way trip). Conversely, SAR resolves azimuth using phase differences resulting from round trip distance differences between aperture elements. By inserting a factor of two and using the aircraft path length *L* as the aperture size in Equation (2.2), the SAR effective beamwidth becomes:

$$\theta_{eff\,3dB} = 0.44\ \lambda / L. \tag{2.4}$$

From Equation (2.1) and the SAR effective beamwidth in (2.4), it follows that the equivalent azimuth resolution can be expressed as:

$$d_{a\ SAR} = 0.44\ R\,\lambda / L. \tag{2.5}$$

To simplify the expression in (2.5), a 4-dB SAR effective beamwidth of λ / D can be used, yielding an azimuth resolution of:

$$d_{a\ SAR} = R\,\lambda / (2L). \tag{2.6}$$

In summary, a side looking SAR, using a physically wide beam, is able to synthesize a narrow beam, whose width and azimuth resolution depends on the range *R* and the aircraft path length *L*. A simple analysis may state that maximizing the distance of path length by extending the integration time would minimize the resolution. However, there is a point at which a further increase in the synthetic aperture length degrades resolution.

1. Unfocused SAR

Assuming that the SAR aircraft path follows a straight line, the range from each point in the path to a distant target differs. As the aircraft path length is increased, the range difference changes quadratically. Due to the fairly short wavelength of the signal, this range difference results in considerable phase rotation. As the path length is increased on an unfocused SAR, there is a limit at which the returns received at the end of the aperture become subtractive, thus degrading the overall performance.

SAR applications compensate this phase degradation by focusing the signals to a specific range. An unfocused SAR does not perform this correction and integrates the return signals within limits where degradation is negligible. Stimson [2] calculates the maximum synthetic aperture length for the unfocused case as:

$$L_{eff} = 1.2\sqrt{\lambda R}. \tag{2.7}$$

By inserting (2.7) in (2.6), the azimuth resolution of the unfocused SAR can be represented as:

$$d_{a\ SAR} = 0.4\sqrt{\lambda R}. \tag{2.8}$$

Equation (2.8) shows that for the unfocused SAR the cross-range resolution is range dependent.

2. Focused SAR

The limitations on the SAR aperture size presented in (2.7) can be eliminated by applying an appropriate phase correction for the return pulse samples according to the pulse location in the synthetic aperture. This phase correction at each array element is given by [2]:

$$\chi = -\left(\frac{2\pi}{\lambda R}\right) r_n^2 \tag{2.9}$$

where r_n^2 is the distance of array element n from the array center. The use of these phase corrections allows a focused SAR to integrate samples through a longer synthetic aperture length L. The integration time can be selected so that the synthetic aperture length L matches the azimuth resolution for a conventional antenna depicted in Equation (2.3), such that:

$$L = R\,\lambda / D. \tag{2.10}$$

The resulting synthetic aperture length of Equation (2.10) can in turn be used in (2.6), yielding a focused SAR azimuth resolution of:

$$d_{a\ SAR} = \frac{R\,\lambda}{2\left[{(R\,\lambda)}\big/{D}\right]} = \frac{D}{2} \tag{2.11}$$

where D is the aperture size of the physical antenna.

The azimuth resolution achievable with a focused SAR seems contradictory in the case of conventional antennas. Fine resolution in focused SAR requires a small physical aperture (large beamwidth) rather than a large one. Furthermore, the resolution for the focused case is independent of the range and signal wavelength.

3. SAR Modes

Several operating modes can be identified depending on the type of target and the orientation of the side looking antenna. They all require different signal processing techniques for compensation, integration and azimuth compression. These modes are strip-map, squint mode, spotlight and inverse SAR.

Strip-map SAR (Figure 2a) is designed to map the ground within a strip perpendicular to the aircraft velocity vector. The antenna is fixed to one side of the aircraft at a right angle and squinted down in elevation. The aircraft flies over a ground track parallel to the strip and sweeps a broad swath. The return signal is processed to select the samples corresponding to the range where the strip of interest is located. Both negative and positive Doppler ground patches are processed.

Squint mode SAR (Figure 2b) trains the antenna to a fixed position forward or backwards from its strip-map right angle position. Here the ground patches have either positive or negative Doppler. By training the antenna forward, SAR does not need to wait for the leading edge of the map to pass by a full array length to obtain a map. A real–time mapping can thus be made over territory not yet flown over. A drawback of this mode is that the azimuth resolution is degraded, such that:

$$d_a = \frac{R \cdot \lambda}{2 \cdot L \cdot \cos(\varphi)} \tag{2.12}$$

where φ is the azimuth angle of the antenna measured from the broadside normal [2].

Spotlight SAR (Figure 2c) trains the antenna in azimuth and elevation in order to continuously point to a specific ground patch where a target of interest is located. Due to the constant steering of the antenna, its beamwidth does not limit synthetic aperture length [3]. Therefore, this mode offers a finer azimuth resolution than the Strip-map and Squint modes.

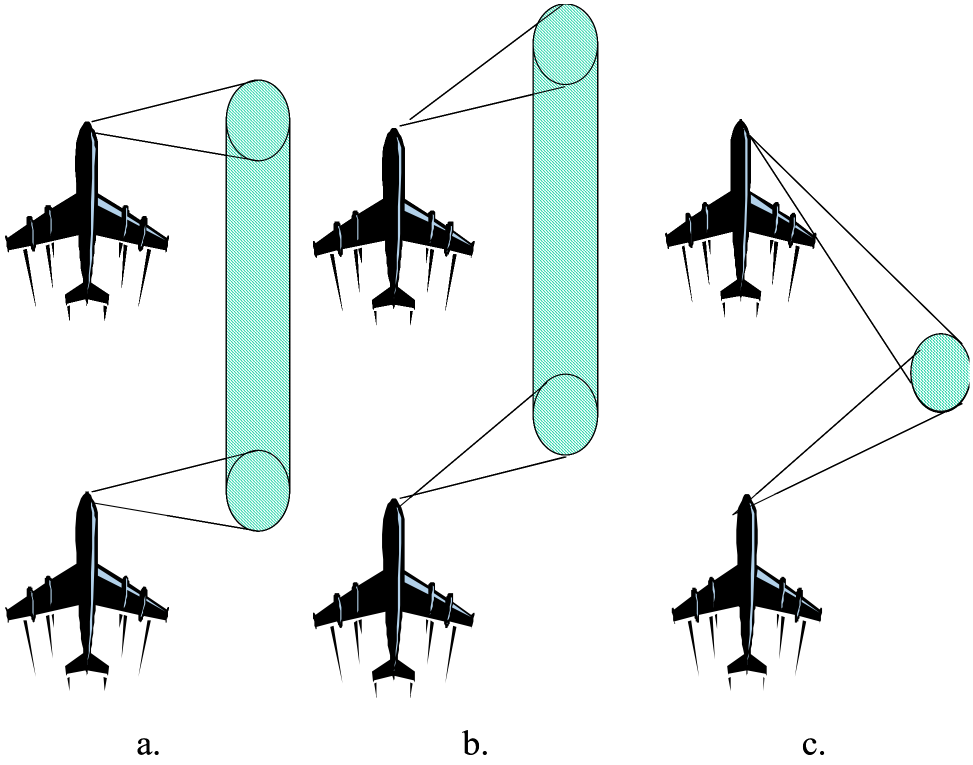


Figure 2 SAR Modes: a) Strip-Map, b) Squint and c) Spotlight (After:[3]).

Inverse SAR (ISAR) is designed to obtain the image of a moving target. In contrast to SAR, the ISAR synthetic aperture is created using the rotational motion of the target instead of using the motion of the radar. Simply put, ISAR can be understood as a spotlight SAR mode, where the target moves and the radar is stationary as shown in

Figure 3. The end result of ISAR is a two-dimensional image in range and Doppler suitable for target recognition.

The ISAR technique has probably been best applied to ship target recognition by airborne radars, where the pitch, roll and yaw movements provide an excellent way to recognize hull shapes, masts and other superstructures. Other applications range from recognition of moving ground vehicles to the imaging of satellites from ground stations.

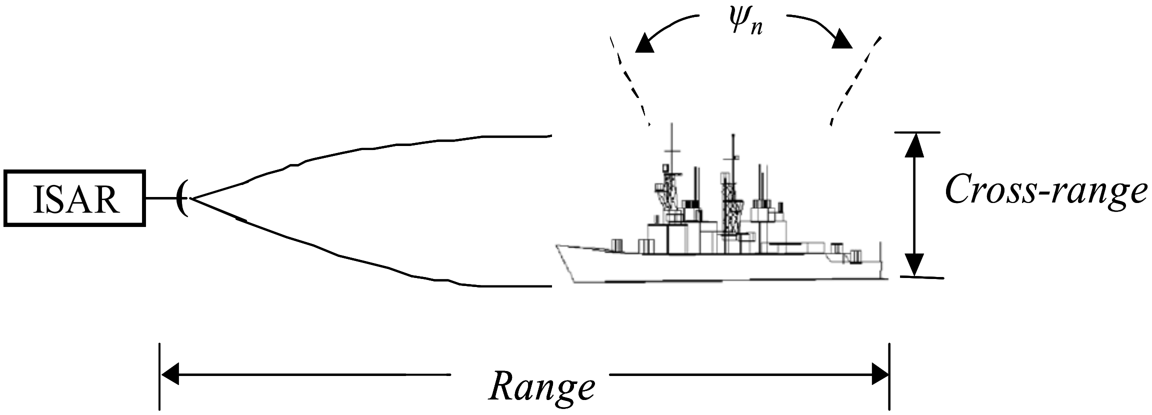


Figure 3 ISAR Concept for Two-Dimensional Imaging Based on Target Rotation.

C. ISAR PRINCIPLES

ISAR target imagery is obtained from the signal processing of the return pulses collected while the target is rotating. The Doppler shift produced by the radial rotational components of the moving target is extracted through the cancellation of the contributions from the velocities of both the radar platform and the target. In addition, ISAR relies on a tracking system to keep the target inside the antenna beam during the data collection period.

The main data processing differences between SAR and ISAR are in the azimuth dimension. In the range dimension, ISAR pulse compression of the received chirp is similar to that of SAR. In the azimuth dimension, however, SAR resolution is a function of the physical antenna size as shown in (2.11). In ISAR, it is a function of integration time (dwell time) [3].

ISAR systems often use a target-based processing technique known as prominent point processing, in which a single large scatter from the target is tracked and used as a reference signal to cancel phase errors common to all targets [4]. The calculation of phase corrections relies on external data from the radar platform, such as flight plan, on-board INS or a ground tracker.

The compensated signals are resolved in range (time of arrival) and in cross-range by their Doppler component using a bank of Doppler filters. In order to achieve a high resolution in both range and cross-range (Doppler), a short-duration pulse with a wide-band characteristic is needed. In other words, pulse compression techniques are required. The chirp and the stepped frequency pulse waveforms satisfy these characteristics.

1. ISAR Resolution

The slant *range resolution* for a chirp waveform is given by:

$$d_r = \frac{c}{2\Delta} \tag{2.13}$$

where c is the velocity of propagation and Δ is the bandwidth of the chirp pulse. Actual values for this resolution are close to one meter. Appendix C includes a brief description of the APS-137 radar with some of its resolution parameters.

If a target rotates at a constant angular rate ω, about an axis perpendicular to the line of sight (LOS), a scatterer at a distance h from the axis has a tangential velocity $\omega \cdot h$. The instantaneous Doppler frequency shift produced by the rotational component of that scatterer is given by:

$$f = \frac{2}{c}\omega h f_C = \frac{2\omega h}{\lambda} \tag{2.14}$$

where f_C is the (center) carrier frequency of the signal. Solving for h in (2.14) the azimuth or cross-range resolution can be expressed as:

$$d_{a\ ISAR} = \Delta h = \frac{\lambda}{2\omega}\Delta f \tag{2.15}$$

where Δf is width of the Doppler filters or Doppler resolution, which in turn is related to the coherent integration time T_I by:

$$\Delta f = \frac{1}{T_I}. \tag{2.16}$$

Therefore, the cross-range resolution can now be expressed in terms of the viewing angle ψ (aspect angle of Figure 3) as:

$$d_{a\ ISAR} = \frac{\lambda}{2\,\omega\,T_I} = \frac{\lambda}{2\,\psi}. \tag{2.17}$$

In conclusion, ISAR range resolution, as with most classical radars, improves by increasing the pulse bandwidth. Conversely, ISAR azimuth resolution improves when the target has a fast angular rotation ω or when the integration time T_I is increased.

2. ISAR Range-Doppler Image

The ISAR image is the result of a collection of echo signals during a small time interval while the target rotates with a small aspect angle. Each scatter returned from the target has a reflectivity value that is plotted on a range versus Doppler frequency rectangular plot. The range of the image lies along the radar LOS and is often referred to as slant range. The cross-range dimension of the image lies in a plane perpendicular to the radar LOS and can be referred to as the Doppler dimension.

Due to the direct proportionality between the cross-range location of a scatterer and its radial Doppler, a Range-Doppler map is equivalent to a two-dimensional image of the target. Although the image cross-range dimension does not represent true distance, its proportionality makes it possible to use it for target recognition purposes.

Figure 4 shows a picture of a ship and its equivalent ISAR image although not taken at the same time. Even though the website publicizing the image does not clarify how it was obtained, it can be estimated that the ISAR aircraft was directed near the bow of the ship. The horizontal distribution of scatterers in the image is obtained from their time delay (range imaging), while the vertical distribution is obtained from the Doppler caused by the pitch movement of the ship in the direction of the aircraft. Thus, the masts of the ship are expected to generate a high Doppler due to their longer distance to the ship's rotation center.

D. RANGE IMAGING

1. Pulse Compression

High range resolution on conventional radars can be achieved using a very short duration pulse. However, the bandwidth of a short pulse is large, which increases the complexity of the radar. Among the various limitations to implementing such a pulse, Skolnik [4] mentions greater demands on the signal processing, higher probability of mutual interference with other radars, less accurate Doppler measurements and energy limitations to achieve long ranges, requiring higher peak power.

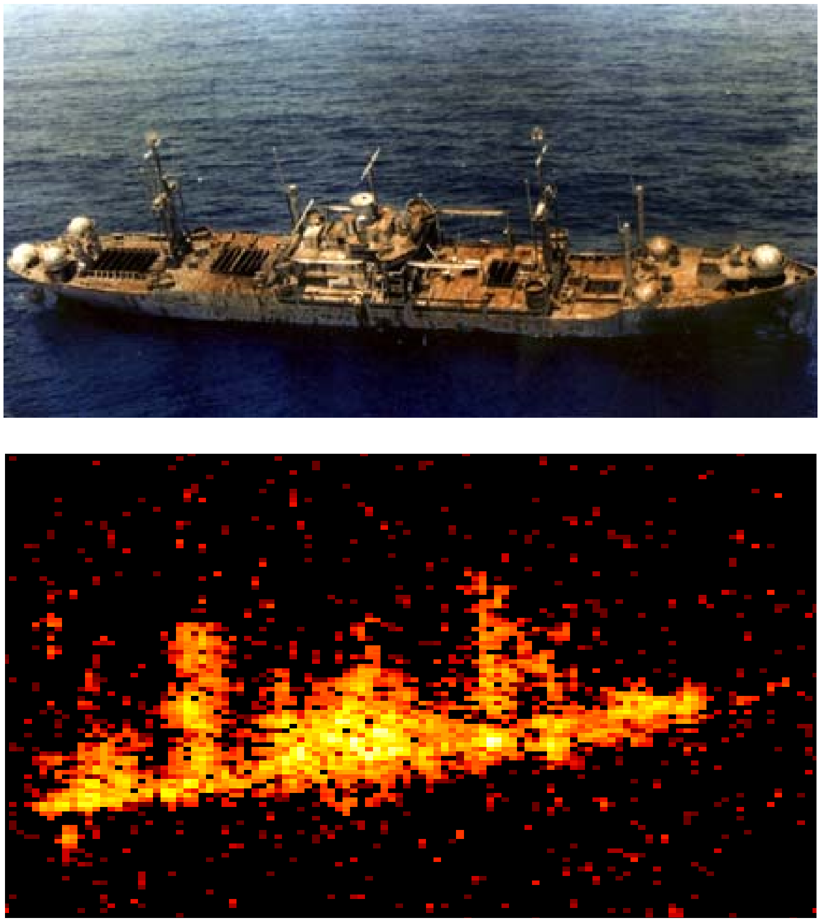


Figure 4 a) Photograph of the USS Crocket. b) ISAR Image of the Same Ship (From: http://radar-www.nrl.navy.mil/Areas/ISAR/)

The benefits of the long duration and the wide band spectrum for a pulse can be combined using *pulse compression*. This process uses a long pulse of width T and applies a frequency or phase modulation to widen the spectrum to a bandwidth Δ. The returning pulse at the receiver is compressed to a shorter pulsewidth $\tau \approx 1/\Delta$, thus improving range resolution by a factor equal to the pulse compression ratio of T/τ.

Various pulse compression techniques have been developed for frequency and phase modulation. Polarization modulation techniques have been employed on an experimental basis [5]. Frequency modulation techniques include linear frequency modulation (LFM or chirp) and stepped frequency waveforms. Phase modulation techniques include binary phase codes, such as Barker and pseudo random codes, and poly-phase codes, such as Frank, P1, P3 and P4 techniques [5].

The DIS countermeasure was designed to deceive an ISAR using LFM pulse compression [6]. The LFM chirp is produced by sweeping the carrier frequency over the length of the pulse *T* by a frequency amount of Δ. Its waveform is described in detail in Chapter III.

2. Range Imaging Model

Considering the one-dimensional scenario of Figure 5 where an imaging radar illuminates a one-dimensional target, the range and reflectivity of its N_r target scatterers can be represented by the range profile function:

$$C_r\left(x\right) = \sum_{m=1}^{N_r} \sigma_m \, \delta\left(x - x_m\right) \tag{2.18}$$

where σ_m is the radar cross-section of the m^{th} scatterer located at the range x_m. Assuming that a chirp signal *s(t)* is used as the transmitted signal in a lossless media, the expected received signal after sampling has the form:

$$I(mT_s) = \sum_{m=1}^{N_r} \sigma_m \, s\left(mT_s - \frac{2\ x_m}{c}\right) \tag{2.19}$$

where T_s is the analog-to-digital converter (ADC) sampling period and $2x_m/c$ is the round trip time delay. Using the ideal target function in (2.18), the received signal in (2.19) can be written as:

$$I(mT_s) = C_r(x) * s(mT_s) = C_r\left(\frac{c\ m\ T_s}{2}\right) * s(mT_s) \tag{2.20}$$

where the * denotes convolution. The target profile function C_r is also linearly transformed from the range domain to the discrete time domain. From the convolution theorem, Equation (2.20) can be expressed as the product of discrete Fourier transforms (DFT) so that the range profile function can be found as:

$$C_r\left(\frac{c\ m\ T_s}{2}\right) = \mathbb{F}^{-1}\left\{\frac{\mathbb{F}\{I(mT_s)\}}{\mathbb{F}\{s(mT_s)\}}\right\} = \mathbb{F}^{-1}\left\{\frac{I(k)}{S(k)}\right\} \tag{2.21}$$

where $\mathbb{F}$ and $\mathbb{F}^{-1}$ denote respectively a DFT and an inverse DFT, normally implemented using fast Fourier transforms (FFT). The *k* variable is an index for frequency resolution cells.

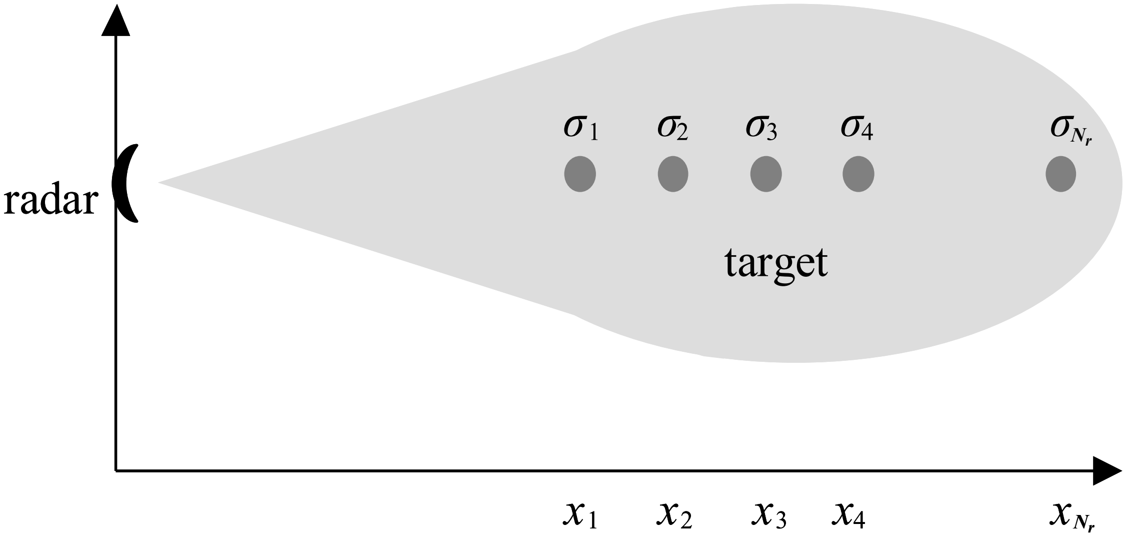


Figure 5 Geometry for One-Dimensional Range Imaging (After: [7]).

3. Range Profile from Match Filtering

The inverse DFT of *I(k) / S(k)* is known as *source deconvolution* and assumes that *S(k)* is nonzero at all frequencies. In practice, radar signals are band-pass signals and thus range imaging is based on matched filtering. Based on matched filter theory, a range profile $C_r^{'}$, slightly different from C_r in (2.21), can be obtained from the cross-correlation received signal with the transmitted signal:

$$C_r^{'}\left(mT_s\right) = I(mT_s) \otimes S(mT_s) \tag{2.22}$$

where $\otimes$ denotes cross-correlation. From the correlation theorem, the DFT of Equation (2.22) can be expressed as [8]:

$$C_r^{'}\left(k\right) = \mathbb{F}\left\{I(mT_s)\otimes s(mT_s)\right\} = I(k)\ S^*(k) \tag{2.23}$$

where the Fourier transform of the cross-correlation is equivalent to the product of Fourier transforms, in which one transformed signal is complex conjugated. Replacing the Fourier domain signal *I(k)* in (2.23) by the DFT of (2.19), the result is:

$$C_r^{'}\left(k\right) = \sum_{m=1}^{N_r} \sigma_m \mathbb{F}\left\{ s\left(mT_s - \frac{2\ x_m}{c}\right)\right\}\ S^*\left(k\right). \tag{2.24}$$

From the inverse DFT of (2.24) the target range profile can be given as:

$$\begin{aligned} C_r^{'}\left(m\ T_s\right) &= \mathbb{F}^{-1}\left\{ \sum_{m=1}^{N_r} \sigma_m \left|S(k)\right|^2 e^{-j2\pi\ k\ f_s \frac{2x_m}{c}} \right\} \\ &= \sum_{m=1}^{N_r} \sigma_m \delta\left(m\ T_s - \frac{2x_m}{c}\right) \mathbb{F}^{-1}\left|S(k)\right|^2 . \end{aligned} \tag{2.25}$$

The target profile obtained from cross-correlation in (2.25) when transformed to the range domain has a similar form as the range profile in (2.18) except for the term $\mathbb{F}^{-1}\left|S(k)\right|^2$, which Soumekh [7] defines as a point-spread function (PSF). Using (2.25), the target range profile obtained from cross-correlation can be related to the ideal deconvolution range profile as:

$$C_r^{'}\left(m\ T_s\right) = C_r\left(m\ T_s\right) * \mathbb{F}^{-1}\left|S(k)\right|^2 . \tag{2.26}$$

From the previous relation, it can be seen that the PSF term determines the time (range) resolvability of the targets. The PSF in the case where the band-pass spectrum possesses an amplitude equal to one has the form of the sinc function [7] within an approximate time interval of $1/\Delta$. The range resolution can therefore be obtained as:

$$d_r = \frac{c}{2}\ \frac{1}{\Delta} = \frac{c\ \tau}{2} \tag{2.27}$$

which agrees with the result presented in (2.13).

The previously described matched filtering technique when applied with chirp signals is essentially a pulse compression. In the following sections, it will be stated as range compression to differentiate it from the azimuth or Doppler compression.

E. ISAR SIGNAL PROCESSING FOR A CHIRP PULSE RADAR

1. Range Compression

The range compression process usually occurs after the received signal is detected in quadrature, sampled and digitized. The resulting digital signal must retain both magnitude and phase of each sample for subsequent coherent processing. A target range profile can be obtained from each pulse of the returned signal using the expression in (2.22), where the multiplication of Fourier transforms is equivalent to a signal cross-correlation.

Figure 6 outlines a block diagram of the fast correlation that implements the range compression process. The ISAR received signal is down-converted in frequency and processed with a Fast Fourier Transform (FFT). A coherent reference signal from the transmitted chirp is also processed with an FFT and then converted into its complex conjugate equivalent. Both signals are multiplied and processed with an inverse Fourier transform to obtain the target range profile.

2. Azimuth Compression

The range compression process produces one range profile for each of the N_p returned pulses that are collected. These N_p range profiles can be arranged in a matrix where each row corresponds to a different returned pulse and each column is comprised of profile elements belonging to the same range bin. This matrix can be expressed as:

$$C_r^{'}\left(m\,T_s,n\right)=\begin{bmatrix} C_{r\;N_P}^{'}\left(m\,T_s\right) & \text{for pulse } N_P \\ C_{r\;N_P-1}^{'}\left(m\,T_s\right) & \text{for pulse } N_P\text{-1} \\ \vdots & \\ C_{r\;2}^{'}\left(m\,T_s\right) & \text{for pulse 2} \\ C_{r\;1}^{'}\left(m\,T_s\right) & \text{for pulse 1} \end{bmatrix} \tag{2.28}$$

where $C_r^{'}\left(t\right)$ are range profiles for different pulses as defined in (2.22).

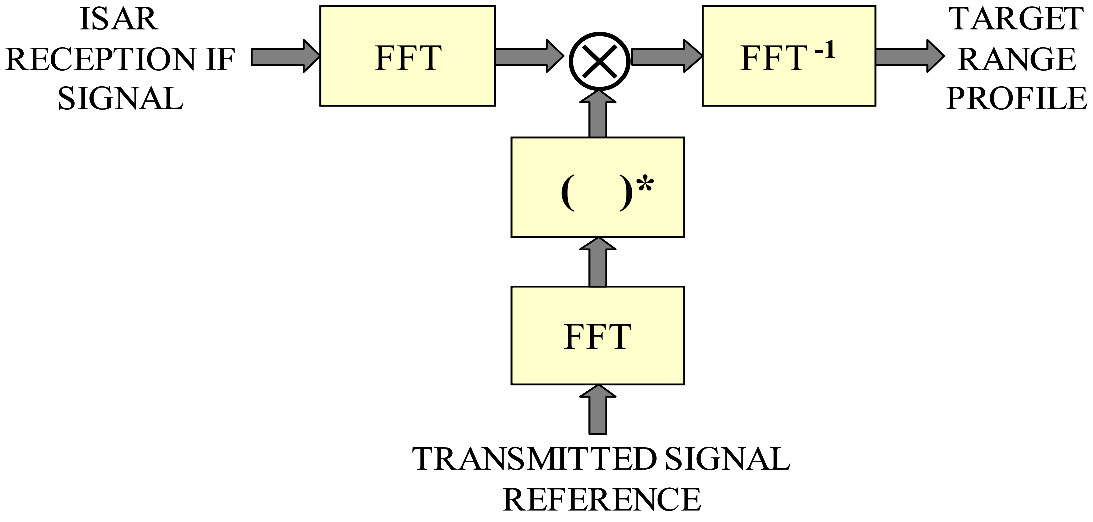


Figure 6 Range Compression Using Fast Correlation.

The process of azimuth compression is basically the FFT of the time history of range profiles. At each range cell location (columns), the time dimension expressed in terms of pulse indices is transformed to the frequency dimension in terms of Doppler cells. Essentially, the FFT of each range profile matrix in the column dimension yields the ISAR range–Doppler image [8]. Thus, the ISAR image can be expressed as:

$$C\left(m,k\right)=FFT\left\{C_r^{'}\left(mT_s,n\right)\right\}=\frac{1}{N_P}\sum_{n=0}^{N_p-1}C_r^{'}\left(mT_s,n\right)e^{-j\omega\,T_s\,k\,n}. \qquad (2.29)$$

The creation of the matrix $C_r^{'}\left(mT_s,n\right)$ by stacking range profiles from different pulses relies on a precise range alignment. Thus, range tracking systems are used to continually align the range profiles $C_r^{'}\left(mT_s\right)$ by adjusting time delays. These systems lock on a prominent scatterer point to set a range trigger delay reference to compensate the rest of the scatterers of the range profile.

In the azimuth dimension, the oscillatory movement of ship targets varies the apparent cross-range dimensions depending on the rotational velocity during the data integration time. Figure 7 shows different ISAR images obtained from the same target at different times. Image 1 shows a ship pitching away from the radar with its masts producing a negative Doppler. Images 2 and 3 show that the data was collected during a period of slow ship rotation. Thus, in accordance with Equation **Error! Reference source not found.**, their azimuth resolution is poor. Images 4, 5 and 6 were generated at an integration period of fast ship rotation. Therefore, the Doppler shift produced by the masts stands out and gives a ship silhouette less difficult to recognize.

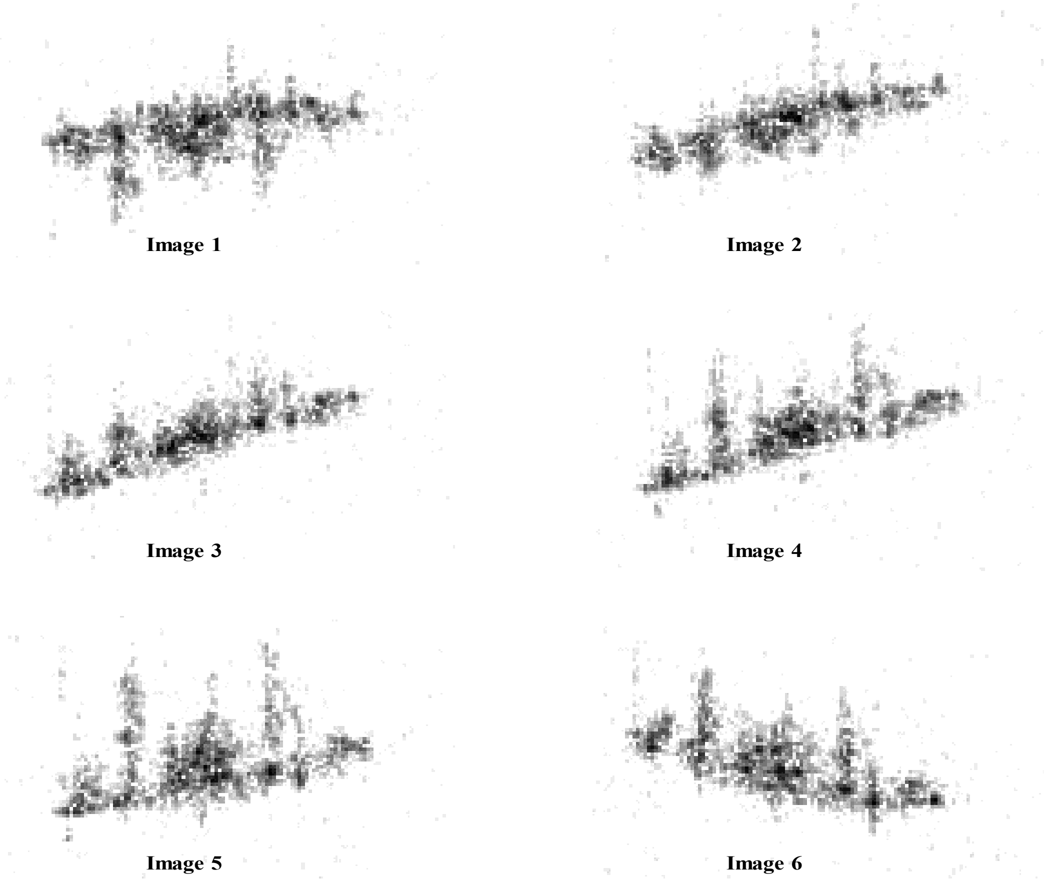


Image 1

Image 2

Image 3

Image 4

Image 5

Image 6

Figure 7 Sequence of Six ISAR Images (From: http://airborne.nrl.navy.mil/)

F. CHAPTER SUMMARY

The concept of SAR and ISAR were presented and explained in terms of their resolution advantage. Next, the range imaging concept was presented in a one-dimensional analysis, where range compression was described in terms of match filtering by implementing fast correlation. Finally, the ISAR processing technique of azimuth compression and the generation of the Range Doppler profile were described. The next chapter presents the DIS concept, for which a primary level of range profiling and ISAR processing is assumed.

THIS PAGE INTENTIONALLY LEFT BLANK

III. THE DIS PROCESS

This chapter presents the DIS concept and its signal processing theory as first described in [10]. Further work on this concept has been discussed in references [6], [9] and [11-18]. Following the theoretical description of the process, the DIS hardware is described at a system level.

A. OVERVIEW

The DIS was designed for electronic attack against linear frequency modulation (LFM) ISARs by emulating the scattering of a false target. In order to effectively deceive an ISAR, the DIS must be able to generate the same scattering effect that a real target would produce on the returning radar signal. The false target should be of a realistic size and its Doppler effects should reflect a feasible target rotational motion. The transmitted power of the DIS deceiving signal should be high enough to cover the true scattering return produced by the DIS platform and low enough to avoid the saturation of the ISAR receiver, and thus revealing the presence of jamming. Figure 1 of Chapter I shows a block diagram summarizing the DIS operational concept to deceive an ISAR platform.

To produce the scattering effect of a false target, the DIS is provided with a set of pulses that are intercepted and down-converted in frequency from an ISAR. The DIS signal processing includes amplitude and phase modulation to recreate, on a pulse-to-pulse basis, the amplitude change and Doppler profile caused by the reflecting surfaces of the target at each location in range and cross-range. The DIS output is upconverted to the carrier frequency and retransmitted back to the radar. A block diagram summarizing the signal processing occurring in the DIS circuit is shown in Figure 8.

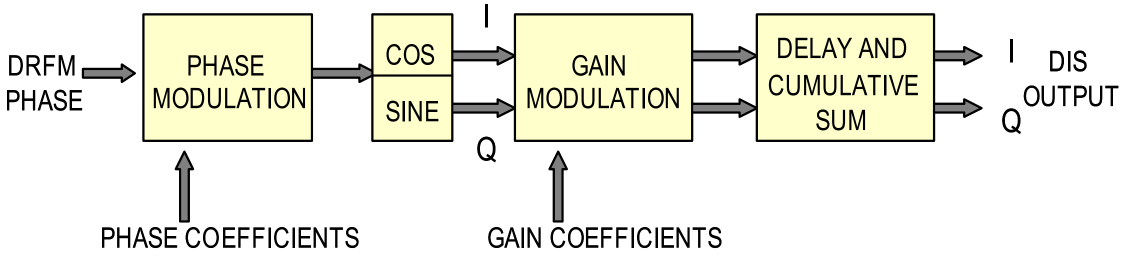


Figure 8 Block Diagram of the DIS.

The ISAR being targeted receives the DIS modulated signal and passes it through a matched filter that compensates for platform motion. The range profiles are obtained for each pulse and collected in a matrix to build a time history of the target rotational movement. An FFT of this time history provides a Doppler mapping of the range profiles, thus creating a two-dimensional profile or image. The image is enhanced, compared with other images, analyzed for features that can be extracted, and processed for automatic target recognition. Figure 9 shows a block diagram summarizing the ISAR processing of the DIS signal and the generation of the false target range-Doppler image.

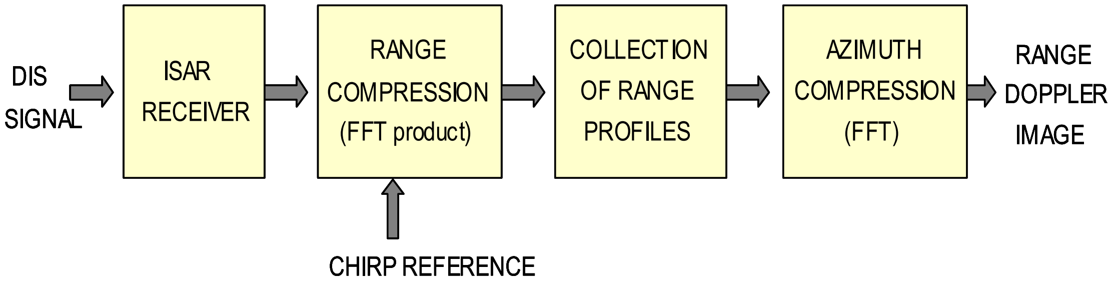

DIS SIGNAL
ISAR RECEIVER
RANGE COMPRESSION (FFT product)
CHIRP REFERENCE
COLLECTION OF RANGE PROFILES
AZIMUTH COMPRESSION (FFT)
RANGE DOPPLER IMAGE

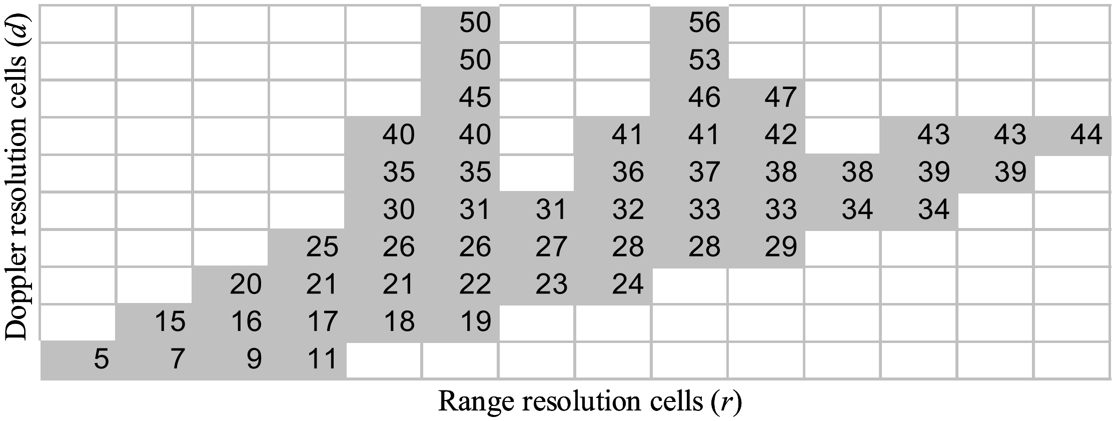
Figure 9 Block Diagram of the ISAR Reception and Image Generation.

