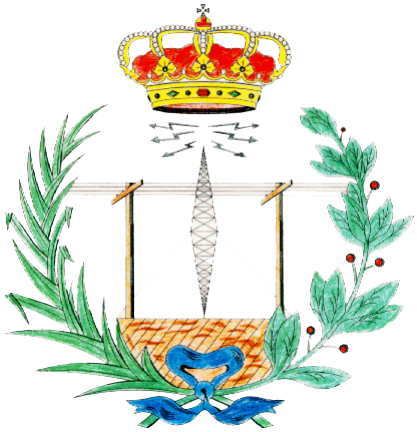


FPGA BASED RADAR ACQUISITION AND PREPROCESSING UNIT

BARCELONA SCHOOL OF TELECOMMUNICATIONS ENGINEERING

TECHNICAL UNIVERSITY OF CATALONIA

CO-DIRECTORS:
Aguasca Solé, Alberto.
Broquetas Ibars, Antoni.
SECRETARY:
Broquetas Ibars, Antoni.
PRESIDENT:
Blanch Boris, Sebastián.
VOCAL:
Ramos Castro, Juan José.

ENGINEER:
Blanco Caamaño, Ramón.

1 ABSTRACT.

In this dissertation, a study and a practical implementation of an acquisition and preprocessing unit are provided for the digitization and suitability of the radar signals. The hardware solution allows the digitization of radar signals at an appropriate speed and subsequently transfers the preprocessed data into a suitable interface. The system must include a memory and a direct connection interface for storage and external communication with a computer, respectively. The final results are obtained by processing the sampled data. This task is performed by a computer with a specific processing software designed by the department. The designed solution must be restrictive in terms of size, weight and portability. A low power hardware is needed in order to feed the unit through external batteries, keeping a suitable performance. From a commercial point of view, a cost effective and high integrated solution is needed.

Blanco Caamaño, Ramón.
ramonblancocaamano@gmail.com

REPOSITORY.

- FPGA_BASED_RADAR_ACQUISITION_AND_PREPROCESSING_UNIT.

Contents

List of Figures

List of Tables

2 INTRODUCTION.

2.1 MOTIVATION.

A radar is a detection system that uses radio waves to determine the range, angle, or velocity of objects. The use of radars in the military field as anti-ballistic or aircraft defence is wide known. In addition, they are being used in special systems that enable the gathering, through an elaborate processing of the radar signal, of information in different fields, such as topography, weather and health, among others.

- Topography: Mapping of high cloud areas, unreachable by means of optical sensors, obtaining world-wide topographic models of high precision.
- Weather: Location of precipitations, calculation of their motion, and estimate of their type: rain, snow, hail, etc.
- Health: Remote sensing techniques that are used to detect vibrations in the body's surface induced by heartbeat and breathing. For example, monitoring cardiac function can be used for diagnosing arrhythmia and mental stress.

These new fields require more compact and portable radar systems focused on the end users. Consequently, the acquisition unit must be suited to these new environments. A high-speed, great performance and reliable unit is needed in radar communications, without forgetting a low power and a light, portable unit suitable for different scenarios. Subsequently, the possible applications the unit could have in the different projects available in the radar department are shown.

- SAR radar drone: A SAR radar called ARBRES-X integrated into a drone has been designed by the department. The main difference between SAR and conventional radars is the use of fewer directive antennas instead of a complete array. A SAR radar moves its antennas creating an effect similar to a complete array. The system could be used to monitor different phenomena, such as object detection or topographic mapping.
- Medical radar for neonates: A radar system used for neonatal medical parameters monitoring has been developed by the department. This is possible thanks to remote sensing techniques which avoid direct contact with the child, unlike the current solutions based on sensor environments.
- Weather radar: The department has developed a pulse-Doppler radar capable of detecting the motion of rain droplets in addition to the intensity of the precipitation. Both type of data can be analyzed to determine the structure of storms and their potential to cause severe weather.

2.2 PROBLEM STATEMENT.

A study and a practical implementation of an acquisition and preprocessing unit are provided for the digitization and suitability of the radar signals. Once sampled, the unit must transfer it into a suitable memory or interface depending on user needs. The final results are obtained by processing the sampled data. This task is performed by a computer with a specific processing software designed by the department.

Figure 1: PROBLEM STATEMENT. RADAR COMMUNICATION SYSTEM.

The radar communication system consists of the following modules. See figure [1].

- **Radar system**: It consists of a transceiver producing electromagnetic waves in the radio domain: a transmitting antenna, a receiving antenna (often the same antenna is used for transmitting and receiving) and a radio-frequency circuitry. Radio waves from the transmitter reflect off the object and return to the receiver, giving information about the object's location and speed.
- **Acquisition and preprocessing unit**: Hardware stage responsible for the processing of sampling signals. It measures the radio wave signal and converts the resulting samples into digital numeric values that can be manipulated by a computer. The main components include A/D converters, interfaces and a specific hardware for signal processing, among others.
- **Processing software computer**: Collection and manipulation of the radar data to produce meaningful information in order to determine the properties of the objects, such as location and speed. The information is processed in any manner detectable by an observer.

2.3 METHODOLOGICAL APPROACH.

The hardware solution must digitize radar signals at an appropriate speed and subsequently transfer the preprocessed data into a suitable interface. The unit must include a memory and a direct connection interface for storage and external communication with a computer, respectively. The designed solution must be restrictive in terms of size, weight and portability. A low power hardware is needed in order to feed the unit through external batteries, keeping a suitable performance. From a commercial point of view, a cost effective and high integrated solution is needed.

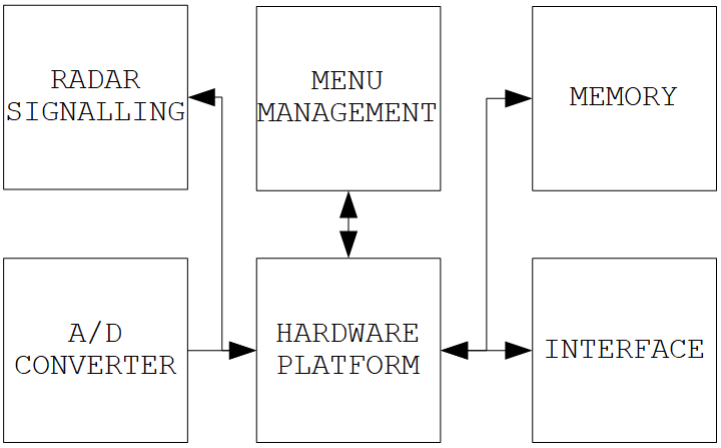


Figure 2: METHODOLOGICAL APPROACH. ACQUISITION AND PREPROCESSING UNIT.

The acquisition and preprocessing unit consists of the following modules. See figure [2].

- **Radar signalling**: Radar unit controller in master mode which manages radar system functionalities. It includes an external clock, a trigger and an arming interrupt.
- **A/D converter**: The radar's RF signal is sampled and quantified, thus it is converted into digital data, giving information about the objects location and speed.
- **Menu management**: User interface which manages the unit operation modes and functionalities. It includes buttons, switches and a display. See table [1].
- **Hardware platform**: Main module of the acquisition unit which manages the different modules. It includes the radar signalling, the acquisition process, the menu controller, the signal preprocessing, the memory controller, and the interface controller, among others. See figure [2].
- **Memory**: It stores the preprocessed digital radar signals.
- **Interface**: It dumps the preprocessed digital radar signals from the memory or directly into a PC, depending on the operation mode selected.

MENU		DESCRIPTION
OPERATION MODES	PC	ACQUISITION OVER INTERFACE WITHOUT MEMORY.
	STANDALONE	ACQUISITION UNTIL MEMORY GOES FULL
FUNCTIONALITIES	RECORD	INITIALIZE THE ACQUISITION SYSTEM.
	SAVE	DUMPS MEMORY INTO THE INTERFACE. NOTE: ONLY AVAILABLE IN STANDALONE MODE.
	RESOLUTION	SIGNAL PREPROCESSING BASED ON SAMPLE WEIGHTING.

Tabla 1: METHODOLOGICAL APPROACH. MENU MANAGEMENT.

2.4 STRUCTURE OF THE DISSERTATION.

In this document, different stages of the thesis development are detailed: problem statement, architecture design, development and implementation, test and validation, among others. Finally, the dissertation concludes with an user guide and the resulting conclusions.

Problem statement. Problem investigation with the aim of allowing designers to better understand the situation, in order to recommend a practical solution.

- **State of art**: Level of development reached in the different technologies that could be used for establishing an appropriate solution. See section[3].
- **Suggested solution.** Formal proposal of a suitable solution, the restrictions that the designed prototype should fulfill are established. See section[4]. See appendix [B].
- **Methodology**: Theoretical concepts of the body of methods and principles associated with the project. In addition, the components used for the development are analyzed. See section[5]. See appendixes [C] & [D].

Architecture design. Formal description and representation of the hardware and software design, organized in a way that supports reasoning about the structure and behaviours of the system. See section [6].

- **System architecture.** The main prototype functionalities are designed in the FPGA device of the development board. It runs tasks such as the management of the radar system and user interface, as well as the

hardware modules. The modules must support necessary operations to control the hardware elements that make up the prototype. Additionally, a suitable architecture is designed for a correct interconnection and management of these modules. See section [6.1].

- **Handshake architecture.** Process of negotiation between two modules through the exchange of signals that establishes the protocol of a communication link. See section [6.2].
- **Modules architecture.** Each of the modules has its own architecture, which generally consists of a control unit and a processing unit. See section [6.3].

Development and implementation. Set of steps of engineering that guide the prototype development. See appendix [E].

- **Main module.** Restructuring of existing source-code without changing their external behavior. It improves code readability and reduces complexity; these can improve source-code maintainability and create a more expressive architecture. See section [E.1].
- **Control modules design.** Development of the Control Units (CU) responsible for the coordination of the hardware elements that make up the prototype. See sections [E.2], [E.3], [E.4], [E.5] & [E.6].

Test and validation. Procedures followed in order to verify that the prototype meets the requirements for intended purpose. These are critical components of a quality management system. See section [7].

- **Unit testing**. Code testing method that enables the simulation of individual units of source-code, or sets of these, to determine whether they are fit for use. See sections [7.1], [7.2], [7.3], [7.4] & [7.5].

Conclusions. Overview of the final prototype. See section [8].

- **Results**: Expected results from the work undertaken. See section [8.1].
- **Review**: General assessment of the work undertaken. See section [8.2].
- **Recommendations**: Proposal of different improvements to the current solution. See section [8.3].
- **User guide**: Drafting of a document that provides practical information about the use of the unit. See appendix [A].

3 STATE OF ART.

3.1 ANALOG-TO-DIGITAL (A/D) CONVERTER.

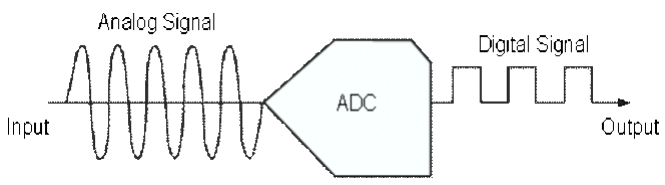


Figure 3: ANALOG-TO-DIGITAL (A/D) CONVERTER. BASIC PRINCIPLES.

An A/D converts a continuous-time/amplitude analog signal to a discrete-time/amplitude digital signal. The conversion involves the quantization mapping of the continuous values into a set of values, often by rounding. Inevitably, the conversion process always adds a certain amount of noise error.