This chapter describes the signal processing occurring in the DIS and its peripherals. The description is divided into the generation of modulation coefficients, the sampling of the ISAR signal and DIS signal processing. It also includes a simplified model of the ISAR receiver signal processing to describe the basics of the target range Doppler image formation. Finally, a brief description of the DIS architecture explains how the signal processing is implemented in hardware. The equations presented in this chapter form the basis of the simulation software used to evaluate the effectiveness of the DIS design as an ISAR countermeasure. These simulation results are presented in Chapter V.

B. GENERATION OF THE DIS MODULATION COEFFICIENTS

In order to synthesize echoes corresponding to a false target, the proper phase and amplitude modulation must be performed for each range bin to synthesize the Doppler effect due to target rotation and the expected radar cross-section value at each scatterer location in the range-Doppler plane.

A false target range-Doppler profile is built by creating a two-dimensional matrix of complex scattering elements. In the case of a ship target, the silhouette can be laid out in an N_r by N_d matrix, where N_r is the number of the resolution cells in the range axis and N_d is the number of Doppler resolution cells in the cross-range axis. For simplicity, the range-Doppler profile is usually set with the ship's longitudinal axis oriented along the range dimension and the ship's vertical axis oriented along the cross-range axis as shown in Figure 10.


Doppler resolution cells (*d*)
Range resolution cells (*r*)

					50			56					
					50			53					
					45			46	47				
				40	40		41	41	42		43	43	44
				35	35		36	37	38	38	39	39	
				30	31	31	32	33	33	34	34		
			25	26	26	27	28	28	29				
		20	21	21	22	23	24						
	15	16	17	18	19								
5	7	9	11										

Figure 10 Range-Doppler Profile Template for a False Target (Cell Doppler *f(r,d)* in Hz).

The amplitude of each scatterer in the range-Doppler profile depends on the radar cross-section at that location. The phase of each scatterer depends on the Doppler shift due to the velocity component along the LOS for each target location. The Doppler shift produces a time dependent phase shift and, thus, is different for each pulse arrival. The mathematical representation for each target return as a function of the transmitted pulse index *n* and the scattering location in the false target profile can be expressed in a tri-dimensional matrix as:

$$T\left(r,d,n\right)=A\left(r,d\right)\ e^{-j2\pi f\left(r,d\right)\ n\ PRI} \tag{3.1}$$

where *r* and *d* are target location indices is terms of range and Doppler resolution cells, *A* is the amplitude of the scatterer due to the radar cross-section at the *(r,d)* location and *f(r,d)* is the Doppler frequency shift that the scatterer exerts on the ISAR signal.

A simplified expression for the return signal can be obtained by adding the complex scatterers along the azimuth (Doppler) dimension. The resulting superposition expression is given by:

$$T'(r,n)=\sum_{d=1}^{N_d} T(r,d,n). \tag{3.2}$$

Equation (3.2) represents a return signal for each pulse *n* at each target range resolution bin *r*. This signal can be used to obtain both the phase and the magnitude modulation coefficients required by the DIS.

The phase of the return signal is calculated for each range cell and pulse index combination using the *angle* function, which is the *arctan* function of the division between the imaginary and the real components, using their signs to solve the $\pm\pi$ ambiguity:

$$\phi_T(r,n)=angle\left(real\left\{T'(r,n)\right\},imag\left\{T'(r,n)\right\}\right). \tag{3.3}$$

The phase modulation effect produced by a target reflection must be understood as a phase increment on a pulse-by-pulse basis. For that reason, the phase of the returned pulse is subtracted from the phase of the previous pulse at each range bin *r* and added to the previous increments to obtain a cumulative sum of increments:

$$\phi_{inc}^{'}(r,n)=\phi_{inc}^{'}(r,n-1)+\phi_T(r,n-1)-\phi_T(r,n). \tag{3.4}$$

The cumulative phase increment describes the ideal phase modulation coefficient needed for image synthesis. The phase increment from Equation (3.4) is scaled and truncated to integer values from 0 to $2^{k_p}-1$. The final expression for the k_p-bit phase coefficient fed to the DIS at each range bin *r* and pulse *n* is given by:

$$\phi_{inc}(r,n)=\left\lfloor \frac{\phi_{inc}^{'}(r,n-1)\ 2^{k_p}}{2\pi} \right\rfloor \tag{3.5}$$

where$\lfloor\ \ \rfloor$is the floor integer rounding function, defined as the greatest integer less than or equal to the argument.

The amplitude modulation effect in the return signal can be synthesized by multiplying the intercepted signal by a gain value equal to the magnitude of the sum of complex scatterers at the same range-bin/pulse combination. Thus, the first step to obtain the gain coefficients is to normalize the magnitude of $T'(r,n)$ in Equation (3.2) so that all magnitudes have a range of values between 0 and 1:

$$T_N^{'}\left(r,n\right)=\frac{\left|T^{'}\left(r,n\right)\right|}{\max\left|T^{'}\left(r,n\right)\right|} \tag{3.6}$$

The DIS amplitude modulation is implemented by shifting the I and Q bit formats by a specific number of binary places. This simple procedure avoids the use of multipliers and reduces processing time. The amount of bit shift *g(r,n)* is set by a control code of k_g bits. Although the available control codes are 2^{k_g}, as it will be explained in the next chapter, some of them may produce the same amount of shift. Since the available gains are power of two values, the quantization of the normalized magnitudes in (3.6) to assign these gains, should also be distributed exponentially. For a gain shifter controlled with a 4-bit format, the 16 available codes only produce 11 unique shifts. One possible gain quantization scheme for this case can be implemented as shown in Table 1:

Magnitude Ranges	**Number of Shifts *g(r,n)***	**Effective Gain $2^{g(r,n)}$**
0.8 to 1.0	10	1024
0.4 to 0.8	9	512
0.2 to 0.4	8	256
0.1 to 0.2	7	128
0.05 to 0.1	6	64
0.025 to 0.05	5	32
0.0125 to 0.025	4	16
0.0063 to 0.0125	3	8
0.0032 to 0.0063	2	4
0.0016 to 0.0032	1	2
0 to 0.0016	0	1

Table 1 Gain Modulation Quantization Scheme.

C. INTERCEPTION AND SAMPLING OF THE ISAR SIGNAL

An LFM ISAR pulse with constant amplitude and upward frequency sweep (chirp) can be expressed as:

$$r(t)=\text{rect}\left(\frac{t}{\tau}\right)\; e^{j2\pi\left(f_c+\Delta\cdot t^2/2\tau\right)} \tag{3.7}$$

where τ is the uncompressed pulsewidth in seconds, Δ is the modulation bandwidth in Hertz and f_c is the transmitted carrier frequency, also in Hertz. The envelope of the chirp pulse is given by the rectangular function $\mathrm{rect}(t/\tau)$, where there is a constant amplitude of one for values of the argument t/τ between -0.5 and +0.5. Figure 11 shows the pulse envelope, the modulating frequency used at each time and the resulting LFM pulse.

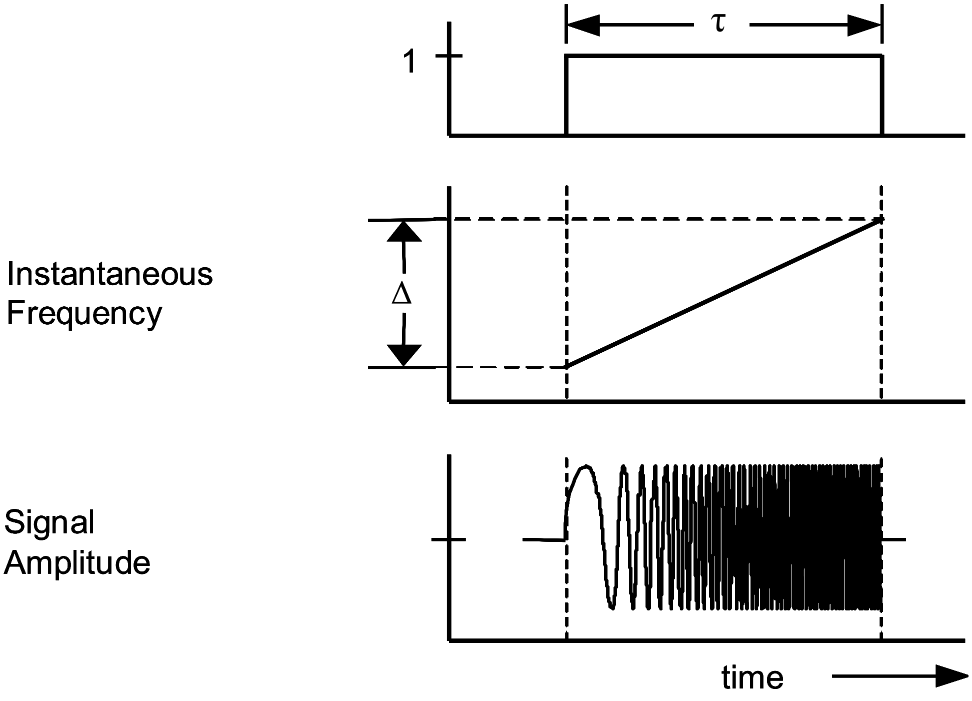


Figure 11 Construction of an LFM Chirp Pulse (From: [6]).

For an ISAR mounted on a moving platform, the signal intercepted by a moving target has an instantaneous phase advance at time *t* given by:

$$\theta(t) = \frac{4\pi f_c}{c}\left(R + v\,t + a\,t^2/2\right) \tag{3.8}$$

where *R*, *v* and *a* are the initial range, initial relative velocity and relative acceleration between the ISAR platform and the target. The corresponding frequency variation due to this phase change is the first derivative of the previous equation given by:

$$f(t) = \frac{2f_c}{c}\left(v + a\,t\right). \tag{3.9}$$

Therefore, the LFM ISAR signal in Equation (3.7), modified for the case of a moving platform and a moving target, can be expressed as:

$$r(t)=\mathrm{rect}\left(\frac{t}{\tau}\right)e^{j2\pi\left(f_c+f+\Delta\, t^2\big/2\tau\right)}. \tag{3.10}$$

The Doppler frequency component *f(t)*, considering realistic platform velocities and ISAR frequencies in the X band (8-12 GHz), is on the order of tens of Hertz. Since the modulation bandwidth is usually on the order of hundreds of MHz, it can be assumed that the intercepted chirp pulse has a negligible Doppler ($f\approx 0$) or constant phase change. From (3.10), the corresponding complex envelope obtained after down converting the intercepted chirp is given by:

$$s(t)=\mathrm{rect}\left(\frac{t}{\tau}\right)e^{j\pi\,\Delta\, t^2/\tau}. \tag{3.11}$$

This complex envelope is stored using a digital RF memory (DRFM). This equipment permits the indefinite and non-degraded storage of radio frequency signals in digital form. The stored signals can be transcribed back into an exact replication of the original signal, which allows coherent transponding of radar signals that are very suitable for EA.

A DRFM is fed with the intercepted ISAR signal and replicates it in terms of phase samples or complex envelope amplitude samples (I and Q), at sample times given by:

$$t=\frac{m}{f_S}+n\ PRI \tag{3.12}$$

where *m* is the index of the intrapulse samples, *n* is the index of each pulse, f_s is the ADC sampling frequency and *PRI* is the ISAR pulse repetition interval in seconds. Although many imaging radars use variable PRI, the PRI stays constant within one integration frame time. Substituting the sample time from Equation (3.12) into (3.11), the sampled phase of the complex envelope can be expressed as:

$$\phi_0\left(m,n\right)=\frac{\pi\,\Delta}{\tau}\left(\frac{m}{f_S}+n\ PRI\right)^2 \tag{3.13}$$

The phase signal in Equation (3.13), as generated by a k_p-bit *phase sampling* DRFM, can be expressed as:

$$\phi(m,n) = \left\lfloor \frac{\phi_0(m,n)\ 2^{k_p}}{2\pi} \right\rfloor \tag{3.14}$$

which is obtained by rescaling the complex envelope phase and truncating it to the integer value. The k_p-bit DRFM phase expression of (3.14) gives 2^{k_p} integer values, ranging from 0 to $2^{k_p}-1$. Therefore, the DRFM phase samples have the same k_p-bit format as the phase modulation coefficients from Equation (3.5). The phase resolution for this DRFM output is given by the number of phase quantization levels $(2\pi)/2^{k_P}$.

For the case of a *complex amplitude-sampling* DRFM, the intercepted signal is stored in terms of the orthogonal components of the complex envelope:

$$D_{0I}(m,n) + j\ D_{0Q}(m,n) = \cos\left(\phi_0(m,n)\right) + j\ \sin\left(\phi_0(m,n)\right). \tag{3.15}$$

The complex envelope, as generated by the two output channels of a k_q-bit amplitude-sampling DRFM, can be expressed as:

$$\begin{aligned} D_I(m,n) &= \left\lfloor D_{0I}(m,n)\ 2^{k_q-1} \right\rfloor \Big/ 2^{k_q-1} \\ D_Q(m,n) &= \left\lfloor D_{0Q}(m,n)\ 2^{k_q-1} \right\rfloor \Big/ 2^{k_q-1}. \end{aligned} \tag{3.16}$$

Either the phase or amplitude DRFM signals can be fed into the DIS chip in order to synthesize the effects of a reflecting target.

D. THE DIS SIGNAL PROCESSING

The DIS is a parallel array of identical complex digital modulators and each one synthesizes the scattering effects for one range bin. The input coefficients necessary to synthesize an image from the DRFM samples consist of the target dependent phase and gain coefficients and the desired target extent provided by a microprocessor. Figure 12 presents a block diagram of the DIS along with the external components. The wideband chirp signal that feeds the block diagram is described by Equation (3.10). After down-conversion, the complex envelope components are represented by Equation (3.11). The phase or amplitude (I and Q) DRFM samples are described by Equations (3.14) and (3.16), respectively. For the case of amplitude DRFM samples, they are processed by a phase conversion circuit.

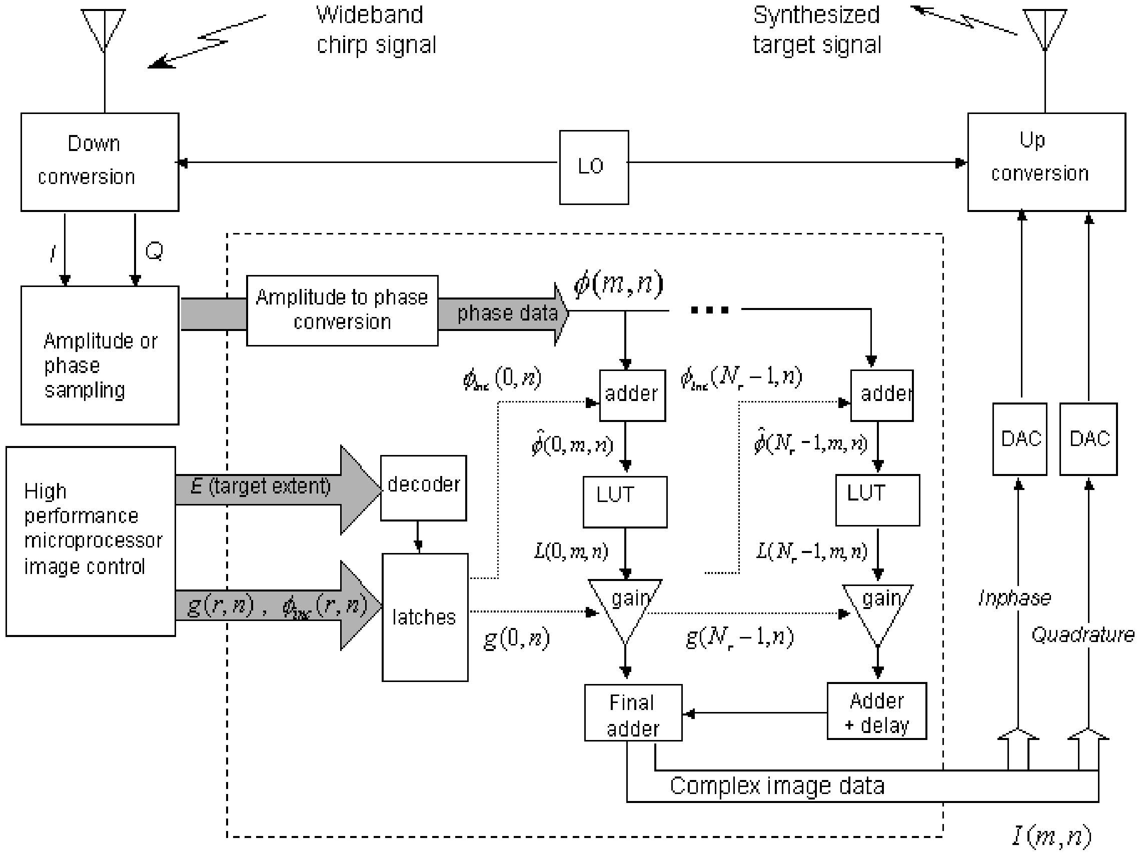


Figure 12   DIS Block Diagram (From: [6]).

For the cases where an amplitude-sampling DRFM is used to feed the DIS, it is necessary to convert the k_q-bit I and Q samples to k_p-bit phase samples. This *arctan* operation is carried out using the Coordinate Rotation Digital Computer (CORDIC) algorithm. CORDIC rotates the phase of $I + jQ$ in the complex plane by multiplying powers of two values using shift-add algorithms. These incremental phase rotations make it possible to calculate trigonometric functions such as *arctan* without the need of multipliers [15].

Each phase sample strobed from the DRFM $\phi(m,n)$ is fed in parallel to all the DIS complex modulators (range bins) and are first added with the false target phase coefficients corresponding to each range bin. The phase coefficient present at each range bin changes for every new ISAR intercepted pulse. That is, at the r^{th} array branch, the m^{th} sample of the n^{th} ISAR pulse is added with the phase coefficient for that range-bin/pulse combination, giving a phase output of:

$$\hat{\phi}(r,m,n) = \phi(m,n) + \phi_{inc}(r,n). \tag{3.17}$$

The modulated phase obtained in (3.17) at each range bin is used to construct a complex signal of unit amplitude given by:

$$L(r,m,n) = \cos\left(\hat{\phi}(r,m,n)\right) + j\ \sin\left(\hat{\phi}(r,m,n)\right). \tag{3.18}$$

The sine and cosine operations for the 2^{k_p} different phase values are implemented using a look-up table (LUT) with a depth of 2^{k_p} and a resolution of k_L bits. The LUT output presented in (3.18) has two output channels: I (cosine) and Q (sine), each with amplitude values ranging from -1 to 1.

The I and Q output channels of the LUT are amplitude modulated with the gain coefficients for each range-bin/pulse combination. The multiplication by the power of two gain coefficients is implemented by shifting bit positions to the left. The gain modulated channels can be expressed as:

$$S(r,m,n) = 2^{g(r,n)} L(r,m,n) = 2^{g(r,n)} e^{j(\phi(m,n)+\phi_{inc}(r,n))} \tag{3.19}$$

where *g(r,n)* is the gain coefficient.

Following the gain modulation, all further range-bin processors are summed with the current processor and passed forward. The final DIS output is taken from range bin one with $\tau f_S + N_r - 1$ complex samples per pulse, where τ is the pulsewidth. The DIS output expression for the m^{th} sample of the n^{th} pulse is given by:

$$I(m,n) = \sum_{r=0}^{N_r-1} S(r,m,n) = \sum_{r=0}^{N_r-1} 2^{g(r,n)} e^{j(\phi(m-r,n)+\phi_{inc}(r,n))}. \tag{3.20}$$

The DIS output samples are fed to a digital-to-analog converter (DAC), coherently modulated with the same local oscillator signal used for reception and transmitted back to the victim ISAR.

E. THE ISAR RECEPTION AND IMAGE GENERATION

In order to evaluate the proper generation of a false target, it is necessary to model the process occurring in the ISAR receiver. ISAR signal processing is described in Chapter II, but the model used in the simulation is simplified to a range compression stage followed by a Doppler (cross-range) compression. This section describes the range and azimuth compression processes in terms of the mathematical equations applied in the simulation software.

The ISAR range compression is implemented using fast correlation between the received DIS signal containing false target echoes and a coherent reference signal. The correlation process is implemented as a product of FFT signals as shown in Figure 6. From equations (3.11) and (3.12), the ISAR reference function is described by a time-periodic chirp-pulse train with samples given by:

$$s(m,n) = \mathrm{rect}\left(\frac{t}{\tau}\right) e^{j\frac{\pi\Delta}{\tau}\left(m/f_s + n\ PRI \right)^2} . \tag{3.21}$$

The FFT of (3.21) determines the transfer function of the pulse compression network for each pulse *n* at the Doppler filter *k*:

$$S(k,n) = \frac{1}{N} \sum_{m=0}^{N-1} s(m,n) e^{-j\omega\ T_s\ k\ m} \tag{3.22}$$

where *N* represents the number of FFT points. For better FFT resolution *N* can be increased by padding zeroes at the end of the reference signal vector in (3.21).

The DIS signal received by the ISAR after down-conversion to base band corresponds to the expression in (3.20). The FFT is given by:

$$I(k,n) = \mathrm{FFT}\left\{ I(m,n) \right\} = \frac{1}{N} \sum_{m=0}^{N-1} I(m,n) e^{-j\omega\ T_s\ k\ m} . \tag{3.23}$$

Multiplying the complex conjugate of the transfer function in (3.22) by the frequency signal in (3.23) and applying an inverse FFT, yields the complex range profile of the false target echoes corresponding to the n^{th} transmitted pulse:

$$C_r\left(m,n\right) = \text{FFT}^{-1}\left\{I\left(k,n\right)S^*\left(k,n\right)\right\} = \frac{1}{N}\sum_{k=0}^{N-1} I\left(k,n\right)S^*\left(k,n\right)e^{j\omega\, T_s\, k\, m}. \quad (3.24)$$

The range profile obtained for each transmitted pulse after range compression is a collection of complex scatterers grouped at each range bin.

The Doppler profile at each range bin is obtained from the magnitude of the FFT of the complex scatterers in the time dimension (pulse indices). Using Equation (2.29), the result of this azimuth compression for all range bins gives the range–Doppler profile of the false target as:

$$C\left(m,k\right) = \left|\text{FFT}\left\{C_r\left(m,n\right)\right\}\right| = \left|\frac{1}{N}\sum_{n=0}^{N_p-1} C_r\left(m,n\right)e^{-j\omega\, T_s\, k\, n}\right| \qquad (3.25)$$

The range-Doppler profile obtained in Equation (3.25) is the end result of the DIS EA effort. This image profile should deceive the ISAR operator and the supporting target recognition algorithms.

F. ARCHITECTURE OF THE DIS MODULE

The DIS module, as presented in Figure 12, is a parallel array of N_r identical modulators controlled by an off-chip microprocessor. Each modulator represents the major reflective surfaces at each range bin of the target profile and thus will be referred to as a range bin. The modulation at each range bin is performed through a pipelined process composed of a phase modulation, a polar-to-rectangular conversion of complex samples with a sine/cosine table, an amplitude modulation and a cumulative sum. The hardware implementation of a range bin depicted in Figure 13 is briefly described in this section. The current DIS chip design has $N_r = 512$ range bins. Additional chips can be cascaded to synthesize larger targets.

1. DIS Phase Modulation

The DIS input signal is a series of DRFM generated phase samples (or I and Q samples converted to phase by a CORDIC circuit) from N_p chirp pulses. Each phase sample is fed simultaneously to all N_r range bins. These samples will be modulated within each range bin with a different phase modulation coefficient updated for every new pulse. This pulse-to-pulse phase modulation (or phase rotation) synthesizes the desired Doppler frequency profile for each range bin.

1. Phase increment
2. Phase data from DRFM
3. Pre-load register
4. Pipeline register
5. 5-bit adder
6. Look Up Table
7. 16-bit adder
8. Gain shifter
9. Input from previous bin
10. Control Logic
11. Output to next bin

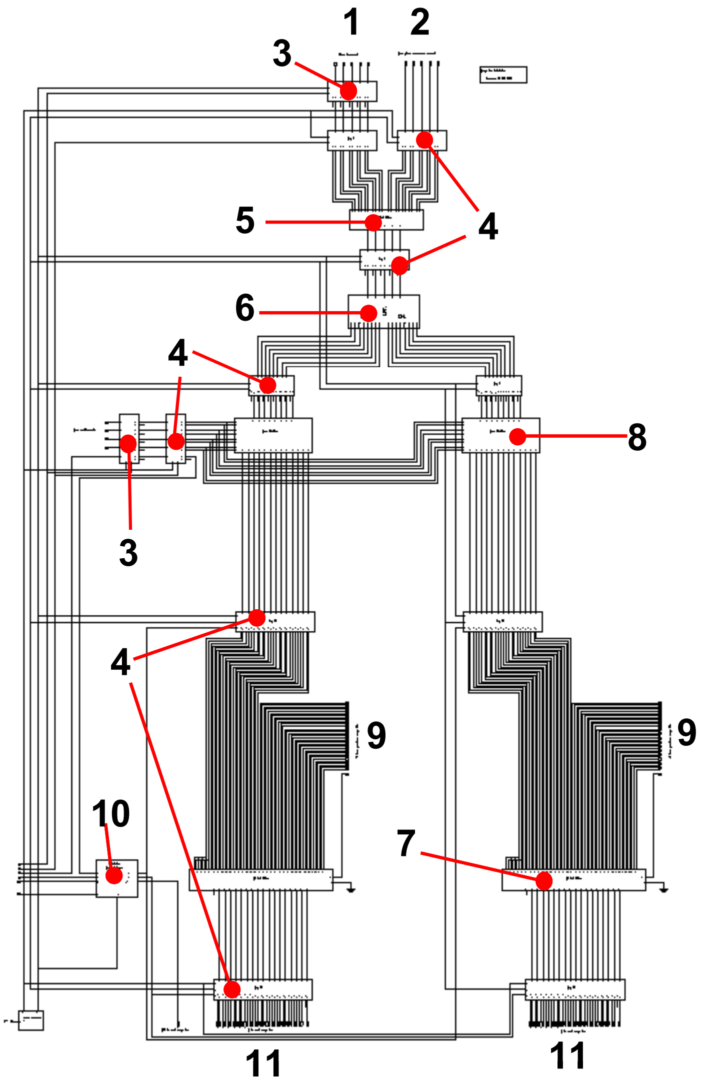


Figure 13 Hardware Diagram for a DIS Range Bin (After:[14]).

As described in Equations (3.5) and (3.14), for k_q=5, both phase signals have a 5-bit format from 00000 to 11111 representing 32 phase levels with increments of 11.25°. Table 2 lists the 32 possible values of this phase format. The phase summation is implemented with a 5-bit phase adder as shown in Figure 14. A control microprocessor programs the phase coefficients which are latched separately into the corresponding range bins at each new PRI.

5-bit format	Phase value	5-bit format	Phase value	5-bit format	Phase value	5-bit format	Phase value
00000	0°	01000	90°	10000	180°	11000	270°
00001	11.25°	01001	102.5°	10001	191.25°	11001	281.25°
00010	22.5°	01010	112.5°	10010	202.5°	11010	292.5°
00011	33.75	01011	123.75°	10011	213.75°	11011	303.75°
00100	45°	01100	135°	10100	225°	11100	315°
00101	56.25°	01101	146.25°	10101	236.25°	11101	326.25°
00110	67.5°	01110	157.5°	10110	247.5°	11110	337.5°
00111	78.75°	01111	168.75°	10111	258.75°	11111	348.75°

Table 2 Equivalence Table for the DIS 5-Bit Phase Format.

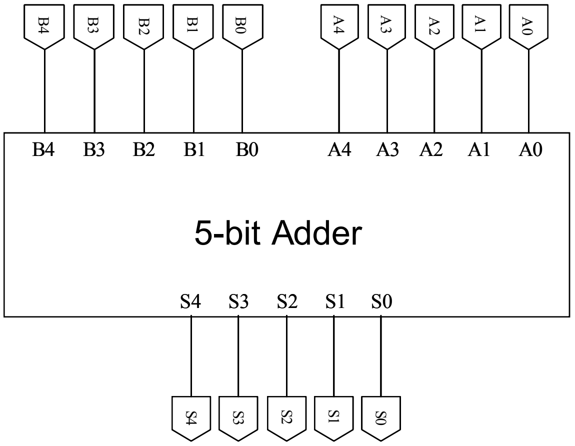


Figure 14 Phase Adder Diagram.

2. DIS Cosine and Sine LUT

The output of the phase modulation adder is a 5-bit format representing the phase of a complex sample with amplitude one. The next step is to convert this complex expression from polar to rectangular format as described in Equation (3.18). The conversion is implemented using the 5-bit phase adder outputs as addresses of a cosine/ sine LUT with a dual channel 8-bit output format. An input/output diagram of the LUT is shown in Figure 15.

The sine/cosine LUT has 32 (2^5) possible pairs of output amplitudes. Each output pair is composed of the I and Q rectangular component of a unitary amplitude complex sample. The I and Q output channel amplitudes are each represented by an 8-bit two's complement format. The format has one sign bit and seven decimal bits from 2^{-1} to 2^{-7} as shown in Table 3. Chapter IV uses additional formats to study the impact of replacing this 8-bit resolution LUT with tables of resolution less than 8-bits.

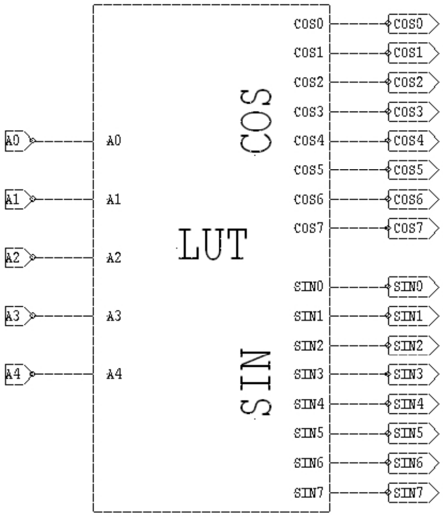


Figure 15 LUT Diagram (From: [14]).

Bit Position	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**
Value	Sign	2^{-1}	2^{-2}	2^{-3}	2^{-4}	2^{-5}	2^{-6}	2^{-7}

Table 3 Bit Format of the LUT Output Amplitudes.

The absence of an integer bit in the LUT format of Table 3 does not allow the representation of a value of '1.0', needed for the phase inputs of 0 and 90 degrees. The amplitude values represented with this format are limited to a range from -1 to $1-2^{-7}$, where the upper limit can be obtained as:

$$s = 2^{-1} + 2^{-2} + 2^{-3} + 2^{-4} + 2^{-5} + 2^{-6} + 2^{-7} = 1 - 2^{-7} = 0.9921875 \ . \qquad (3.26)$$

In order to provide a solution for the cosine and sine values of 0 and 90 degrees, the sine and cosine LUT values are scaled down by a factor of *s*, equivalent to the upper limit found in (3.26). Using the scaled LUT values, the sine output for the phase angle equivalent to 90 degrees is given by:

$$\sin\left(8 \cdot 11.25°\right) = 1 \approx L_{SIN}\left(01000\right) = s \qquad (3.27)$$

where L_{SIN} is the LUT sine output and 01000 is the 5-bit phase representation for 90 degrees as given in Table 2.

3. DIS Amplitude Modulation

The signals in the I and Q channels are amplitude modulated by multiplying each binary signal by the gain modulation coefficients $2^{g(r,n)}$. Similar to the phase modulation, the gain coefficients are fed separately for each range bin and updated for every new ISAR pulse (each PRI). The use of a multiplier at this stage is avoided due to fast speed requirements. Instead, the gain modulation is implemented by shifting bit locations. A gain shifter module shifts the binary digits by *g(r,n)* places (from 0 to 10) using four arrays of 2-bit to 1-bit multiplexers. An input/output diagram of the gain shifter is shown in Figure 16. Details of the multiplexer implementation can be found in Fouts et al. [9].

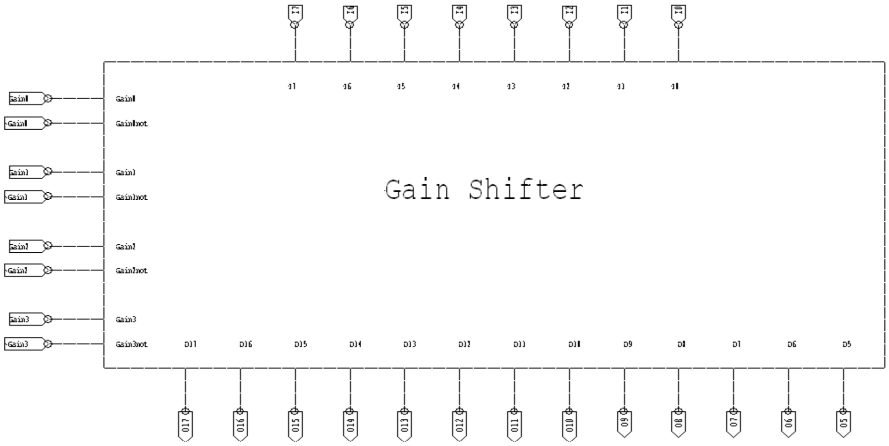


Figure 16 Gain Shifter Diagram (From:[14]).

The left side of the gain shifter diagram in Figure 16 shows four control inputs and the negated equivalents. These inputs represent a set of 16 control codes to select gains according to the list included in Table 4. The missing control codes of 4, 8, 9, 10 and 12 are discarded because they produce a shift also produced by a lower control code. For instance, the control codes 0011 (3) and 0100 (4), when applied to the multiplexer array, both produce a shift of three binary digits.

The outputs of the I and Q channels after gain modulation are amplitudes from -1028 to $1028-2^{-7}$. They are represented in an 18-bit two's complement format with one sign bit, ten integer bits from 2^9 to 2^0 and seven decimal bits from 2^{-1} to 2^{-7}. The format described is then truncated to the five least significant bits (LSB). Thus, each channel is composed of a 13-bit format with a sign bit, ten integer bits and two decimal bits, as listed in Table 5.

Control Code	**Size of Shift *g(r,n)***	**Multiplication Factor $2^{g(r,n)}$**
0	0	1
1	1	2
2	2	4
3	3	8
5	4	16
6	5	32
7	6	64
11	7	128
13	8	256
14	9	512
15	10	1024

Table 4 Gain Shifter Control Codes.

Bit Position	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**
Value	Sign	2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0	2^{-1}	2^{-2}

Table 5 Bit Format of the Gain Modulation Output after Truncation of 5 LSBs.

The use of bit shifting from 0 to 10 binary places, instead of a multiplier, restricts the gains to a binary exponential distribution from 1 to 1028. This gain distribution was the main design consideration in choosing the gain coefficient quantization scheme given in Table 1.

4. DIS Cumulative Sum

The complex outputs for all range bins are added to form a consolidated output (return signal) where the scattering effects of the different range locations of the target are superimposed. The cumulative sum of the modulated outputs start from the output of the last range bin being added with the adjacent range-bin output. On the next clock pulse, the result of this sum is added with the next adjacent range bin until all range-bin outputs are added. The implementation of this cumulative sum is shown in Figure 17.