Performance factors. They can be evaluated through several factors, the most important of which are:

- Signal-to-noise ratio (SNR): The SNR reflects the average number of non-noise bits in any particular sample (effective number of bits or ENOB).
- Bandwidth: It can determined by evaluating the sampling rate, i.e. the number of times per second the analog source is sampled to generate discrete values.

Common types. Each A/D architecture has its own distinct strengths and weaknesses.

- Flash A/D: It can reach very high speeds by running a bank of comparators that operate in parallel, each for a defined voltage range. As a result, flash converters tend to be large and expensive compared to other technologies. They are well-suited for broad-band applications such as signal processing.
- Semi-flash A/D: It applies two separate flash converters, each with a resolution of half the bits of the semi-flash converter. One of the flash converter handles the most significant bits, while the other handles the least significant ones. Semi-flash converters take twice as long as flash converters, although they are still very fast.
- Successive approximation (SAR): These A/D can be identified by their successive approximation registers (SAR). They compare input voltage and the output of an internal D/A converter, successively judging whether the input is above or below a narrowing range midpoint. The process continues until they reach the desired resolution. SAR converts are slower than flash technology, but they offer higher resolution without the cost of flash converters.
- Pipelined ADC: Also known as sub-ranging quantizers, they are similar in concept to SAR converters, but more refined. While SAR converters progress thought step by step by going to the next most significant bit, pipelined converters adhere to the following process.

  1. It performs a coarse conversion.
  2. Then, it compares that conversion to the input signal.
  3. The A/D performs a finer conversion.

  Pipelined designs typically offer a middle ground between SAR and flash converters, balanced speed, high resolution and size.
- Sigma-delta A/D: They are slow compared to other technologies, but they offer the highest resolution of all A/D types. As a result, sigma-delta converters are well-suited for narrow-band applications such as Hi-Fi audio systems.

About: Research updates, see reference [4].

3.2 HARDWARE PLATFORMS.

3.2.1 DIGITAL SIGNAL PROCESSOR (DSP).

A DSP is a specialized microprocessor, typically built on MOS integrated circuit (IC) chips, whose architecture is optimized for digital signal processing needs. They are widely used in digital audio and image processing and

in telecommunications as radars, sonars, electronic devices, etc. A DSP can properly measure, filter or compress continuous real-time signals in terms of power and efficiency, which makes it an ideal technology for portable devices. In addition, DSPs provide an efficient architecture to simultaneously fetch multiple data and instructions. Most general-purpose microprocessors can also execute digital signal algorithms successfully, but they may not be suitable for real-time operations. DSPs instructions sets are often irregular, while traditional instructions sets, used in general-purpose processors, are made up of a wider variety of operations. The instructions sets are optimized for signal processing as FFT operations. Obviously, the developed software is more efficient when resorting to DSPs solutions.

3.2.2 APPLICATION-SPECIFIC INTEGRATED CIRCUIT (ASIC).

An ASIC is an customized hardware design, typically built on MOS integrated circuit (IC) chips for a particular use, rather than intended for general-purpose use; e.g., a customized design for digital audio processing or high-rate communication systems.

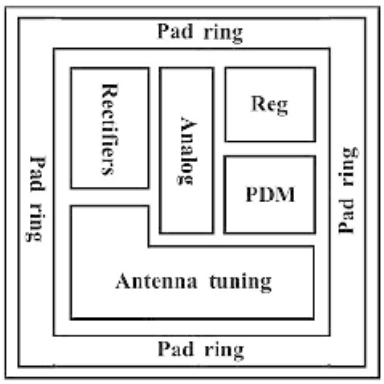


Figure 4: APPLICATION-SPECIFIC INTEGRATED CIRCUIT (ASIC). BLOCK DIAGRAM.

Over the years, integration and design tools have improved exponentially, allowing the implementation of high complex designs for a wide variety of functionalities. ASICs often include microprocessors, and memory blocks as ROM or RAM, among others. An ASIC that contains one or more processor cores is often referred to as SoC (system-on-chip). Designers typically use a hardware description language (HDL), such as Verilog or VHDL, for ASICs development.

Common types. There are basically three types of ASIC chip designs:

- Full custom design: In this design, the resistors, transistors, digital logic, capacitors and analog circuits are all positioned in the circuit layout. Generally full custom designs are referred to as handcrafted designs; e.g., a microprocessor is a full custom integrated circuit (IC). Maximum performance, minimized area and highest flexibility are the major features of this design. However, the manufacturing and design of this device is very expensive.
- Semi-custom design: This design makes use of components coming from a standard library. Thus, all logic cells are pre-designed and only some layers can be customized. Owing to the use of pre-designed logic cells, ASIC design tends to be easier without losing too much performance.
- Programmable ASIC: Field-programmable gate arrays (FPGA) are included in this group. In the following section, the FPGA will be described in detail.

3.2.3 FIELD-PROGRAMMABLE GATE ARRAY (FPGA).

A FPGA is an integrated circuit (IC) designed to be configured by a designer after manufacturing, hence the term field-programmable. FPGAs can be easily programmed on a computer through a hardware description language (HDL), such as Verilog or VHDL. As FPGA speed increased, power consumption decreased, and thus prices also decreased, which lead to FPGAs replacing their equivalent ASICs as definitive designs. Obviously, FPGAs are still suited for prototyping designs. FPGAs contain an array of programmable logic blocks and a hierarchy of reconfigurable interconnections that allow the blocks to be wired. Logic blocks can be configured to perform as simple logic gates, from AND and XOR to complex combinational functions.

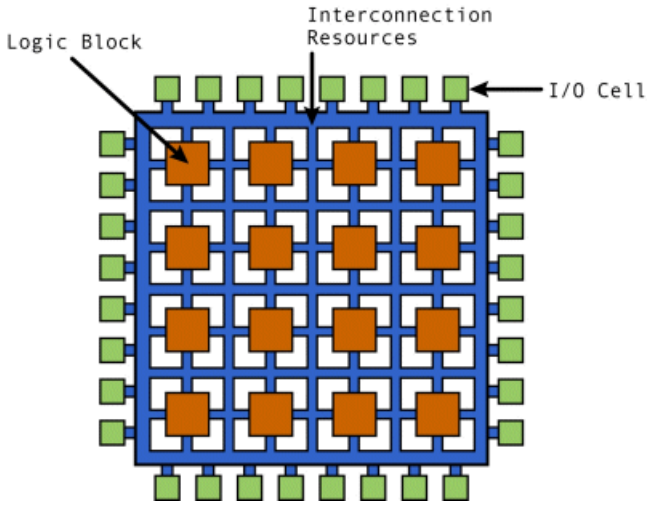


Figure 5: FIELD-PROGRAMMABLE GATE ARRAY (FPGA). BLOCK DIAGRAM.

The architecture consists of configurable logic blocks, configurable I/O blocks and programmable interconnections. Likewise, a clock circuitry is provided for each logic block. Additional logic resources, such as ALUs, memory, and decoders, may also be available.

Common types. Configuration bitstreams can be stored in FPGA through various technologies. The majority of FPGAs are based on SRAM (Static RAM).

- SRAM-based FPGAs: Logic cell configuration data are stored in a static memory, organized as an array of latches. Since SRAM is volatile, these FPGAs must be programmed upon start. The FPGA reads the configuration data from an external source, such as a flash memory or a computer, via a boundary-scan (JTAG). Vendors examples: Xilinx Virtex, Xilinx Spartan, Altera Stratix, Altera Cyclone.
- Flash-based FPGAs: Logic cell configuration data are stored in a flash memory. This technology has the advantage of being less power consuming than the previous one. Vendors examples: Actel Igloo, Actel ProASIC3.
- Antifuse-based FPGAs: An antifuse is a device that does not conduct current initially, but it can be burned to enable this feature. Obviously, the FPGA cannot be reprogrammed. Vendors example: Actel Axcelerator.

3.3 INTERFACES.

3.3.1 ETHERNET.

Ethernet is a family of computer networking technologies commonly used in local area networks (LAN), metropolitan area networks (MAN) and wire area networks (WAN). The original 10BASE5 Ethernet uses a coaxial cable such as shared medium, while the newer Ethernet variants use twisted pair and optic fiber links with switches.

NAME	**100BASE-TX**	**1000BASE-T**	**2.5GBASE-T**	**5GBASE-T**	**10GBASE-T**
SPEED	100 Mbps	1000 Mbps	2.5 Gbps	5 Gbps	10 Gbps
PAIRS	2	4			
LINES/ DIRECTION	1	4			
BIT/HZ	3.2	4	6.25		
LINE CODE	4B5B MLT-3 NRZ-I	TCM 4D-PAM-5	64B65B PAM-16 128-DSQ		
SYMBOL RATE	125 MBd		200 MBd	400 MBd	800 MBd
BW	31.25 MHz	62.5 MHz	100 MHz	200 MHz	400 MHz
DISTANCE	100 m				
CABLE	CAT. 5	CAT. 5e		CAT. 6	CAT. 6A

Tabla 2: ETHERNET. STANDARD VARIANTS.

Ethernet divides data streams into shorter pieces called frames. Each frame contains a source/destination address and error-checking data. Higher-layer protocols, such as TCP, could manage the retransmission of lost frames. As per the OSI model, Ethernet provides services up to and includes the data link layer, a 48-bit MAC address was adopted by other IEEE networking standards (e.g., IEEE 802.11 Wi-Fi.).

3.3.2 UNIVERSAL SERIAL BUS (USB).

USB is an industry standard that establishes specifications for cables and connectors, as well as protocols for connections, communications and power supply between devices.

VERSION	**2.0**	**2.0 REVISED**	**3.0**	**3.1**	**3.2**	**4**
SPEED	480Mbps HIGH SPEED	480Mbps HIGH SPEED	5 Gbps SUPERSPEED	10 Gbps SUPERSPEED+	20 Gbps SUPERSPEED+	40 Gbps SUPERSPEED+ THUNDERBOLT3
STD.	TYPE A	TYPE A	TYPE A	TYPE A	DEPRECATED	
	TYPE B	TYPE B	TYPE B	TYPE B	DEPRECATED	
	TYPE C	TYPE C	TYPE C	TYPE C	TYPE C	TYPE C
MINI	MINI A	MINI A	DEPRECATED			
	MINI B	MINI B	DEPRECATED			
	NONE	MINI AB	DEPRECATED			
MICRO	NONE	MICRO A	DEPRECATED			
	NONE	MICRO B	MICRO B	DEPRECATED		
	NONE	MICRO AB	DEPRECATED			

Tabla 3: UNIVERSAL SERIAL BUS (USB). STANDARD VARIANTS.

USB connections are directed: a host device includes a downstream port that connects to the upstream ports of the devices. Only downstream ports provide power. This topology was chosen to prevent electrical overloads and equipment damage. Thus, USB cables have different ends: A and B, with different physical connectors for each. Each format has a connector and receptacle defined for each of the A and B ends. In practice, the A end is usually the standard format, and the B side varies between standard, mini an micro. The mini and micro formats also include USB On-The-Go (OTG) that allows USB devices to act as a host enabling other USB devices to be attached to them (e.g., a flash drive connected to a smartphone.).

3.4 MEMORIES.

3.4.1 RANDOM-ACCESS MEMORY (RAM).

RAM is a volatile memory, typically built on MOS integrated circuit (IC) chips, and is widely used in computers to store working data and machine code. A RAM memory enables data to be read or written in the same amount of time, irrespective of the physical location inside the memory. In contrast, direct-access memories, such as HDDs or SDs, depend on their physical location due to design limitations. RAM contains multiplexing circuitry to connect the data lines to the addressed storage for reading or writing entries. Usually, more than one bit of data is accessed by the same address, e.g. 8/16-bit RAM.
The main types of RAM memory are the following:

- Static random-access memory (SRAM).
- Dynamic random-access memory (DRAM).