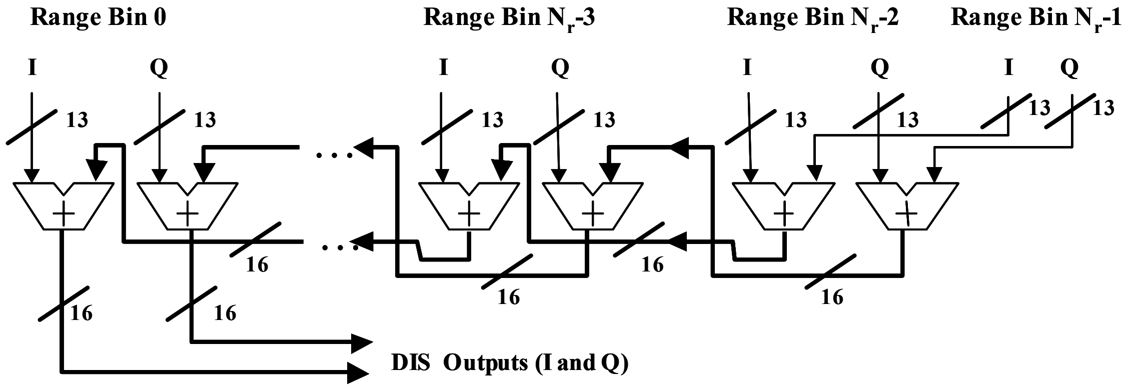
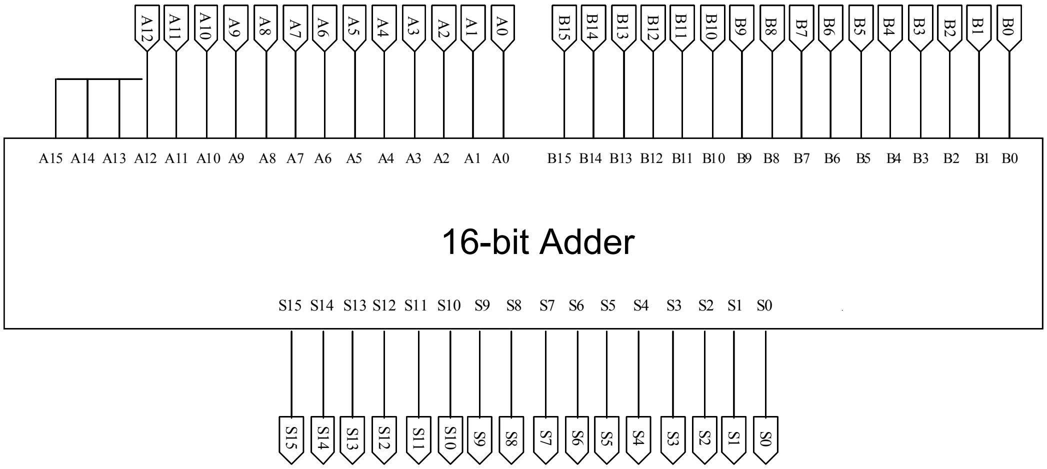


Figure 17 Diagram of the Output Adder Cumulative Sum.

The hardware implementation of the cumulative sum uses a 16-bit adder for each channel (I and Q) in each range bin. A diagram of this adder with the input and output connections is shown in Figure 18. The 13-bit truncated signal (gain modulation output) with the format described in Table 5 is fed into one of the adder inputs. To complete the three missing bit locations, the sign bit at location 13 is also fed at locations 14, 15 and 16. The replication of the sign bit in the three most significant bits is equivalent to a 'zero' padding for a positive number and a 'one' padding for a negative number. Both cases do not alter the value of the data in two's complement form. The second adder input is fed with the 16-bit output from the cumulative sum of previous range bins.



A12 A11 A10 A9 A8 A7 A6 A5 A4 A3 A2 A1 A0
B15 B14 B13 B12 B11 B10 B9 B8 B7 B6 B5 B4 B3 B2 B1 B0
A15 A14 A13 A12 A11 A10 A9 A8 A7 A6 A5 A4 A3 A2 A1 A0
B15 B14 B13 B12 B11 B10 B9 B8 B7 B6 B5 B4 B3 B2 B1 B0
16-bit Adder
S15 S14 S13 S12 S11 S10 S9 S8 S7 S6 S5 S4 S3 S2 S1 S0
S15 S14 S13 S12 S11 S10 S9 S8 S7 S6 S5 S4 S3 S2 S1 S0

Figure 18 16-bit Adder Diagram.

The DIS I and Q output channels have the 16-bit format shown in Table 6. The DIS sends the outputs channels to the digital-to-analog converters (DAC). The analog signals are then up-converted in frequency and transmitted to the victim radar.

Bit Position	**16**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**
Value	Sign	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}	2^{-1}	2^{-2}

Table 6 Bit Format of the DIS Output (per Channel).

5. DIS Control Signals

The DIS is controlled by an off-chip microprocessor that generates appropriate control signals. Description of these DIS control signals follow:

a. Use Range Bin (URB)

This signal is used to enable or disable a range bin as appropriate, according to the target extent or the existence of low radar cross-section segments. If the URB signal is a logic one, the range bin is included in the cumulative sum chain of Figure 17. When a logic zero is applied in the URB signal, the range bin is closed. This means that it feeds a zero to the output adder. Thus, a closed range bin propagates the cumulative sum to the next range bin without incrementing it.

b. Phase Sample Valid (PSV)

This signal indicates when data coming from the DRFM is valid. This signal is only enabled during the pulse duration and disabled during the interpulse period.

c. Output Data Valid Out (ODV_{out})

This signal is used to cascade additional DIS chips in order to synthesize very large targets. The signal indicates that the output of the last range-bin processor in the summation chain of the previous DIS chip is valid.

d. Output Data Valid In (ODV_{in})

This signal is used in conjunction with the ODV_{out} signal to cascade DIS chips. The ODV_{out} signal from a cascaded chip is fed to the next chip as the ODV_{in} signal. For the case of the first chip, the ODV_{in} signal is always enabled.

e. Program Range Bin (PRB)

This signal causes the phase and gain modulation coefficients to be loaded into the range bins. This signal clocks a preload buffer register to update these values every PRI.

f. Use New Programming (UNP)

This signal causes the phase and gain rotation registers to load the respective modulation coefficients from the pre-load registers every PRI.

6. DIS Clock

The DIS chip is a synchronous digital integrated circuit. An off-chip clock signal is required to control the flow of data through the internal logic and to synchronize the DIS chip with the DRFM providing the input data samples, and the DAC receiving the output of the DIS.

G. CHAPTER SUMMARY

The DIS concept was described mathematically from the generation of the input signals, including the signal processing at the complex range-bin modulators, and finishing with the generation of the output-jamming signal. The process that takes place at the ISAR processor (range and azimuth compression) generates a range Doppler image of the intended false target. A brief description of the DIS hardware was included to understand the pipelined process that occurs at each range-bin modulator. In the next chapter, this process is represented with a software model to simulate the DIS input signals, the signal processing, and the follow-on ISAR processing that results in the target image generation.

IV. DIS SIMULATION ALGORITHM

This chapter describes a software model created to simulate the performance of the DIS system design against chirp-pulse ISARs. The software model is implemented with MATLAB version 6.1. Rather than analyzing the software code in a per line basis, only the major signal processing steps will be addressed, presenting the code lines involved for each case. This chapter must be read in conjunction with the complete software codes included in Appendix A in order to understand the origin of variables that are defined before the presented code lines.

The MATLAB code is composed of three main files and a number of auxiliary files. The three main files must be run in sequence. First, the *extract_v5.m* file generates the modulation coefficients for a false target. Next, the *mathost_v5.m* file simulates the DRFM samples used by the DIS and the reference signal used at the ISAR receiver. Finally, the *simhwchk_v5.m* file simulates the DIS signal processing, generates the deceiving signal and simulates the ISAR compression process to produce a target range-Doppler image.

A. GENERATION OF FALSE TARGET MODULATION COEFFICIENTS

1. Loading of a Target Doppler Template

The *extract_v5.m* file loads the range-Doppler template of a target as presented in Figure 10. The template contains the main target scatterers arranged according to their range location *r* and Doppler shift value *f(r,d)*. The *r* coordinate is the range position within the ship structure in terms of range cell units, usually in the horizontal dimension. The *d* coordinate is the cross-range or azimuth location in terms of the Doppler resolution units or Doppler filters, usually in the vertical dimension.

The size of the DIS hardware array (number of range-bin processors) is determined by the target extent in range cell units. The current software model has been set to simulate DIS array sizes for 2, 32, 128 and 512 range-bin processors. This is achieved by loading false target templates from the data files *target02.mat*, *target32.mat*, *target128*.mat and *target512.mat* that are created from the files *doppler_02_target.m*, *dop-*

pler_08.m, *doppler_128.m*, *doppler_32.m* and *doppler_512.m*. The creation of new files with target templates allows the simulation of additional DIS array sizes.

The *extract_v5.m* file starts with the definition of the *numtaps* variable as the number of range-bin processors in the DIS hardware array or target size. The target Doppler template file is chosen and loaded depending on the value of the *numtaps* variable. The Doppler frequency values are saved into a matrix labeled as *freq*. A plot of this matrix is displayed by the routine in order to corroborate the target template that will be synthesized. Figure 19 shows the display of the *freq* matrix, indicating the range Doppler arrangement of the scatterers for a 32-range cell target.

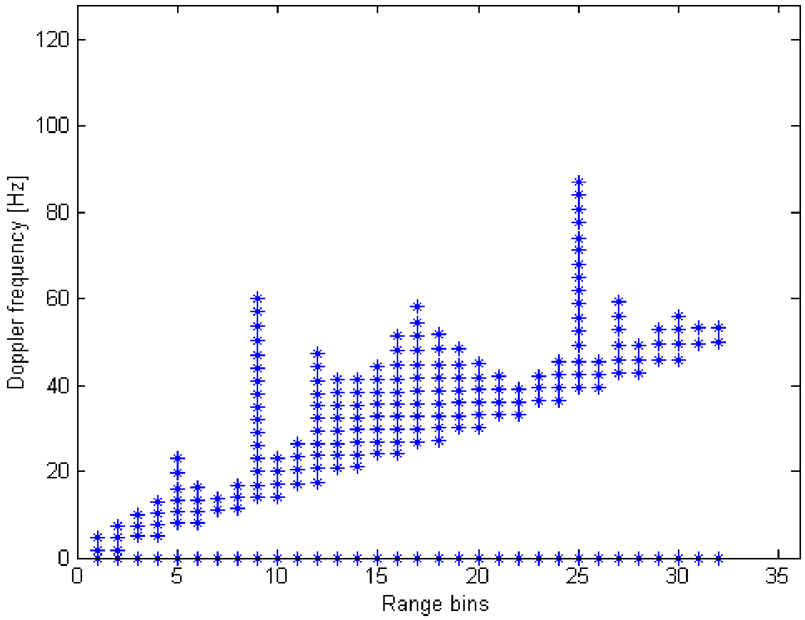


Figure 19 Scatterer Distribution for a 32-Range-bin Target Using 128 Doppler Cells.

2. Generation of a Complex Scatterer Expression

The Doppler frequency values from the *freq* matrix can be used in conjunction with time and magnitude values to produce a complex scatterer expression in polar form. Hence, an array of target scatterers *T(r,d)* for each pulse is obtained using a common magnitude matrix *A(r,d)*, equivalent to the RCS of the scatterers, and phase matrices obtained from the product of 2π, the Doppler shift *f* contained in the *freq* matrix and pulse dependant time values:

$$T(r,d,t) = A(r,d)\, e^{-j\,2\pi f(r,d)\,t}. \tag{4.1}$$

The time used in the phase product of (4.1) must be the leading edge time of the respective pulse, thus it can be expressed as the product $n\ PRI$, where n is the pulse index. Replacing this product in the time expression of (4.1), the array of scatterers can be expressed as a function of the pulse index n resulting in the *T(r,d,n)* three-dimensional scatterer array previously defined in Equation (3.1).

The ISAR total integration time is obtained from the product of the PRI and the number of pulses integrated at the radar receiver N_p, equivalent to the number of Doppler resolution cells. The value for N_p is defined by the *dp_pts* variable. The integration time is calculated as $N_p\ PRI$ and saved under the variable name of *tstop* at the beginning of the *extract_v5.m* file. Knowing the integration time period and the number of pulses involved, it is possible to define a vector with the leading edge time for every pulse as:

$$t = n\,PRI \qquad \text{for } \ n = 1..N_p. \tag{4.2}$$

This time vector is defined in the simulation model as *ti* in the following code lines:

```
tstop = pri*dp_pts;                        % Pulse train duration (integration time)
ti = linspace(0,tstop,dp_pts);             % Vector with pulse leading edge times
```

The *T(r,d,t)* expression in (4.1) can be defined as *T(r,d,n)* using the *ti* vector. The *T(r,d,n)* expression contains a complex scattering matrix for each of the N_P pulses and is saved as *target*. This three-dimensional expression can be reduced to a two-dimensional matrix $T'(r,n)$, by adding the scatterers coherently in the cross-range or azimuth dimension. The two-dimensional result $T'(r,n)$ is saved as the *targetSum* matrix and represents a collection of range profiles (rows) for each pulse *n*. This matrix is equivalent to the matrix of range profiles formed after the ISAR range compression output. Thus, the process of generating $T'(r,n)$ is inverse to the ISAR azimuth compression. The calculation of *T(r,d,n)* (*target*) and $T'(r,n)$ (*targetSum*) is performed using a three loop structure as shown in the following code lines:

```
% GENERATION OF COMPLEX SCATTERER EXPRESSIONS
A = ones(numtaps,dp_pts);                  % Scatterer Amplitudes (uniform)
targetSum = zeros(numtaps,dp_pts);
for i1 = 1:numtaps,                        % At each range bin
   target = zeros(dp_pts,length(ti));
   for i2 = 1:numfreq(i1),                 % At each scatterer
      for i3 = 1:length(ti),               % At each integrated pulse
         target(i2,i3) = A(i1,i2) * exp(-j*2*pi*freq(i1,i2)*ti(1,i3));
      end                                           % Creates complex scatterer
    end
    for i4 = 1:numfreq(i1)                          % Combined signal p/rbin p/pulse
      targetSum(i1,:) = targetSum(i1,:) + target(i4,:);
    end
  end
```

where i1, i2 and i3 are loop variables and *numfreq(i1)* is the number of scatterers co-existing within the same range bin *i1*.

3. Magnitude Coefficients

The magnitudes of the array of range profiles $T'(r,n)$ are normalized and quantized into 11 levels. Each level is assigned to a gain coefficient. The quantization levels are exponentially distributed between threshold values given by:

$$Thr_s = 2^{-(11-s)} f_q \tag{4.3}$$

where s goes from 1 to 10, and f_q is an arbitrary factor between 1 and 2. The resulting quantization scheme is shown in Table 7 for an f_q value of 1.5. The threshold values slightly differ from those in the quantization example in Table 1, where they were generated with an f_q value of 1.6. The choice of the f_q value has a direct effect on the number of times that the gain coefficient has the maximum value of 1024. Therefore, this factor can be used to control the number of overflow occurrences in the DIS output adder. For example, if an f_q of 1.8 is used, the gain coefficient of 1024 will be only assigned to the scatterer locations where the normalized magnitude of $T'(r,n)$ has a value from 0.9 to 1.0. Thus, the probability of obtaining an overflow is greatly reduced.

Magnitude Range	Number of Shifts *g(r,n)*	Effective Gain $2^{g(r,n)}$
0.75 to 1.0	10	1024
0.375 to 0.75	9	512
0.1875 to 0.375	8	256
0.0938 to 0.1875	7	128
0.0469 to 0.0938	6	64
0.0234 to 0.0469	5	32
0.0117 to 0.0234	4	16
0.0059 to 0.0117	3	8
0.0029 to 0.0059	2	4
0.0015 to 0.0029	1	2
0 to 0.0015	0	1

Table 7 Quantization Scheme for DIS Magnitude Coefficients.

The gain quantization scheme described in Table 7 is implemented in the simulation program using the following code lines:

```
% GENERATION OF GAIN MODULATION COEFFICIENTS
amp = abs(targetSum);                          % Magnitude of combined signal.
gain = ones(numtaps,dp_pts);                   % Default gain coefficient is 1.
fq = 1.5;                                      % Arbitrary threshold factor
for i1 = 1:numtaps,
   for i2 = 1:dp_pts,
      normAmp = amp(i1,i2)/max(max(amp));      % Magnitudes are normalized.
      for s=1:10,
           if normAmp > (2^-(11-s))*fq,        % Thresh: 0.0015....,0.375,0.75
               gain(i1,i2) = 2^s;              % Gain:   2,4,8,16,......512,1024
         end                                   % Assigns 11 power-of-two gains.
      end
   end
end
```

Figure 20a shows the resulting amplitude distribution of gain coefficients for a test target of 32 range cells with an integration of 128 pulses. To prove that the exponential assignment of gain coefficients is evenly generated, a logarithmic plot of the gain coefficient distribution is shown in Figure 20b.

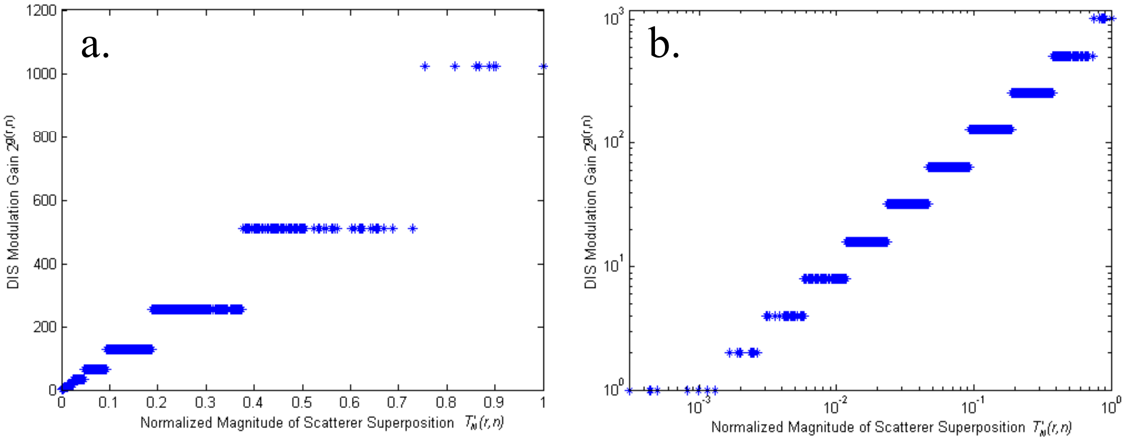


Figure 20 Quantization Scheme for Gain Coefficients of a 32-Range-bin Target.

The plot of gain coefficients can also be rearranged with respect to their range cell/pulse location. The resulting plot is shown in Figure 21. This plot shows occurrences of the highest gain coefficient of 1024 at the combination of pulses 1, 2, 63, 64, 65 and 66, and range cells 9 and 25 corresponding to the target mast locations in Figure 19. These peak gain locations are potential sources of overflow in the output adder bit format. In addition, the figure shows that the distribution of gain coefficients is symmetric between pulses 1 through 64 and 65 through 128.

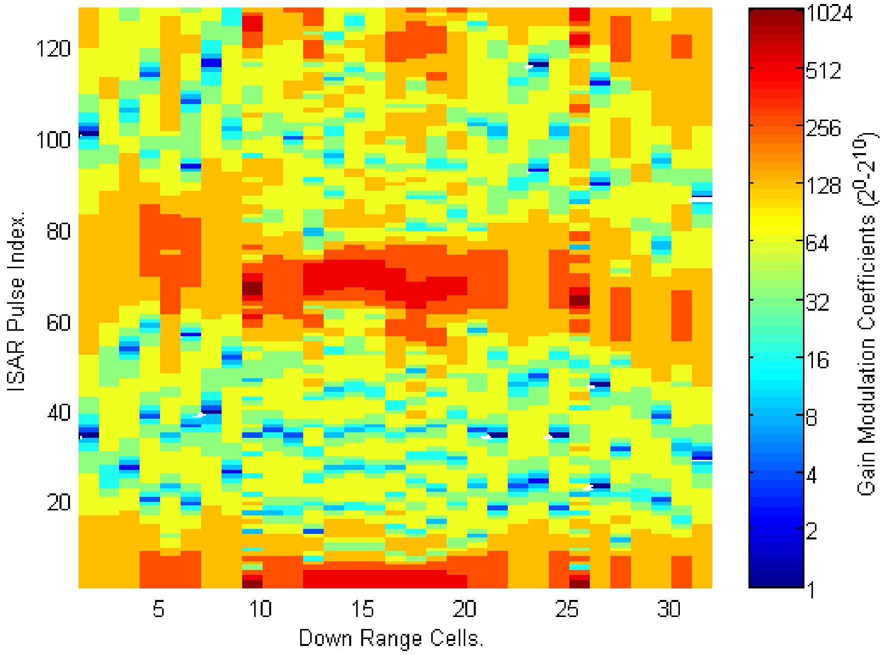


Figure 21 Gain Coefficient Distribution on a Range Cell-Pulse Index Map.

4. Phase Coefficients

The phase of the array of range profiles is extracted by applying the *angle()* function to the *targetSum* array. The resulting phase values are then converted into a scale from 0 to 31.99. For each scatterer phase, a phase increment is calculated with respect to the scatterer of the previous Doppler location. The first Doppler location is assigned a phase value of zero. At each scatterer location, the cumulative sum of phase increments results in a relative phase value with respect to the first Doppler location. These relative phase values are stored in a matrix labeled as *phaseinc*. The DIS phase modulation coefficients are obtained from the quantization of the values of the *phaseinc* matrix into 32 integer phase values from 0 to 31. These coefficients are stored in a matrix labeled *phasecoeff*. The calculation of the *phasecoeff* phase coefficient matrix is implemented is the following code lines:

```
% GENERATION OF PHASE MODULATION COEFFICIENTS
vphase1  = angle(targetSum);                      % Phase of combined signal
vphase2  = vphase1*2^nbitsdop/(2*pi);             % Adjusts values from 0 to 31.99
phaseinc = zeros(numtaps,dp_pts);
```

```
for i6 = 1:numtaps,                              % Turn phases into phase increments
    for i7 = 1:dp_pts,
        if i7==1,
            phaseinc(i6,i7) = phaseinc(i6,i7)+vphase2(i6,i7);
        else                                     % Finds phase increments bet/pulses
            phaseinc(i6,i7) = phaseinc(i6,i7-1)+vphase2(i6,i7-1)-vphase2(i6,i7);
        end
    end
end
for i8 = 1:dp_pts,
    for i9 = 1:numtaps,                          % 5-bit quantization in 32 levels.
        phasecoeff(i9,i8) = fix(rem(phaseinc(i9,i8)+32,32));
    end
end
```

Similar to Figure 21, the phase modulation coefficients can be plotted with respect to their range cell/pulse location. The resulting plot is shown in Figure 22. This distribution shows that the phase variation becomes faster for higher range bins. In addition, there is a pseudo symmetry between the phase coefficients from pulses 1 through 64 and pulses 65 through 128.

5. Storage of Coefficients

The gain and phase coefficient values are stored in a text file labeled as *paraMULTIq4Ship.txt* after a header of three identification values. The first three stored quantities are the number of intrapulse samples (500 for the default values of 0.5 us pulsewidth and 1 GHz sampling frequency), the number of Doppler cells (default is 128) and the target extent in range cells (default is 32). For the default case of 32 range bins and 128 Doppler cells, the total number of gain coefficients is 4096. For the case of these default settings, the gain modulation coefficients are stored from the 4th location to the 4099th location. The phase modulation coefficients are stored from the 4100th location to the 8195th location. The storage of the gain and phase modulation coefficients in the *paraMULTIq4Ship.txt* file is implemented throughout the following code lines:

```
% STORAGE OF MODULATION COEFFICIENTS
f4 = fopen('paraMULTIq4Ship.txt','w');           % Open file to store coefficients
fprintf(f4,'%d\r\n',rag_pts);                    % Stores # of chirp pulse samples
fprintf(f4,'%d\r\n',dp_pts);                     % Stores # of Doppler bins
fprintf(f4,'%d\r\n',numtaps);                    % Stores # of tap lines (target extent)
for aa=1:dp_pts,
   for bb=1:numtaps,
      fprintf(f4,'%d\r\n',gain(bb,aa));          % Stores gain coefficients
   end
end
for aa = 1:dp_pts,
   for bb = 1:numtaps,
      fprintf(f4,'%d\r\n',phasecoeff(bb,aa));          % Stores phase coefficients
   end
end
fclose(f4);
```

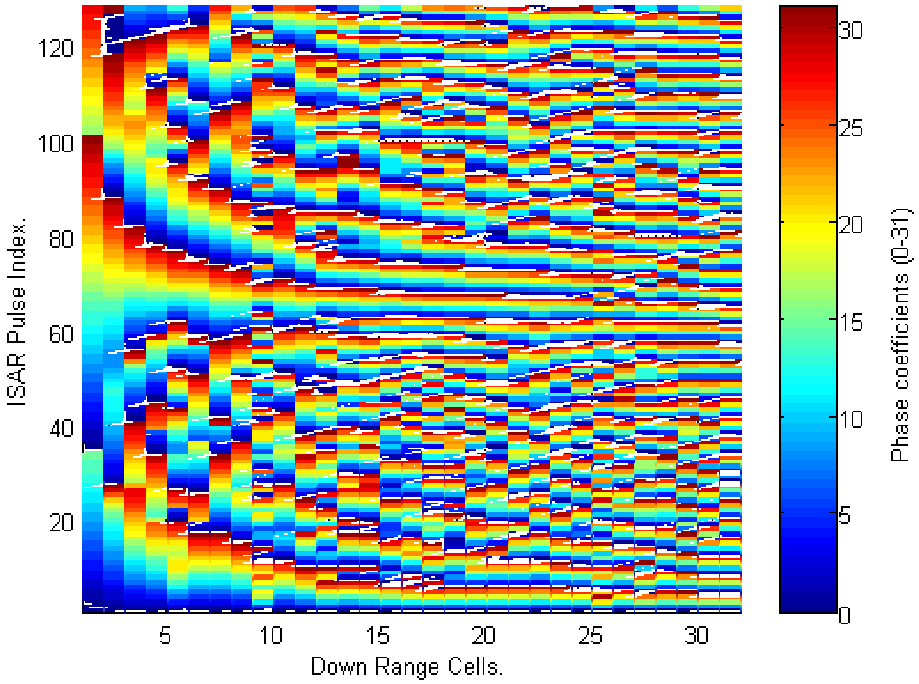


Figure 22   Phase Coefficient Distribution on a Range Cell-Pulse Index Map.

B. SIMULATION OF DIS INPUT AND ISAR PULSE COMPRESSION REFERENCE SIGNALS

1. DIS Input Signal

The DIS input signal is a digitized version of the ISAR chirp pulse that is used as the main input for the DIS. Each pulse is fed into the DIS as a batch of DRFM phase samples. Due to the squared time term in the exponent of the chirp phase in (3.11), as time increases the phase values become very large rotated values. If the Doppler effect is ignored as stated in that equation, the chirp phase as a function of time can be represented as:

$$\phi_o = 2\pi k \left({t^2}/{2} \right) = \pi k t \qquad (4.4)$$

where k is the compression factor equivalent to the ratio of the bandwidth Δ over the pulsewidth τ. The resulting phase-value samples for each pulse are stored in a vector labeled as *oldphase*. These rotated values converted to equivalent phase values within the 0 to 360 degrees interval and then quantized into 32 integer phase levels from 0 to 31. The quantized values correspond to the 5-bit representation of the phase sampling DRFM output. The quantized phase samples for all the integrated pulses are stored in a matrix labeled as *pri_rg_phaseq*. These DIS phase inputs are arranged by rows for each different radar pulse. A baseband digital representation of a chirp pulse can be obtained by plotting one row of the *pri_rg_phaseq* matrix. The resulting chirp waveform is plotted in Figure 23.

The reference signal that is used at the receiver to perform the range compression of each returned pulse can be obtained for the *oldphase* vectors. These vectors are stored as columns of a matrix named *cref*. The process used to generate the *pri_rg_phaseq* DIS input matrix and the *cref* range compression reference matrix is implemented with the following code lines:

```
% GENERATION OF DRFM PHASE SAMPLES
pri_rg_phaseq = zeros(dp_pts,rag_pts);          % Initialize DRFM phase matrix
p = 2*pi/(2^nbitsdop);                          % Phase quantization factor
pri = 1/prf;
for idx1 = 1:dp_pts,                            % Loop per pulse
  t1 = t0 + idx1*pri;                           % PRI time shift for all samples
  oldphase = 2*pi*(k*t1.*t1)/2;                 % Chirp pulse phase samples
  oldphase = rem(oldphase,2*pi);                % limit phase from 0 to 2pi
  pri_rg_phaseq(idx1,:) =fix(oldphase/p);       % Phase quantized in 2^n levels
  crefc(:,idx1) = exp(j*oldphase.');            % ISAR reference
end
```

where *nbitsdop* has a value of 5 to obtain 2^5 phase levels.

2. Storing of DRFM Samples and Reference Signal

The phase samples contained in the *pri_rg_phaseq* matrix are stored row by row in the *rawint.txt* file. Each row of this text file corresponds to the phase samples of one radar pulse. The last element of the matrix is stored followed by a carrier return character to skip to the next line of the data file. The code lines involved are listed below:

```
% STORAGE OF DRFM PHASE MATRIX :
f2 = fopen('rawint.txt','w');                   % Opens file for phase storage
for i = 1:dp_pts,                               % Loop for all pulses.
  int_raw = pri_rg_phaseq(i,1:rag_pts-1);
  fprintf(f2,'%d,',int_raw);                    % Stores pulse phase samples
  int_raw = fpri_rg_phaseq(i,rag_pts);
  fprintf(f2,'%d\r\n',int_raw);                 % Store last sample of the pulse
end;
fclose(f2);
```

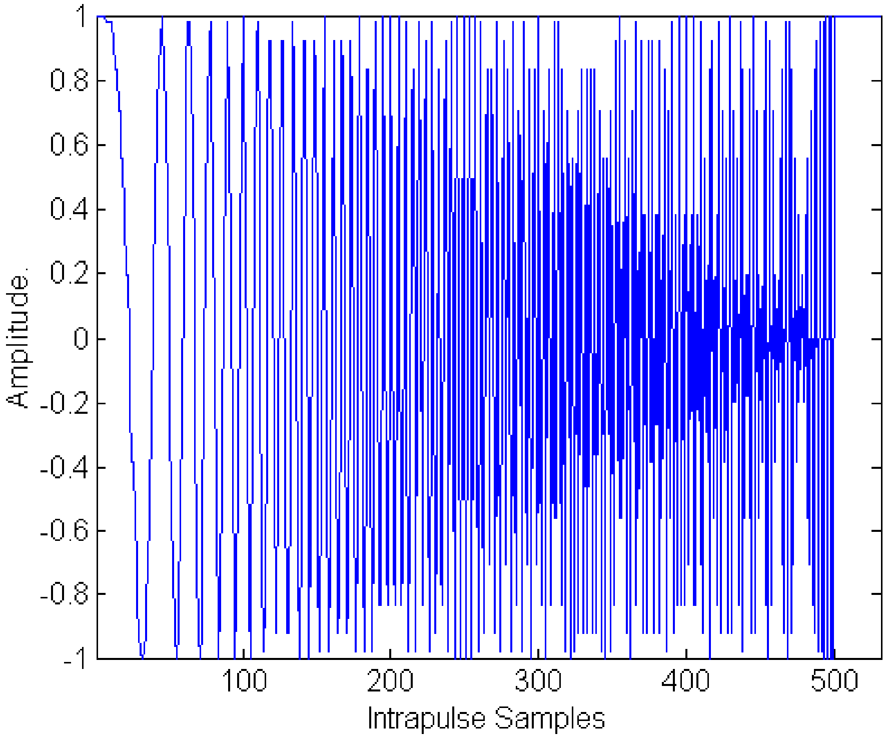


Figure 23 Baseband Plot of the Intercepted ISAR Pulse as Generated by a DRFM.

The coherent reference for each pulse needs to be stored in order to be available for the *simhwchk_v5.m*. This file performs, among other things, the cross-correlation to simulate the ISAR range compression. Since the cross-correlation uses the complex conjugate of the Fourier transform of the reference (Figure 6), the complex conjugation and the transform process is performed before storing the signals. To approximate the ideal infinite signal condition, the FFT is performed using $2\,rg\_pts-1$ points, where *rg_pts* is set arbitrarily as the maximum between the number of samples within a radar pulse and ten times the target extent in range bins. The FFT of the reference is then complex conjugated and stored in a matrix labeled as *crefc,* where each column corresponds to the conjugated FFT vector of a pulse reference.

```
% GENERATION OF ISAR REF. FOR RANGE COMPRESSION - FAST CORRELATION
cref = conj(fft(crefc,2*rg_pts-1));                    % FFT of ISAR reference
save pc_ref cref t1                                    % Save FFT of ISAR reference
```

3. Execution of Follow-on Routines

At the end of the *mathost_v5.m* file, there are instructions to execute the *simhwchk_v5.m* and the *Image_Analysis.m*. The first routine emulates the DIS and ISAR compression. The second routine is an image quality evaluation performed by comparison between the current output image and an ideal infinite resolution quantization image. This analysis is only executed for cases where the target has a size of 32 range bins and the number of Doppler resolution cells is 128, as the ideal image is made available only for that case. The code lines involved in this procedure are listed below:

```
% DIS SIMULATION and IMAGE GENERATION AN ANALYSIS
simhwchk_v5;                                                    % DIS & ISAR processing
if numtaps==32 & nDopplerCell==128 & prf==200,
   Image_Analysis;                                              % Comparison with ideal image
end;
```

C. SIMULATION OF DIS AND ISAR SIGNAL PROCESSING

The DIS and ISAR simulations are implemented by the *simhwchk_v5.m* file. The input parameters for the execution of this file are the DRFM phase samples from the ISAR intercepted pulses and the DIS modulation coefficients. The DIS processes the ISAR pulses sequentially. Within each pulse, the phase samples are also processed in the same manner. The simulation model emulates the DIS using a loop structure that is repeated for each received pulse. Within that loop, an additional looped process is repeated for each intrapulse sample. The DIS hardware processes each intrapulse sample simultaneously (in parallel) at all range bins. Similarly, the simulation model emulates this process using vectors with one element at each range tap. The samples at the range-bin outputs are properly delayed and added to build the DIS output modulated pulses. The ISAR simulation is modeled with a range compression of the DIS output pulses, followed by an azimuth compression. The end result is a matrix with intensity values representing the target range Doppler image. A flow diagram summarizing the processes that are modeled by the *simhwchk_v5.m* file is shown in Figure 24.

1. Loading of Modulation Coefficients

The modulation coefficients are loaded from the *paraMULTIq4Ship.txt* file. This file also contains the values for the number of samples in the ISAR chirp pulse, the number of pulses integrated by the radar and the target extent in range bins. The number of pulse samples is saved under the variable name *nRangeCell* and is used to control the total number of times the inner loop is set for each pulse. The number of pulses integrated, which also corresponds to the number of Doppler resolution cells, is saved under the variable name *nDopplercell* and is used to control the number of times the outer loop is used. The target extent is saved under the variable name *numtaps* and determines the length of the vectors that are used to simulate the parallel process at the range bins.

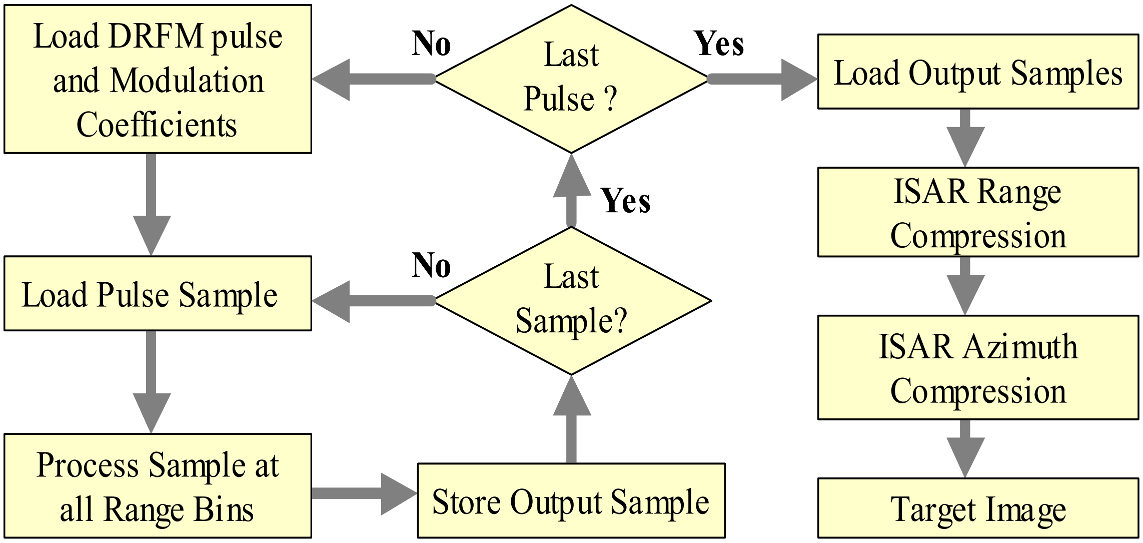


Figure 24 Summarized Flow Diagram of the *simhwchk_v5.m* File.