Both memories can be written and read repeatedly, and both types lose their content when the system is turned off. However, dynamic RAM must be constantly refreshed, while static RAM retains its content indefinitely. Therefore, static memories should only be rewritten again when a change in their content is required.

Double Data Rate Synchronous Dynamic Random-Access Memory (DDR SDRAM). It is a dynamic RAM capable of performing twice write and read operations for each clock cycle. The interface uses double pumping, transferring data on both the rising and falling edges of the clock signal.

$$BANDWIDTH = CLOCK\_RATE \cdot 2 \cdot BITS\_TRANSFERRED \tag{1}$$

For example, a 64b @ 100MHz bus will have a bandwidth of 1600MB/s. See equation[1].

VERSION		**DDR**	**DDR2**	**DDR3**	**DDR4**
CLOCK RATE (MHz)		200	533	1066	1600
TRANSFER RATE (MT/s)		400	1066	2133	3200
BANDWIDTH (MB/s)		3200	8533	17066	25600
VOLTAGE (V)		2.6	1.8	1.5/1.35	1.02/1.05
PINS	DIMM	184	240	240	288
	SO-DIMM	200	200	204	260

Tabla 4: RANDOM-ACCESS MEMORY (RAM). STANDARD VARIANTS.

3.4.2 SECURE DIGITAL (SD).

SD is a proprietary non-volatile memory card format widely used in portable devices, better known due to its speed, considerably superior to that of its predecessors. SD includes five card families, available in three different sizes. The five families are the original standard-capacity (SDSC), the high-capacity (SDHC), the extended-capacity (SDXC), the ultra-capacity (SDUC) and the SDIO, which combines input/output functions with data storage. The three form factors are the original, the mini and the micro size.

FAMILY		**SDSC**	**SDHC**	**SDXC**	**SDUC**
CAPACITY	MIN.	128MB	2GB	32GB	2TB
	MAX.	2GB	32GB	2TB	128TB
UHS RATE INITIAL STANDARD		STANDARD	HIGH	UHS (I,II)	UHS (I,II,III) SD EXPRESS
FILE SYSTEM		FAT16	FAT32	FAT32/exFAT	exFAT

Tabla 5: SECURE DIGITAL (SD)(I). STANDARD VARIANTS - CARDS FAMILIES.

Electrically passive adapters allow for a smaller card to fit into a device built for larger cards. The card's small footprint makes it an ideal storage medium for portable devices.

BUS	**SPEED**	**DUPLEX**	**SDSC**	**SDHC**	**SDXC**	**SDUC**	**VERSION**
STANDARD-SPEED	12.5 MB/s	NONE	YES	YES	YES	YES	1.01
HIGH-SPEED	25 MB/s	NONE					2.00
UHS-I	50 MB/s	HALF/FULL	NO				3.01
	104 MB/s	HALF					
UHS-II	156 MB/s	FULL					4.00
	312 MB/s	FULL					
UHS-III	312 MB/s	FULL					6.0
	624 MB/s	FULL					
SD EXPRESS	985 MB/s	FULL					7.0

Tabla 6: SECURE DIGITAL (SD)(II). STANDARD VARIANTS - BUS SPEEDS.

3.4.3 SOLID-STATE DRIVE (SSD).

A SSD drive employs integrated circuit (IC) assemblies as memory to store data persistently, typically through a flash memory. Compared to electromechanical drives (HDDs), SSDs are more resistant to physical shocks, run silently, have quicker access time and lower latency. The hybrid SSDHs drives combine features of SSDs and HDDs, using both flash memory and a HDD driver, in order to improve drive performance. While the price of SSDs drives has continued to decline over time, SSDs are still more expensive in terms of storage unit than HDDs. Traditional interfaces, such as SATA, and standard HDD form factors can be used in SSDs drives. Newer form factors, such as mSATA and M.2, and higher speed interfaces, such as NVMe over PCIe, have increased the SSD performance surpassing HDD technology.

ABOUT	SSD	HDD
PRICE PER CAPACITY	EXPENSIVE.	CHEAP.
STORAGE CAPACITY	UP TO 200 TB.	UP TO 16TB.
STORAGE RETENTION	ONE YEAR.	VERY LONG PERIOD.
START-UP TIME	INSTANTANEOUS.	SEVERAL SECONDS.
SEQUENTIAL ACCESS	3500MB/s.	200MB/s.
FS FRAGMENTATION	NO NEED.	REQUIRED.
STRENGTH	NO MOVING PARTS; VERY RESISTANT.	MOVING PARTS; SUSCEPTIBLE TO SHOCK, VIBRATION, ETC.
NOISE	SILENT.	WHIRRING.
WEIGHT & SIZE	M.2 FACTOR; SMALL & LIGHT.	STD HDD FACTOR; LESS SMALL & LIGHT.
POWER CONSUMPTION	LOW; RAM-BASED EXCEPT.	NORMAL.

Tabla 7: SOLID-STATE DRIVE (SSD). COMPARISON OF SSD AND HDD.

SSDs properties vary depending on the number of bits stored on each cell; single-bit cells (SLC) are generally the most reliable, durable, fast and expensive compared to 2/3-bits cells, MLC and TLC, respectively. Finally, quad-bit cells (QLC) are used in consumer devices. Finally, quad-bit cells (QLC) are used in consumer devices where extreme properties are not required. In this regard, QLC are the cheapest solution. In addition, RAM-based SSDs drives can be used for high speed needs when data persistence after power loss is not required. However, NAND-based SSDs drives will slowly leak charge for long periods without power. For this reason, SSDs drives are not suitable for archival storage.

3.5 SUMMARY.

Currently the use of DSPs in signal processing applications is widespread. However, when it comes to particular speeds and a certain complexity, its performance may not be enough due to the CPU instructions cycle. In high-performance processing cases, there is no other alternative but to use specific hardware technologies, such as FPGA and ASIC. The flexibility in the configuration of FPGA makes it suitable for applications that need to be modified frequently; as opposed to ASIC which is suited for permanent applications. This feature is also what makes FPGA the ideal choice for prototyping purposes.

ABOUT	FULL CUSTOM	SEMI-CUSTOM	FPGA
DENSITY	HIGH	MEDIUM	LOW
PERFORMANCE	HIGH	MEDIUM	LOW
DESIGN TIME	LONG	MEDIUM	SHORT
COST	HIGH	MEDIUM	LOW
TEST	DIFFICULT	LESS DIFFICULT	EASY
VOLUME	HIGH	MEDIUM	LOW

Tabla 8: STATE OF ART(I). SUMMARY - HARDWARE PLATFORMS.

The different technologies available to incorporate high-speed interfaces into the acquisition unit are analyzed. The selected interface should be able to overcome the restrictions imposed by the system requirements and archi-

tecture.

ITEM	STANDARD	REQUIRED
A/D CONVERTER	12b @ 40MHz 480 Mbps (60MB/s)	SAR/PIPELINED
HW PLATFORM	FPGA	SRAM/FLASH.
INTERFACES	ETHERNET	1000BASE-T.
	USB	2.0. (3.0 RECOMMENDED)
MEMORIES	RAM	DDR3 SDRAM
	SD	SDXC CARD, UHS-I HALF. (UHS-II RECOMMENDED)
	SSD	GENERIC MODEL.

Tabla 9: STATE OF ART(II). SUMMARY.

4 SUGGESTED SOLUTION.

4.1 REQUIREMENTS CAPTURE.

The designed solution should overcome, at least, the following system requirements restrictions.

Technical requirements. It includes a document containing all the prototype requirements. Its aim is to allow designers to understand what the prototype should do. See appendix[B].

4.1.1 RADAR REQUIREMENTS.

- Digitization and storage of narrow/broad-band radar signals up to several MHz. The resolution must be at least 12 bits.
- The acquisition interval, as well as the transfer interval, must be between 1-2ms and it must be established according to the carrier frequency range and radar clock. During this interval, a N samples packet is acquired.
- The prototype must manage suitably the radar signalling according to 3.3V CMOS logic levels. An external clock, a trigger and an arming interrupt are included as sync signals.
- In a standalone mode, the prototype must be capable of store at least 10,000 packets.

4.1.2 HARDWARE REQUIREMENTS.

- An A/D converter is necessary for the digitization of the radar signals. Its architecture must be SAR/pipelined, and its performance must be at least 12b at 40MHz.
- The development board must be in possession of the necessary specifications for the prototyping of the radar signal processing unit. Specifications for development boards include processor type, form factor, number of ports, port type, memory, and on-board peripherals, among others.

- SRAM/FLASH FPGA technology must be used as a hardware platform. Its high-performance and flexibility make it the ideal choice for development. Its resources must be suitable for radar signals processing.
- An Ethernet interface is necessary for the preprocessed digital radar signals dumped from memory or directly from a computer, depending on the operation mode selected. Its technology must be equal to or higher than the 1000BASE-T standard.
- A RAM memory is necessary as intermediate volatile memory between the acquisition and transfer stage. Its technology must be equal to or higher than the DDR3 SDRAM standard.
- A SD memory is necessary for the storage of preprocessed digital radar signals. Its technology must be equal to or higher than the UHS-I SDHC standard.

4.1.3 SOFTWARE REQUIREMENTS.

- The prototype must have in place two operating modes.
  - Standalone mode: The unit must digitize the radar signals until the SD memory goes full.
  - PC mode: The unit must digitize the radar signals continuously through the Ethernet interface.
- The prototype must contain a signal processing algorithm based on down-sampling in order to manage the acquisition resolution.
- The prototype must have a menu for the management of the system functionalities.
  - Record: Initialization of the acquisition system.
  - Save: In standalone mode, it dumps the SD memory into a computer through the Ethernet interface.
  - Resolution: Decimation rate of the down-sampling algorithm.

4.1.4 MECHANICAL REQUIREMENTS.

- The prototype chassis must be restrictive in terms of size, weight and portability. Additionally, it must be resistant to adverse weather conditions.
- The prototype must have the inputs/outputs suitable for radar signalling. Likewise, it must include the interfaces necessary for the system functionalities, such as buttons an switches.
- It could be desirable to have in place an user-friendly interface as a display menu.

4.2 FORMAL PROPOSAL

The suggested solution is a hardware system for an acquisition and preprocessing unit capable of digitizing radar signals at an appropriate speed and subsequently transfer the preprocessed data into a suitable interface and memory. In order to meet the technical requirements, a designed solution must be built from the following stages, whose modules must fulfill its established tasks.

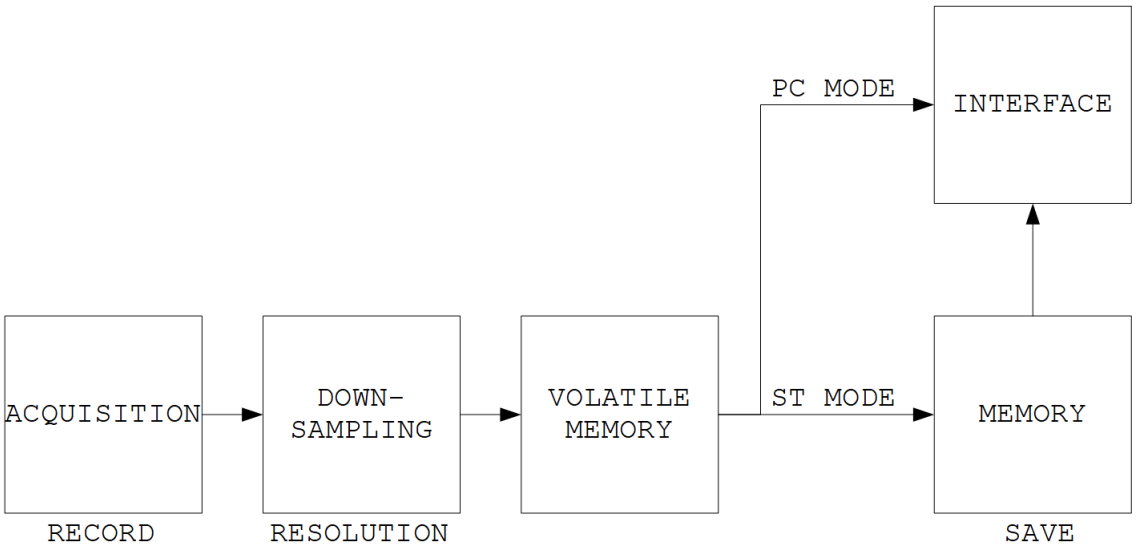


Figure 6: FORMAL PROPOSAL. REFERENCE MODEL.

The radar system transmits a periodic signal in order to receive an echo signal which provides information about the objects. The carrier range is established according to the radar resolution.

4.2.1 ACQUISITION INTERVAL.

During the ascending frequency period, a packet is acquired and processed according to the external radar clock. Obviously, the packet size is determined by the carrier range and sampling rate. Once finished, the processed data are stored into a volatile memory, such as a DDR3 memory.

- **Acquisition:** An A/D based module, responsible for the digitization and management of the radar system. Record: It initializes the acquisition module.
- **Down-sampling:** A signal processing module based on a down-sampling algorithm. Resolution: It manages the decimation rate.

4.2.2 TRANSFER INTERVAL.

During the decreasing frequency period, the stored data are dumped from the volatile memory into the interface or memory, depending on the operation mode.

- **Memory:** A SD based memory, responsible for the storage of the processed data. Save: In standalone mode, it dumps the stored data into the interface.
- **Interface:** An Ethernet interface based module, responsible for the download of the processed data on a PC.

4.3 DEVELOPMENT KIT.

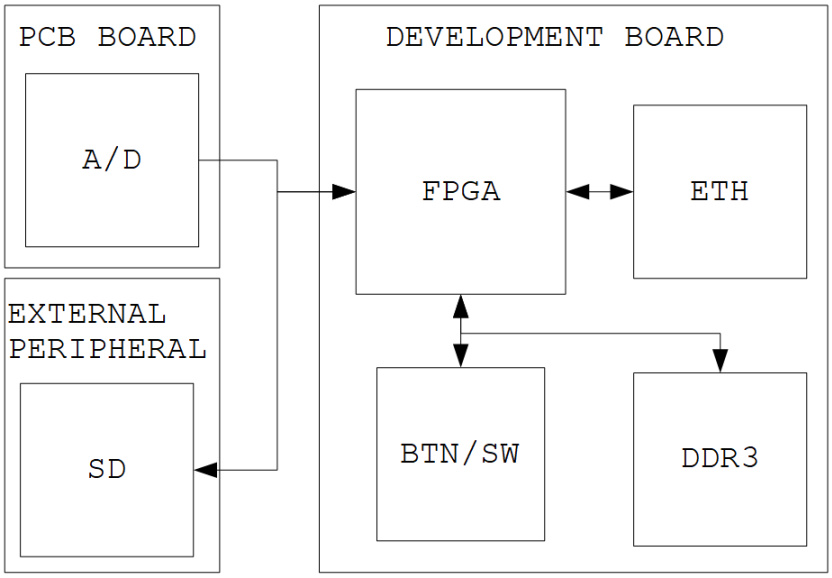


Figure 7: DEVELOPMENT KIT. REFERENCE MODEL.

The suggested prototype consists of a hardware system whose modules were strictly selected according to technical requirements. A formal representation of the modules used is organized in a way that supports reasoning about the structure of the prototype.

- **PCB board:** Developed by the department, this module is responsible for the management of the radar system. It integrates an A/D converter that digitizes the radar signals. Additionally, it manages the system signalling through the following sync signals: arming, trigger and external clock.
- **External peripheral:** A PMOD based module is used as external peripheral to incorporate SD memories into the prototype. In addition, a SPI variant is used as interface to the UHS-I SDHC standard.
- **Development board:** A FPGA based development board has been selected as main hardware for the prototype design for flexibility reasons. This module is responsible for the development of the main prototype functionalities. It includes tasks such as data acquisition, processing and storage, as well as communication interfaces. Furthermore, it acts as a manager of the external peripherals, such as PCB board or external SD drive.

Unfortunately, as a designer, I do not have full flexibility to select the ideal modules for prototyping. In addition, I am compelled to comply with the restrictions of previous laboratory projects for retro-compatibility reasons. Nevertheless, the Arty A7 development platform is powerful enough to satisfy most of the prototype requirements. As a drawback, high-speed interfaces, such as SATA, 1000BASE-TX or USB 3.0, are not available. Platform limitations must be assumed; the on-board 100BASE-TX peripheral will be used as Ethernet interface, while an external PMOD peripheral will operate as SD memory.

ITEM		ID		ABOUT
PCB BOARD	LAB.	A/D	ANALOG DEVICES AD9224	12b @ 40MHz, PIPELINED
EXTERNAL PERIPHERAL	DIGILENT PMOD SD	SD		SDHC CARD, HIGH SPEED (25 MB/s)
DEVELOPMENT BOARD	DIGILENT ARTY A7	FPGA	XILINX ARTIX-7 XC7A35T	LOGIC CELLS: 33280 DSP SLICES: 90 MEMORY: 1800 TRANSCEIVERS: 4 I/O PINS: 250
		DDR3	MICRON DDR3L	256MB DDR3L 16b @ 667MHz.
		ETH		FAST ETHERNET, 10/100BASE-T
		BTN/SW		4x BUTTONS 4x SWITCHES

Tabla 10: DEVELOPMENT KIT. USED COMPONENTS.

5 METHODOLOGY.

5.1 USED CONCEPTS.

5.1.1 RADAR FUNDAMENTALS.

RADAR SYSTEM.

A FMCW based radar system is used as radar sensor to obtain information about a target scene. Additionally, it is compatible with synthetic-aperture radars (SAR).

- **FMCW radar**: It is a special type of radar sensor which radiates continuous transmission power, like a simple continuous wave (CW) radar. In contrast, FMCW radar can change its operating frequency during the measurement: i.e., the transmission signal is modulated in frequency or in phase. Therefore, the difference in frequency or phase between the actually transmitted and the received signal is measured instead. See appendix [C].
- **SAR radar**: It is a type of radar used to create two-dimensional images or three-dimensional reconstructions of objects, such as landscape. It uses the motion of the radar antenna over a target region to provide finer spatial resolution than conventional beam-scanning radar. SAR radars are typically mounted on a moving platform, such as a drone.

ANTENNA ARRAY.

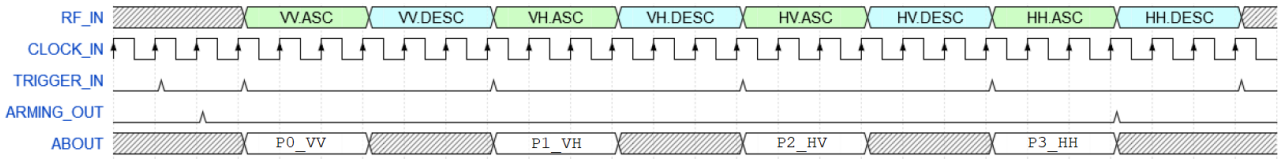


Figure 8: RADAR FUNDAMENTALS. RADAR SIGNALLING.

ITEM		ABOUT
ANTENNAS ARRAY	VV	RX VERTICAL AND TX VERTICAL.
	VH	RX VERTICAL AND TX HORIZONTAL.
	HV	RX HORIZONTAL AND TX VERTICAL.
	HH	RX HORIZONTAL AND TX HORIZONTAL.

Tabla 11: RADAR FUNDAMENTALS(I). ANTENNA ARRAY.

The antenna array consists of a set of multiple connected antennas, which work together as a single antenna, in order to transmit or receive radio waves. The radio waves radiated by each individual antenna combine and superpose, adding together to enhance the radiation of the power in desired directions and cancelling to reduce the power radiated in other directions. Similarly, when used for receiving, the separate radio waves from the individual antennas combine in the receiver through the correct phase relationship in order to enhance the signal received from the desired directions and cancel signals from undesired directions. In the suggested solution, each combination of the antenna array is treated as an individual unit that will be acquired and processed into a single packet. Once the information is downloaded into a PC, a specific software designed by the department will process the information as appropriate.

ID.		ABOUT	
SIGNALS	RF	IN	MODULATION FREQUENCY.
	CLOCK	IN	EXTERNAL CLOCK.
	TRIGGER	IN	CONTROL SIGNAL FOR ASCENDING FREQUENCY INDICATION.
	ARMING	OUT	RADAR RESET/ENABLE.
CONFIGURATION	NDATA	NUMBER OF SAMPLES PER PACKET. NOTE: 4096.	
	NPACKETS	TOTAL NUMBER OF PACKETS. NOTE: 32768.	

Tabla 12: RADAR FUNDAMENTALS(II). RADAR SIGNALLING.

RADAR SIGNALLING.

The radar signalling is responsible for the management of the radar system. It includes an external clock, a trigger and an arming interrupt.

- **External clock**: Provided by the radar system, it acts as an A/D converter clock reference.
- **Trigger**: Ascending frequency control signal, it acts as a timing mark for the radar signals digitization.
- **Arming**: It allows the prototype to reset or enable the radar system, depending on its operation modes and functionalities.

RADAR SIGNALS (RF).

A low-frequency range mixed signals are received, resulting from the differences in frequency or phase between the actually transmitted and the received signals instead. As discussed in appendix [C], the carrier range will be arranged according to radar requirements.

- **Ascending frequency**: A N samples packet is digitized and processed according to the radar signalling. Obviously, the packet size is determined by the prototype configuration according to the carrier range and sampling rate. Once finished, the processed data are stored into a volatile memory, such as a DDR memory. In standalone mode, the number of packets is limited for memory reasons.
- **Decreasing frequency**: The stored data are dumped from the volatile memory into the interface or memory depending on the operation mode.

5.1.2 ENGINEERING FUNDAMENTALS.

The overall organization of this section establishes some of the basic premises spread throughout this thesis. These are the basic engineering concepts which constitute the foundation of the prototype development. See appendix[D].

- **Down-sampling:** Also known as decimation, it is a term associated with the process of re-sampling in a multi-rate digital signal processing system. Synonymous with compression, it describes an entire process of bandwidth and sample-rate reduction. See section[D.1].
- **Standard AXI**: The Advanced extensible Interface (AXI), part of ARM AMBA, is a parallel high-performance, synchronous, high-frequency, multi-master, multi-slave communication interface, mainly designed for on-chip communication, see reference [17]. See section[D.2].
- **Standard SPI**: It is a synchronous serial data protocol used by embedded systems for communicating with one or more peripheral devices quickly over short distances, see reference [16]. See section[D.3].
- **Standard 100BASE-TX**: It is the predominant standard of Fast Ethernet, and runs over two wire-pairs inside a category 5, or above, cable. Each network segment can reach a maximum cabling distance of 100 metres. One pair is used for each direction, providing full-duplex operation with 100 Mbps of throughput. See section[D.4].
- **TCP/IP model**: It was designed to describe the functions of a communication system by dividing the communication procedure into smaller and simpler components. The TCP/IP model contains four layers, i.e.: application layer, transport layer, internet layer and link layer. See section[D.5].