After the first three values are loaded from the *paraMULTIq4Ship.txt* file, the gain coefficients can be found starting from the fourth value contained in the file. For the case of a 32-bin target and 128 integrated pulses, the gain coefficients can be extracted by reading the values from location 4 through 4099. These coefficients are stored in a 4096-element vector named *gainIn*. The elements of this vector are rearranged into a 128 by 32 element matrix labeled as *gain*. The rows of this matrix contain coefficients belonging to the same pulses, while the columns represent same range bins. The phase modulation coefficients can be read immediately after the last gain coefficient, from location 5000 through 8195. These coefficients are stored in a 4096-element vector named *phi*. Similarly to the gain coefficient case, the *phi* phase coefficient vector is rearranged into a 128 by 32 element matrix labeled as *phasInc*, where coefficients from the same pulse can be found in the same rows. The loading of modulation coefficients is implemented with the following code lines:

```
% LOADING OF MODULATION COEFFICIENTS
fid = fopen('paraMULTIq4Ship.txt','r');           % Opens file w/DIS coefficients
tmp = fscanf(fid,'%f');                           % Loads file contents
nRangeCell = tmp(1);                              % number of pulse samples
nDopplerCell = tmp(2);                            % number of pulses integrated
numtaps = tmp(3);                                 % number of DIS range taps
gainIn = tmp(4:4+numtaps*nDopplerCell-1);         % Extracts gain coefficients
gain = reshape(gainIn,numtaps,nDopplerCell);      % Arranges them in a matrix
gain = gain';                                     % Transpose
phi = tmp(4+numtaps*nDopplerCell:end);            % Extracts gain coefficients
phasInc = reshape(phi,numtaps,nDopplerCell);      % Arranges them in a matrix
fclose(fid);
```

2. Loading of DRFM Phase Signal

The simulated DRFM phase samples are stored in the *rawint.txt* file. The file has the values arranged in rows and separated by commas, with each row representing a different pulse. Once the phase samples are loaded, they are stored in a matrix labeled as *raw*, where each row also represents a different pulse. The loading of the DRFM signal is obtained with the following code lines:

```
% LOADING OF DRFM SAMPLES
raw = zeros(nDopplerCell,nRangeCell);             % Initialize DRFM phase matrix
fid = fopen('rawint.txt','r');                    % Opens file w/DRFM samples
for r = 1:nDopplerCell,                           % Loop per pulse
   for i2 = 1:nRangeCell-1,                       % Loop per sample
      raw(r,i2)=fscanf(fid,'%f',1);               % Loads DRFM phase sample
      comma = fscanf(fid,'%c',1);                 % Loads comma character
   end
   raw(r,nRangeCell) = fscanf(fid,'%f',1);        % Last pulse phase sample
end
fclose(fid);
```

In order to reset the range-bin inputs after the last DRFM sample has been fed to the DIS, the DRFM phase samples are appended with zeros at the end of each pulse. This is accomplished by concatenating a matrix with zeros to the right of the *raw* matrix. The resulting matrix is labeled as *phaDRFM*. The code lines used for the concatenation of zeros are shown below:

```
[row,col] = size(raw);
phaDRFM = [raw,zeros(row,numtaps-1)];             % Trailing zeros to clear taps
```

3. Phase Modulation

The two loops are implemented to iterate the DIS process for each pulse and for each sample within a pulse. The first step that takes place within the loops is the phase addition of DRFM phase samples with phase coefficients. The phase samples are fed in parallel to all range bins. Thus, it is necessary to transform each phase value into a vector of equal value with the size equal to the total number of range bins. The resulting vector is labeled as *tapIn*. The phase coefficients are different for each range bin and might be changed each time the arriving samples correspond to a new radar pulse. To produce the phase modulation, the *tapIn* vector containing repetitions of the DRFM sample is added with a row of the *phasInc* phase coefficient matrix. The resulting vector is labeled *phAddout* and contains the modulated phase for all range bins. These phase values are later used to access the LUT. The code lines involved in the DIS phase modulation are shown below:

```
% DIS PHASE MODULATION
 tapIn(1:numtaps) = phaDRFM(batchCnt,inPlsCnt);        % Loads DRFM Phase in
 phAddOut= tapIn(1:numtaps)'+phasInc(:,batchCnt);      % Phase adder output
```

4. Look-up Table (LUT)

The output of the phase modulation represents the phase value of a unitary magnitude complex expression. The sine and cosine LUTs convert this complex expression into in-phase and quadrature phase amplitudes. For the simulation model, 8-bit resolution LUTs are stored in vector form using two text files labeled as *cosine8.txt* and *sine8.tx*t. Additional text files were also created to represent LUTs with lower bit resolutions. The contents of the *cosine8.txt* and *sine8.tx*t LUT text files are presented in Table 8. These LUT files are loaded using the following code lines.

```
load -ascii cosine8.txt                                % Loads LUT tables
load -ascii sine8.txt
```

Once the LUT files are loaded, their contents are available as two 32-element vectors labeled respectively as *cosine8* and *sine8*. Each element of these vectors represents one possible LUT output. The LUT input phase values are integer numbers from 0 to 31. the output I and Q values are contained in vectors with indices from 1 to 32. Therefore, the downloading of a specific LUT output can be implemented with the LUT vector name with an index value equivalent to the *phAddout* modulated phase added by one as in

Equation (3.15). The output value is calculated simultaneously for all range bins and is stored in a column vector labeled as *lutOut*. The calculation of the LUT output is obtained using the following code lines:

```
tmp = rem(phAddOut+depthLUT,depthLUT)+1;          % Phase limit bet 0-31
lutOut = cosine8(tmp) + j*sine8(tmp);             % LUT table output
```

	cosine8.txt	**Cosine ROM**	**sine8.txt**	**Sine ROM**
1	0.9922	01111111	0	00000000
2	0.9766	01111101	0.1953	00011001
3	0.9141	01110101	0.3828	00110001
4	0.8281	01101010	0.5547	01000111
5	0.7031	01011010	0.7031	01011010
6	0.5547	01000111	0.8281	01101010
7	0.3828	00110001	0.9141	01110101
8	0.1953	00011001	0.9766	01111101
9	0	00000000	0.9922	01111111
10	−0.1953	11100111	0.9766	01111101
11	−0.3828	11001111	0.9141	01110101
12	−0.5547	10111001	0.8281	01101010
13	−0.7031	10100110	0.7031	01011010
14	−0.8281	10010110	0.5547	01000111
15	−0.9141	10001011	0.3828	00110001
16	−0.9766	10000011	0.1953	00011001
17	−0.9922	10000001	0	00000000
18	−0.9766	10000011	−0.1953	11100111
19	−0.9141	10001011	−0.3828	11001111
20	−0.8281	10010110	−0.5547	10111001
21	−0.7031	10100110	−0.7031	10100110
22	−0.5547	10111001	−0.8281	10010110
23	−0.3828	11001111	−0.9141	10001011
24	−0.1953	11100111	−0.9766	10000011
25	0	00000000	−0.9922	10000001
26	0.1953	00011001	−0.9766	10000011
27	0.3828	00110001	−0.9141	10001011
28	0.5547	01000111	−0.8281	10010110
29	0.7031	01011010	−0.7031	10100110
30	0.8281	01101010	−0.5547	10111001
31	0.9141	01110101	−0.3828	11001111
32	0.9766	01111101	−0.1953	11100111

Table 8 Contents of the 8-Bit Resolution LUT Files and LUT ROM.

5. Gain Modulation

The I and Q LUT outputs for all range bins are stored in the *lutOut* complex vector. The gain modulation is performed by multiplying these I and Q values by the gain coefficient for the corresponding range-bin/pulse combination. For the case of the n^{th} pulse, the gain modulation is implemented with the element-by-element multiplication of the *lutOut* vector with the n^{th} row from the *gain* matrix. The modulated amplitudes of the

I and Q channels are stored in a complex vector labeled as *gainOut*. The following code line performs the gain modulation:

```
gainOut = gain(batchCnt,:)'.*lutOut;                    % Gain modulation
```

Once the intrapulse sample has been modulated by the DIS, the range-bin outputs must maintain an output equal to zero. This shutting down of range-bin taps is accomplished in the hardware design using the control signals described in Chapter III. The simulation model implements this action by setting the value of the *gainOut* vector to zero after the last pulse sample has been processed. The following code lines perform the shutting down of range-bin taps:

```
if inPlsCnt > nRangeCell,                     % Shuts down taps
    gain0ut(1:(inPlsCnt-nRangeCell)) = 0;     % after pulse passes
end
```

6. Output Adder and Overflow Analysis

The chain of output adders described in Figure 17 perform a cumulative sum of the I and Q range-bin outputs, applying one sample delay between them. The I and Q range-bin output samples represented by the *gainOut* complex vector are collected as rows in a matrix labeled as *tapOut*. Hence, the rows of this matrix represent outputs from the same DRFM samples and the columns represent outputs from the same range bins. To implement one sample delay on an output element, the row location in the *tapOut* matrix must be increased by one. Similarly, all elements loaded in the *tapOut* matrix are delayed by shifting a specific amount of rows depending on their range-bin number. In other words, the range-bin output *gainOut* vectors are loaded in the *tapOut* matrix as diagonals.

The amplitude of the I and Q range-bin outputs are represented by the bit format shown in Table 5. This bit format limits the minimum resolution to the value of the LSB. The DIS simulation emulates this resolution limit by truncating the I and Q channel amplitude defined in (3.19) as:

$$S'(r,m,n)=\left\lfloor \frac{S(r,m,n)}{\mathrm{LSB}\big(S(r,m,n)\big)} \right\rfloor \mathrm{LSB}\big(S(r,m,n)\big) \tag{4.5}$$

where the LSB function is equal to the value of the least significant bit in the binary format of the argument. The simulation defines an I and Q bit format LSB as 0.25, storing it under the variable name of *o_res*. The code used to implement the delay between range-bin outputs and the truncation for the bit format resolution is listed below:

```
% OUTPUT ADDER DELAY AND MINIMUM RESOLUTION
for idx3=1:numtaps,
   delay = idx3-1;                                % Out.adder register delay bet/taps
   tapOut(inPlsCnt+delay,idx3)=fix(gainOut(idx3)/o_res)*o_res;
end                                               % Applies minimum register resolution
```

The sum of range-bin outputs can be performed by adding the row elements from the *tapOut* matrix. This complex value result is equivalent to the I and Q components of one sample of one DIS output pulse. As described in Figure 17, the sum between delayed range-bin outputs is implemented serially. The output of each sum is represented by the bit format of Table 6. Thus, it is only able to represent amplitude values limited by:

$$I\left(m,n\right)_{Lower\_limit} = -2\,\mathrm{MSB}\left(I\left(m,n\right)\right) \tag{4.6}$$

$$I\left(m,n\right)_{Upper\_limit} = 2\,\mathrm{MSB}\left(I\left(m,n\right)\right) - \mathrm{LSB}\left(I\left(m,n\right)\right) \tag{4.7}$$

where the MSB function is equal to the value of the most significant bit in the binary format of the argument.

Whenever an I or Q sum output is forced to represent a value outside these limits there is an overflow in the adder format. The main effect of the format overflow is that the MSB produces a carry that toggles the next bit, thus switching the sign of the two's complement bit format. The resulting erroneous outputs for cases where the lower or upper limits are surpassed are given by:

$$I\left(m,n\right) = I\left(m,n\right)_{Lower\_overflow} + 2\,\mathrm{MSB}\left(I\left(m,n\right)\right) \tag{4.8}$$

$$I\left(m,n\right) = I\left(m,n\right)_{Upper\_overflow} - 2\,\mathrm{MSB}\left(I\left(m,n\right)\right). \tag{4.9}$$

The simulation model performs the cumulative sum storing the I and Q adder outputs in a complex variable named *sum1*. After each sum the real and imaginary values of the cumulative sum are stored as *Iout* and *Qout*, and checked in case they fall outside the overflow limits in (4.6) and (4.7). For the 16-bit format implemented in the DIS hardware and shown in Table 6, the limits are –8196 and +8195.75. Whenever the real and imaginary components *Iout* and *Qout* fall outside these limits, the cumulative sum represented by *sum1* is corrected to apply the erroneous overflow outputs given in (4.8) and (4.9). In addition, a counter represented by the variable name *overflow* is incremented every time an overflow occurs. Once the cumulative sum has been completed, the resulting DIS output sample is stored in a vector labeled as *partial_tapsum.* This vector stores the DIS output samples that belong to the same pulse. Thus, it stores one complete output pulse. The code lines that implemented the range-bin output cumulative sum and overflow simulation for each sample are listed below:

```
% OUTPUT ADDER CUMULATIVE SUM AND OVERFLOW
sum1=0;
for tt=min(inPlsCnt,numtaps):-1:max(1,inPlsCnt-nRangeCell),
    sum1 = sum1 + tapOut(inPlsCnt,tt);
    Iout=real(sum1);                              % Overflow simulation using +/-limit
    Qout=imag(sum1);
    if Iout >= limit,                             % Positive overflow (in-phase)
        sum1 = Iout-2*limit + j*Qout;             % Adder output w/overflow
        overflow = overflow + 1;                  % Increases counter
    elseif Iout < -limit,                         % Negative overflow (quad-phase)
        sum1 = Iout+2*limit + j*Qout;             % Adder output w/overflow
        overflow = overflow + 1;                  % Increases counter
    end
    if Qout >= limit,                             % Positive overflow (in-phase)
        sum1 = Iout + j*(Qout-2*limit);           % Adder output w/overflow
        overflow = overflow + 1;                  % Increases counter
    elseif Qout < -limit,                         % Negative overflow (quad-phase)
        sum1 = Iout + j*(Qout+2*limit);           % Adder output w/overflow
        overflow = overflow + 1;                  % Increases counter
    end
    peak=max(max(abs(Iout),abs(Qout)),peak);      % End of overflow simulation.
    partial_tapsum(inPlsCnt,1) = sum1;
 end
```

7. DIS Output

After the last sample of the pulse is processed, the output pulse represented by the samples contained in the *partial_tapsum* vector is stored as a row in a matrix labeled as *finalAdderOut.* The resulting matrix contains the complete DIS output complex signal with each row representing one of the modulated pulses. Figure 25 displays a plot of one DIS output pulse obtained from the simulation model. The figure is obtained by plotting the first row of the *finalAdderOut* matrix (first output pulse).

The DIS output I and Q channels represented by the *finalAdderOut* complex matrix are fed to I and Q DACs. These converters should ideally have the same bit resolution as the DIS outputs to avoid information losses. In addition, the DAC speed should be high enough to keep up with the required clock frequency. For cases where a DAC has less bits that the 16-bit DIS output, only the most significant DIS output bits are connected. The exclusion of output bits produces a truncation that is dependent on the value of the DIS output bit position that is connected to the DAC LSB, such that:

$$I'(m,n) = \left\lfloor \frac{I(m,n)}{\mathrm{LSB}(DAC)} \right\rfloor \mathrm{LSB}(DAC). \tag{4.10}$$

The value of the DIS bit connected to the DAC LSB depends on the number of DAC bits N_B and is given by:

$$\mathrm{LSB}(DAC) = 2^{14-N_B}. \tag{4.11}$$

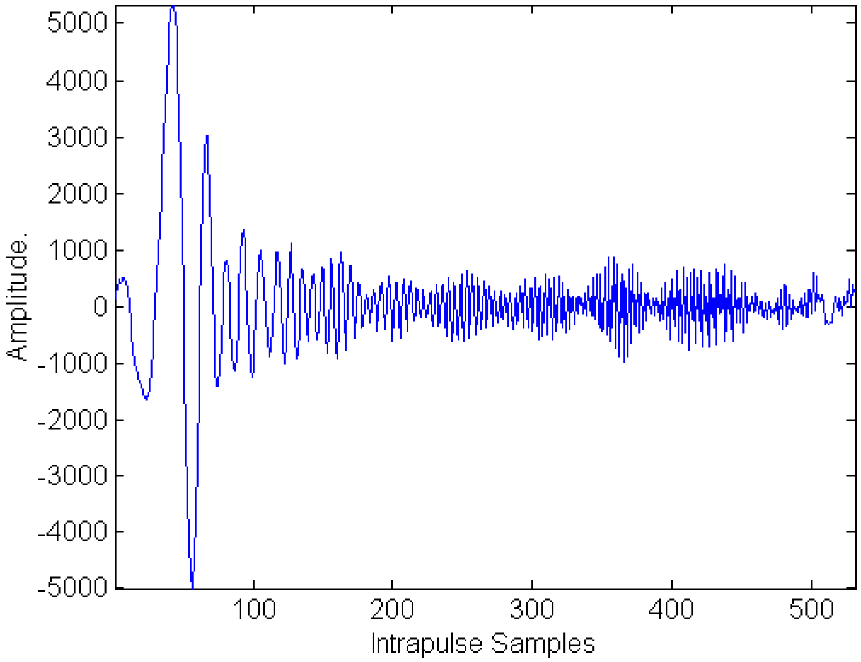


Figure 25 Plot of the DIS Output Modulated Chirp

The DIS model simulates the effect of feeding the DIS output channels through DACs of different bit resolutions using Equation (4.10) and defining the LSB of the DAC as the variable name *dac_res*. The following code lines show the implementation of the DAC loss:

```
% DIS OUTPUT GENERATION AND DAC
finalAdderOut(batchCnt,:)=partial_tapsum.';          % DIS output mod. chirps
dac_res = 0.25;                                      % DAC resolution 0.25 for 16bit
finalAdderOut = fix(finalAdderOut/dac_res)*dac_res;
figure
plot( real( finalAdderOut(1,:) )),axis tight         % Plots 1 DIS output chirp pulse
title('Plot of the DIS output chirp pulse');
```

8. ISAR Range Compression

The ISAR range compression is modeled in a pulse-by-pulse basis as the cross-correlation between a DIS output pulse and the reference of the transmitted pulse. This cross-correlation is performed with the product of the FFT of both signals. The FFT of the transmitted pulse reference for each pulse is obtained from the *pc_ref.mat* Matlab data file that is created during the execution of the *mathost_v5.m* file. A plot of this FFT signal for one pulse is shown in Figure 26. The FFT of a DIS output pulse is obtained by calculating the FFT of the respective row of the *finalAdderOut* matrix. A plot of this FFT signal is displayed in Figure 27. Both FFT output vectors are multiplied element by element and saved under a temporary vector *tmp.* This Fourier product signal is shown in Figure 28.

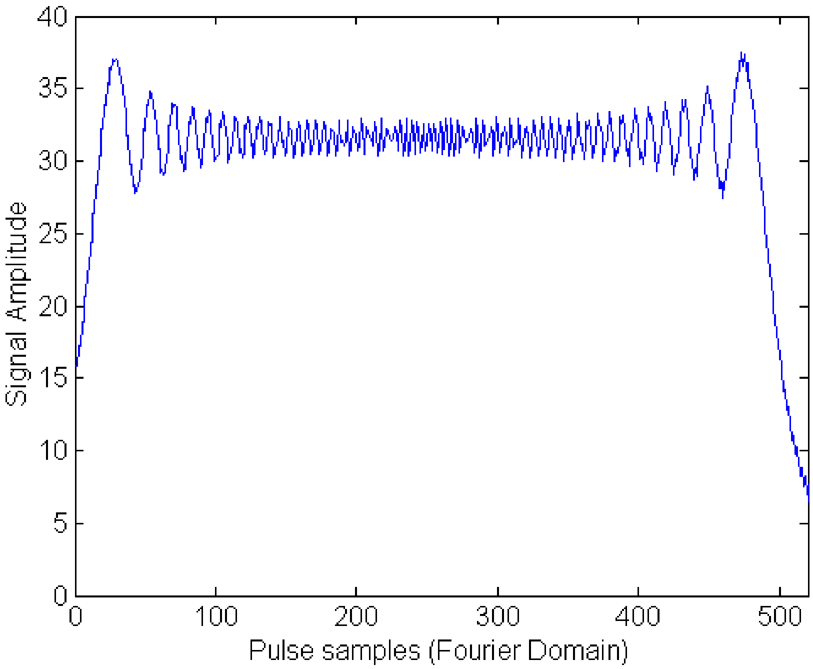


Figure 26 FFT of the Chirp Reference Signal.

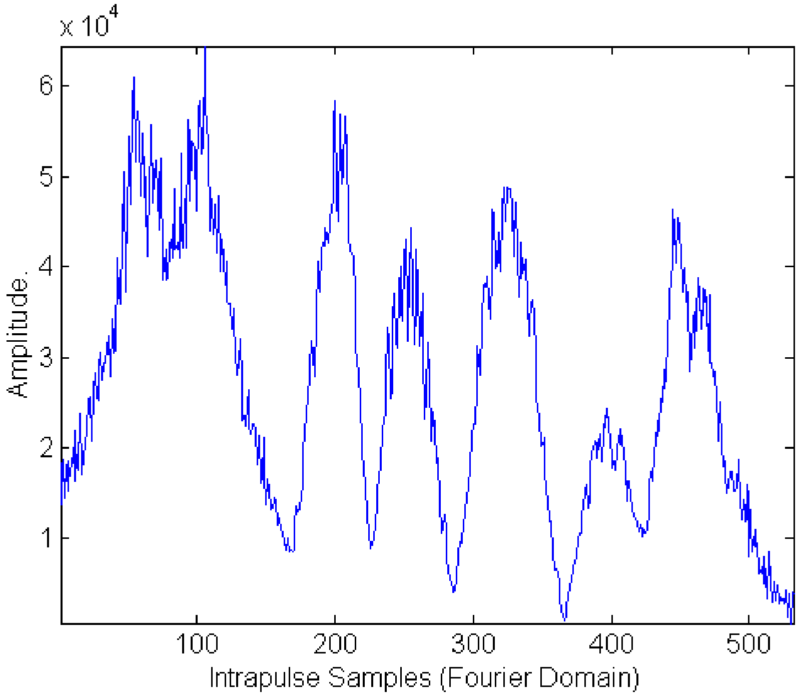


Figure 27 FFT of the DIS Output Signal.

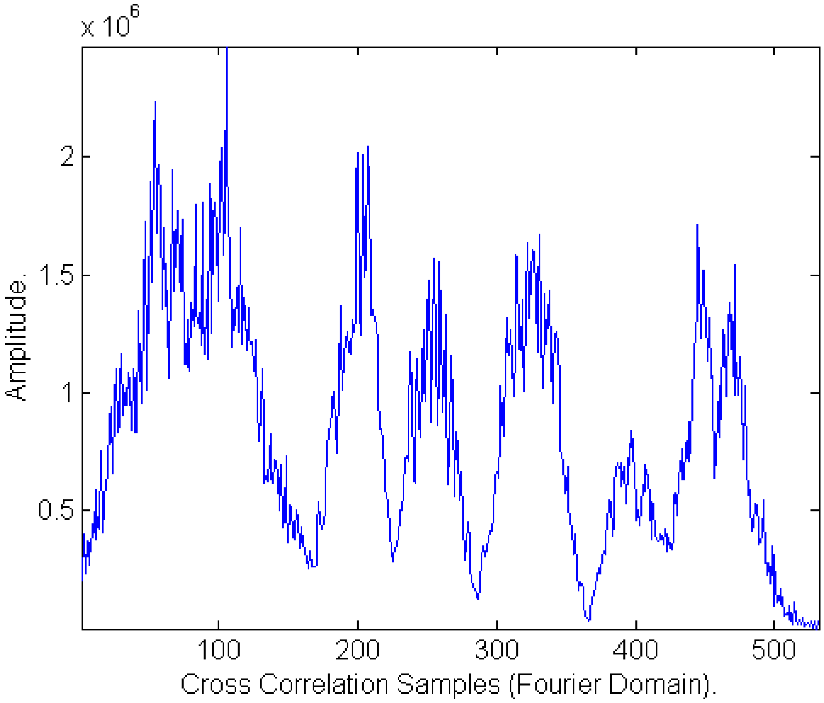


Figure 28 FFT Product Signal at the Fast Cross-correlation Output.

The output of the cross-correlation for one pulse is obtained by calculating the inverse FFT of the FFT product, as in (3.24). The inverse FFT of the *temp* vector is saved under a second temporary vector *tmp2*. This vector contains a one dimensional range profile for one returned pulse. A plot of the *tmp2* vector is presented in Figure 29, demonstrating that the target range profile can be extracted from the first N_R samples, where N_R is the target extent in range bins. The range profile vector for each pulse is extracted and saved as a row in a new matrix labeled as *priRgMapShift*.

The cross-correlation process implemented as an FFT product followed by an inverse FFT is repeated for every returned pulse using a loop structure. The end result is the *priRgMapShift* matrix containing a collection of range profiles from different pulses. The code lines used in the range compression model are listed below:

```
% ISAR PROCESSING – RANGE COMPRESSION
load pc_ref                                              % Loads FFT of ISAR reference
priRgMapShift = zeros(fix(numtaps*9/8),nDopplerCell);
   % Range compression:
pcRefMapShift = fft(finalAdderOut.',2*rg_pts-1).';       % FFT of DIS output
for idx = 1:nDopplerCell,                                % For all pulses.
    tmp  = cref(:,idx).*pcRefMapShift(idx,:).';          % Fast correlat.(FFT product)
    tmp2 = ifft(tmp);                                    % InvFFT to obtain range profile
    priRgMapShift(:,idx) = tmp2( 1:fix(numtaps*9/8) );   % Collect range profile as a row
end
```

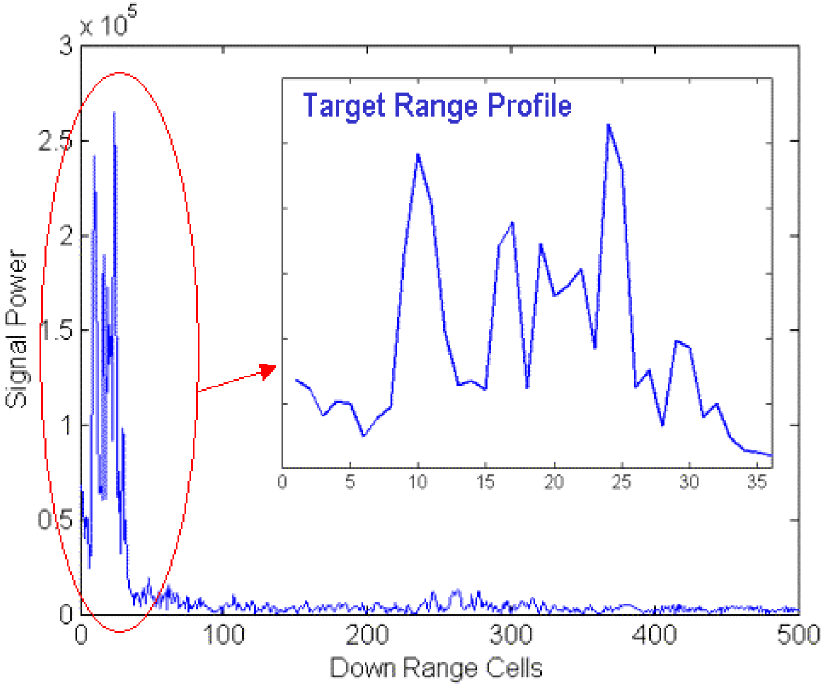


Figure 29 Fast Cross-correlation Output Including the Target Range Profile.

Figure 30 shows a collection of 16 range profiles extracted from the simulation model. They were obtained by plotting the amplitude of the first 16 rows from the *priRgMapShift* complex matrix.

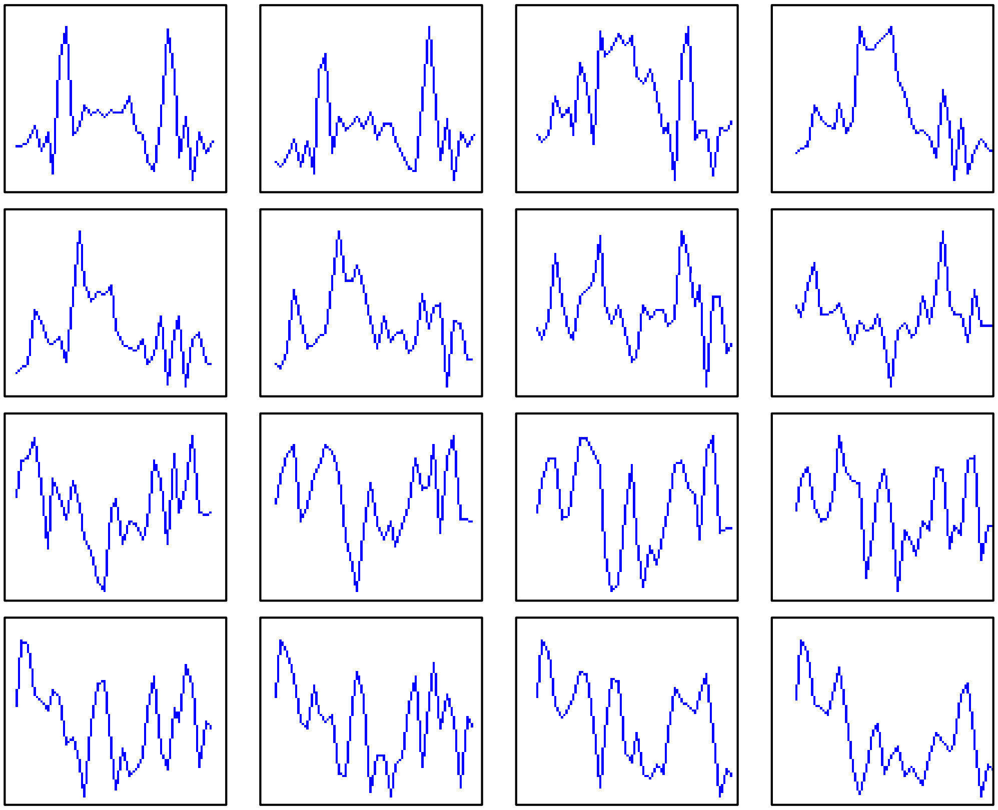


Figure 30 Range Profiles from 16 Different Pulses (Vertical Axis Not Scaled).

The magnitude of the *priRgMapShift* complex matrix can be plotted in order to view the range profiles vertically stacked. Figure 31 presents this surface plot with the intensities mapped in a color scale. Due to scaling differences, only the highest magnitude range profiles stand out in the figure.

9. ISAR Azimuth Compression and Image Generation

The *priRgMapShift* matrix contains the one-dimensional profile from each returned pulse. Thus, this matrix can be interpreted as a time history of range profiles, where time is represented by the pulse index. The azimuth compression process consists of an FFT along the time dimension in order to perform a conversion to the frequency domain. This is equivalent to an FFT of the *priRgMapShift* matrix calculated along the column dimension (pulse indices). The scatterers at each range bin are rearranged in Doppler resolution cells depending on their frequency component. The end result is a matrix whose magnitude distribution matches the range Doppler image of the false target that is loaded from the Doppler template file. The magnitude matrix is stored under the name of *dpRgMapShiftMOD*. This matrix is truncated to the first N_r columns (target extent in range cells) and N_p rows (number of integrated pulses) and saved under the matrix name *quantimage*. The code lines involved in this azimuth compression process and truncation of the final image matrix are listed below:

```
% ISAR PROCESSING – AZIMUTH COMPRESSION
dpRgMapShiftMOD = abs(fft(priRgMapShift.',nDopplerCell));
quantimage = dpRgMapShiftMOD(1:nDopplerCell*100/prf,1:fix(numtaps/8*9));
save finresoldata quantimage                                    % Saves Range Doppler profile
```

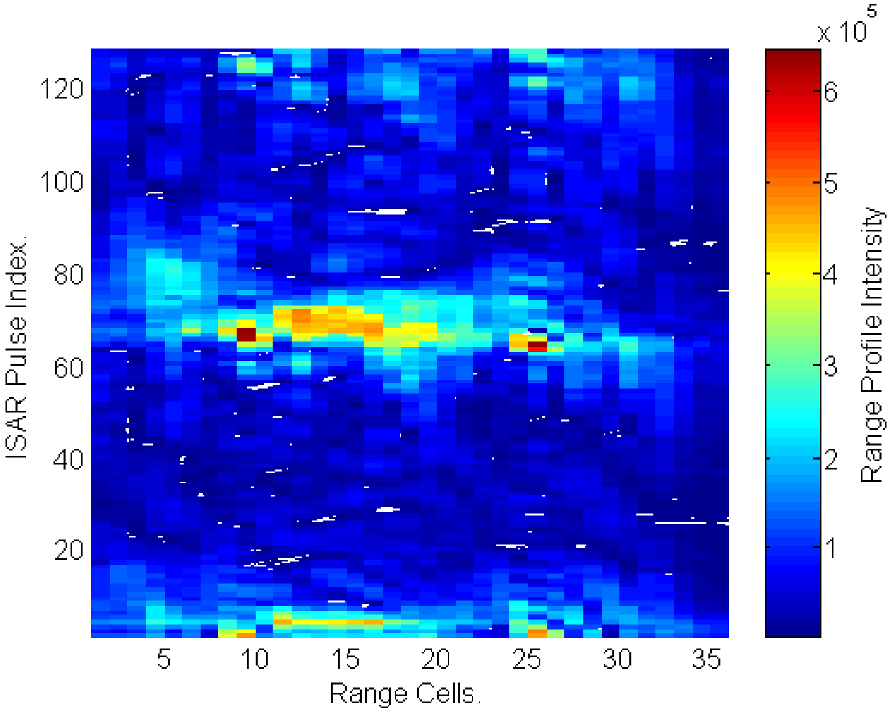


Figure 31 Surface Plot of Range Profile Matrix.

To plot the target range Doppler image using frequency values in the azimuth axis, it is necessary to relate the Doppler filter number n with the Doppler frequency f. This relation is given by:

$$f = \frac{PRF}{n}. \tag{4.12}$$

Using a Doppler filter vector from 1 to 128 and applying (4.12), a Doppler frequency vector is obtained and saved under the name *dopp*. Similarly, a range axis vector is defined from 1 to the number of columns of the *quantimage* matrix. This vector is saved under the name *rang*. The range Doppler image for a target of 32 range cells and 128 integrated pulses is presented in Figure 32 and can be compared with the intended target profile in Figure 19. The code lines used to define the axes and plot the target image are listed below:

```
% ISAR PROCESSING – IMAGE DISPLAY
dopp = (1:size(quantimage,1))*prf/nDopplerCell;          % Doppler axis
rang = 1:size(quantimage,2);                             % Range axis
colormap(hot(100))
surf(rang,dopp,quantimage), shading interp, lighting phong, view(0,90);
```

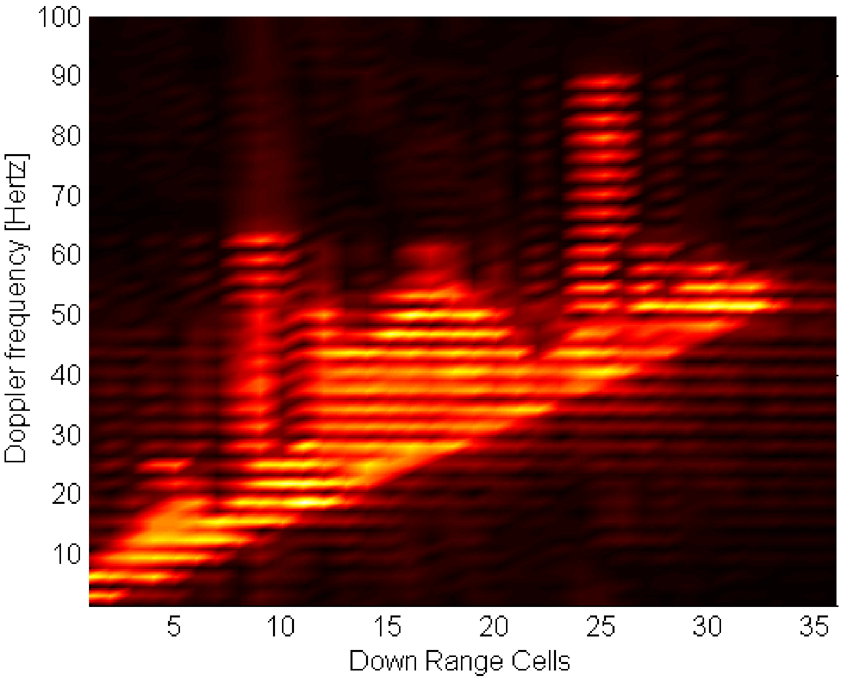


Figure 32 Range Doppler Image for a False Target of 32 Range Cells.

D. IDEAL INFINITE RESOLUTION MODEL

1. Definition

The ideal model is arbitrarily defined as the DIS model that has been modified to include the following assumptions:

- The phase coefficients are not limited to integer values (no quantization is used)

- The phase samples generated by the DRFM are not limited to integer values (no quantization is used)
- The LUT stage is replaced by a Matlab calculation of Sine and Cosine
- The gain coefficients are linearly assigned assuming that the DIS uses a gain multiplier instead of a binary shifter. Thus, coefficients are not limited to integer values (no quantization is used).
- The limiting resolution of 0.25 due to the LSB at the I and Q channel formats is non-existent
- There is no overflow at the output adder
- The DAC used at the output channels of the DIS has infinite resolution

It must be pointed out that the term infinite resolution is idealistic in the sense that the simulation is always bounded to the resolution of the MATLAB program and the computer.