5.2 USED COMPONENTS.

5.2.1 PCB BOARD: ANALOG DEVICES AD9224.

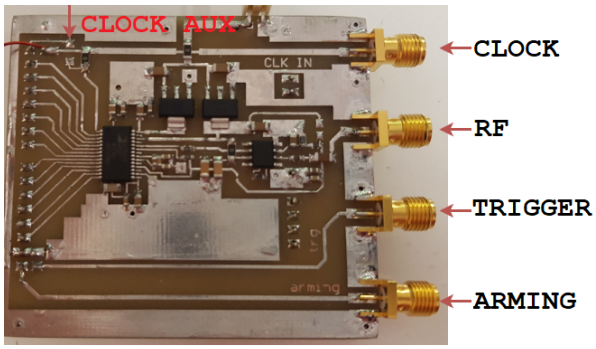


Figure 9: PCB BOARD. A/D ACQUISITION SYSTEM.

An A/D based PCB board has been developed by the department. Its main task is the management of the radar system as follows.

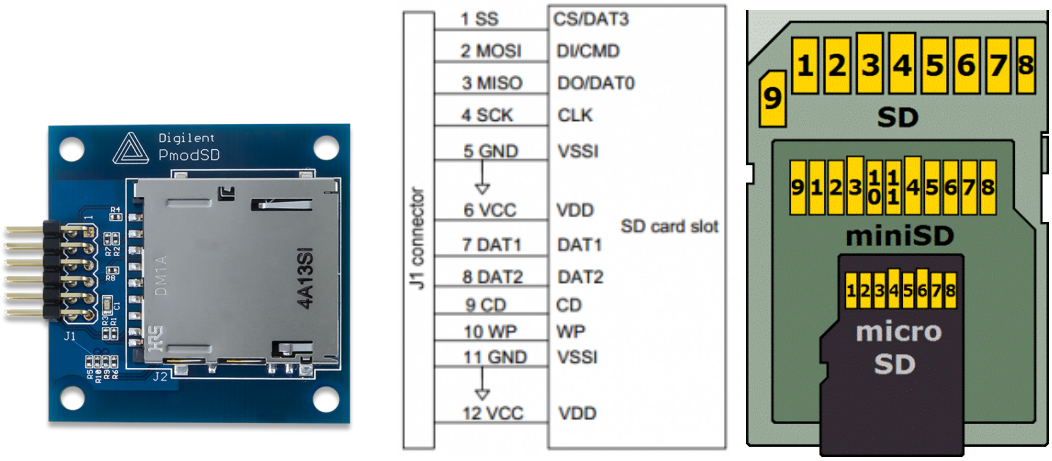
- **Management**: It manages the radar system through the following sync signals: arming, trigger and external clock.
- **Digitization**: The radar signals are sampled and quantified through an A/D convert obtaining digital data. This information will be processed in order to extract information about the location and speed of the objects.

The AD9224 is a monolithic, single supply 12-bit, 40MSPS A/D converter with an on-chip, high performance sample-and-hold amplifier and voltage reference. The AD9224 employs a multi-stage differential pipelined architecture with output error correction logic to provide 12-bit accuracy at 40MSPS data rates and guarantees no missing codes over the full operating temperature range. See reference [6].

ANALOG DEVICES AD9224: FEATURES AND BENEFITS.	
LOW POWER DISSIPATION, 415 mW.	SFDR: 81 dB.
SINGLE +5V SUPPLY.	OUT-OF-RANGE INDICATION.
NO MISSING CODES GUARANTEED.	STRAIGHT BINARY OUTPUT DATA.
DNL: ±0.33 LSB.	28-LEAD SSOP PACKAGE.
ON-CHIP SAMPLE-AND-HOLD AMPLIFIER AND VOLTAGE REFERENCE.	COMPATIBLE WITH 3.3 V LOGIC.
SNR: 68.3 dB.	

Tabla 13: PCB BOARD. ANALOG DEVICES AD9224.

5.2.2 EXTERNAL PERIPHERAL: DIGILENT PMOD SD.



(a) (b) (c)

Figure 10: EXTERNAL PERIPHERAL. DIGILENT PMOD SD. (A) REFERENCE, (B) PIN CONFIGURATION, (C) CARD CONFIGURATION.

STD.	MICRO	PMOD	SPI BUS	1-bit SD BUS	4-bit SD BUS
1	2	CS/DAT3	nCS/SELECT	CD/DETECTION	DAT3/DATA3
2	3	DI/CMD	DI/DATA IN	CMD/RESPONSE	
3	NONE	VSSI	VSS/GROUND		
4	4	VDD	VDD/POWER		
5	5	CLK	CLK/CLOCK		
6	6	VSSI	VSS/GROUND		
7	7	DO/DAT0	DO/DATA OUT	DAT0/DATA0	
8	8	DAT1	nIRQ/INTERRUPT NC/UNUSED		DAT1/DATA1 nIRQ/INTERRUPT
9	1	DAT2	NC/UNUSED		DAT2/DATA2

Tabla 14: EXTERNAL PERIPHERAL. DIGILENT PMOD SD.

A PMOD based module is used as an external peripheral in order to incorporate the SD memory into the development board. The Digilent PMOD SD enables the reading from and the writing of cards without file-system or memory size limitations, allowing the storage of large amount of data. SD cards support several combinations of bus types and transfer modes based on SPI variants. Once the host-card negotiation is established, the interface is selected so the pins numbering is the same for all card families. See reference [7].

Mandatory. Speed: 12.5 - 25 MB/s.

- SPI bus: Serial Peripheral Interface (SPI) bus is primarily used by embedded microcontrollers. This bus type supports only a 3.3V interface. A host license is not required.
- One-bit SD bus: Commands/data channel are separated, a proprietary transfer format is used.

Optional. UHS-I speed: 50 - 104 MB/s; UHS-II speed: 156 - 312 MB/s.

- UHS-I (four-bit bus): It uses extra pins, in addition to some reassigned pins. This is the same protocol as the one-bit SD bus mode, it employs one command channel and four lines as data channel for faster transfer.
- UHS-II (two differential lines): It uses two low-voltage differential interfaces to transfer commands and data.

5.2.3 DEVELOPMENT BOARD: DIGILENT ARTY A7.

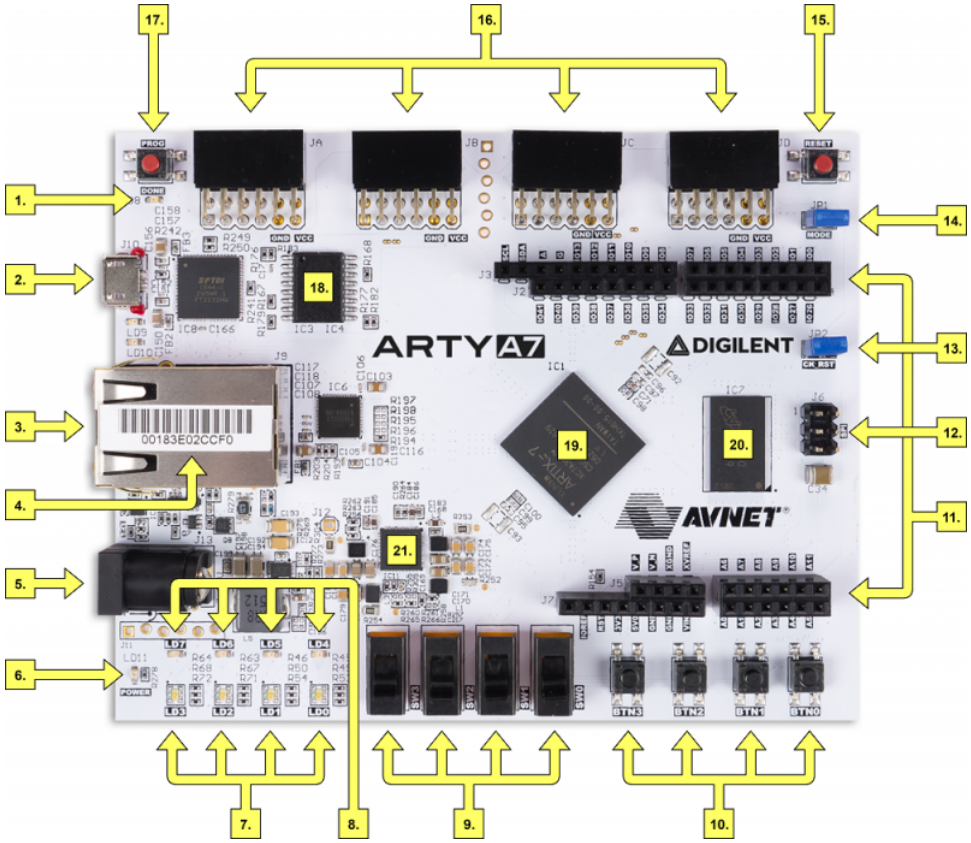


Figure 11: DEVELOPMENT BOARD. DIGILENT ARTY A7.

ID.	DESCRIPTION	ID.	DESCRIPTION
1	PROGRAMMING LED.	12	ARDUINO/CHIPKIT SHIELD SPI CONNECTORS.
2	USB JTAG/UART/POWER PORT.	13	CHIPKIT PROCESSOR RESET JUMPER.
3	10/100BASE-TX ETHERNET.	14	FPGA PROGRAMMING MODE.
4	MAC ADDRESS STICKER.	15	CHIPKIT PROCESSOR RESET.
5	EXTERNAL SUPPLY, 7-15V.	16	4x PMOD CONNECTORS. NOTE: JA, JB, JC, JD.
6	POWER LED.	17	FPGA PROGRAMMING RESET.
7	4x LED.	18	16MB QUAD-SPI FLASH MEMORY.
8	4x RGB LED.	19	XILINX ARTIX-7 FPGA, XC7A35TICSG324-1L.
9	4x SWITCHES. NOTE: SW0, SW1, SW2, SW3.	20	MICRON 256MB DDR3L MEMORY 16b @ 667MHz.
10	4x BUTTONS. NOTE: BTN0, BTN1, BTN2, BTN3.	21	DIALOG SEMICONDUCTOR DA9062 POWER SUPPLY.
11	ARDUINO/CHIPKIT SHIELD CONNECTORS.(IOs)		

Tabla 15: DEVELOPMENT BOARD. DIGILENT ARTY A7(I).

The Arty A7 is a development platform designed around the Artix-7 FPGA from Xilinx. Its flexibility makes it capable of adapting to different project functionalities. A wide variety of digital buses are available, e.g. SPI, I2C and Ethernet. In addition, Arty A7 is fully compatible with Vivado Design Suite. See reference [8].

- **Artix-7 FPGA**. It provides a high performance-per-watt fabric, transceiver line rates, DSP processing and integration in cost-optimized devices. Featuring the Microblaze soft processor and 1066 Mbps DDR3L support, this family is a suitable option for a wide variety of software-defined radio projects, such as radar systems. See figure [16].
- **Micron 256MB DDR3L.** The development board has an integrated DDR3L memory that must be properly connected to the FPGA. Therefore, a suitable interface must be developed to manage the reading, writing and control of memory operations.
- **10/100BASE-TX Ethernet.** The development board has an integrated Ethernet peripheral that must be used as external interface. Therefore, a suitable controller must be developed to manage the reading, writing and control of interface operations.

RESOURCES		**ARTIX-7 XC7A35T**
LOGIC	LOGIC CELLS	32280
	SLICES	5200
	CLB FLIP-FLOPS	41600
MEMORY	MAXIMUM DRAM	400 Kb
	BLOCK RAM/FIFO W/ ECC	50 (36 Kb EACH)
	TOTAL BLOCK RAM (Kb)	1800 KB
CLOCK MANAGEMENT		5
DSP		90
GTP TRANSCEIVERS		4 (6.6 Gb/s MAX. RATE)
I/O PINS		250

Tabla 16: DEVELOPMENT BOARD. DIGILENT ARTY A7(II). ARTIX-7 XC7A35T FPGA RESOURCES.