2. Software Implementation

Copies of the three main simulation files are made and renamed as *extract_v5_inf_res.m, mathost_v5_inf_res.m* and *simhwchk_v5_inf_res.m.* These files are moved to a new directory. This subsection describes the code that must be applied to simulate the ideal model.

a. Phase Coefficients

The phase coefficients for the DIS model are obtained from Equation (3.5) resulting in integer values from 0 to 31. In contrast, the coefficients used in the ideal model are not rounded. Thus, the symbol $\lfloor \quad \rfloor$ is not used. The resulting phase coefficient values range from 0 to 31.999. To generate phase coefficients for the ideal model, the following loop structure from the *extract_v5_inf_res.m* file are disabled:

```
for i8 = 1:dp_pts,
        for i9 = 1:numtaps,                              % 5-bit quantization in 32 levels
                phasecoeff(i9,i8) = fix(rem(phaseinc(i9,i8)+32,32));
        end
end
```

and substituted by the code line:

```
phasecoeff = phaseinc;                                   % Added For Infinite Resolution
```

b. DRFM Phase

Similarly to the phase coefficient case, the DRFM phase samples defined in (3.14) are not rounded for the ideal case. The resulting DRFM phase sample values also range from 0 to 31.999. The code change required for the ideal model is the elimination of the *fix* rounding function from the following *mathost_v5_inf_res.m* code line:

```
pri_rg_phaseq(idx1,:) =fix(oldphase/p);               % Phase quantized in 2^n levels
```

c. Sine and Cosine LUT

The sine and cosine of the phase modulation output in the ideal model is achieved using the MATLAB *sin* and *cos* functions. The input phase values need to be previously converted from the 0 to 31.99 scale to radian values. The ideal model LUT output is obtained by enabling the *simhwchk_v5_inf_res.m* file code line:

```
lutOut=cos(phAddOut*2*pi/32)+sqrt(-1)*sin(phAddOut*2*pi/32);   % Ideal LUT
```

and disabling the code line:

```
tmp = rem(phAddOut+depthLUT,depthLUT)+1;              % Phase limited between 0-31
lutOut = cosine8(tmp) + j*sine8(tmp);                 % 8-bit LUT table output
```

d. Gain Coefficients

The gain coefficients in the ideal model are linearly assigned by multiplying the normalized magnitude by the maximum shifter gain of 1024. This ideal gain assignment can be expressed as:

$$2^{g(r,n)} = 1024\ T_N^{'}\left(r,n\right) \tag{4.13}$$

where $T_N^{'}\left(r,n\right)$ is the normalized magnitude of the superposition of scatterers in (3.6). Equation (4.13) is implemented in the *simhwchk_v5_inf_res.m* file by enabling the following code line:

```
gain = 1024*amp/max(max(amp));                        % Added for infinite resolution
```

and disabling the following loop structure:

```
gain = ones(numtaps,dp_pts);                          % Default gain coefficient is 1.
for i1 = 1:numtaps,
   for i2 = 1:dp_pts,
      normAmp = amp(i1,i2)/max(max(amp));             % Magnitudes are normalized.
      for s=1:10,
         if normAmp > (2^-(11-s))*3/2,                % Thresh: 0.0015....,0.375,0.75
            gain(i1,i2) = 2^s;                        % Gain:   2,4,8,16,......512,1024
         end                                          % Assigns 11 power-of-2 gains
      end
   end
end
```

e. I and Q Channel Resolution Phase Coefficients

The resolution of the I and Q channels in the ideal model is not restricted by a bit format. Thus, the implementation to simulate the use of an LSB must be eliminated, which is accomplished in the *simhwchk_v5_inf_res.m* file by enabling the code line:

```
tapOut(inPlsCnt+delay,idx3) = gainOut(idx3);
```

and disabling the code line:

```
tapOut(inPlsCnt+delay,idx3) = fix(gainOut(idx3)/o_res)*o_res;
```

f. Output Adder Overflow

The output adder in the ideal model has an unlimited output amplitude. Thus, the overflow simulation included in the *simhwchk_v5_inf_res.m* file must be eliminated. This code change is achieved by disabling the following code lines:

```
Iout=real(sum1);                              % Overflow simulation using +/-limit
Qout=imag(sum1);
if Iout >= limit,                             % Positive overflow (in-phase)
    sum1 = Iout-2*limit + j*Qout;             % Adder output w/overflow
    overflow = overflow + 1;                  % Increases counter
elseif Iout < -limit,                         % Negative overflow (quad-phase)
    sum1 = Iout+2*limit + j*Qout;             % Adder output w/overflow
    overflow = overflow + 1;                  % Increases counter
end
if Qout >= limit,                             % Positive overflow (in-phase)
    sum1 = Iout + j*(Qout-2*limit);           % Adder output w/overflow
    overflow = overflow + 1;                  % Increases counter
elseif Qout < -limit,                         % Negative overflow (quad-phase)
    sum1 = Iout + j*(Qout+2*limit);           % Adder output w/overflow
    overflow = overflow + 1;                  % Increases counter
end
peak=max(max(abs(Iout),abs(Qout)),peak);      % End of overflow simulation.
```

g. DIS output DAC

The ideal model assumes that the DIS output channels are passed through ideal DACs with infinite bit resolution. This model can be implemented by disabling the following code lines from the *simhwchk_v5_inf_res.m* file:

```
dac_res = 0.25;                               % DAC resolution 0.25 for 16bit
finalAdderOut = fix(finalAdderOut/dac_res)*dac_res;
```

h. Storage of the Ideal Model Target Image

The ideal model range Doppler image that is obtained by the *simhwchk_v5_inf_res.m* file is saved under the variable name *image* in the *infresoldata.mat* data file. This change is implemented by enabling the following code lines from the *simhwchk_v5_inf_res.m* file:

```
quantimage = dpRgMapShiftMOD(1:nDopplerCell*100/prf,1:fix(numtaps/8*9));
save infresoldata image                         % Saves Range Doppler profile
```

and disabling the code lines:

```
quantimage = dpRgMapShiftMOD(1:nDopplerCell*100/prf,1:fix(numtaps/8*9));
save finresoldata quantimage                         % Saves Range Doppler profile
```

E. CHAPTER SUMMARY

The simulation model presented in this chapter is able to generate the DIS input signals, simulate the signal processing that takes place in the chip and produce the jamming output samples. The model also simulates an ISAR processor that takes the DIS output samples to produce a range Doppler image of the target. The simulation of the DIS signal processing incorporated the use of the different bit formats, in order to properly recreate the quantization and resolution errors involved in each processing stage.

A second version of the DIS model was created by using full resolution signals and ignoring the quantization effects derived from the use of finite-bit formats. In the next chapter, this ideal model is used as a reference to compare output images in order to study the DIS hardware resolution and quantization effects.

THIS PAGE INTENTIONALLY LEFT BLANK

V. DIS ARCHITECTURE QUANTIZATION AND OVERFLOW ANALYSIS

The simulation algorithm described in Chapter IV can be used to generate range-Doppler images from a variety of DIS design alternatives. This chapter studies the performance of different DIS quantization (different hardware implementations). The analysis is performed by comparing the output image with the ideal model. The effects of individual DIS hardware stages are isolated and measured in terms of output image quality. As a result, the DIS hardware contributions to the overall image error are identified.

A. IMAGE ANALYSIS TOOLS

Two processes are developed to compare the hardware image model with the ideal image model. These image analysis tools are the noise calculation and the two-dimensional correlation coefficient. Both methods identify the images as matrices of intensities of the same dimensions and produce an index to quantify matrix similarity.

1. Mean Square Error (MSE) Calculation

The MSE method compares the image distribution of two images by obtaining the average difference between squared intensities at equivalent locations. The first step before comparing image intensities is to normalize both matrices. This is accomplished by dividing each matrix by the highest intensity value and obtaining new values ranging from 0 to 1. Having both matrices under this condition, the ideal matrix can be subtracted from the real matrix, giving the following error voltage:

$$V_N\left(m,k\right)=\frac{C_{ideal}(m,k)}{\max\left(C_{ideal}\right)}-\frac{C_{DIS}(m,k)}{\max\left(C_{DIS}\right)}. \qquad (5.1)$$

If the matrix intensities C_{ideal} and C_{DIS} in (5.1) are understood as voltage values, the subtraction of these two normalized matrices is equivalent to a voltage noise error. The deviation of the real image C_{DIS} from the ideal image C_{ideal} is due to the quantization and overflow of the bit formats.

The error voltage in (5.1) is squared and averaged over all the pixel positions of the image. Thus, the result is equivalent to the average noise energy and is given by:

$$e = \frac{1}{N_r N_p} \sum_{m=0}^{N_r-1} \sum_{k=0}^{N_p-1} \left(V_N \left(m,k \right) \right)^2 \tag{5.2}$$

where N_p is the number of image matrix rows (Doppler cells) and N_r is the number of image matrix columns (range bins). The average error energy e will be referred as the MSE coefficient. This quantity ranges from 0 to 1, with values very close to 0 for the current DIS model (low quantization noise).

2. Image Correlation Coefficient

A two-dimensional correlation coefficient is a mathematical tool that can be used as a measure of similarity between two-matrices of the same size. As with the MSE coefficient method, both the ideal and test images can be interpreted as matrices of voltage intensities. The first step of this calculation is to subtract each matrix's average intensity. This is equivalent to taking the DC voltage component out of the signal, and leaving both images with bipolar intensities and a zero mean.

A new matrix R_1 is formed with the element-by-element product of the two zero mean matrices:

$$R_1\left(m,k\right) = \left(C_{ideal}\left(m,k\right) - \overline{C_{ideal}} \right) \left(C_{DIS}\left(m,k\right) - \overline{C_{DIS}} \right) \tag{5.3}$$

where the over bar denotes the mean. Two additional matrices R_2 and R_3 are formed as:

$$R_2\left(m,k\right) = \left(C_{ideal}\left(m,k\right) - \overline{C_{ideal}} \right)^2 \tag{5.4}$$

$$R_3\left(m,k\right) = \left(C_{DIS}\left(m,k\right) - \overline{C_{DIS}} \right)^2 . \tag{5.5}$$

For each of these three new matrices, a sum of all elements can be obtained as:

$$r_1 = \sum_{m=0}^{N_r-1} \sum_{k=0}^{N_p-1} R_1\left(m,k\right) \tag{5.6}$$

$$r_2 = \sum_{m=0}^{N_r-1} \sum_{k=0}^{N_p-1} R_2\left(m,k\right) \tag{5.7}$$

$$r_3 = \sum_{m=0}^{N_r-1} \sum_{k=0}^{N_p-1} R_3\left(m,k\right). \tag{5.8}$$

The ratio of the r_1 coefficient in (5.6) can be divided by the square root of the product of r_2 and r_3 in (5.7) and (5.8) to obtain a two-dimensional correlation coefficient of the two original image matrices C_{ideal} and C_{DIS}. The resulting two-dimensional coefficient can be summarized as [19] :

$$r = \frac{r_1}{\sqrt{r_2\ r_3}} = \frac{\sum_{m=0}^{N_r-1}\sum_{k=0}^{N_p-1}\left(C_{ideal}\left(m,k\right)-\overline{C_{ideal}}\right)\left(C_{DIS}\left(m,k\right)-\overline{C_{DIS}}\right)}{\sqrt{\left(\sum_{m=0}^{N_r-1}\sum_{k=0}^{N_p-1}\left(C_{ideal}\left(m,k\right)-\overline{C_{ideal}}\right)^2\right)\left(\sum_{m=0}^{N_r-1}\sum_{k=0}^{N_p-1}\left(C_{DIS}\left(m,k\right)-\overline{C_{DIS}}\right)^2\right)}}. \quad (5.9)$$

The correlation coefficient in (5.9) can be implemented in MATLAB with the function *corr2(A,B)*, where A and B are matrices of the same size. This correlation function can be used for many template-matching applications, such as face detection or speech recognition [20]. The two-dimensional correlation coefficient values range from 0 to 1, with values very close to 1 for the DIS models. This indicates the image is very similar to the ideal model image.

3. Image Analysis Routine

A simple image analysis routine is performed during the DIS simulation immediately after the ISAR range-Doppler image generation. The analysis routine compares the recently generated image with the ideal-case image obtained from a previous simulation. The comparison is in terms of the MSE coefficient and the two-dimensional correlation coefficient analysis.

The image analysis routine is listed in Appendix A and is saved as the *image_analysis.m* file. This routine assumes that the real and ideal images have been stored as matrices of the same size. The real image is loaded from the *quantimage* variable stored in the *finresoldata.dat* data file and generated by the *simhwchk_v5.m* file. The ideal image is loaded from the *image* variable stored in the *infresoldata.dat* data file and generated by the *simhwchk_v5_inf_res.m* file. Once loaded, both image matrices are normalized. The following code lines implement this intensity normalization:

```
normimage       = image/max(max(image));              % Normalizes real image.
normquantimage = quantimage/max(max(quantimage));     % Normalizes ideal image.
```

The normalized matrices are subtracted and the resulting matrix is squared at each element location. The matrix containing the squared errors is saved as *noiseEnergy*. The squared error at each pixel location is found with the following loop structure:

```
for m=1:dp_pts/2,
   for n=1:numtaps*1.0625,                              .
     noiseEnergy(m,n) = (normimage(m,n)-normquantimage(m,n))^2; % Pixel square error
   end;
end;
```

The MSE coefficient is then found as the average of all the elements of the *noiseEnergy* matrix. The calculation is implemented summing all the elements of *noiseEnergy* and dividing the sum with the product of the number of rows and the number of columns. This average is calculated in the following code lines:

```
[a,b] = size(noiseEnergy);
noise_ave = sum(sum(noiseEnergy))/(a*b)                 % MSE coefficient.
```

The two-dimensional correlation coefficient is implemented using the Matlab function *corr2( )*, using the real and the ideal image matrices as the two parameters. This Matlab function implements the calculation presented in Equation (5.9). As shown in the following code line, the correlation coefficient is stored as *corel_coeff*:

```
corel_coeff = corr2(image,quantimage)                   % 2D correlation coeff.
```

Finally, the software routine plots the image distribution of the squared error (*noiseEnergy* matrix), with the purpose of identifying possible concentrations of errors at specific range bins or Doppler cells.

B. PHASE QUANTIZATION ANALYSIS

This section analyzes an ideal model that is modified to implement the DIS phase modulation stage using 5-bit phase formats. The three files containing the ideal model are copied to new files under the names of *extract_para_ship_v5_isol_quant.m, mathost_v5_isol_quant.m* and *simhwchk_v5_isol_quant.m.* This last file is modified in order to store the non-ideal output image as the variable *quantimage* in the *finresoldata.dat* data file.

1. Identification of Error Sources

The intercepted ISAR signal is fed to the DIS in terms of DRFM generated phase samples as described in (3.14). This collection of phase samples is represented by the unsigned 5-bit format in Table 2. The use of this format is a source of quantization error induced in the phase information. The maximum phase error due to the 5-bit format is given by:

$$\triangle\phi_0 = 360^\circ / 2^5 = 11.25^\circ. \tag{5.10}$$

The 5-bit DRFM phase input is incremented by the phase modulation coefficient of the same bit format and error. Thus, the resulting phase sum from this modulation has a resolution error ranging from 0 to 22.5 degrees. Assuming the DRFM phase samples and the phase coefficients have a uniform probability of distribution, the error of the modulated phase should have a mean of 11.25 degrees. The incidence of the phase error is reflected in the sine and cosine LUT table. At this next stage, the phase samples carrying errors higher than the 11.25-degree average produce an LUT output off by one level. As shown in Table 8 from Chapter IV, the I and Q amplitude difference between adjacent 8-bit LUT levels ranges from 0.015 to 0.195 (equivalent to 0.7 to 10 percent error).

2. Implementation of the Software Simulation

The *extract_para_ship_v5_isol_quant.m* file is modified to add the use of 5-bit formats in the DIS phase modulation stage. The new model is run and the image effects are compared with the ideal model image. The image analysis tools described in Section A are used to quantify the isolated effect of the 5-bit phase quantization. The inclusion of 5-bit phase modulation coefficients is done enabling the following loop structure:

```
for i8 = 1:dp_pts,
        for i9 = 1:numtaps,                          % 5-bit quantization in 32 levels
                phasecoeff(i9,i8) = fix(rem(phaseinc(i9,i8)+32,32));
        end
end
```

and disabling the following line:

```
phasecoeff=phaseinc;                                 % Added For Infinite Resolution
```

The inclusion of the 5-bit format in the DRFM phase is done by adding the fix rounding function to the chirp phase generated in the *mathost_v5_isol_quant.m* file. This can be done by disabling the code line:

```
  pri_rg_phaseq(idx1,:) =oldphase/p;                 % Phase quantized in 2^n levels
```

and enabling the code line:

```
  pri_rg_phaseq(idx1,:) =fix(oldphase/p);            % Phase quantized in 2^n levels
```

3. Simulation Results

The MSE coefficient obtained when the effects of the 5-bit phase format in the phase modulation are added is $9.8486 \cdot 10^{-5}$. The two-dimensional correlation coefficient obtained for the same case is 0.99931. The small value in the MSE coefficient or noise energy indicator and the correlation coefficient value vary in the vicinity of one, indicating that the quantization effects of the 5-bit phase format are minimum. In fact, the differences between the resulting image and the ideal case cannot be identified with the human eye. The image obtained from this model and the distribution of the MSE are included in Figure 33.

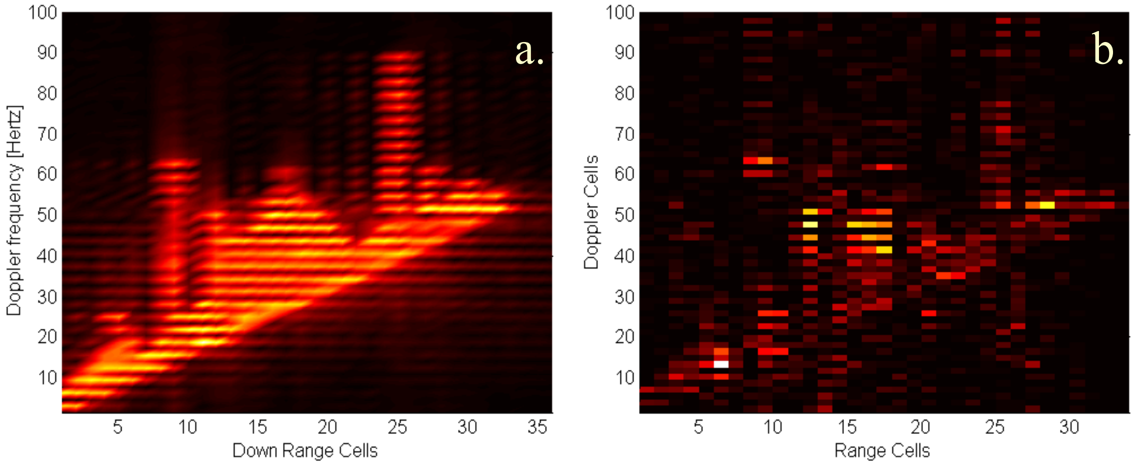


Figure 33 a) Image Generated by an Ideal DIS with Phase Quantization. b) Image Distribution of the Phase Quantization MSE.

C. LUT RESOLUTION AND SCALING EFFECTS

1. Identification of Error Sources

The output of the DIS phase modulation circuit is converted into I and Q amplitude components using cosine and sine calculations. Due to the high-speed system requirements, an LUT table with 8-bit resolution is used. The LUT is a 5-bit decoder, where each of the 32 possible phases that can be represented by the format in Table 2 has an equivalent sine and cosine value. The errors induced by the use of an LUT with 8 bits of resolution to calculate cosine and sine can be classified as resolution or quantization errors.

The LUT resolution errors originate from the output format of the I and Q channels. These LUT outputs have an 8-bit two's complement format with a sign bit and 7 fractional bits. Thus, the sine and cosine calculations have a maximum resolution error of 2^{-7}. This value in the output scale from -1 to $+1$ represents a 0.4 percent error. If the 8-bit LUT table is replaced by a 7, 6, 5, 4 or 3-bit table, the maximum resolution error increases to $2^{-6}, 2^{-5}, 2^{-4}$, 2^{-3} and 2^{-2}, respectively. Also, these values represent a 0.8, 1.6, 3.2, 6.4 and 12.8 percent error, respectively.

2. Implementation of the Software Simulation

The LUT quantization effects are isolated by reversing the changes done in the phase quantization analysis to return to the ideal model and then adding an LUT to replace the sine and cosine calculations. The inclusion of the LUT table is implemented changing the code in the *simhwchk_v5_isol_quant.m* file to disable line:

```
lutOut=cos(phAddOut*2*pi/32)+sqrt(-1)*sin(phAddOut*2*pi/32);   % Ideal LUT
```

and to enable the code lines listed below:

```
tmp = rem(phAddOut+depthLUT,depthLUT)+1;              % Phase limited between 0-31
lutOut = cosine8(tmp) + j*sine8(tmp);                 % 8-bit LUT table output
```

where *cosine8* and *sine8* are 32-element array variables storing the contents of the 8-bit LUT. An additional code line must be added to round up the infinite resolution phase (non-integers) to be used as integer addresses for the LUT. The following code line must be added immediately before the line defining the *lutOut* vector:

```
tmp = fix(tmp);                                       % Needed to isolate LUT effects
```

3. Simulation Results

The new model was run and the results are similar to the phase quantization case. The 8-bit LUT quantization showed a very small MSE coefficient ($6.1069 \cdot 10^{-6}$) and a correlation coefficient very close to one (0.999916). Thus, the effects of the 8-bit LUT quantization can be considered minimal. The image generated by the model and the MSE distribution is shown in Figure 35.

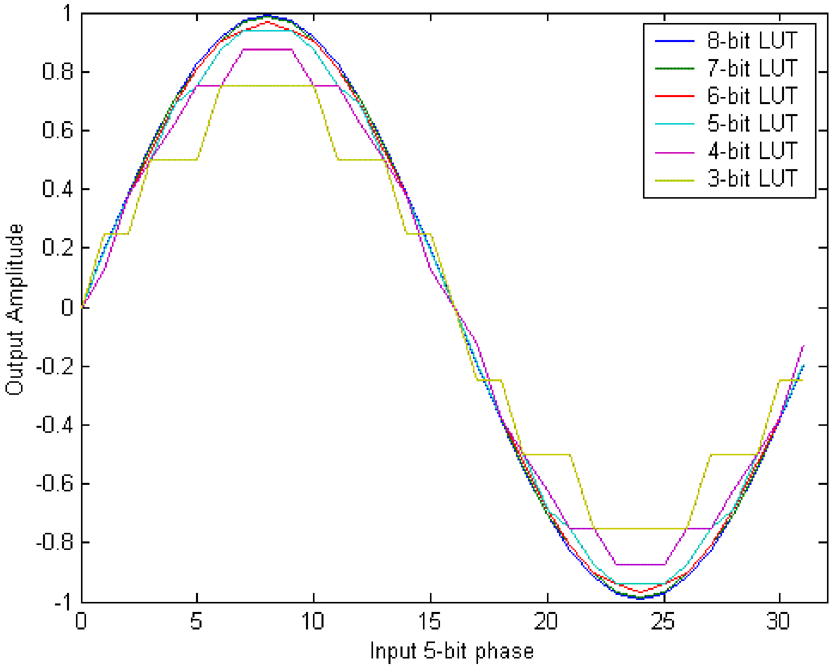


Figure 34 Plot of the Sine Curve Using 32 Phase Values in Different LUTs.

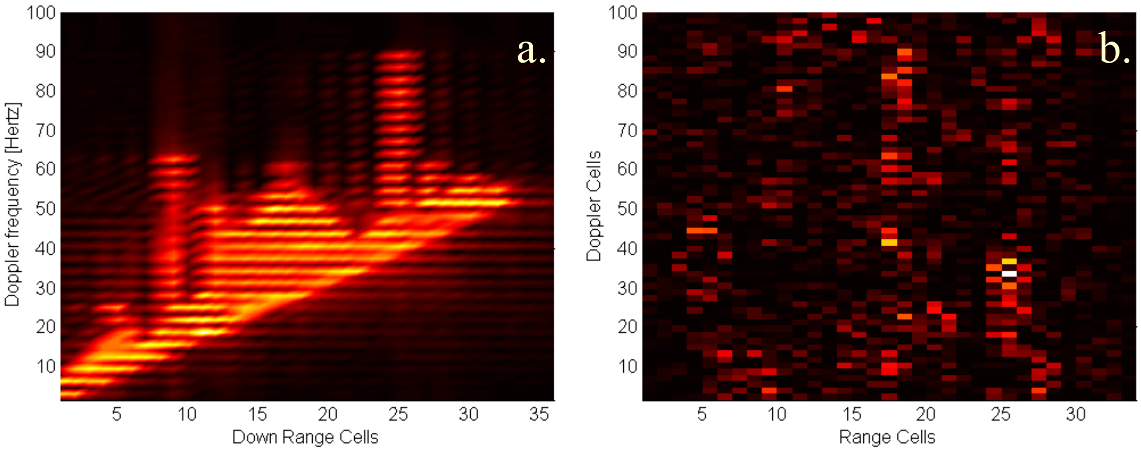


Figure 35 a) Image Generated by an Ideal DIS with LUT Quantization. b) Image Distribution of the LUT Quantization MSE.

To evaluate the effect of additional LUT configurations, the arrays *cosine8* and *sine8* are replaced to implement LUT options from 7 bits through 3 bits. The results are summarized in Table 9. The DIS images for the different LUT cases are not presented here since they are too similar to visually recognize their differences.

LUT resolution	MSE coefficient	2D Correl.Coefficient
8-Bit	$6.1069 \cdot 10^{-6}$	0.9999160
7-Bit	$6.1382 \cdot 10^{-6}$	0.9999155
6-Bit	$6.2004 \cdot 10^{-6}$	0.9999149
5-Bit	$6.4654 \cdot 10^{-6}$	0.9999112
4-Bit	$6.7981 \cdot 10^{-6}$	0.9999057
3-Bit	$1.1416 \cdot 10^{-5}$	0.9998420

Table 9 Image Quality Statistics for Different LUT Resolutions.

In summary, the simulations showed that the quantization effects of the LUTs are negligible. Even at the extreme case of cosine/sine calculations using a 3-bit resolution LUT where only seven output amplitude levels are obtained, the effects are minimal.

D. GAIN QUANTIZATION ANALYSIS

1. Identification of Error Sources

In the gain modulation stage of the DIS pipeline, the I and Q outputs of the LUT are multiplied by a power-of-two gain ($2^{g(r,n)}$). This multiplication is implemented using a binary shifter using the control codes listed in Table 4 from Chapter III. As described in that chapter, the generation of these coefficients is performed for each range-bin/pulse combination. The default quantization to obtain the gain coefficients follows the scheme shown in Table 1, where normalized magnitudes in the range from 0 to 1 are respectively assigned gains from 2^0 to 2^{11}. This quantization scheme is shown as a staircase shape curve in Figure 36. Ideally, the gain values should be linearly related to the normalized magnitudes, and thus following the diagonal line of that figure. However, the exponentially nature of the gain shifter limits the gain assignment to exponentially distributed levels.

2. Implementation of the Software Simulation

The gain quantization effects are isolated by reverting the previous changes to return to the ideal model and adding new code modifications to the *extract_para_ship_v5_isol_quant.m* file. The following code line is disabled to eliminate the ideal gain quantization:

```
gain = 1024*amp/max(max(amp));                    % Added for infinite resolution
```

and the following loop structure is enabled to return to the default gain quantization scheme:

```
gain = ones(numtaps,dp_pts);                    % Default gain coefficient is 1.
for i1 = 1:numtaps,
   for i2 = 1:dp_pts,
      normAmp = amp(i1,i2)/max(max(amp));       % Magnitudes are normalized.
      for s=1:10,
         if normAmp > (2^-(11-s))*3/2,          % Thresh: 0.0015....,0.375,0.75
            gain(i1,i2) = 2^s;                  % Gain:   2,4,8,16,......512,1024
         end                                    % Assigns 11 power-of-two gains.
      end
   end
end
```

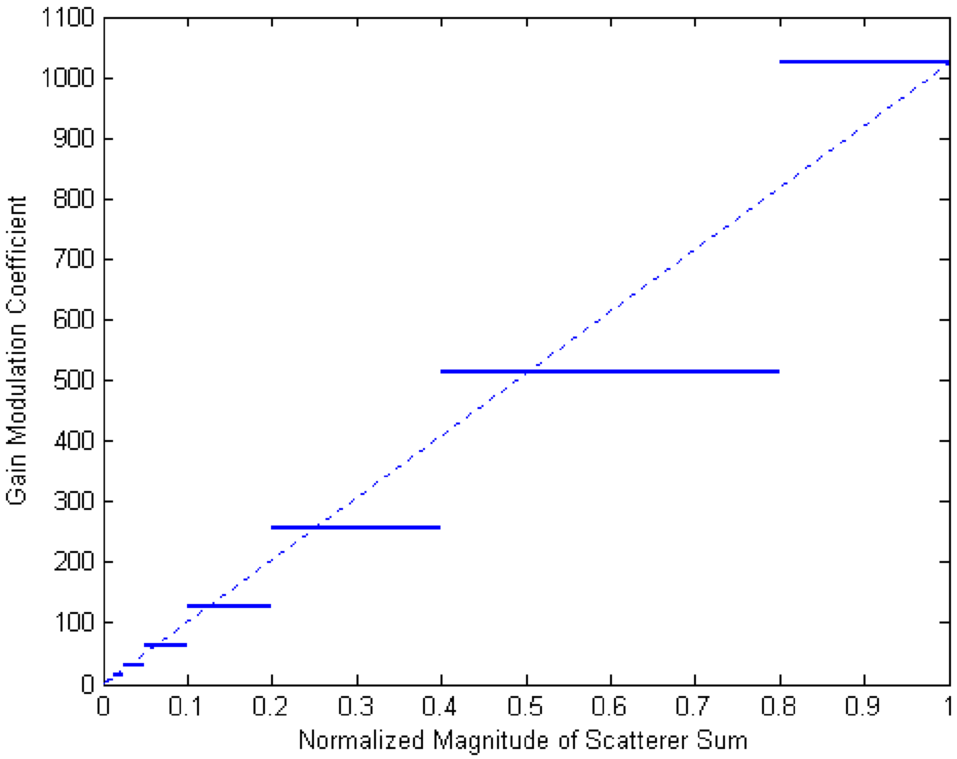


Figure 36 Quantization Scheme for Gain Modulation Coefficients.

3. Simulation Results

The new model is run yielding an MSE coefficient of $5.220 \cdot 10^{-4}$ and a correlation coefficient of 0.99311. The model image using the DIS gain quantization and the MSE distribution are shown in Figure 37.

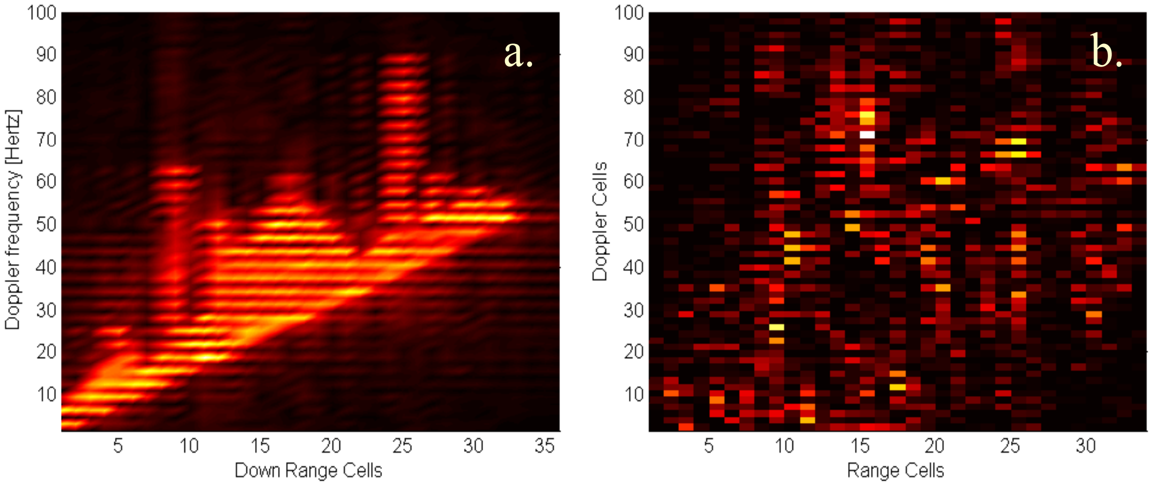


Figure 37 a) Image Generated by an Ideal DIS with Gain Quantization. b) Image Distribution of the Gain Quantization MSE.

E. I AND Q CHANNEL RESOLUTION ANALYSIS

1. Identification of Error Sources

The format of the I and Q LUT channels after truncation of the 5 LSBs has a 13-bit two's complement format as shown in Table 5. The LSB of that format is 2^{-2}. Thus, all the possible I and Q magnitudes are multiples of 0.25. The resolution error in the DIS performance is quantified in this section.

2. Implementation of the Software Simulation

After reverting the changes implemented in the previous model to return to the ideal model, new code modifications are performed to the *simhwchk_v5_isol_quant.m* file in order to implement a truncation by multiples of 0.25. That truncation is included by enabling the following code line:

```
tapOut(inPlsCnt+delay,idx3) = fix(gainOut(idx3)/o_res)*o_res;
```

and by disabling the following code line:

```
tapOut(inPlsCnt+delay,idx3) = gainOut(idx3);
```

where the variable *o_res* is the bit format LSB with a value of 0.25.

3. Simulation Results

The model is run with a 32-range-bin target and 128 radar pulses. The image generated had an MSE coefficient of $0.023378 \cdot 10^{-6}$ and the two-dimensional correlation with the ideal image had a coefficient of 0.9999998. The simulation results indicate that the resolution effects of the LSB in the I and Q channels are negligible. The image obtained in this simulation is shown along with the MSE distribution in Figure 38.

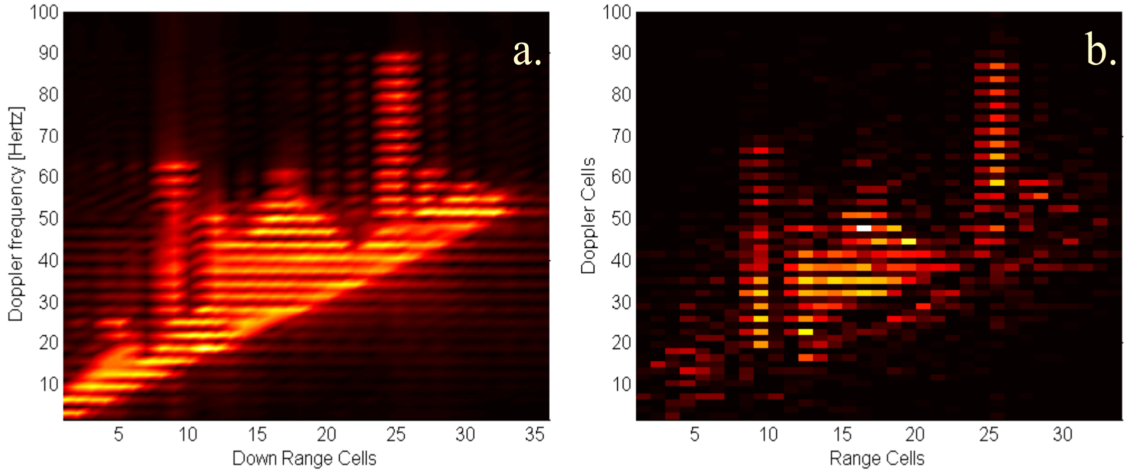


Figure 38 a) Image Generated by an Ideal DIS with LSB Resolution Effects in the I and Q Channels. b) Image Distribution of the MSE.

F. OUTPUT ADDER OVERFLOW ANALYSIS

1. Identification of Error Sources

The two's complement output format of the 16-bit output adder can represent values from -2^{13} to $2^{13}-2^{-2}$ (-8192 to 8191.75), as shown in Table 6. Although the output values that are originated in each range bin vary from -1028 to 1027.75 (cases for maximum gain), it is feasible that the cumulative sum of range bins exceeds the -8192 and 8191.75 output adder limits. An overflow occurs in the output adder when the cumulative sum of the range-bin values goes beyond these limits.

The main effect of the output adder overflow is that the sum of the most significant bits (2^{12}) produces a carry that changes the sign of the two's complement format. This sign change effect was studied in order to predict the output value for overflow cases. It was found that the result of an overflow due to the sum of a value A and a value B can be expressed as:

$$\left(\mathrm{A{+}B}\right)_{Overflow} = \left(\mathrm{A{+}B}\right) + 2\cdot 8192 \quad \text{if } \left(A+B\right) < -8192 \tag{5.11}$$

$$\left(\mathrm{A{+}B}\right)_{Overflow} = \left(\mathrm{A{+}B}\right) - 2\cdot 8192 \quad \text{if } \left(A+B\right) > +8191.75 \tag{5.12}$$

where (5.11) and (5.12) represent a negative and a positive overflow respectively.

2. Implementation of the Software Simulation

The code changes performed for the previous model are reversed to return to the ideal mode. Next, the *extract_para_ship_v5_isol_quant.m* file code is changed to implement equations (5.11) and (5.12) by enabling the following code lines:

```
Iout=real(sum1);                              % Overflow simulation using +/-limit
Qout=imag(sum1);
if Iout >= limit,                             % Positive oflow (in-phase)
   sum1 = Iout - 2*limit+j*Qout;              % Adder output w/overflow
   overflow = overflow + 1;                   % Increases counter
elseif Iout < -limit,                         % Negative oflow (quad-pha)
   sum1 = Iout + 2*limit+j*Qout;              % Adder output w/overflow
   overflow = overflow + 1;                   % Increases counter
end
if Qout >= limit,                             % Positive oflow (in-phase)
   sum1 = Iout + j*(Qout-2*limit);            % Adder output w/overflow
   overflow = overflow + 1;                   % Increases counter
elseif Qout < -limit,                         % Negative oflow (quad-pha)
   sum1 = Iout + j*(Qout+2*limit);            % Adder output w/overflow
   overflow = overflow + 1;                   % Increases counter
end
peak=max(max(abs(Iout),abs(Qout)),peak);      % End of oflow simul.
```

where the variable *limit* was previously set to 8192 and the variable *peak* is used to register the highest overflow value.

3. Simulation Results

The simulation of this model is run using a 32-range-bin target and 128 pulses. After comparing the model image with the ideal quantization image, the MSE coefficient obtained was $0.43933\cdot 10^{-3}$ and the two-dimensional correlation coefficient was 0.99407. For this particular target, the number of overflows is 22 with a maximum value of 8443.5. The image of this model and the MSE distribution is shown in Figure 39.

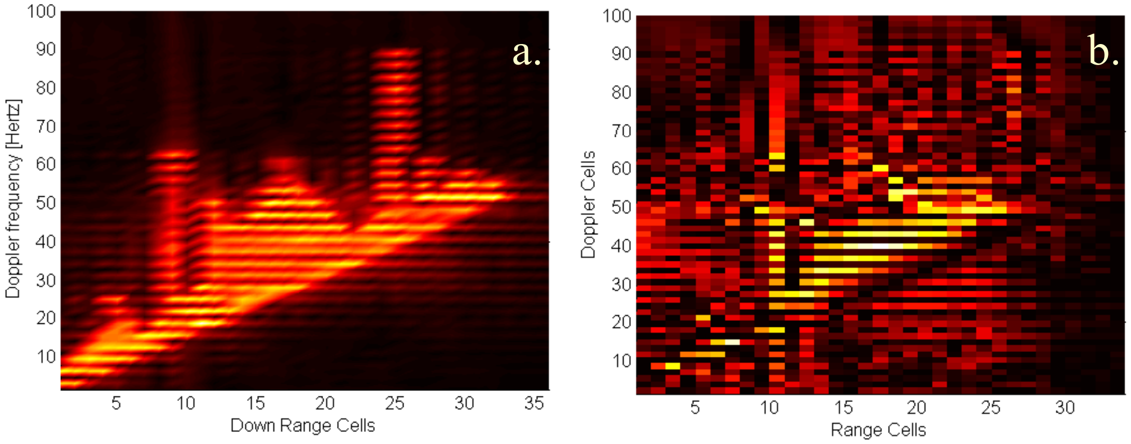


Figure 39 a) Image Generated by an Ideal DIS with Output Adder Overflow. b) Image Distribution of the MSE Due to Overflow.

G. DIS HARDWARE MODEL SIMULATION

1. Identification of Resolution Error Sources

This model simulates the DIS hardware performance including all possible quantization and resolution effects contained in the DIS being fabricated, except only 32 range-bin processors are being evaluated instead of 512. The model includes the use of a 5-bit phase format, an 8-bit resolution LUT, gain coefficient quantization from Table 1, the minimum resolution of the I and Q channels and the overflow effects of the output adders. This model also assumes the DIS output is connected to 16-bit DACs. Thus, truncation does not occur due to bit format mismatch. The simulation is performed using the defaults simulation files: *extract_para_ship_v5.m, mathost_v5.m* and *simhwchk_v5.m*.

2. Simulation Results

The default model was run with a 32-range-bin target and integrating 128 pulses. The simulation yielded an MSE of $0.074854 \cdot 10^{-3}$ and a two-dimensional correlation coefficient of 0.990004. The number of overflow counts is 22 with a peak value of 8410.

H. RESOLUTION EFFECTS OF OUTPUT DAC CONVERSION

1. Identification of Resolution Error Sources

The resolution effects of the I and Q channels (Section E) minimally impacted the DIS performance. Therefore, it may be expected that truncating the DIS output by the least significant bits (LSB) would have the same negligible effect. The DIS output channels have a 16-bit format. The intention here is to simulate the hardware model of Section G added with the effects of coupling the DIS output with a 14, a 12 and a 10-bit DAC, and thus truncating different amounts of LSBs.

2. Implementation of the Software Simulation

The *simhwchk_v5.m* file is changed in order to set the value of the DAC minimum resolution (*dac_res* variable) according to Table 10. The implementation of this code change ensures that all I and Q amplitudes in the DIS output are exact multiples of the resolution value *dac_res*.

DAC resolution	**Resolution value**
16 bit	0.25
14 bit	1
12 bit	4
10 bit	16
8 bit	64
6 bit	256

Table 10 Minimum DIS Resolution for Different DAC Configurations.

3. Simulation Results

The 16-bit DAC model is equivalent to the hardware model of Section G, where the DAC does not produce any truncation effect. Changing the DAC to 14-bit, where the 2 DIS output LSBs are discarded, yielded an MSE coefficient of $0.74799 \cdot 10^{-3}$ and a two-dimensional correlation coefficient of 0.990005. For 12-bit DAC, 4 output LSBs are discarded and the results are respectively $e = 0.73572 \cdot 10^{-3}$ and $r = 0.990078$. For 10-bit DAC, 6 LSBs are discarded obtaining an MSE coefficient of $e = 0.72228 \cdot 10^{-3}$ and a two-dimensional coefficient of $r = 0.990082$. There is a slight apparent image quality improvement when reducing the DAC resolution for some cases. However, this is attributable to an error cancellation effect that can be neglected because of the small magnitude. For an 8-bit DAC case, $e = 1.200*10^{-3}$ and $r = 0.98605$. As shown in Figure 40, only when DAC resolution is reduced to 4 bits is there a visually notable image deterioration with r = 0.9279. These results clearly show that most of the vital information of the DIS output is included in the 4 MSBs and that DACs of 6 or 8 bits can be used without affecting the system performance. As a final note, the DAC must sample at the rate of the DIS clock frequency, which dictates the range resolution of the device.

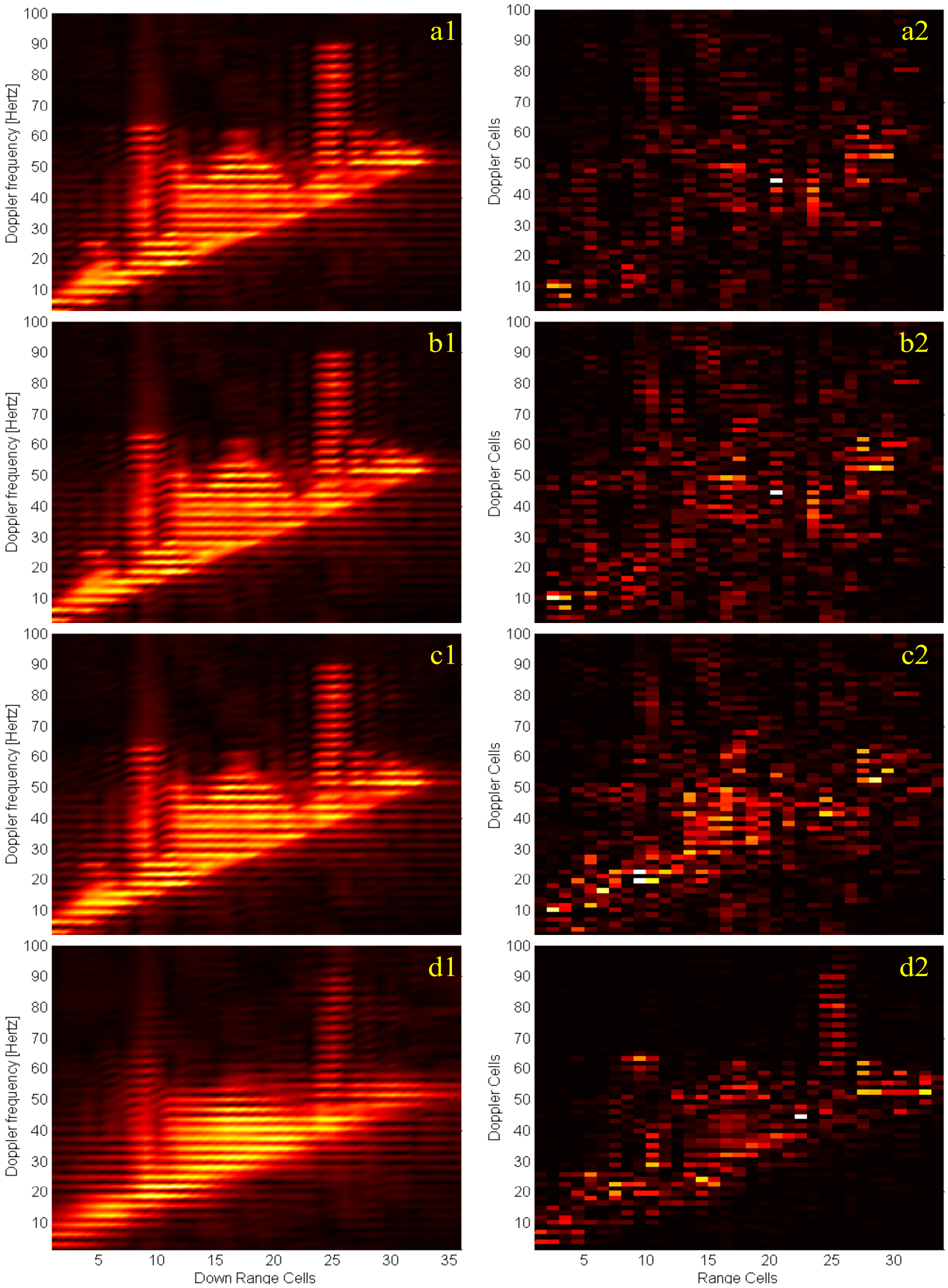


Figure 40 Target Image and MSE for Different Output DACs: (a) 16-Bit, (b) 8-Bit, (c) 6-Bit and (d) 4-Bit.

I. CHAPTER SUMMARY

The effects of the bit format quantization and resolution on the DIS output images were studied for different models in terms of the similarity to an ideal image. The isolated effects from the 5-bit phase format, the sine/cosine LUT, the 4-bit gain control code format, the 13-bit I and Q channel format and the output adder overflow were modeled independently. Only the effects of gain quantization and adder overflow errors stood out. The rest were mathematically negligible.

The hardware architecture is modeled considering all bit format effects (Section G). This model is basically the true simulation of the DIS performance. The effects of the combined quantizations on the image errors were minimum (r = 0.990) and not visually recognizable. Finally, the output truncating effects of using DACs with different bit resolution were simulated. From these results, it can be concluded that the DIS output information is concentrated in the first 8 MSBs from the 16-bit format. To support this fact, the discarding of the 8 LSB out of the output format resulted in image effects that were mathematically negligible.

A summary of the image effects of the different simulated models is included in Table 11. In general, the DIS simulation models proved that the architecture design is robust and able to endure the effects of error originated from the use of values truncated by bit formats.

Effect Simulated	**MSE from Ideal Image**	**2D Correlation Coefficient**
Phase 5-bit format	$0.0098486 \cdot 10^{-3}$	0.999310
Sine/Cosine 8-bit LUT	$0.0061069 \cdot 10^{-3}$	0.999916
Sine/Cosine 7-bit LUT	$0.0061382 \cdot 10^{-3}$	0.999915
Sine/Cosine 6-bit LUT	$0.0062004 \cdot 10^{-3}$	0.999914
Sine/Cosine 5-bit LUT	$0.0064654 \cdot 10^{-3}$	0.999911
Sine/Cosine 4-bit LUT	$0.0067981 \cdot 10^{-3}$	0.999905
Sine/Cosine 3-bit LUT	$0.011416 \cdot 10^{-3}$	0.999842
Gain control 4-bit format	$0.522000 \cdot 10^{-3}$	0.993110
I and Q 13-bit format	$0.0000233 \cdot 10^{-3}$	0.999999
Output Adder Overflow	$0.43933 \cdot 10^{-3}$	0.994074
Hardware model with 16-bit DAC	$0.74854 \cdot 10^{-3}$	0.990004
Hardware model with 14-bit DAC	$0.74799 \cdot 10^{-3}$	0.990005
Hardware model with 12-bit DAC	$0.73572 \cdot 10^{-3}$	0.990078
Hardware model with 10-bit DAC	$0.72228 \cdot 10^{-3}$	0.990082
Hardware model with 8-bit DAC	$1.200 \cdot 10^{-3}$	0.986059

Table 11 Quantization and Resolution Error Effects on the DIS Image.

THIS PAGE INTENTIONALLY LEFT BLANK

VI. DIS PERFORMANCE UNDER DIFFERENT ISAR PARAMETERS

This chapter explains the ISAR requirements for PRF and coherent integration time and relates these parameters to the Doppler resolution obtained at the ISAR target image. The simulation model is run to verify the expected results and the effect of these parameters under a different set of values.

A. ISAR PRF AND INTEGRATION TIME REQUIREMENTS

1. ISAR PRF Limits

As with any radar design, ISAR PRF must be low enough to avoid range ambiguities and high enough to avoid Doppler ambiguities. The limitation for maximum PRF is given by [2] :

$$PRF_{\max} = \frac{c}{2R_{FP}} \tag{6.1}$$

where R_{FP} is the range extension of the real antenna footprint. For ISARs that are focused on a target (spotlight), the range window is relatively short. For a range footprint of 1km, the maximum PRF is 150,000 pulses per second (pps). To avoid Doppler ambiguities, the PRF must exceed the maximum Doppler frequency spread. For airborne applications, the minimum ISAR PRF is defined as [8]:

$$PRF_{\min} = \Delta f_d \cong \frac{2\,V_R\,\theta_{NNaz}}{\lambda} \tag{6.2}$$

where V_R is the velocity magnitude of the aircraft and θ_{NNaz} is the azimuth antenna real beamwidth. Using worst-case values with a speed of 205m/s (400 knots), a beamwidth of 0.0175 radians (1 degree) and a wavelength of 0.03m (10 GHz), the resulting minimum PRF is 240 pps. This minimum PRF can be understood as the Nyquist sampling limit. Radar pulses are equivalent to Doppler samples (complex). Thus, the PRF, acting as a sampling rate, must be equal to the highest Doppler frequency difference.

2. ISAR Coherent Integration Time

The coherent integration time is the time period that the radar uses to collect complex samples from pulse returns. At the end of the integration time, the radar samples are processed as a group. In Equation (2.16), it is demonstrated that the Doppler resolution is the inverse of the coherent integration time.

$$\Delta f_D = \frac{1}{T_I}. \tag{6.3}$$

Thus the azimuth or cross-range resolution improves (decreases) with longer integration times. Since the increase in integration time is directly related to the number of samples processed, the limit is set by the signal processing capacity of the ISAR processor.

B. DIS PERFORMANCE UNDER DIFFERENT PRFS AND INTEGRATION TIMES

1. DIS Dependence from ISAR Parameters

The ISAR pulse samples processed by the DIS are fed from a DRFM. Therefore, the DRFM controls the rate at which these samples are processed. This sampling rate is based on the Nyquist criteria and depends on the maximum frequency value of the analog signal before sampling. The PRF defines the rate at which the ISAR pulses arrive at the DIS. Each radar pulse is received by the DIS in the form of a sequence of complex samples. Therefore, from the DIS point of view, the ISAR PRF is transparent. In contrast, the number of pulses integrated by the ISAR is very relevant for the DIS, since it determines the number of modulation coefficients that must be used at each range bin to synthesize a column of the range-Doppler image. The DIS uses different modulation coefficients for every pulse. If the ISAR receiver is designed to integrate N_p pulses, then the DIS must repeat the set of modulation coefficients every N_p pulses. The number of integrated ISAR pulses is directly related to the PRF and integration time such that:

$$N_p = PRF\ T_I. \tag{6.4}$$

Therefore, DIS dependence on the ISAR PRF and integration time are tied together and cannot be analyzed separately. For a fixed PRF, it can be expected that if the number of integrated pulses is increased, the integration time must increase, thus the Doppler resolution is improved. Similarly, for a fixed number of integrated pulses, if the PRF is increased, the Doppler resolution is also increased. Table 12 shows various combinations of PRFs and the number of integrated pulses, along with their equivalent integration times and the resulting Doppler resolution.

PRF	Number of Integrated Pulses	Integration Time	Doppler Resolution
50 pps 100 pps 200 pps 400 pps	256 pulses 128 pulses 64 pulses 32 pulses	5.12 s	0.20 Hz
100 pps 200 pps 400 pps 800 pps	256 pulses 128 pulses 64 pulses 32 pulses	2.56 s	0.39 Hz
200 pps 400 pps 800 pps 1600 pps	256 pulses 128 pulses 64 pulses 32 pulses	1.28 s	0.78 Hz
400 pps 800 pps 1600 pps 3200 pps	256 pulses 128 pulses 64 pulses 32 pulses	0.64 s	1.56 Hz
800 pps 1600 pps 3200 pps 6400 pps	256 pulses 128 pulses 64 pulses 32 pulses	0.32 s	3.12 Hz
1600 pps 3200 pps 6400 pps 12800pps	256 pulses 128 pulses 64 pulses 32 pulses	0.16 s	6.24 Hz

Table 12 Doppler Resolution for Various ISAR Parameters.

2. Simulation Runs

The DIS performance is modeled for each of the six resolution cases listed in Table 12. To run the model, the number of integrated pulses is set to 128 pulses while the PRF is increased from 100 pps to 800 pps. In addition, the pulsewidth is adapted for each case in order to keep a constant duty cycle. Figure 41 presents the target range Doppler images for each resolution case. For this specific target, a maximum Doppler resolution close to 2 Hz is required for a good target profile.

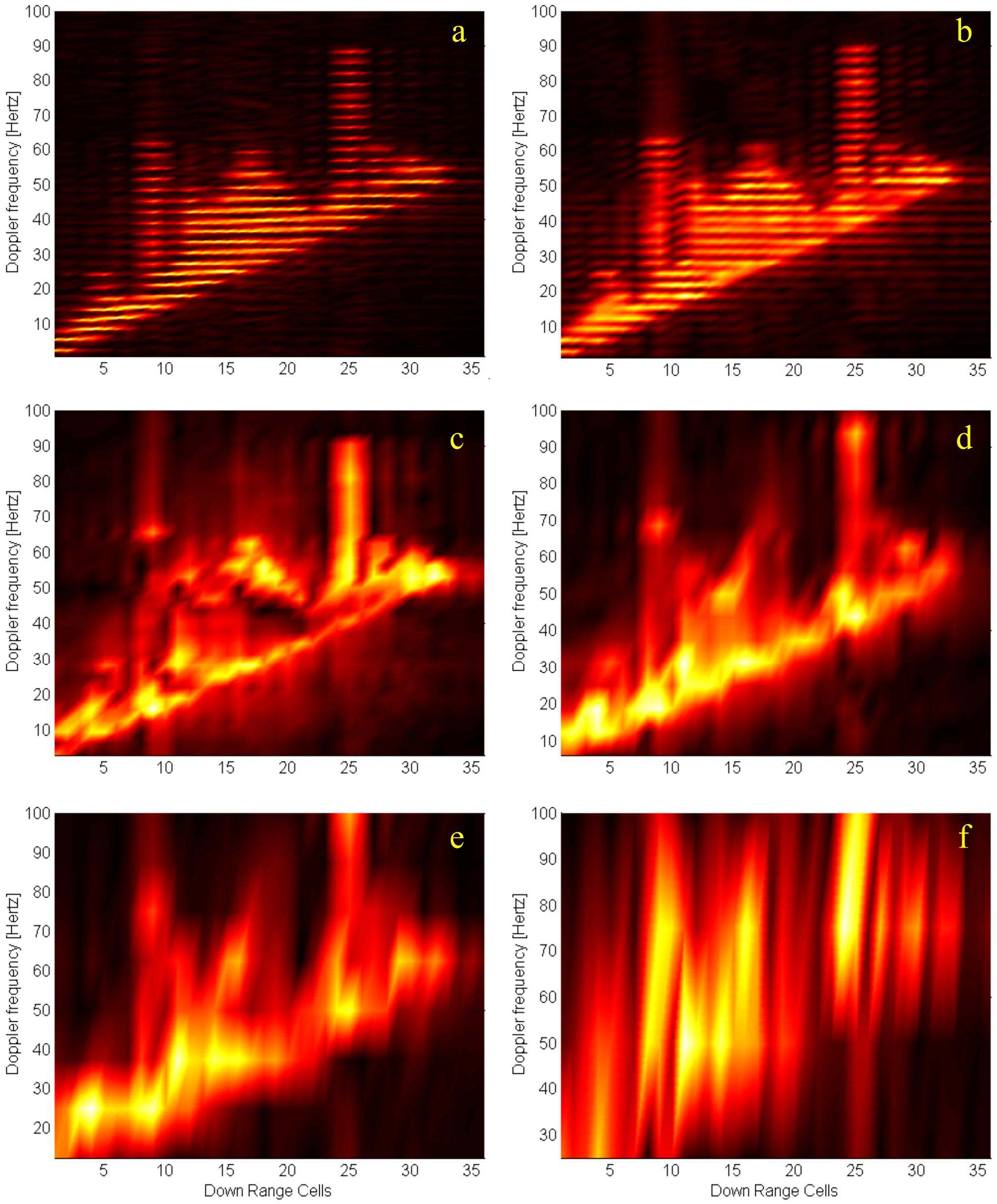


Figure 41 DIS Image for Different Doppler Resolutions: (a) 0.2 Hz, (b) 0.39 Hz, (c) 0.78 Hz, (d) 1.56 Hz, (e) 3.12 Hz and (f) 6.24 Hz.

3. PRI Multiplicity

During the simulation runs, it was observed that the use of a duty cycle, in which the PRI was not an exact multiple of the pulsewidth, did not properly synthesize the target image. The default values for PRF and PW used in the previous chapters were 200 pps and 0.5 us respectively. If the PRF is increased from the default value to 201 pps, the chirp wave is not properly constructed by the routine in the *mathost_v5.m* file. Figure 42 shows a plot of the chirp wave when the ISAR PRF is 201 pps. This chirp pulse suffers a phase reversal near three quarters of the pulsewidth. After this ISAR pulse is processed by the DIS, the resulting output modulated pulse returned to the radar is that presented in Figure 43. The resulting target range Doppler image is not properly synthesized by the ISAR routine as can be seen in Figure 44. This image presents ambiguities in the Doppler dimension, resulting in the display of three strong targets

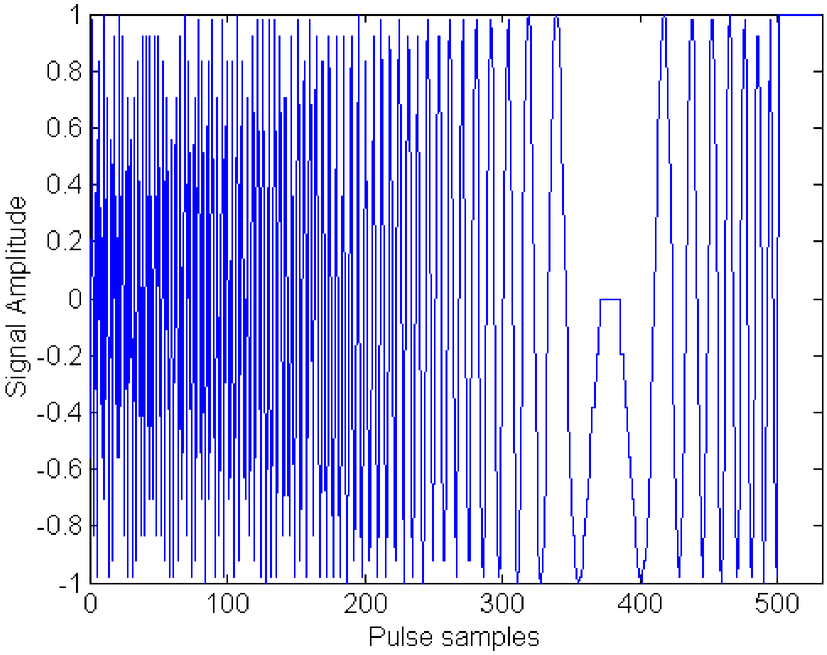


Figure 42 Erroneous Generation of a Chirp Pulse of 0.5us PW and 201pps PRF.

The range Doppler image obtained with the default PRF of 200 pps and 128 pulses is that shown in Figure 41b. The PRI / PW ratio for this case is 10000, equivalent to a duty cycle of $1 \cdot 10^{-4}$. For the case where the PRF is 201 pps, the PRI / PW ratio is 9950.2 and the duty cycle is slightly increased to $1.05 \cdot 10^{-4}$.

To better understand the phenomena of Doppler ambiguities when the duty cycle is changed, different target images are generated by increasing the PRF from 200 pps and keeping the pulsewidth fixed at 0.5 us. These images are displayed in Figure 45 for different cases. The quality of the resulting target images is summarized in Table 13. Of special interest is the image obtained in Figure 45b with a PRF of 220 pps. This plot shows a target image with a better range resolution than the default plot used throughout this thesis. From the simulations results listed in that table, it is not clear when the Doppler ambiguity is produced. However, this problem does not occur if the PRI/PW ratio has an integer value. Such is the case with a coherent transmitter.

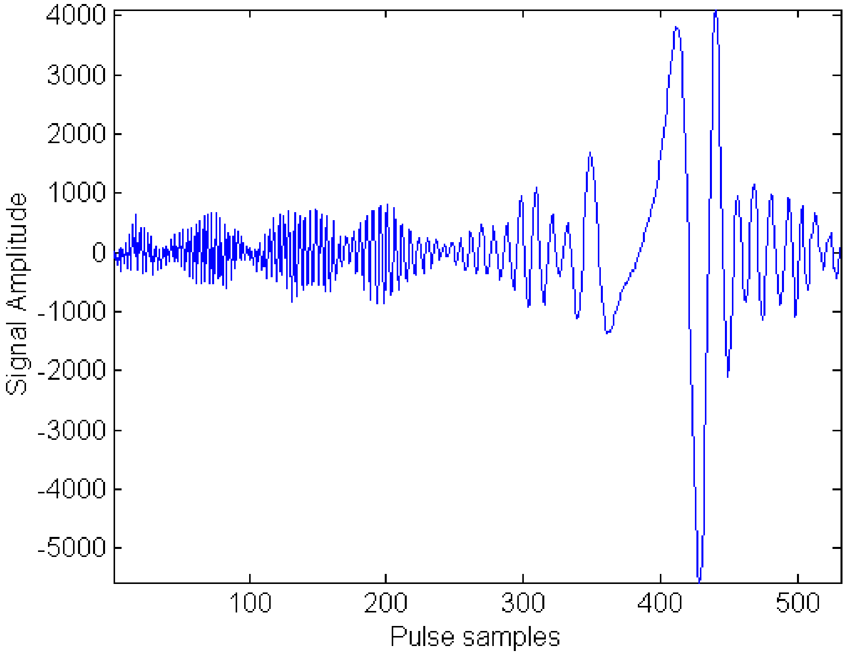


Figure 43 DIS Output Pulse Corresponding to the Chirp Wave in Figure 42.

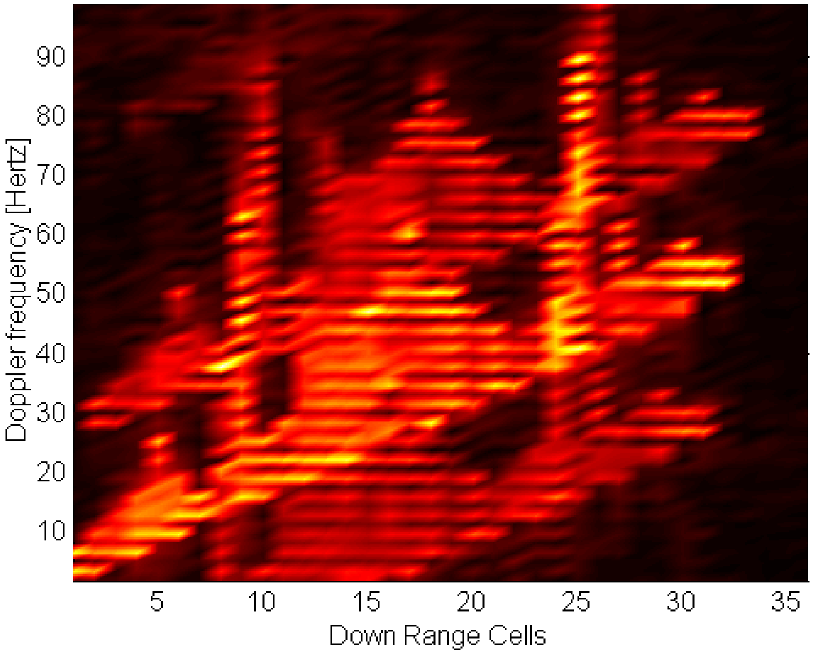


Figure 44 DIS Image for an ISAR with 0.5us PW and 201pps PRF.

PRF	PRI / PW	Duty Cycle	Doppler Resolvability
210	9523.2	$1.05 \cdot 10^{-4}$	Ambiguous
220	9090.9	$1.1 \cdot 10^{-4}$	Good
230	8695.7	$1.15 \cdot 10^{-4}$	Ambiguous
240	8333.3	$1.2 \cdot 10^{-4}$	Good
250	8000	$1.25 \cdot 10^{-4}$	Good
260	7.6923	$1.3 \cdot 10^{-4}$	Ambiguous

Table 13 Doppler Resolvability for Different ISAR Duty Cycles

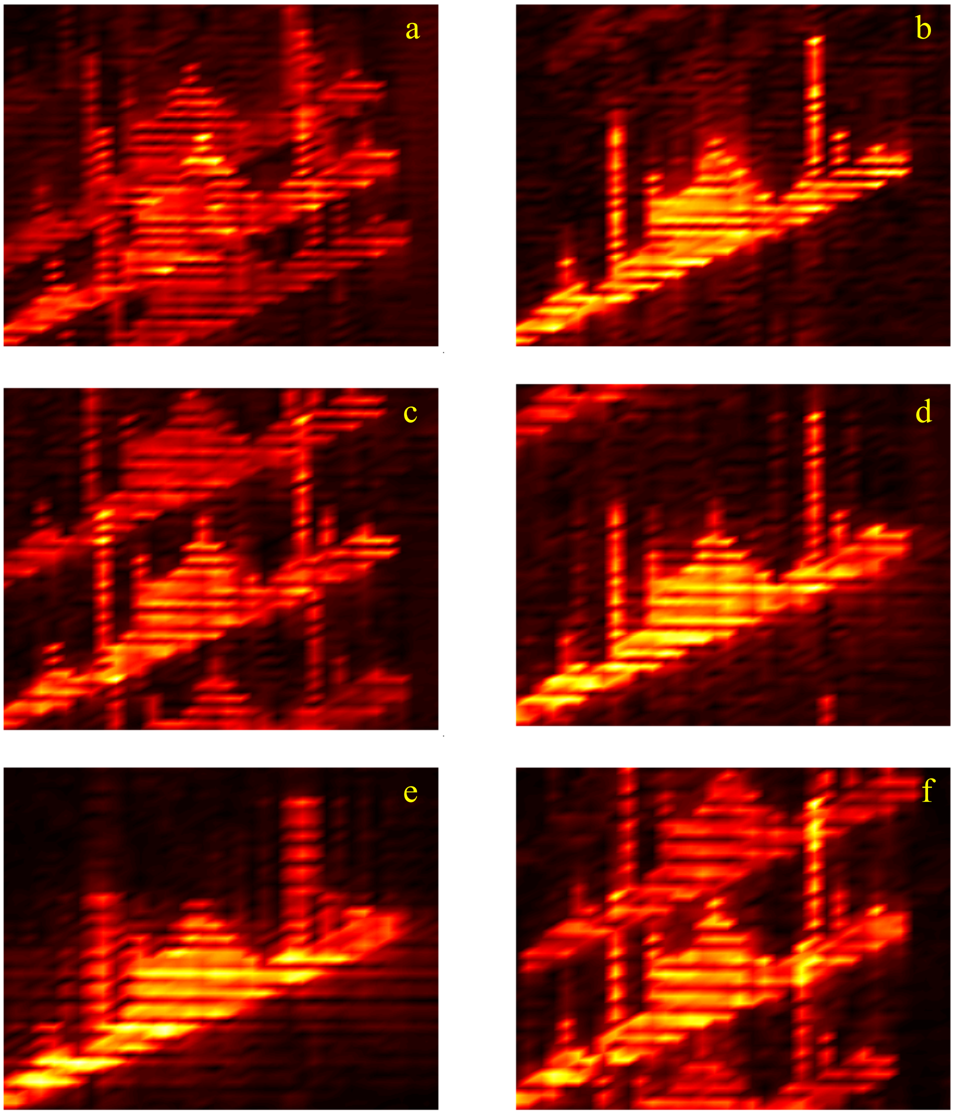


Figure 45 DIS Image for Different PRF Cases with a Fixed 0.5us PW: (a) 210 pps, (b) 220 pps, (c) 230 pps, (d) 240 pps, (e) 250 pps and (f) 260 pps.

C. CHAPTER SUMMARY

The simulation model was used to test the DIS performance and the ISAR image generation under different PRFs and coherent integration times. Two main conclusions can be extracted from this study. The first is that Doppler resolution of the image is inversely proportional to the integration time, which is equal to the product of the radar PRF and the number of pulses integrated. Since both parameters are radar dependent, the Doppler resolution of the target image is independent from the DIS hardware. The second is that an ISAR with a PRI that is not an exact multiple of the pulse width is subject to phase incoherence, which in turn, induces Doppler ambiguities in the target profile. In the next chapter, the target image matrix is analyzed in terms of the dynamic range intensity in order to relate it to the DIS output.

VII. DIS OUTPUT AND IMAGE INTENSITY DYNAMIC RANGE

The DIS output is a dual channel I and Q signal, the magnitude of which is dependent on the cumulative effect of the phase and amplitude modulations exerted at different range bins. The purpose of this chapter is to study the dynamic range of the output signal amplitude and the effect on the ISAR image dynamic range. The study is supported by simulation results of the synthesized image for a particular target using a fixed number of integrated radar pulses. This target is the same 32-range-bin target used in the study of quantization and resolution effects in Chapter V.

A. DIS OUTPUT DYNAMIC RANGE

The I and Q DIS output channels are generated from the 16-bit output adder. However, since the image is an array of scatterers arranged by range and Doppler, the intensity distribution cannot be related to the output of the 16-bit adder, where the contribution of different range bins have already been superimposed. Therefore, the DIS signal must be analyzed at the output of each range bin.

The I and Q range-bin outputs are represented by a 13-bit format given in Table 5. Since the format uses a two's complement arrangement, the magnitudes of the I and Q amplitudes are represented by 12 bits. The amplitude dynamic range, based on the number of unsigned bits *x*, is given by:

$$DR = 20\,\log_{10}\left(2^{x}\right). \tag{7.1}$$

Thus, for the range-bin output $x = 12$, the dynamic range is 72.2 dB. It is expected that the cumulative sum that takes place in the follow-on stage produces a superposition of phasors that can be successfully unwrapped in the ISAR pulse compression.

Using the 32-bin test target with the integration of 128 pulses, the DIS model is run to extract the output signal. The amplitude of two range-bin channels is captured from the simulation model and displayed in Figure 46. The two selected signals were respectively modulated by gain coefficients of 1024 and 1. Thus, they contain samples with the maximum and minimum signal magnitudes present at any range-bin output.

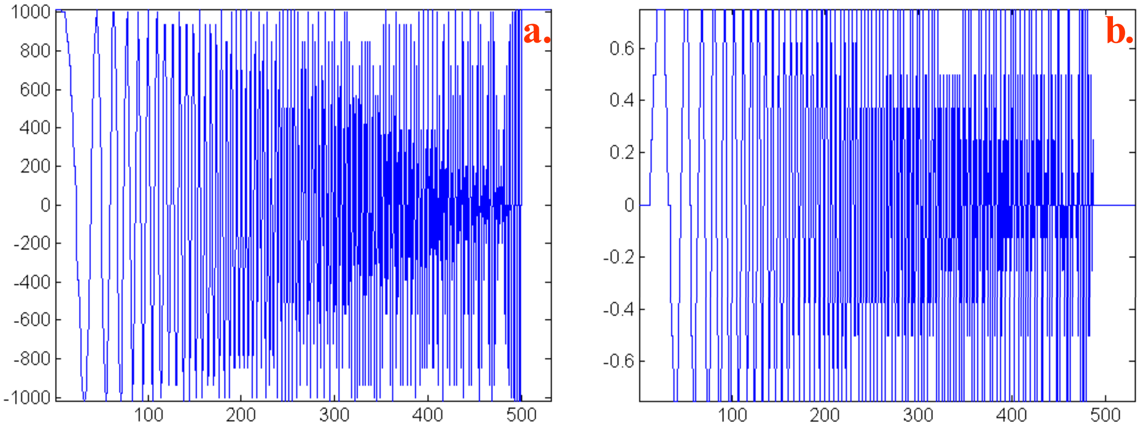


Figure 46 Range-bin Outputs Signals for a Gain of 1024 (a.) and for a Gain of 1 (b.).

From Table 3 of Chapter III, it was shown that the non-zero magnitudes of the LUT output samples range from 2^{-7} to $1-2^{-7}$. For the current range-bin/pulse combination the gain coefficient is 1024 and the binary samples are shifted 10 binary places. Following gain modulation, the samples are truncated, discarding the 5 LSBs from 2^{-3} to 2^{-7}. The resulting range-bin output non-zero magnitudes vary from $2^{-7}2^{10}$, equivalent to eight, to a value of $\left(1-2^{-7}\right)2^{10}$, equivalent to 1016. These magnitude limits can be seen in the range-bin output plots shown in Figure 46a. For the case of a gain of 1, no shift of binary places in the range bin occurs. Therefore, after a truncation of the 5 LSBs the non-zero magnitude varies from 2^{-2} to $1-2^{-7}$. The extracted range-bin output signal in Figure 46b is bounded by these limits.

The amplitude dynamic range defined in (7.1) can also be found graphically. The magnitude of the range-bin output signals can be logarithmically plotted as shown in Figure 47. The output dynamic range is then found from the difference between the peak logarithmic magnitude for the highest gain case (1024) and the lowest logarithmic magnitude for the lowest gain case (1). In terms of the bounding limits that can be extracted from the plots in Figure 47, the range-bin output dynamic range is given by:

$$DR_{RangeBin\_OUT} = \left(60.2\right) - \left(-12\right) = 72.2 \text{ dB} \tag{7.2}$$

which matches the theoretical values obtained with Equation (7.1).

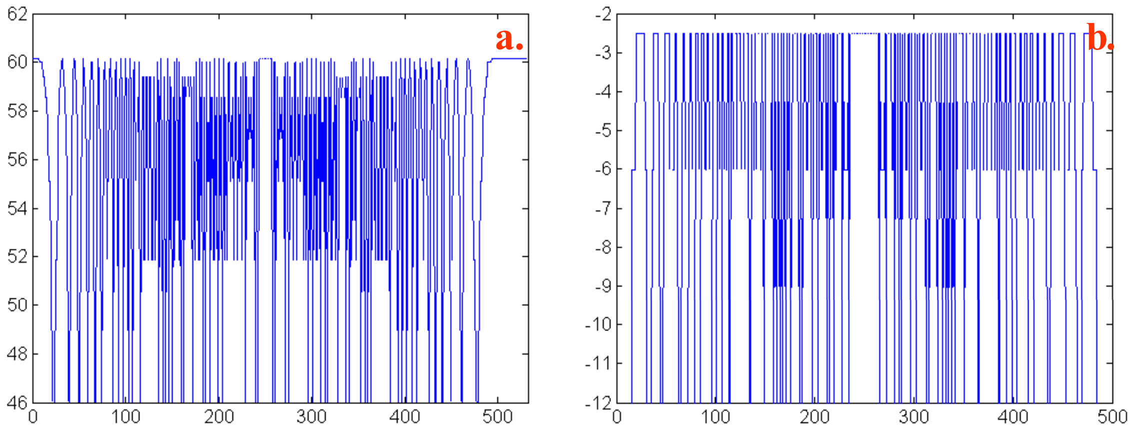


Figure 47 Logarithmic Plot of the Magnitudes of the Range-bin Signals in Figure 46.