5.3 DEVELOPMENT TOOLS.

5.3.1 ELECTRONIC INSTRUMENTATION.

The set of hardware tools used for the project development are the following.

- **Signal generator**: It is an electronic device that generates repeating or non-repeating electronic signals in either the analog or the digital domain. These generated signals are used as a stimulus for electronic measurements, typically used in designing and testing of electronic devices.
- **Oscilloscope**: It is a type of electronic test instrument that graphically displays varying signal voltages, usually as a two-dimensional plot of one or more signals as a function of time.

5.3.2 SOFTWARE DEVELOPMENT KIT (SDK).

The set of software tools used for the project development are the following.

- **Vivado Design Suite**: It is a software suite produced by Xilinx for the synthesis and analysis of HDL designs, superseding Xilinx ISE with additional features for system on a chip development and high-level synthesis. See reference [9].

  - Vivado Synthesis: It is a compiler responsible for transforming an RTL-specified design into a gate-level representation. Synthesis is timing-driven and optimized for memory usage and performance. It supports mixed-languages, such as Verilog or VHDL.
  - Vivado Simulator: It is a compiled simulator that supports mixed-languages, such as Verilog or VHDL, TCL scripts, encrypted IP and enhanced verification. It does not have a design size, instance or line limitations and it allows the running of unlimited instances of mixed-language simulations.

- **Intellectual Property (IP) core**: A IP core, or IP block, is a reusable unit of logic, cell or integrated circuit (IC) design that belongs to one party. The term is derived from the licensing of the patent and/or source code copyright that exists in the design. IP core can be used as a building block within FPGA logic design.

  - ChipScope Integrated Logic Analyzer (ILA): It is a customizable logic analyzer core that can be used to monitor any internal signal of the design. The ILA core includes many advanced features of modern logic analyzers, including boolean trigger equations, trigger sequences, and storage qualification. Because the ILA core is synchronous to the design being monitored, all design clock constraints that are applied to the design are also applied to the components inside the ILA core. See reference [10].
  - Memory Interface Generator (MIG): It is a free software tool used to generate memory controllers and interfaces for Xilinx FPGAs. Memory interface generates VHDL design files, UCF constraints, simulation files and implementation script files to simplify the design process. See reference [11].

- **VHDL**: It is a hardware description language used in electronic design automation to describe digital and mixed-signal systems such as FPGAs, see reference [12].

- **TCL**: It is a standard language used in the semiconductor industry for the application of programming interfaces, and it is used by testing designs. See reference [13].

- **Git**: It is a distributed version-control system for tracking changes in source code during software development. It is designed for coordinating work among programmers, but it can be used too to track changes in any set of files. Its goals include speed, data integrity, and support for distributed, non-linear workflows. See reference [14].

- **MATLAB:** It is a multi-paradigm numerical computing environment and proprietary programming language developed by MathWorks. MATLAB allows matrix manipulations, plotting of functions and data, implementation of algorithms, creation of user interfaces, and interfacing with programs written in other languages. See reference [15].

6 ARCHITECTURE DESIGN.

6.1 SYSTEM ARCHITECTURE.

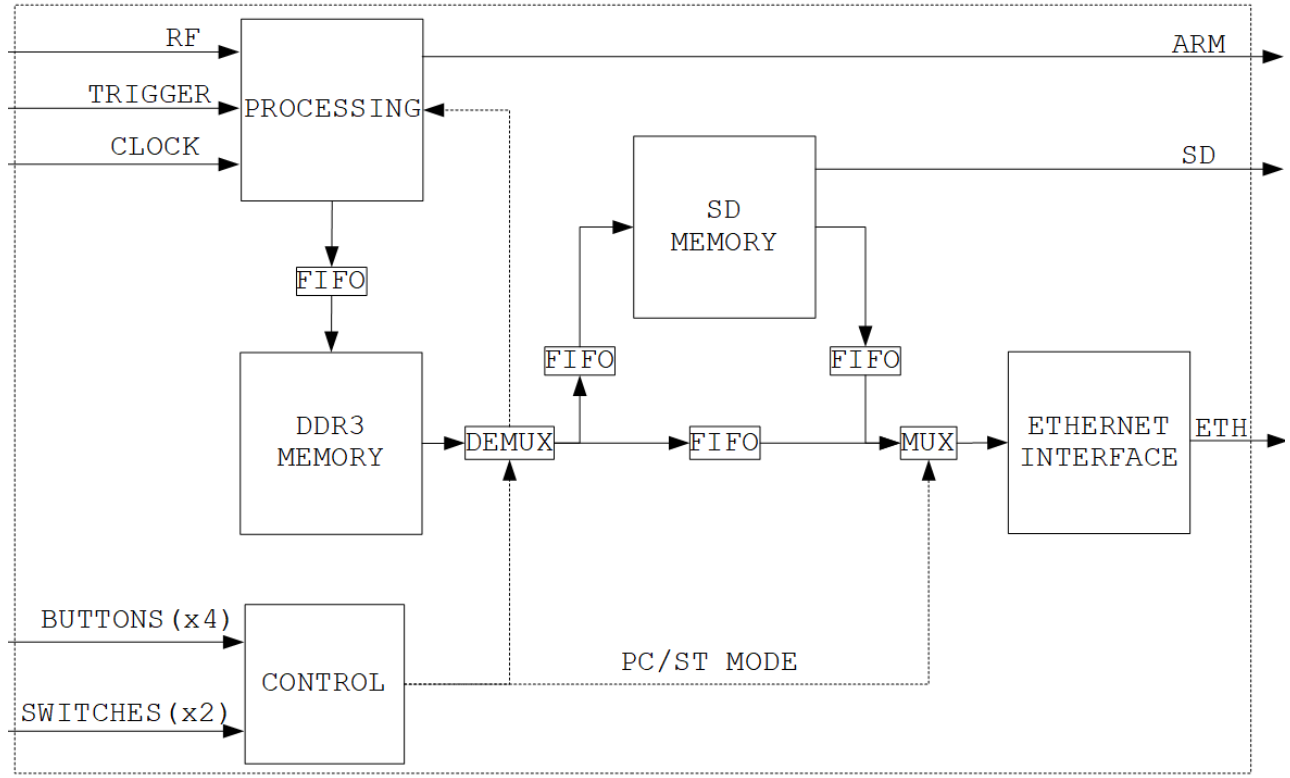


Figure 12: SYSTEM ARCHITECTURE. SYSTEM DESCRIPTION.

The main prototype functionalities are designed in the FPGA device of the development board. It runs tasks such as the management of the radar system and user interface, as well as the hardware modules. The modules must support necessary operations to control the hardware elements that make up the prototype. Additionally, a suitable architecture is designed for a correct interconnection and management of these modules.

- **Radar system management.** The prototype must adequately manage the radar system, as well as the radar signalling and signals (RF), see section [5.1.1]. Its main task is the integration of the external A/D peripheral into the development board, see sections [5.2.1] & [5.2.3]. Requirements sections [4.1.1] & [4.1.2].
- **User interface.** The prototype must adequately manage the operation modes and functionalities through the buttons and switches available on the development board, see section [5.2.3]. Requirements sections [4.1.3] & [4.1.4].

6.1.1 HARDWARE MODULES.

As stated above, the modules act as controllers of the hardware elements that make up the prototype. They must adequately manage the operations needed to control these hardware elements, whether they are on-board or external peripherals of the development platform. Furthermore, a proper handshake architecture is designed as a communication protocol between the hardware modules, see section [6.2]. Each of the modules has its own architecture, which generally consists of a control unit and a processing unit, see section [6.3].

- **Control:** It coordinates the different modules that make up the system architecture, as well as the modules interconnection. Its main task is the integration of the unit operation modes and functionalities through the buttons and switches of the development board, see section [5.2.3]. Additionally, the external A/D peripheral acts as a master controller through the arming interrupt, see section [5.2.1]. Its design is based on a simple state machine, see section [E.2]. Validation and test section [7.1].

- **Processing:** It coordinates the tasks related to the digitization and processing of the radar signals, as well as the external A/D peripheral management as slave controller, see section [5.2.1]. In order to achieve this, the signalling is used as the radar trigger an external clock. Furthermore, it provides a signal processing algorithm based on down-sampling, which is controlled by resolution decimation rate, see section [D.1]. Its design is based on a complex state machine, see section [E.3]. Validation and test section [7.2].

- **DDR3 memory:** It coordinates the operations needed to control the integrated DDR3L memory on the development board. Its main task is to act as intermediate volatile memory between the acquisition and transfer stages. Due to the fact that the DDR3L memory is external to the FPGA, a suitable interface has been designed as communication interface between them, see section [5.2.3]. In orde to achieve this, an AXI interface has been created using a Vivado MIG tool, see section [D.2]. Its design is based on a control unit and a processing unit, see sections [6.3] & [E.4]. Validation and test section [7.3].

- **SD memory:** It coordinates the operations needed to control the external SD memory PMOD peripheral, see section [5.2.2]. Its main task is the storage of the processed data into a non-volatile SD memory. Additionally, in standalone mode, it dumps the stored data into the Ethernet interface. Due to the fact that the SD memory is external to the FPGA, a suitable interface has been designed as communication interface between them, see section [5.2.3]. In order to achieve this, a SPI variant is used as interface to the UHS-I SDHC standard, see sections [D.3]. Its design is based on a control unit and a processing unit, see sections [6.3] & [E.5]. Validation and test section [7.4].

- **Ethernet interface:** It coordinates the operations needed to control the integrated Ethernet interface on the development board. Its main task is the download of the processed data on a PC. Due to the fact that the Ethernet memory is external to the FPGA, a suitable interface has been designed as communication interface between them, see section [5.2.3]. In order to achieve this, 100BASE-TX standard is used as communication protocol, see section [D.4]. Its design is based on a control unit and a processing unit, see sections [6.3] & [E.6]. Validation and test section [7.5].

6.2 HANDSHAKE ARCHITECTURE.

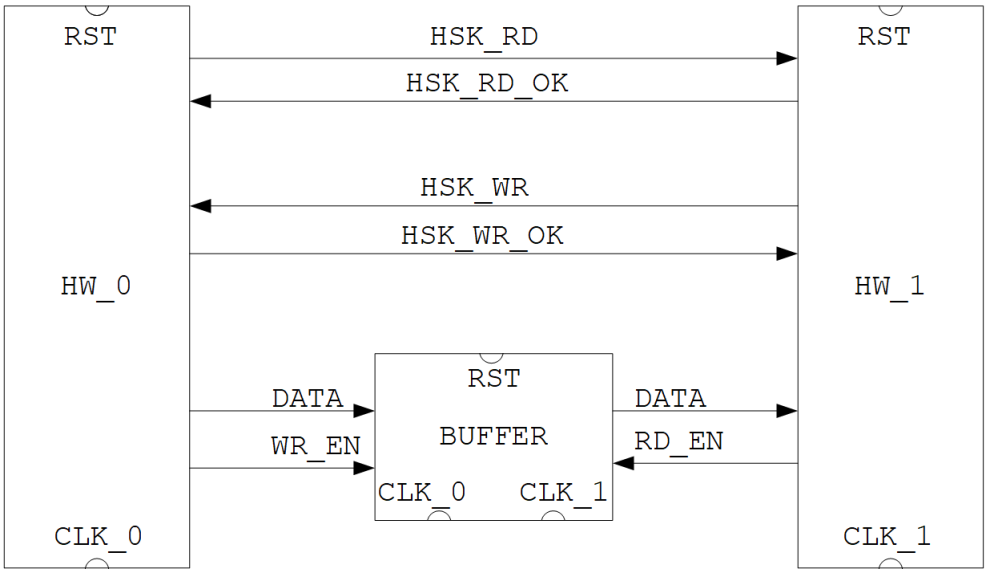


Figure 13: HANDSHAKE ARCHITECTURE. SYSTEM DESCRIPTION.