B. ISAR PROCESSING DYNAMIC RANGE

The output signals of different range bins are superimposed by the chain of output adders from Figure 17 in Chapter III. The resulting cumulative sum is then converted to an analog signal and transmitted to the ISAR. The radar down converts the signal and performs range compression. To simplify this analysis, it is assumed that both the atmospheric attenuation and the receiver processing do not affect the DIS signal dynamic range. The last assumption is very practicable, since a dynamic range greater than 70 dB is typical in radar receivers.

ISAR range compression in the simulation model is implemented using cross-correlation (Chapter II, Figure 6). The DIS signal is cross-correlated with the ISAR reference by multiplying them in the Fourier domain, previously complex conjugating one of them. Since both signals represent voltage values, the range compression output signal should represent a power signal. The output of the range compression is equivalent to a one dimensional target profile on a pulse-by-pulse basis. The range compression outputs for 128 pulses were extracted from the simulation model. These 128 target range profiles are superimposed in one plot in Figure 48. Based on the maximum and minimum magnitudes from the range profiles, the dynamic range at the range compression output can be obtained as:

$$DR_{RangeCompression} = 10\,\log_{10}\left(4.9\cdot 10^{5}\right) - 10\,\log_{10}\left(670\right) = 28.6\text{ dB}. \qquad (7.3)$$

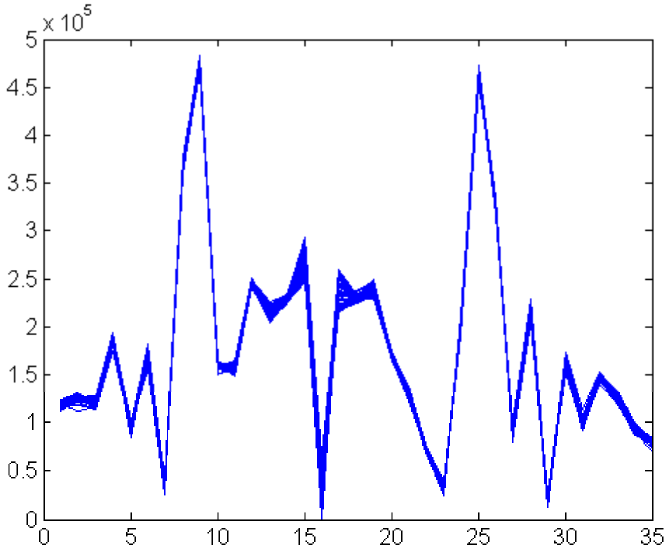


Figure 48 Plot of Superimposed Target Range Profiles for 128 Pulses.

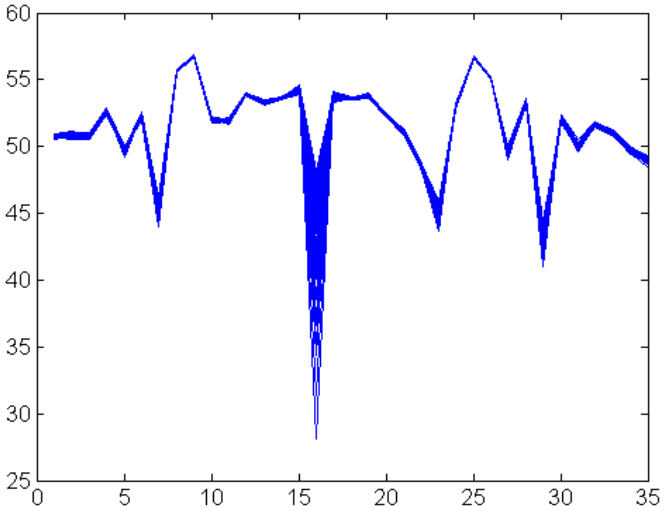


Figure 49 Logarithmic Plot of the Target Range Profiles.

The superimposed range profile is plotted logarithmically in Figure 49, where the 28.6 dB of dynamic range can be observed from the difference between the maximum and minimum magnitude. It must be pointed out that the intensities at a single range profile of the range compression output are superimposed in terms of Doppler. That is, the complex value at each location in the one-dimensional profile is the result of the addition of scatterers with different Doppler frequencies. Since the Range Doppler image does not superimpose scatterers with different Doppler components, the dynamic range obtained in (7.3) cannot be used to derive the image dynamic range.

The collection of range compression outputs form a matrix to represent the time history of the target range profile. The azimuth compression produces a Fourier transform of this matrix along the time dimension. Thus, the scatterers are rearranged according to their Doppler components. The result is the target range-Doppler image. Figure 50 shows the range-Doppler image along with one-dimensional views from the two axes.

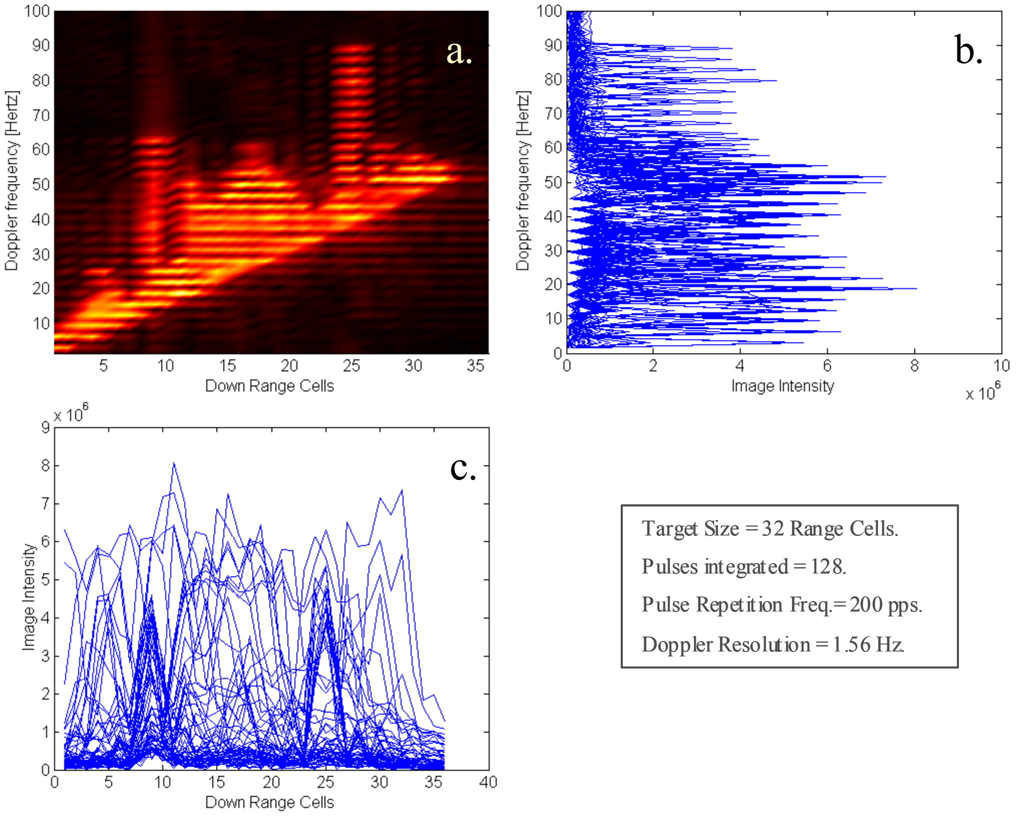


Figure 50 Range Doppler Image: a) 2D View. b) 1D View Along the Doppler Axis. c) 1D View Along the Range Axis.

From Figure 50c the maximum intensity is $8 \cdot 10^6$. The minimum value extracted from the simulation model is 761.7. Therefore, the dynamic range for the range Doppler image can be obtained as:

$$DR_{RD\_Image} = 10\ \log_{10}\left(8 \cdot 10^6\right) - 10\ \log_{10}\left(761\right) = 40.2\ \text{dB} \tag{7.4}$$

The ISAR image dynamic range in (7.4) is higher than that calculated for the DIS range-bin outputs in (7.2). This can be explained since the range-bin output samples are not resolved in Doppler as the ISAR image pixels. It can be concluded that the DIS dynamic range cannot be related to the image dynamic range, unless there is a unique Doppler component per range bin. This can be simulated using a point target.

C. IMAGE DYNAMIC RANGE OF A POINT TARGET

A single scatterer with a unitary radar cross-section and a Doppler component of 50 Hz is defined as the false target. The model is run integrating 128 pulses. Both the target template and the induced range Doppler image at the ISAR are shown in Figure 51.

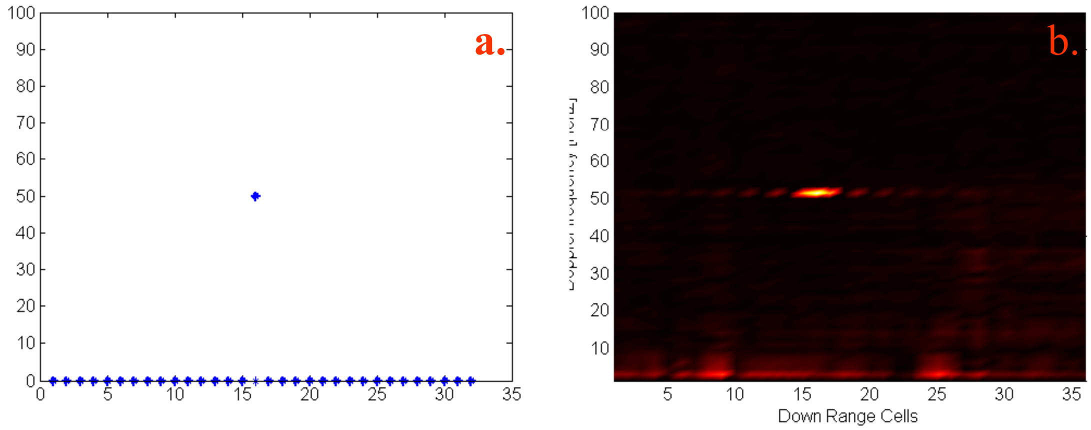


Figure 51 Unitary RCS Point Target: a) Target Template. b) Range-Doppler Image.

The generated point target appeared in the ISAR range Doppler profile at the 33rd Doppler filter, corresponding to 50 Hz. A smearing effect in the range dimension is apparent. This effect can be observed in detail by extracting the magnitudes across the 33rd Doppler filter as shown in Figure 52.

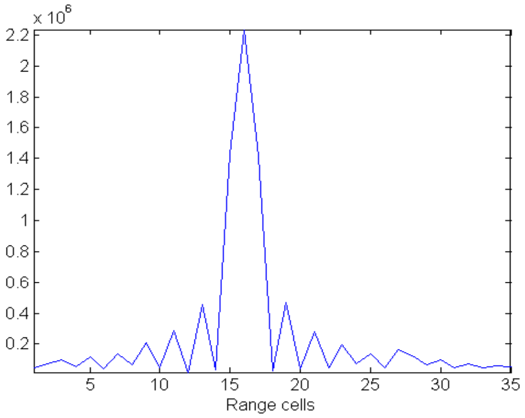


Figure 52 Magnitude Plot of the Doppler Filter Containing the Point Target

The smearing effect across the range dimension is due to the superposition of complex scatterer samples from different range bins at the DIS output adders. Ideally, this range superposition is reversed at the ISAR range compression but imperfections in the idealistic approximation of the inverse Fourier transform shifts the range location of a few samples following the Sinc distribution noted in Figure 52. This effect is identified in Equation (2.26) as the PSF of source deconvolution. The smearing effect is noted better in the three plots shown in Figure 53. While the range dimension in Figure 53c shows a wide spike at the scatterer location, the Doppler dimension shows narrow spike in Figure 53b.

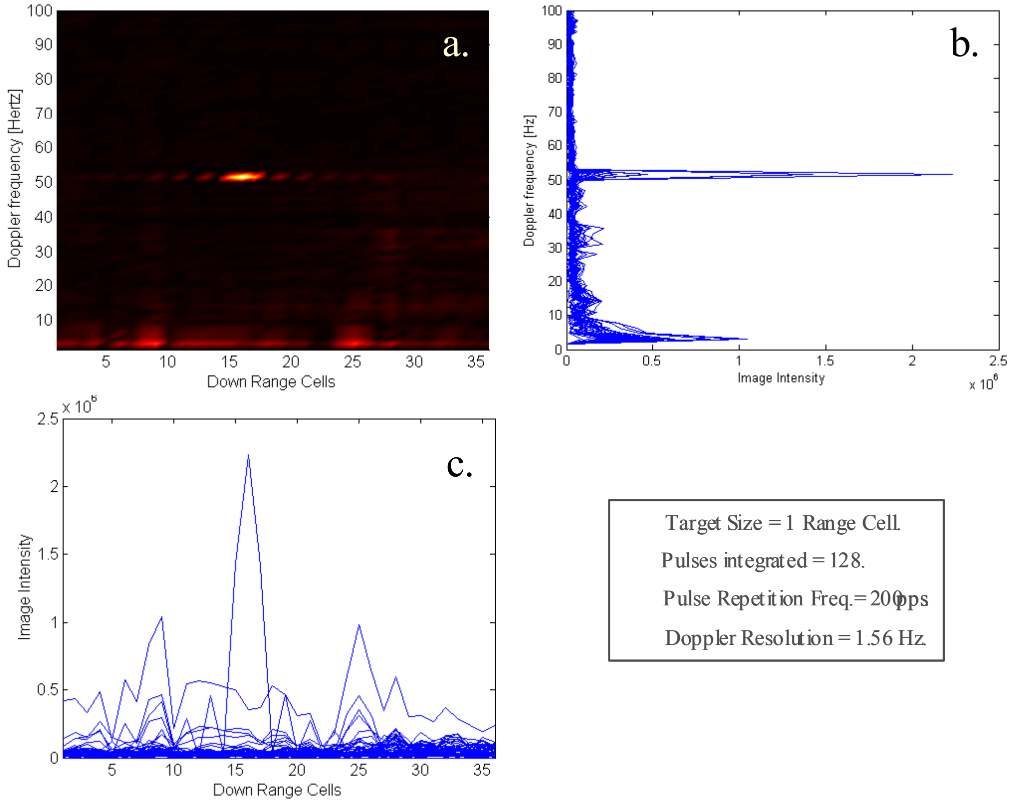


Figure 53 Point Target Range Doppler Image. a) 2D.View. b) 1D. View Along the Doppler Axis. c) 1D.View Along the Range Axis.

A logarithmic plot of the point target image intensities is presented in Figure 54. From the maximum and minimum intensity values, the Dynamic range for the point target image can be obtained as:

$$DR_{Point\ Target\ Image} = 64 - 25 = 39 \text{ dB} \tag{7.5}$$

which is only 1 dB smaller from the dynamic range image of a ship-like target in (7.4). It can be observed that the target scatter distribution barely affects the image intensity dynamic range.

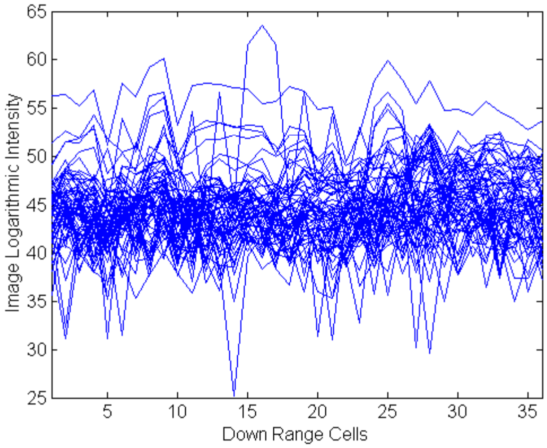


Figure 54 One Dimensional Logarithmic Plot of the Point Target Image.

D. CHAPTER SUMMARY

This chapter studied the amplitude range of the DIS samples in order to relate them to the dynamic ratio of the ISAR azimuth compression output, which in turn determines the intensity contrast in the range-Doppler image. The analysis was done for both a test ship target and a point source target. The results showed that the dynamic range is reduced from 72 dB in the DIS samples at the output of the DIS range bins, to 40.4 dB in the Range Doppler image. Similar results are obtained for both the ship target and the point source target.

The simulation studies presented in Chapter V, VI and VII used a 32-range-bin target to reduce the long simulation times that would involve larger targets. In the next chapter, the current version 512-range-bin chip is simulated.

VIII. SYNTHESIS OF A 512-RANGE CELL TARGET

The existing DIS simulation models used previous to this thesis do not run properly with target extents having more than 256 range cells. This chapter explains the code changes that are implemented to extend the simulation capabilities to any target size and describes a simulation run for a 512-range cell target.

A. SOFTWARE MODEL MODIFICATION REQUIREMENTS

The original software code of the *mathost.m* file created a chirp wave where the first sample is repeated a specific number of times depending on the number of range bins of the DIS array (target extent). The repetition of the first phase originates from an earlier DIS model in which the DRFM samples were fed serially, modeling the sample delay between taps at their inputs. The DIS model was eventually changed to the current model where the DRFM samples are fed in parallel to all range bins and the delay between them is implemented in the output adder.

The default values for the pulsewidth and sampling rate are 0.5 us and 1 GHz respectively. The resulting number of intra pulse samples is their product, equivalent to 500 samples. For the case of a 32-range-bin target, the old model repeated the first phase sample 32 times. Thus, the chirp that is processed by the DIS routine, and later by the ISAR compression routines had an overhead of 32 samples (6%). Nonetheless, the model worked correctly and synthesized a clear image of the intended target. However for a target of 512 range cells, there are more overhead samples than the actual chirp phase samples (50.6%). The initial solution to this problem was to increase the number of intra pulse samples by increasing the sampling rate. This solution demanded an excessive processing load with long simulation time. The definite solution for this problem is to eliminate the overhead of repeated samples and apply code changes in the *simhwchk_v5.m* file to ensure that each range bin has a zero output until the time previous range bin has already generated a returned scattering sample.

B. SIMULATION RUN

A Doppler template for a 512-cell target is defined in the *target512.mat* data file. Once loaded, the simulation model displayed the scatterer arrangement, as shown in

Figure 55. The scatterer complex values are processed using Equations (3.2) through (3.6) to obtain magnitude and phase modulation coefficients.

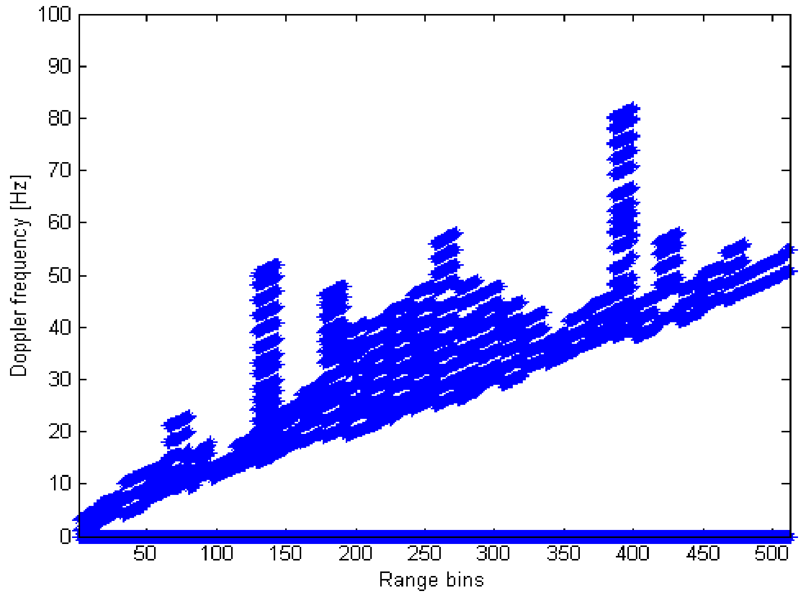


Figure 55 Scatterer Distribution for a 512-Range-bin Target Using 128 Doppler Cells.

The distribution of magnitude coefficients is arranged in a range vs. pulse index map in Figure 56. This plot does not show the symmetry condition observed for the gain coefficient distribution for a 32-range cell target.

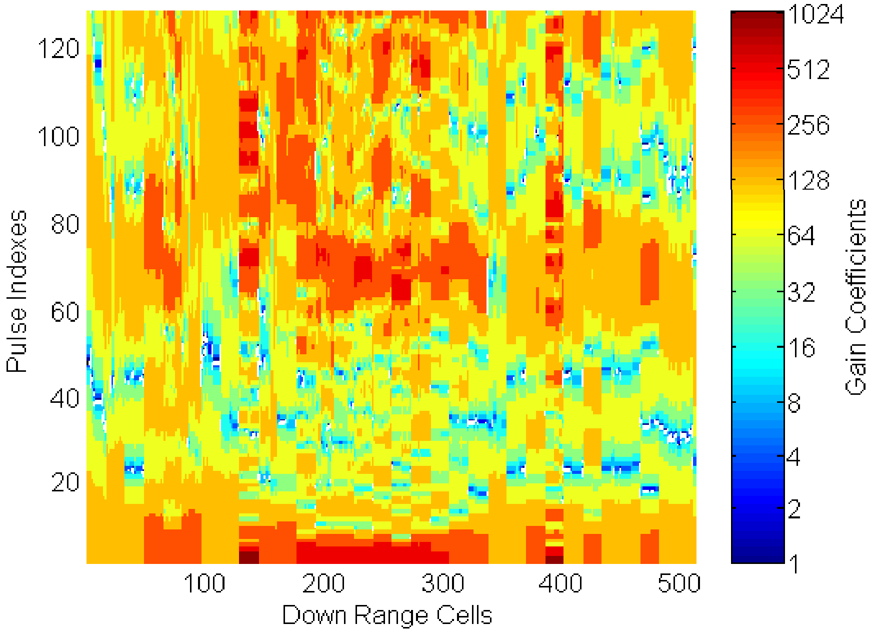


Figure 56 Gain Coefficient Distribution on a Range Cell-Pulse Index Map.

The range vs. pulse index distribution of the phase modulation coefficients is presented in Figure 57.

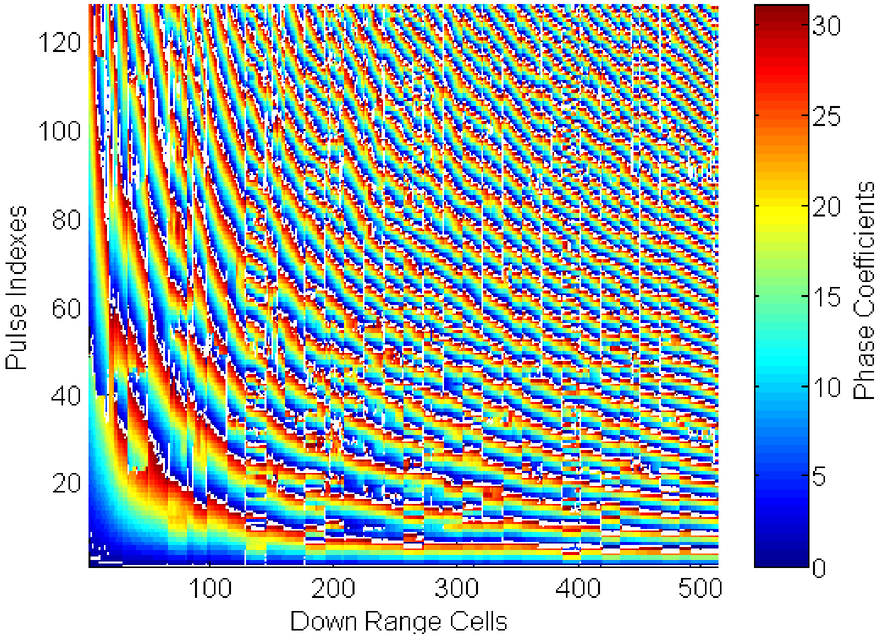


Figure 57 Phase Coefficient Distribution on a Range Cell-Pulse Index Map.

The *mathost_v5.m* file generates a chirp-pulse train, with each pulse having the waveform as displayed in Figure 58.

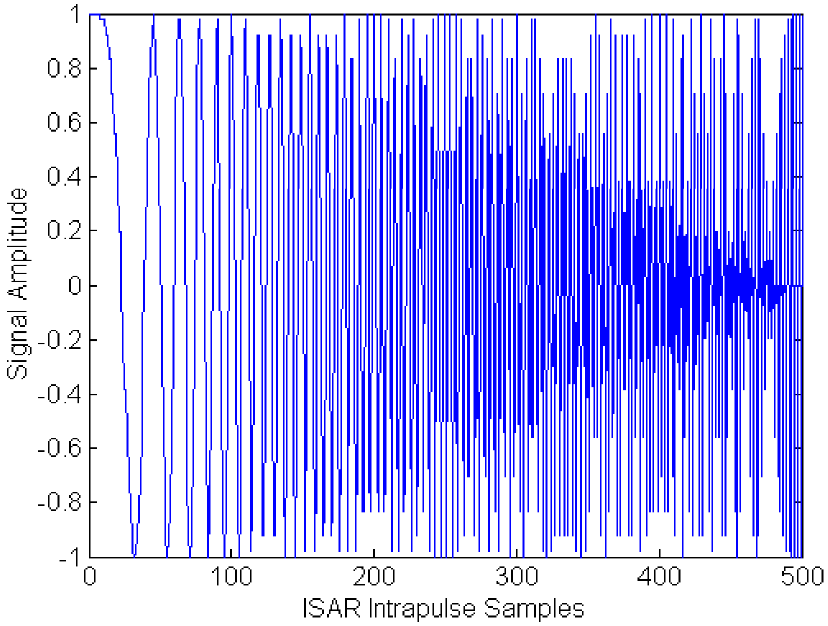


Figure 58 Baseband Plot of the Intercepted ISAR Pulse as Generated by a DRFM.

The *simhwchk_v5.m* file processes the 128 pulses of the pulse train applying the gain and phase modulation coefficients depending on the range tap of the DIS array and the index of the radar pulse being processed. The resulting DIS output signal is a pulse train with modulated waveforms. Figure 59 presents the DIS output waveform for the first pulse.

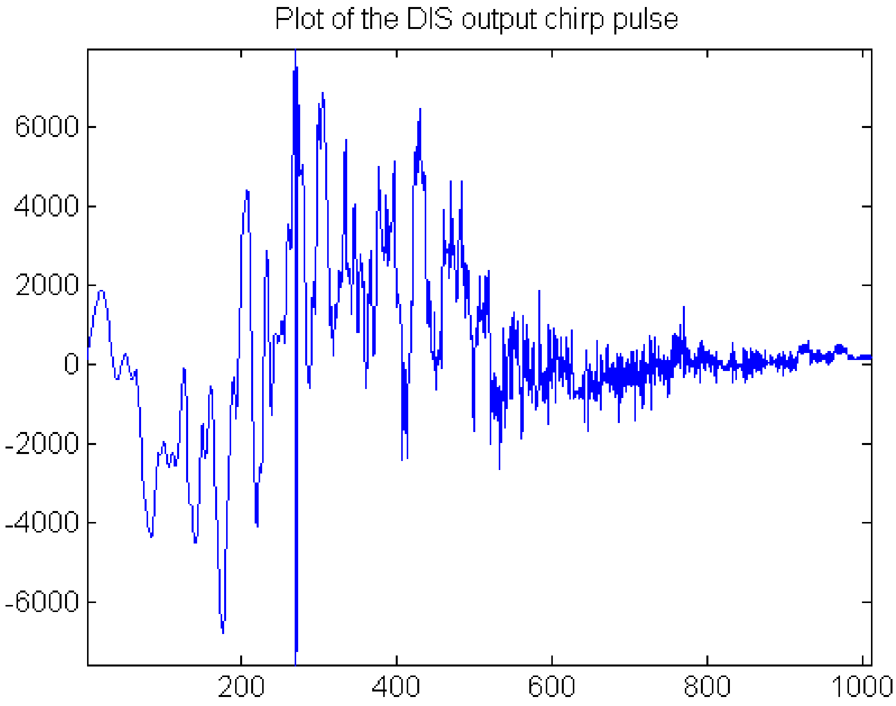


Figure 59 Plot of the DIS Output Modulated Chirp

The ISAR routine of the *simhwchk_v5.m* file processes each pulse returned to the DIS with a range compression process. For the case of the simulation model, this process is implemented with a fast cross-correlation as described in Figure 6. The FFT of the first returned signal is presented in Figure 60.

The FFT of the chirp reference signal is presented in Figure 61. After complex conjugation of the latter, both FFT signals are multiplied element by element. The resulting Fourier domain signal is presented in Figure 62.

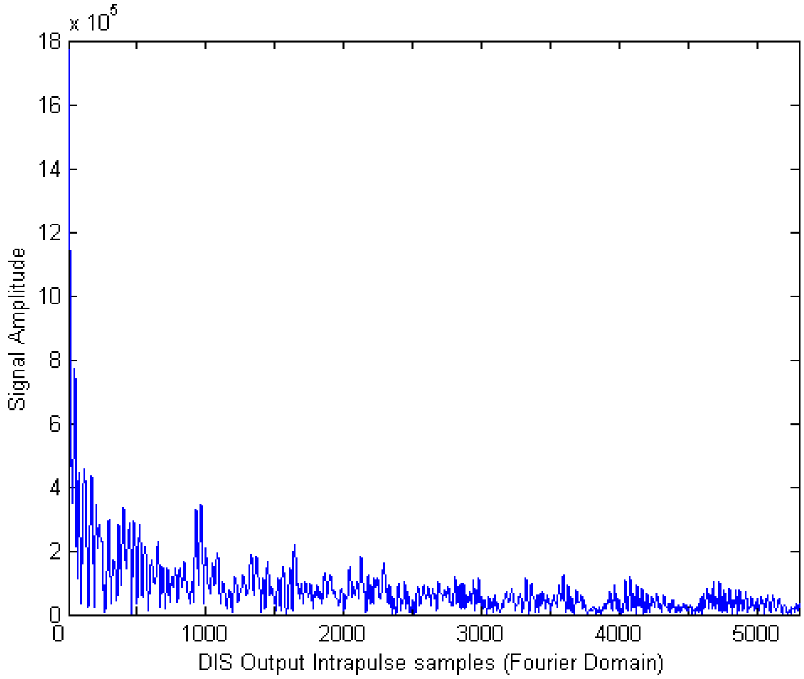


Figure 60 FFT of the DIS Output Signal.

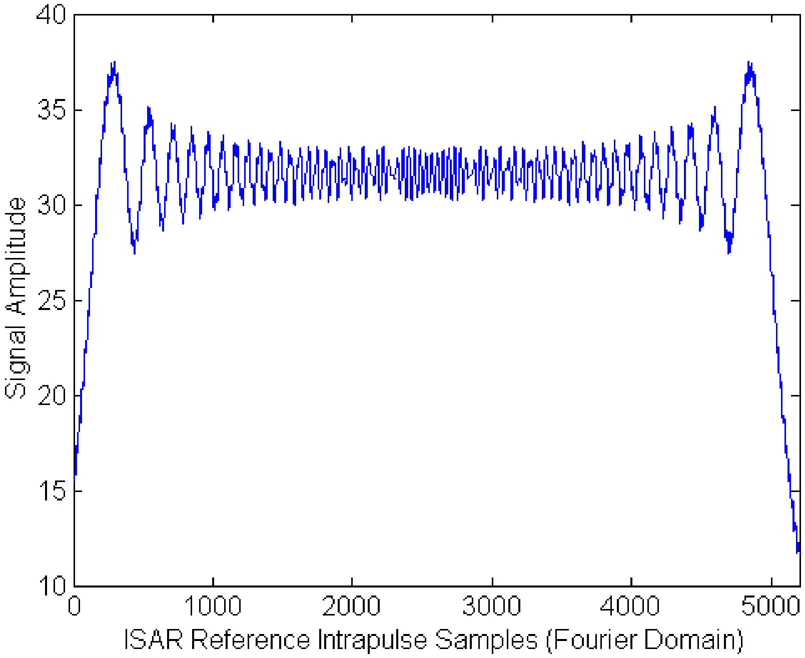


Figure 61 FFT of the Chirp Reference Signal.

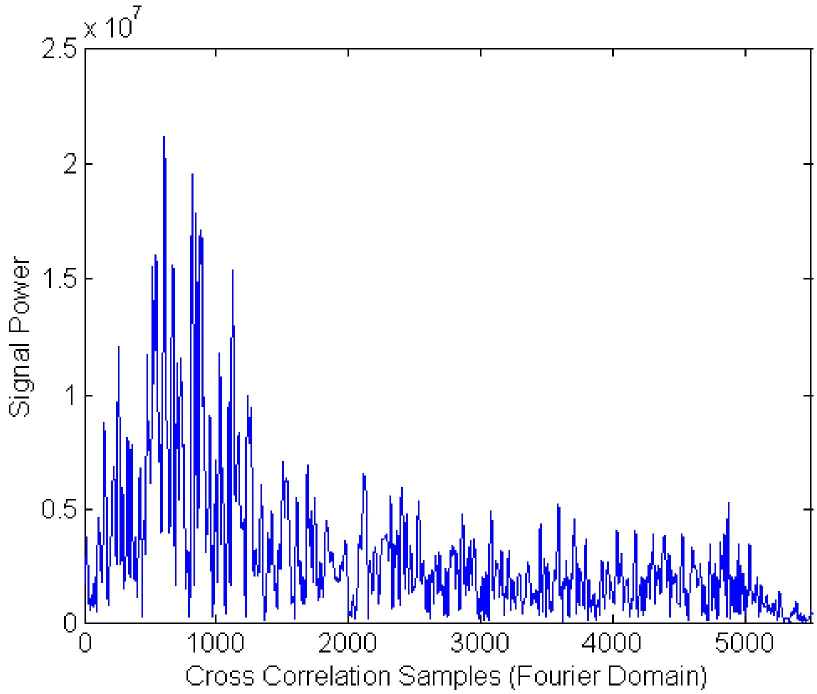


Figure 62 FFT Product Signal at the Pulse Fast Correlation Output.

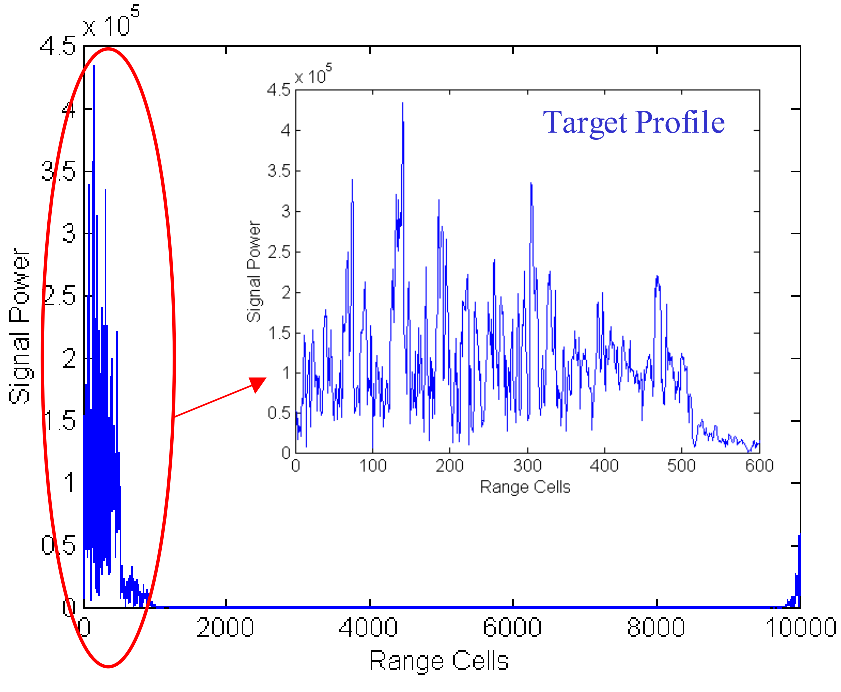


Figure 63 Fast Correlation Output Showing the Target Range Profile.

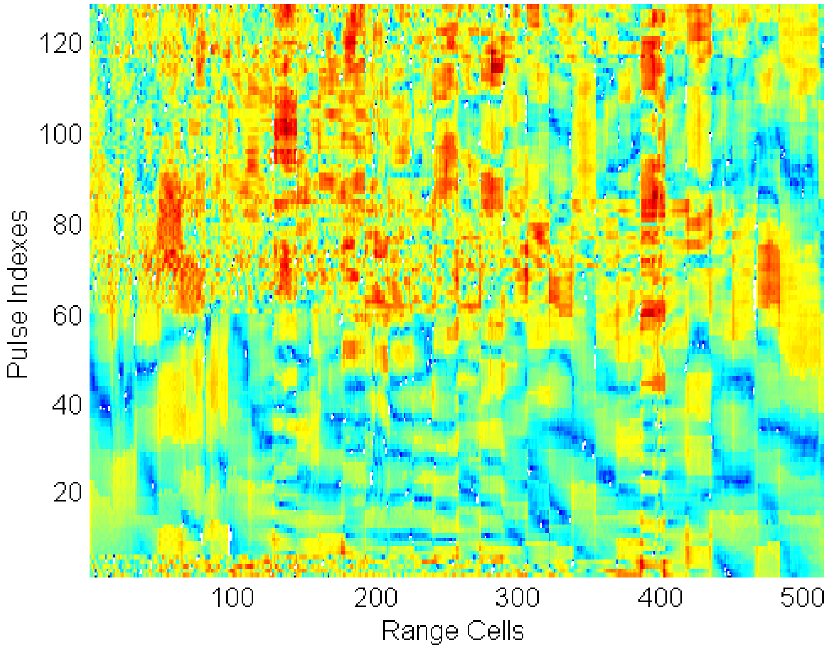


Figure 64 Surface Plot of Range Profile Matrix.