A handshake is an automated process of negotiation between two modules through the exchange of signals which establishes the protocol of a communication link. This link consists of a buffer with its respective data bus and control signals.

- **Read signalling**: It manages the buffer read operations of a module in a daisy chain architecture. When a previous module (HW_0) completes its buffer write operations, it triggers buffer read operations of the subsequent module (HW_1). Note: HSK_RD and HSK_RD_OK as read trigger/acknowledgement, respectively.
- **Write signalling**: It manages the buffer write operations of a module in a daisy chain architecture. When a subsequent module (HW_1) completes its buffer read operations, it triggers buffer write operations of the previous module (HW_0). Note: HSK_WR and HSK_WR_OK as write trigger/acknowledgement, respectively.

6.3 MODULES ARCHITECTURE.

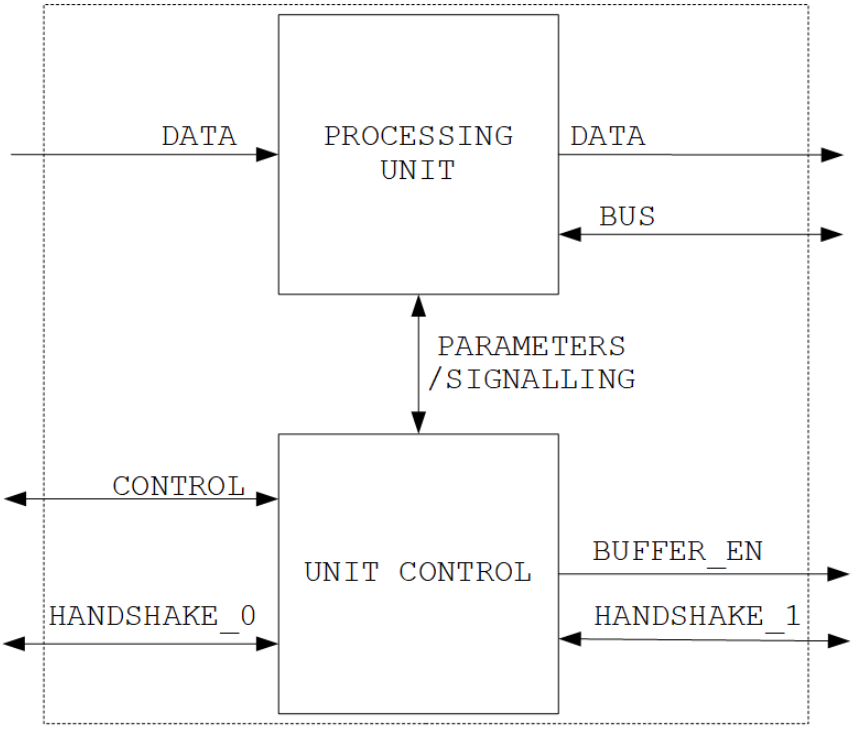


Figure 14: MODULE ARCHITECTURE. SYSTEM DESCRIPTION.

The structure of the designed modules is the following:

- **Unit Control (UC):** It is a component of a hardware module that directs the operation of the processing unit. The UC searches, decodes and executes instructions using the processing unit. It consists of a complex machine whose main components are the combinational logic circuit, the sequential logic circuit, the state control circuit and the control signal recognition.
- **Processing Unit (PU):** It is the electronic circuitry within a hardware module that executes the instructions that make up a hardware controller. The PU performs arithmetic, logic, controlling and input/output(IO) operations managed by the control unit.

MODULE	CONTROL UNIT				PROCESSING UNIT		SIGNALLING/
	BUFF0.	BUFF1.	HSK0	HSK1	DATA	BUS	PARAMETERS
DDR3 MEMORY	X	X	X	X	X	X	X
SD MEMORY	X	X	X	X	X	X	X
ETHERNET INTERFACE	X	NONE	X	NONE	X	X	X

Tabla 17: MODULE ARCHITECTURE. SYSTEM DESCRIPTION.

6.3.1 DDR3 MEMORY.

UNIT CONTROL (UC).

It coordinates the hardware controller, as well as the handshake negotiation process between the hardware modules that make up the prototype. Its main task is the management of the signalling and data of the hardware controller.

Its design is based on a complex state machine, see section [E.4].

PROCESSING UNIT (PU).

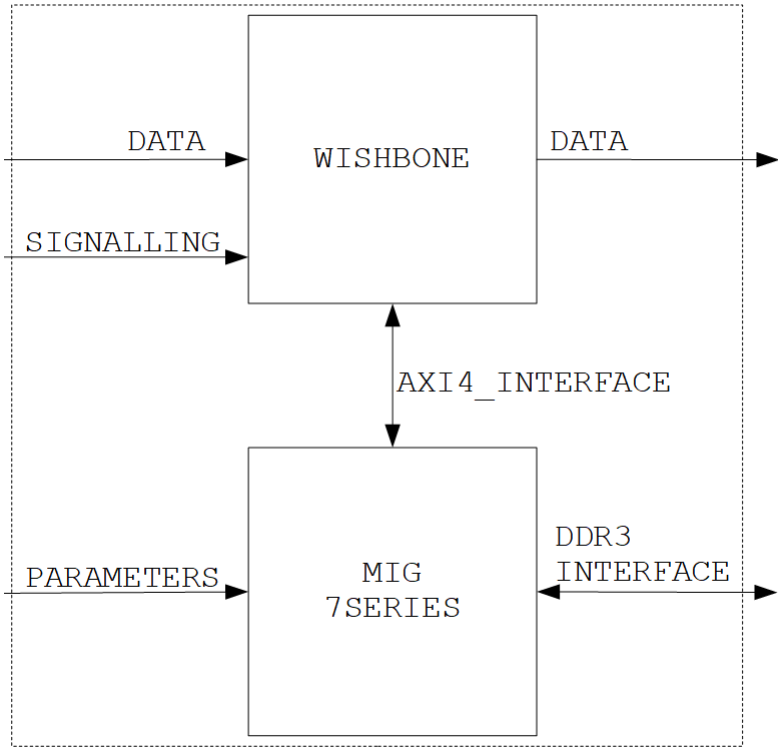


Figure 15: DDR3 MEMORY. PROCESSING UNIT. SYSTEM DESCRIPTION.

The hardware controller to be developed consists of the following modules.

- **MIG 7series**: The development board contains an integrated DDR3L memory that must be properly connected to the FPGA. Due to the fact that the DDR3L memory is external to the FPGA, a suitable interface has been designed as communication interface between them. To achieve this, an AXI interface has been created using the Vivado MIG IP tool, see reference [11]. See section [D.2].
- **Wishbone**: The AXI4 interface is mainly designed for on-chip communication, resulting in a complex bus to configure directly without a microcontroller. For simplicity reasons, an open-source IP Wishbone bus was selected to use as AXI4 converter. Thus, the interface is reduced to only 8 channels, while the AXI4 interface requires the use of 35 channels. Finally, the unit control (UC) can coordinate the operations needed to control the integrated DDR3L memory using a 8 channel bus. See reference [18].

6.3.2 SD MEMORY.

UNIT CONTROL (UC).

It coordinates the hardware controller, as well as the handshake negotiation process between the hardware modules that make up the prototype. Its main task is the management of the signalling and data of the hardware controller. Its design is based on a complex state machine, see section [E.5].

PROCESSING UNIT (PU).

As hardware controller, an open-source IP block has been used by way of solution as a SD interface which supports the UHS-I SDHC standard. See reference [19].

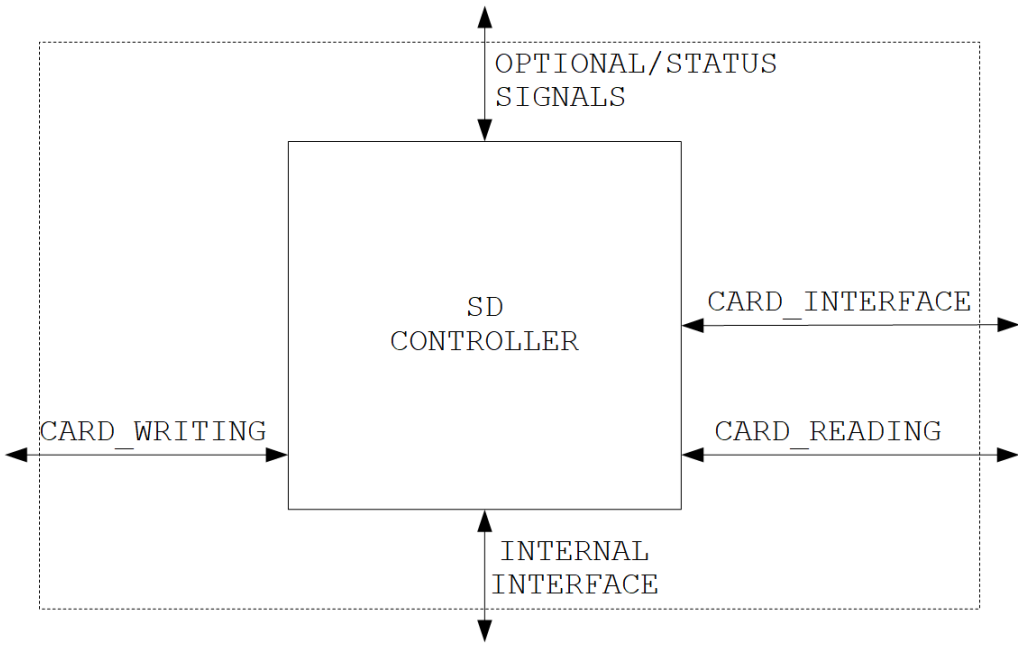


Figure 16: SD MEMORY. PROCESSING UNIT. SYSTEM DESCRIPTION.

The IP block interface is the following.

- **Card interface**: The SPI standard is used as a synchronous serial protocol, so that the development board can communicate with the external peripheral PMOD. This task is performed by the sclk, ss, mosi and miso signals, as well as the card present and write protection signals when required. The addressing is made by the sector number, 512B blocks. Additionally, the CRC error-checking is used in SPI transfers. See section [D.3].
- **Internal interface**:
  - clk (in): 50Mhz (or lower) clock. SPI clock is half of this.
  - addr (in): The 32bit sector address to be transferred.
  - sd_busy (out): It indicates whether the controller can accept a new command.
  - reset (in): To initialize the controller.
  - sd_error (out): It indicates that an error has occurred during initialization or transfer.
  - sd_error_code (out): It provides more information on the error.
- **Card reading**:
  - rd (in): Read trigger signal, it must be asserted for the entire transfer.
  - dout (out): 8bit data coming from the card.
  - dout_avail (out): It indicates that data are present on dout.
  - dout_taken (in): It indicates that data out has been accepted.

- rd_multiple (in): Read Multiple sectors trigger signal, assert for entire transfer

- **Card writing**:
  - wr (in): Write trigger signal, it must be asserted for the entire transfer.
  - din (in): 8bit data to be sent to the card.
  - din_valid (in): It indicates that data is present on din.
  - din_taken (out): It indicates that the controller has taken the data from din.
  - wr_multiple (in): Write Multiple sectors trigger signal, assert for entire transfer.
  - erase_count (in): 8bit count of pre-erased sectors for write multiple only

- **Optional/status signals**:
  - sd_type (out): 2bit card status.
  - sd_fsm (out): 8bit FSM state indication.