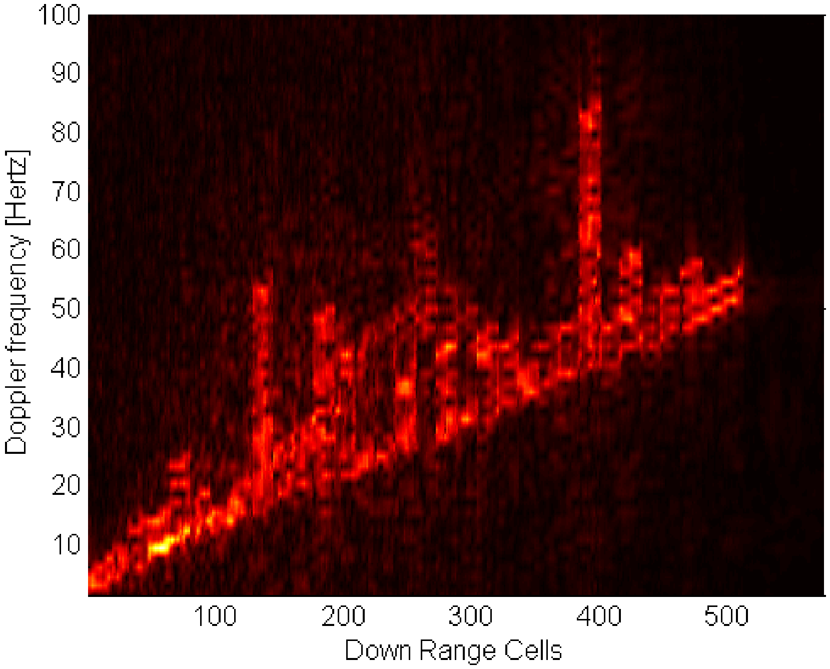


Figure 65 Range Doppler Image for a False Target of 512 Range Cells.

C. CHAPTER SUMMARY

This chapter presented the simulation model for the current DIS chip design with 512 range-bin processors. From the values selected for DRFM sampling frequency (1 GHz) and radar PW (5us), the resulting number of samples at each pulse is 500. Previous DIS software models were unable to simulate a scenario where the target extent had as many range bins as the number of samples in a radar pulse. The software improvements applied to this model made possible a simulation of a target whose extent in range cells (512) was superior to the number of radar pulse samples (500). It is expected that the model would perform similarly for larger targets.

The new version of the DIS software model is able to represent the current DIS hardware design. The simulation results at this level showed that the hardware design works properly. The results must be supported by lower level DIS hardware simulations using VHDL.

IX. CONCLUSIONS AND RECOMMENDATIONS

This thesis investigated the theoretical operation of the DIS integrated circuit as an ISAR countermeasure and developed a software simulation model to test the performance. The software model simulates the main signal processing steps that take place in the DIS and later in the ISAR receiver and processor. The extraction and plotting of signals processed by the simulation model provided a good tool to study the DIS design in depth.

A. CONCLUSIONS

- The DIS software simulation model provided with this thesis is better than the predecessors in being closer in the resemblance to the actual hardware, faster to execute and shorter in code extension. Thus, the DIS design team has a more efficient and easier to understand simulation tool.
- A method to evaluate the DIS performance based on image comparison from an ideal model was integrated. The performance under the different simulation scenarios could be evaluated based on the quality of the output image and the similarity to the ideal model image. The image comparison tool made it possible to isolate individual processing stages in order to find the major contributors to the overall image error. This tool can also be used to measure the impact of future design changes at any stage of the DIS process.
- The output adder overflow was identified as the main source of image error. This thesis suggests that the overflow can be managed by solely controlling the thresholding values that assign the highest DIS gain coefficient value (1024), thus preventing any hardware design change.
- Software improvements in the simulation code enabled the modeling of a target with an extension of 512 range cells. These code changes eliminated a theoretical limit for the simulation even bigger targets.

- The simulation model was efficiently used to produce input and output vectors for VHDL simulations for DIS arrays of different sizes. Output results of both the MATLAB simulation model and the VHDL model matched perfectly.

- The software model has been described thoroughly in order to make it easier to understand when implementing future code changes.

B. RECOMMENDATIONS

- The software code can still be improved by substituting for some of the iterative loop structures with vector and matrix operations. These changes would increase the percentage of parallel processing operations, thus greatly improving the simulation time. Additional time saving changes include the reduction of intermediate variables, the use of column operations (vice rows) and the reduction of conditional (if-then) operations. Care must be taken to keep the software code easy to understand if excessive reduction in code lines is achieved.

- The performance of the DIS against stepped chirp pulses can be added to this study. From the quantization analysis obtained from chirp-pulse cases, it can be predicted that the DIS should be able to deceive stepped chirp pulses that resemble a discrete linear FM chirp and scrambled frequency stepping pulses (discrete chirp with scrambling of pulse samples). In the particular case of interpulse frequency stepping where each pulse has a different constant frequency, the DIS should not be effective.

APPENDIX A. MATLAB CODES

1. DIS SIMULATION FILES – VERSION 5

a. extract_para_ship_v5.m

```
%**************************************************************************************************
%  extract_para_ship_v5.m                                                                          *
%                                                                                                  *
%  - Extracts modulation coefficients (phase and magnitude) from a false                           *
%  target to be used by DIS simulation files.                                                      *
%  - Runs 'doppler_XX_target.m' to create a Range-Doppler target profile.                          *
%  - Saves modulation coefficients in 'paraMULTIq4Ship.txt'.                                       *
%                                                                                                  *
%  Modified by F.LeDantec 7/7/02 based on previous version from P.E.Pace                           *
%                                                                                                  *
%**************************************************************************************************

clear all;
global dp_pts rg_pts rag_pts numtaps t0 k prf nbitsdop

numtaps = 32                        % # of taps (target extent)
dp_pts  = 128                       % # of Doppler bins (number of pulses integrated)
bw2     = 500e6                     % Bandwidth of the chirp signal, delta
fs      = 2 * bw2                   % Sampling frequency
prf     = 200;                      % Pulse repetition frequency

nbitsdop = 5;                       % # of bits used for phase quantization
pwc   = 1/(bw2);                    % Compressed pulsewidth
pw    = 0.5e-6*200/prf;             % Uncompressed pulsewidth
pri   = 1/prf;                      % Pulse repetition interval
k     = bw2/pw;                     % Pulse compression rate (deltaBW/uncompr.pWidth)
Ts    = 1/fs;                       % Sampling time
t0    = Ts:Ts:pw;                   % Interpulse sample index
rag_pts = length(t0);               % Number of samples per chirp pulse (range cells)
rg_pts = max(rag_pts,numtaps*10); % # of range FFT pts including padding
p = 2*pi/(2^nbitsdop);              % Quantization factor
tstop = pri*dp_pts;                 % Pulse train duration (integration time)
ti = linspace(0,tstop,dp_pts);      % Vector with pulse leading edge times

switch numtaps
case 2,
   doppler_02;                      % Creates and saves a 2 range-bin profile
   load target02                    % Loads profile
case 32,
   doppler_32;                      % Creates and saves a 32 range-bin profile
   load target32                    % Loads profile
case 128,
   doppler_128;                     % Creates and saves a 32 range-bin profile
   load target128                   % Loads profile
case 512,
   doppler_512;                     % Creates and saves a 512 range-bin profile
   load target512                   % Loads profile
end
```

```
if dp_pts < 1024,
   freq = freq/10;                             % Doppler profile is reduced (slower rotation)
end;

% PLOT OF THE FALSE TARGET RANGE-DOPPLER PROFILE :

figure;
colordef black
plot(freq,'y*');
nt  = num2str(numtaps);
nd = num2str(dp_pts);
title([ nt  '-range bin False Target Profile - '  nd  ' Radar Pulses' ])
xlabel('Range bins');
ylabel('Doppler frequency [Hz]');
axis([0 fix(numtaps/8*9) 0 128]);
axis square


% CREATION OF A COMBINED SIGNAL AT EACH RANGE GATE, BY SUPERPOSITION OF
% COMPLEX RETURNS, TO OBTAIN A 1 DIM RANGE PROFILE FOR EACH PULSE :

A = ones(numtaps,dp_pts);                     % Scatterer Amplitudes (uniform)
targetSum = zeros(numtaps,dp_pts);
for i1 = 1:numtaps,                           % At each range bin
   target = zeros(dp_pts,length(ti));
   for i2 = 1:numfreq(i1),                    % At each scatterer
      for i3 = 1:length(ti),                  % At each integrated pulse
         target(i2,i3) = A(i1,i2) * exp(-j*2*pi*freq(i1,i2)*ti(1,i3));
      end                                     % Creates complex scatterer
   end
   for i4 = 1:numfreq(i1)                     % Combined signal p/rbin p/pulse
      targetSum(i1,:) = targetSum(i1,:) + target(i4,:);
   end
end

% CREATION OF GAIN MODULATION COEFFICIENTS :

amp = abs(targetSum);                         % Magnitude of combined signal.
gain = ones(numtaps,dp_pts);                  % Default gain coefficient is 1.
fq = 1.5;                                     % Arbitrary threshold factor
for i1 = 1:numtaps,
   for i2 = 1:dp_pts,
      normAmp = amp(i1,i2)/max(max(amp));     % Magnitudes are normalized.
      for s=1:10,
         if normAmp > (2^-(11-s))*fq,         % Thresh: 0.0015....,0.375,0.75
            gain(i1,i2) = 2^s;                % Gain:   2,4,8,16,......512,1024
         end                                  % Assigns 11 power-of-two gains.
      end
   end
end

% CREATION OF PHASE MODULATION COEFFICIENTS :

vphase1  = angle(targetSum);                  % Phase of combined signal
vphase2  = vphase1*2^nbitsdop/(2*pi);         % Adjusts values from 0 to 31.99
```

```
phaseinc = zeros(numtaps,dp_pts);
for i6 = 1:numtaps,                              % Turn phases into phase increments
  for i7 = 1:dp_pts,
           if i7==1,
              phaseinc(i6,i7) = phaseinc(i6,i7)+vphase2(i6,i7);
           else                                  % Finds phase increments bet/pulses
      phaseinc(i6,i7) = phaseinc(i6,i7-1)+vphase2(i6,i7-1)-vphase2(i6,i7);
           end
  end
end
for i8 = 1:dp_pts,
   for i9 = 1:numtaps,                           % 5-bit quantization in 32 levels.
      phasecoeff(i9,i8) = fix(rem(phaseinc(i9,i8)+32,32));
   end
end

% MODULATION COEFFICIENTS ARE STORED SERIALLY, SO THEY CAN BE LOADED BY
% THE DIS SIMULATION PROGRAM :

f4 = fopen('paraMULTIq4Ship.txt','w');           % Stores modulation coefficients
fprintf(f4,'%d\r\n',rag_pts);                    % Store # of chirp pulse samples
fprintf(f4,'%d\r\n',dp_pts);                     % Store # of Doppler bins
fprintf(f4,'%d\r\n',numtaps);                    % Store # of taplines (target extent)
for aa=1:dp_pts,
   for bb=1:numtaps,
      fprintf(f4,'%d\r\n',gain(bb,aa));          % Stores gain coefficients
   end
end
for aa = 1:dp_pts,
   for bb = 1:numtaps,
      fprintf(f4,'%d\r\n',phasecoeff(bb,aa)); % Stores phase coefficients
   end
end
fclose(f4);

% PLOT OF THE GAIN QUANTIZATION SCHEME
figure
plot(amp/max(max(amp)),gain,'*');                % Plots gain coefficient distrib.
```

b. mathost_v5.m

```
%****************************************************************************************************
%  mathost_v5.m                                                                                    *
%                                                                                                  *
%  - Creates and stores an intercepted chirp pulse as generated by a                               *
%  phase sampling DRFM.                                                                            *
%  - Creates the reference signal required for ISAR range compression.                             *
%  - Runs 'simhwchk_v5.m' to execute the DIS and ISAR simulation.                                  *
%  - Runs 'image_analysis.m' to compare image w/infinite resolution case.                          *
%                                                                                                  *
%  Modified by F.LeDantec 7/7/02 based on previous version from P.E.Pace                           *
%                                                                                                  *
%****************************************************************************************************

global dp_pts rg_pts numtaps rag_pts k t0 prf nbitsdop
```

```
% GENERATION OF DRFM SIGNAL CONTAINING PHASE SAMPLES FOR ALL PULSES :
pri_rg_phaseq = zeros(dp_pts,rag_pts);          % Initializes DRFM phase matrix
p = 2*pi/(2^nbitsdop);                          % Phase quantization factor
pri = 1/prf;
for idx1 = 1:dp_pts,                            % Loop per pulse
  t1 = t0 + idx1*pri;                           % PRI time shift for pulse samples
  oldphase = 2*pi*(k*t1.*t1)/2;                 % Chirp pulse phase samples
  oldphase = rem(oldphase,2*pi);                % limit phase from 0 to 2pi
  pri_rg_phaseq(idx1,:) = fix(oldphase/p);      % Phase quantized in 2^nbit levels
  crefc(:,idx1) = exp(j*oldphase.');            % Store phase for ISAR reference
end

% STORAGE OF DRFM PHASE MATRIX :
f2 = fopen('rawint.txt','w');                   % Opens file for DRFM ph. storage
for i = 1:dp_pts,                               % Loop for all pulses.
  int_raw = pri_rg_phaseq(i,1:rag_pts-1);
  fprintf(f2,'%d,',int_raw);                    % Stores pulse phase samples
  int_raw = pri_rg_phaseq(i,rag_pts);
  fprintf(f2,'%d\r\n',int_raw);                 % Stores last pulse phase sample
end;
fclose(f2);

% GENERATION OF ISAR REFERENCE FOR RANGE COMPRESSION (FAST CORRELAT.) :
cref = conj(fft(crefc,2*rg_pts-1));             % FFT of ISAR reference
save pc_ref cref t1                             % Saves FFT of ISAR reference.

% DIS SIMULATION, IMAGE GENERATION AN ANALYSIS :
simhwchk_v5;                                    % DIS simulation & ISAR procesing
if numtaps==32 & nDopplerCell==128 & prf==200,
    Image_Analysis;                             % Comparison with ideal image
end;
```

c. simhwchk_v5.m

```
%****************************************************************************************************
%  simhwchk_v5.m                                                                                   *
%                                                                                                  *
%  Simulates the DIS architecture by:                                                              *
%        - Loading its inputs from files 'paraMULTIq4Ship.txt' and 'rawint.txt'                    *
%        - Processing the modulations of the tap array.                                            *
%        - Generating the output chirp.                                                            *
%  Simulates the ISAR processing and plots the range-Doppler profile of                            *
%  the false target as seen in the radar screen.                                                   *
%                                                                                                  *
%  Modified by F.LeDantec 7/7/02 based on previous version from P.E.Pace                           *
%                                                                                                  *
%****************************************************************************************************

clear
global rg_pts prf

fid = fopen('paraMULTIq4Ship.txt','r');         % Opens file w/DIS coefficients
tmp = fscanf(fid,'%f');                         % Loads file contents
nRangeCell = tmp(1);                            % number of pulse samples
```

```
nDopplerCell = tmp(2);                                   % number of pulses integrated
numtaps = tmp(3);                                        % number of DIS range taps
gainIn = tmp(4:4+numtaps*nDopplerCell-1);                % Extracts gain coefficients
gain = reshape(gainIn,numtaps,nDopplerCell);             % Arranges them in a matrix
gain = gain';                                            % Transpose
phi = tmp(4+numtaps*nDopplerCell:end);                   % Extracts gain coefficients
phasInc = reshape(phi,numtaps,nDopplerCell);             % Arranges them in a matrix
fclose(fid);
raw = zeros(nDopplerCell,nRangeCell);                    % Initializes DRFM phase matrix
fid = fopen('rawint.txt','r');                           % Opens file w/DRFM samples
for r = 1:nDopplerCell,                                  % Loop per pulse
   for i2 = 1:nRangeCell-1,                              % Loop per sample
      raw(r,i2)=fscanf(fid,'%f',1);                      % Loads DRFM phase sample
      comma = fscanf(fid,'%c',1);                        % Loads comma character
   end
   raw(r,nRangeCell) = fscanf(fid,'%f',1);               % Last pulse phase sample
end
fclose(fid);
[row,col] = size(raw);
phaDRFM = [raw,zeros(row,numtaps-1)];                    % Trailing zeros to clear taps

colordef white
set(0,'defaultAxesFontSize',12);                         % Font size for graph labels
figure                                                   % Plots one intercepted pulse
plot( real( exp(j*raw( 1,1:nRangeCell) /32*2*pi ) ) )
title('Plot of an intercepted ISAR chirp as generated by a DRFM');

depthLUT = 32;                                           % Size of the LUT table
load -ascii cosine8.txt                                  % Loads LUT tables
load -ascii sine8.txt
lutOut  = zeros(numtaps,1);                              % Initializes LUT output
tapOut  = zeros(nRangeCell + (numtaps-1),numtaps);
o_res   = 0.25;                                          % Minimum output bit resolution
overflow= 0;                                             % Initializes overflow counter
peak    = 0;                                             % Initializes maximum amplitude
limit   = 8192;                                          % Adder overflow limit 2^13
partial_tapsum = zeros(nRangeCell+numtaps-1,1);

for batchCnt = 1:nDopplerCell,                           % Loop per ISAR pulse
   disp(['Processing Pulse ',num2str(batchCnt)]);
   for inPlsCnt = 1:(nRangeCell+numtaps-1),              % Per out plse sample
      tapIn(1:numtaps) = phaDRFM(batchCnt,inPlsCnt);     % Loads DRFM Pha in//
      phAddOut= tapIn(1:numtaps)'+phasInc(:,batchCnt);   % Pha adder output
      tmp = rem(phAddOut+depthLUT,depthLUT)+1;           % Phas limit bet 0-31
      lutOut = cosine8(tmp) + j*sine8(tmp);              % LUT table output
      gainOut = gain(batchCnt,:)'.*lutOut;               % Gain modulation
      if inPlsCnt > nRangeCell,                          % Shuts down taps
         gain0ut(1:(inPlsCnt-nRangeCell)) = 0;           % after pulse passes
      end
      for idx3=1:numtaps,
         delay = idx3-1;                                 % Out.adder register delay bet/taps
         tapOut(inPlsCnt+delay,idx3)=fix(gainOut(idx3)/o_res)*o_res;
      end                                                % Applies minimum register resolution
      sum1=0;
      for tt=min(inPlsCnt,numtaps):-1:max(1,inPlsCnt-nRangeCell),
        sum1 = sum1 + tapOut(inPlsCnt,tt);
        Iout=real(sum1);                                  % Overflow simulation using +/-limit
        Qout=imag(sum1);
        if Iout >= limit,                                 % Positive overflow (in-phase)
           sum1 = Iout-2*limit + j*Qout;                  % Adder output w/overflow
           overflow = overflow + 1;                       % Increases counter
        elseif Iout < -limit,                             % Negative overflow (quad-pha)
           sum1 = Iout+2*limit + j*Qout;                  % Adder output w/overflow
           overflow = overflow + 1;                       % Increases counter
        end
        if Qout >= limit,                                 % Positive overflow (in-phase)
           sum1 = Iout + j*(Qout-2*limit);                % Adder output w/overflow
           overflow = overflow + 1;                       % Increases counter
        elseif Qout < -limit,                             % Negative overflow (quad-pha)
           sum1 = Iout + j*(Qout+2*limit);                % Adder output w/overflow
           overflow = overflow + 1;                       % Increases counter
        end
        peak=max(max(abs(Iout),abs(Qout)),peak);          % End of overflow simulation.
        partial_tapsum(inPlsCnt,1) = sum1;
      end
   end
   finalAdderOut(batchCnt,:)=partial_tapsum.';            % DIS output mod. chirps
end
overflow                                                  % Number of DIS overflows
peak                                                      % Maximum DIS amplitude
dac_res = 0.25;                                           % DAC resol 0.25 for 16bit
finalAdderOut = fix(finalAdderOut/dac_res)*dac_res;
figure
plot( real( finalAdderOut(5,:) )),axis tight              % Plots 1 DIS output chirp pulse
title('Plot of the DIS output chirp pulse');

% ISAR PROCESSING :

load pc_ref                                               % Loads FFT of ISAR reference
priRgMapShift = zeros(fix(numtaps*9/8),nDopplerCell);
   % Range compression:
pcRefMapShift = fft(finalAdderOut.',2*rg_pts-1).';        % FFT of DIS output
for idx = 1:nDopplerCell,                                 % For all pulses.
    tmp  = cref(:,idx).*pcRefMapShift(idx,:).';          % Fast correlat.(FFT product)
    tmp2 = ifft(tmp);                                     % InvFFT to obtain range profile
    priRgMapShift(:,idx) = tmp2( 1:fix(numtaps*9/8) );    % Combined range profile per pulse
end
   % Azimuth compression:
   % FFT of matrix of range profiles in the pulse index dimension (time)
   % gives the target range Doppler profile.
dpRgMapShiftMOD = abs(fft(priRgMapShift.',nDopplerCell));
quantimage = dpRgMapShiftMOD(1:nDopplerCell*100/prf,1:fix(numtaps/8*9));
save finresoldata quantimage                              % Saves Range Doppler profile

% ISAR IMAGE PLOTS (Range Doppler profile):

dopp = (1:size(quantimage,1))*prf/nDopplerCell;           % Doppler axis
rang = 1:size(quantimage,2);                              % Range axis
colordef black                                            % Background color
figure;
Ncontours = 120;                                          % # of plot contour lines
contour(rang,dopp,quantimage,Ncontours);                  % Contour Plot
```

```
%title('Range-Doppler profile of a false target');
xlabel('Down Range Cells');
ylabel('Doppler frequency [Hertz]');
axis tight
figure;
colormap(hot(100))
surf(rang,dopp,quantimage), shading interp, lighting phong, view(0,90);
%title('Range-Doppler profile of a false target');
xlabel('Down Range Cells');
ylabel('Doppler frequency [Hertz]');
axis tight
```

d. image_analysis.m

```
%****************************************************************************************************
%  image_analysis.m                                                                                  *
%                                                                                                    *
%  - Load the matrix representing the real DIS range-Doppler image.                                  *
%  - Load the matrix representing the ideal DIS image range-Doppler.                                 *
%  - Calculates the Mean Squared Error coefficient (average noise energy)                            *
%  - Calculates the Two-dimensional correlation coefficient.                                         *
%  - Plots the noise energy image distribution.                                                      *
%                                                                                                    *
%  Fernando LeDantec 7/7/02.                                                                         *
%                                                                                                    *
%***************************************************************************************************%

dp_pts  = 128;                                        % Number of Doppler cells
numtaps = 32;                                         % Number of range cells
load finresoldata                                     % Loads real DIS image
load infresoldata                                     % Loads ideal DIS image

normimage      = image/max(max(image));               % Normalizes real image.
normquantimage = quantimage/max(max(quantimage));     % Normalizes ideal image.

for m=1:dp_pts/2,
   for n=1:numtaps*1.0625,                             .
     noiseEnergy(m,n) = (normimage(m,n)-normquantimage(m,n))^2;        % Pixel square error.
   end;
end;

[a,b] = size(noiseEnergy);
noise_ave = sum(sum(noiseEnergy))/(a*b)               % MSE coefficient.

corel_coeff = corr2(image,quantimage)                 % 2D correlation coeff.

figure;                                               % Plots Noise energy.
dopp = (1:a)*prf/nDopplerCell;                        % Doppler axis
rang = 1:b;                                           % Range axis
colormap(hot(100))
surf(rang,dopp,noiseEnergy),lighting phong, view(0,90);
title('Range-Doppler Map of the Quantization and Overflow Noise');
xlabel('Range Cells');
ylabel('Doppler Cells');
axis tight;
```

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX B. DIGITAL RF MEMORY

1. DRFM THEORY

Most military radars currently operate using coherent processing, where the signal phase is preserved for detection and performance and low vulnerability to repeater jamming. Among EA target-like jamming techniques are those that radiate a signal similar to the signal scattered by the target. Selected signal parameters are then changed to deceive or confuse the radar operator. The advent of coherent radar processing proved to be very powerful to sustain non-coherent deception and confusion jamming techniques with older repeaters. Jamming technology was forced to use coherent techniques. The development of the Digital Radio Frequency Memory (DRFM) provided a technological breakthrough to achieve this effort.

The DRFM concept was developed to store a signal and replicate it coherently. Thus, a DRFM can be used for coherent jamming techniques. Figure 66 shows a block diagram representation of a DRFM. The first stage is the frequency down-conversion of the signal into an intermediate frequency (IF). This is implemented using two very stable local oscillators (LO) in quadrature that generate two channels I and Q. High stability of these oscillators is essential in order to preserve the phase information of the signal. Both channels are fed into ADCs for sampling at intervals of a few microseconds. The samples are quantized into digital formats and stored in digital Random Access Memory (RAM). Once the signal is in memory, it is regenerated after any user-defined delay. The signal I and Q samples are loaded from RAM, fed to DACs and up-converted in frequency using the same LO signal.

The signal samples stored in RAM can be managed for a wide variety of ECM application. The DIS described in this thesis certainly exploits this capacity. Other optional processing techniques include the generation of noise, synthesis of a different in-band frequency, and modulation of false targets.

The conversion of microwave signals into stream of bits is limited by the speed of ADCs and semiconductor memories. Hence, the use of an intermediate frequency is essential to match the converter sampling and RAM access speed. For example, a 10-GHz signal can be down-converted to a 1 GHz intermediate frequency. According to the Nyquist criterion, the sampling frequency should be twice the signal frequency. Therefore, the IF signal should be sampled at 2 GHz. However, since the sampling is done simultaneously in the I and Q channels, each of them can sample at half of the required sampling frequency of 1 GHz. This sampling frequency matches current RAM access speeds**.**

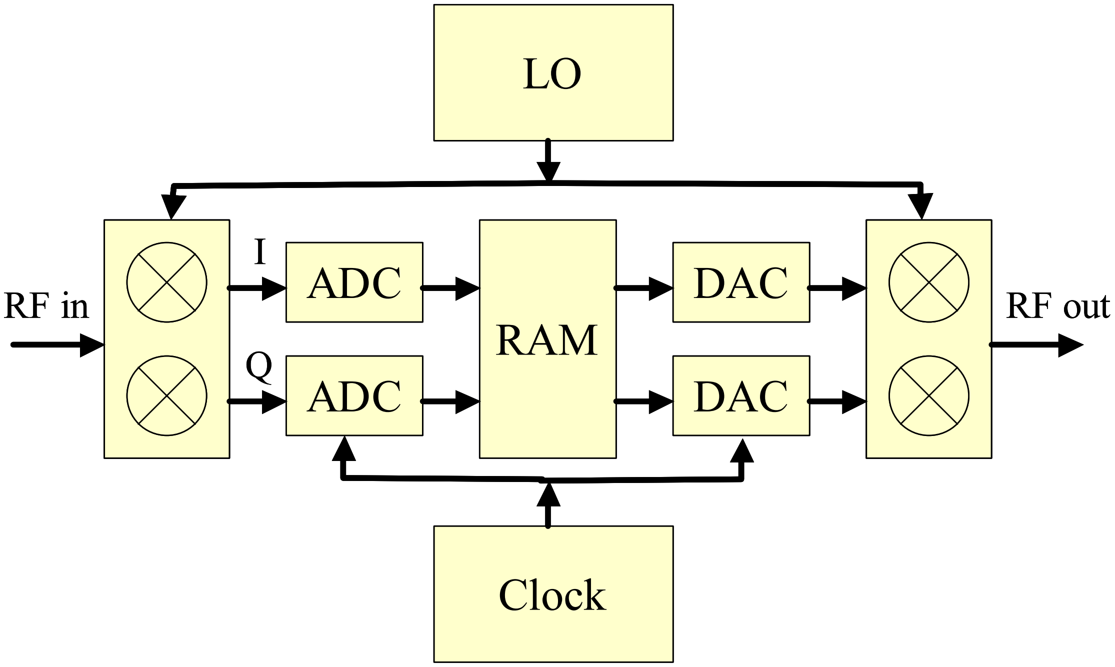


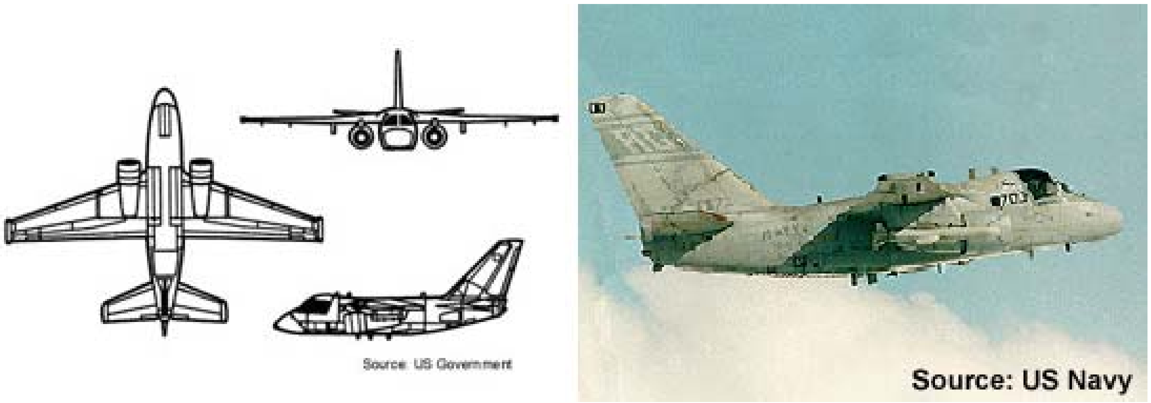
Figure 66 Functional Diagram for a DRFM.

APPENDIX C. THE AN/APS-137 RADAR

The APS-137 is a maritime surveillance radar used by the U.S. Navy onboard the S-3B anti-submarine warfare aircraft and will soon be installed onboard the P-3C maritime patrol aircraft. The system, manufactured by Raytheon, provides long range detection and ship classification capability using ISAR imaging among other operational modes. The radar operating modes include surface surveillance (land, sea and littoral), navigation, periscope detection, strip-map SAR, spotlight SAR and ISAR. The processed signals are displayed in a variety of options: PPI, B-scan, A-scan range profiling, SAR map and ISAR imagery.

Periscope detection is achieved using a high PRF and a high antenna scan rate. The high resolution obtained provides periscope detections to a range up to 26 nm while flying at 1500 feet. The ISAR processor differentiates this radar from its predecessor, the APS-116, used on S-3A aircraft. The ISAR processor, coupled with onboard classification aids, is a powerful tool to identify rolling or pitching surface targets before an anti-surface strike. ISAR requires a tight beam be held on the target ship to measure the Doppler shifts as the target rotates.

The APS-137 radar uses two antennas, a narrow beam parabolic antenna for ISAR applications in a search light mode, and a flat plate array for sector scan and normal surveillance.



Source: US Government

Source: US Navy

Figure 67 The S-3B Viking.

APS-137 Parameters [21]:

Frequency:	9.3-10.1 GHz.
Pulsewidth:	5 us (periscope detection)
	12 us (SAR modes)
	10 us (all other modes)
Pulsewidth compression:	2.5ns (periscope detection)
	6 to 13 ns (ISAR mode)
	14 ns (search modes)
	240 ns (navigation mode)
PRF:	388 pps (search/navigation)
	500 to 1000 (SAR-ISAR)
	1854 pps (periscope detection)
Best resolution:	0.9 m (ISAR)
	3 m (SAR)
Azimuth coverage:	360°
Average power:	200 W (search/navigation).
	500 W (ISAR).
	350 W (SAR).
	460 W (periscope detection).

APS-137 Antennas:

Antenna rotation:	6,60 and 300 rpm.
Parabolic antenna:	1.3° H beamwidth.
	4.5° V beamwidth.
	35 dB Gain
Planar array antenna:	1.05° H beamwidth.
	15° V beamwidth.
	32.5 dB Gain.

Figure 68 APS-137 Antenna and Processing Modules. (From: Raytheon Co.)

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF REFERENCES

[1] Neri, F., *Introduction to Electronic Defense Systems*, pp. 25-26, Second Edition, Artech House, Norwood MA, 2001.

[2] Stimson, G. W., *Introduction To Airborne Radar*, pp. 393-438, Second Edition, SciTech Publishing, Mendham NJ, 1998.

[3] Carrara, W., *Spotlight Synthetic Aperture Radar,* Artech House, Norwood MA, 1995.

[4] Skolnik, M., *Introduction to Radar Systems*, McGraw-Hill, New York NY, 1990.

[5] Eaves, J. L., *Principles of Modern Radar*, Chapman & Hall, New York NY, 1987.

[6] Pace, P. E., Fouts D. J., Karow C., Ekestrom S. R., "A Digital False Target Synthesizer for Countering ISAR," *IEE Proceedings – Radar, Sonar and Navigation*, Accepted for Publication, July 2002.

[7] Soumekh, M., *Synthetic Aperture Radar Signal Processing*, John Wiley & Sons, New York NY, 1999.

[8] Wehner, D. R., *High-Resolution Radar*, Artech House, Norwood MA, 1995.

[9] Fouts, D. J., Pace P. E., Karow C., Ekestrom S. R., "A Single-Chip False Target Radar Image Generator for Countering Wideband Image Radars," *IEEE Journal of Solid State Circuits*, To be published, June 2001.

[10] Pace, P. E., *Advanced Techniques for Digital Receivers,* Artech House, Norwood MA, 2000.

[11] Yeo, S., "A Digital Image Synthesizer for ISAR Counter-targeting," Master's Thesis, Naval Postgraduate School, Monterey CA, 1998.

[12] Ekestrom S. R., Karow C., "An All-Digital Image Synthesizer for Countering High Resolution Radars," Master's Thesis, Naval Postgraduate School, Monterey CA, 2000.

[13] Kirin, K. M., "VLSI Design of Sine/Cosine Lookup Table for Use with Digital Image Synthesizer ASIC," Master's Thesis, Naval Postgraduate School, Monterey CA, 2001.

[14] Guillaume, C. H., "Circuit Design and Simulation for a Digital Image Synthesizer Range-bin Modulator," Master's Thesis, Naval Postgraduate School, Monterey CA, 2002.

[15] Altmeyer, R.C., “Design, Implementation, and Testing of a VLSI High Performance ASIC For Extracting the Phase of a Complex Signal,” Master’s Thesis, Naval Postgraduate School, Monterey CA, 2002.

[16] Bergon, H., “VHDL Modeling and Simulation for a Digital Target Imaging Architecture for Multiple Large Targets Generation,” Master’s Thesis, Naval Postgraduate School, Monterey CA, 2002.

[17] Pace, P. E., Surrat R. E., Yeo S., “Signal Synthesizer and Method Therefore,” Patent NC 79,429, Filed September 1, 1999.

[18] Fouts, D. J., P.E. Pace, “False Radar Target Digital Image Synthesizer,” Patent NC 83,887, Submitted May 1, 2002 (to be filed).

[19] *Image Processing Toolbox User’s Guide*, The Mathworks, Natick MA, 1999.

[20] Rademacher, D., “Introduction to Computer Vision and Image Processing,” Colorado School of Mines, 2001.

[21] *Jane’s Radar and Electronic Warfare Systems*, Jane’s Information Group, Surrey, United Kingdom, 2001-2002.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia.

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California.

3. Chairman, Code EC
   Department of Electrical and Computer Engineering
   Naval Postgraduate School
   Monterey, California.

4. Dr. Phillip Pace, Code EC/PC
   Department of Electrical and Computer Engineering
   Naval Postgraduate School
   Monterey, California.

5. Dr. Douglas Fouts, Code EC/FS
   Department of Electrical and Computer Engineering
   Naval Postgraduate School
   Monterey, California.

6. Dr. David Jenn, Code EC/JN
   Department of Electrical and Computer Engineering
   Naval Postgraduate School
   Monterey, California.

7. LT. Fernando LeDantec
   Chilean Navy
   Valparaiso, Chile.

8. Technical Department
   Chilean Naval Mission
   Washington, D.C.

9. Dr. Peter Craig
   Office of Naval Research
   Washington, D.C.

10. Dr. Joseph Lawrence
    Office of Naval Research
    Arlington, Virginia.

11. Mr. James Talley
Office of Naval Research
Arlington, Virginia.

12. Mr. Michael Monsma
Office of Naval Research
Arlington, Virginia.

13. Dr. John Montgomery
Naval Research Laboratory
Washington, D.C.

14. Mr. Alfred DiMattesa
Naval Research Laboratory
Washington, D.C.

15. Dr. Frank Klenn
Naval Research Laboratory
Washington, D.C.

16. Mr. Gregory Hrin
Naval Research Laboratory
Washington, D.C.

17. Mr. Daniel Bay
Naval Research Laboratory
Washington, D.C.

18. Mr. Brian Edwards
Naval Research Laboratory
Washington, D.C.

19. Mr. George Farmer
Naval Research Laboratory
Washington, D.C.

20. Dr. Preston Grounds
Naval Research Laboratory
Washington, D.C.

21. Mr. Greg Tavik
Naval Research Laboratory
Washington, D.C.