6.3.3 ETHERNET INTERFACE.

UNIT CONTROL (UC).

It coordinates the hardware controller, as well as the handshake negotiation process between the hardware modules that make up the prototype. Its main task is the management of the signalling and data of the hardware controller. Its design is based on a complex state machine, see section [E.6].

PROCESSING UNIT (PU).

As hardware controller, an open-source IP block has been used by way of solution to send raw data from the development board through the Ethernet interface. See reference [20].

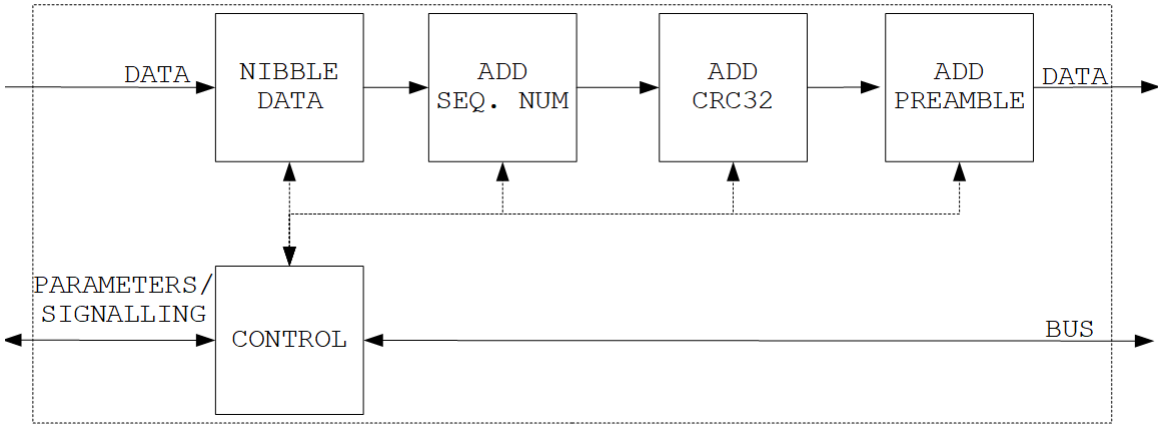


Figure 17: ETHERNET INTERFACE. PROCESSING UNIT. SYSTEM DESCRIPTION.

ID.	ABOUT	DEFAULT
ETH_SRC_MAC	MAC ADDRESS OF THE SENDER OF THE PACKET.	DE:AD:BE:EF:01:23
ETH_DST_MAC	MAC ADDRESS OF THE RECEIVER OF THE PACKET.	PC MAC ADDRESS
IP_SRC_ADDR	IP ADDRESS OF THE SENDER OF THE PACKET.	10.10.10.10
IP_DST_ADDR	IP ADDRESS OF THE RECEIVER OF THE PACKET.	10.10.10.1
UPD_SRC_PORT	UDP PORT OF THE SENDER OF THE PACKET.	4096
UDP_DST_PORT	UDP PORT OF THE RECEIVER OF THE PACKET.	4096

Tabla 18: ETHERNET INTERFACE. PROCESSING UNIT. CONFIGURATION PARAMETERS.

The IP block architecture is the following.

- **Control**: It coordinates the different modules that make up the IP block according to the parameters and signalling of the main control and control unit modules, respectively. Likewise, it is responsible for the interconnection of the modules that make up the processing unit. Its main task is the development of the 1000BASE-TX standard used as a communication protocol, see section [D.4].
- **Nibble data**: This module builds an UDP packet according to the TCP/IP model, specifically it is in charge of the transport and network layers of the protocol stack. Therefore, the following fields are addressed: IP header, UDP header and application data. Additionally, the MAC addresses and Ethernet length field are included. See section [D.5].
- **Add sequence number**: This module adds a two-byte sequence at the start of the application data. A verification field is added to guarantee the delivery, ordering or duplicate protection of packages.
- **Add crc32**: This module adds the CRC error-checking code of the Ethernet packet.
- **Add preamble**: This module adds the preamble and SFD fields of the Ethernet packet.

7 TEST AND VALIDATION.

The prototype development is checked through quality management procedures. These include tasks such as the architecture validation, as well as the modules test that make up the prototype through simulation analysis. All this should be made in a manner that allows the designer to guarantee the fulfillment of the prototype requirements. Due to the complexity of this issue, the prototype is analyzed module by module through unit testing.

UNIT TESTING.

Code testing method by which individual units of source-code, or sets of these, are simulated to determine whether they are fit for use. The modules repositories are the following.

- UT_CONTROL_MODULE: See sections [7.1] & [7.1.2].
- UT_PROCESSING_MODULE: See sections [7.2] & [7.2.2].
- UT_DDR3_MEMORY_MODULE: See sections [7.3] & [7.3.2].
- UT_SD_MEMORY_MODULE: See sections [7.4] & [7.4.2].
- UT_ETHERNET_INTERFACE_MODULE: See sections [7.5] & [7.5.2].

SIMULATION AND RTL ANALYSIS.

They are used as test and validation tools, as follows.

- **Register-transfer level (RTL)**: It is a design abstraction which models a synchronous digital circuit in terms of the flow of digital signals between hardware registers, and the logical operations performed on those signals. See sections [7.1.1], [7.2.1], [7.3.1], [7.4.1], & [7.5.1]
- **Behavioral simulation**: It allows the verification of syntax and functionality without timing information. During design development, most verification is accomplished through behavioral simulation. Errors identified early in the design cycle are inexpensive to fix compared to functional errors identified during silicon debug. See sections [7.1.3], [7.2.3] & [7.5.3].

7.1 CONTROL MODULE.

7.1.1 REGISTER-TRANSFER LEVEL.

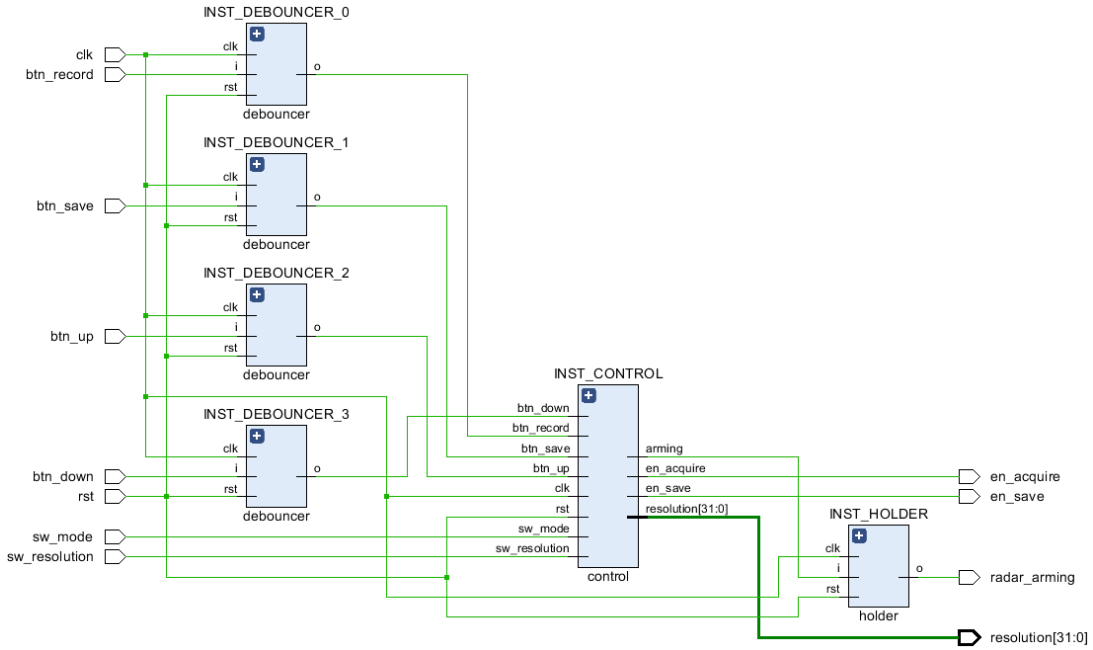


Figure 18: CONTROL MODULE. REGISTER-TRANSFER LEVEL.

The module to simulate consists of the following elements.

- **Control:** As mentioned in section [6.1.1], it coordinates the different modules that make up the system architecture, as well as the modules interconnection. Its main task is the integration of the prototype operation modes and functionalities through the buttons and switches of the development board. Additionally, the external A/D peripheral management as master controller through the arming interrupt. Its design is based on a simple state machine, see section[E.2].
- **Debouncer:** The bouncing is the non-ideal behavior of any button which generate multiple transitions of a single input. Hence, in order to solve this issue, a debouncing circuit is used.
- **Holder:** It is a circuit used to hold short-temporal input signals.

7.1.2 UNIT TESTING.

TEST		DESCRIPTION	DEPENDENCY
RST.	UT.1	IT INITIALIZES FSM RESET. OPERATION: DEFAULT VALUES.	NONE.
	UT.2	IT DISABLES ACQUIRE PROCESS.	UT.1, UT4.
ARM.	UT.3	IT INITIALIZES FSM ARMING. OPERATION: RADAR RESET/ENABLE.	NONE.
REC.	UT.4	IT ENABLES/DISABLES FSM RECORD. OPERATION: ACQUIRE PROCESS.	UT.3
SAVE	UT.5	IT ENABLES/DISABLES FSM SAVE. OPERATION: SAVE PROCESS.	NONE.
	UT.6	ONLY AVAILABLE IN STANDALONE(ST) MODE.	UT.5
	UT.7	IT DISABLES ACQUIRE PROCESS.	UT.4, UT.5
RES.	UT.8	IT INITIALIZES FSM UP/DOWN. OPERATION: SET RESOLUTIONS.	NONE.
	UT.9	ONLY AVAILABLE IN RESOLUTION MODE.	UT.8
	UT.10	IT DISABLES ACQUIRE PROCESS.	UT.4, UT.8

Tabla 19: CONTROL. UNIT TESTING.

Hereafter, the results obtained from the unit testing carried out through simulation analysis are explained, see section [7.1.3]. Also see the TCL commands used, section [E.2]. In order to test the external A/D peripheral management, an appropriate electronic instrumentation must be used.

Unit Testing(I). UT. 1-4. When the reset switch is active, the module remains in an idle state restoring the default parameters. If the acquire process is enabled, it is blocked immediately, and if necessary, the module will reset the radar system. In ordinary mode, the acquire process is enabled or disabled according to the record button. To perform these tasks, the module uses the acquire enable and arming interrupt signals. See table [19].

Unit Testing(II). UT. 5-7. In standalone (ST) mode, when the mode switch is active, the save process is enabled or disabled according to the save button. Similar to the previous U.T. 1-4, if the acquire process is enabled, it is blocked immediately, and if necessary, the module will reset the radar system. To perform these tasks, the module uses the acquire enable, save enable and arming interrupt signals. See table [20].

Unit Testing(III). UT. 8-10. When the resolution switch is active, the resolution mode is enabled and the decimate rate is set according to the resolution (UP/DOWN) buttons. Similar to the previous U.T.1-4, if the acquire process is enabled, it is blocked immediately, and if necessary, the module will reset the radar system. To perform these tasks, the module uses the acquire enable, resolution and arming interrupt signals. See table [21].

7.1.3 BEHAVIORAL SIMULATION.

UNIT TESTING(I). UT. 1-4.

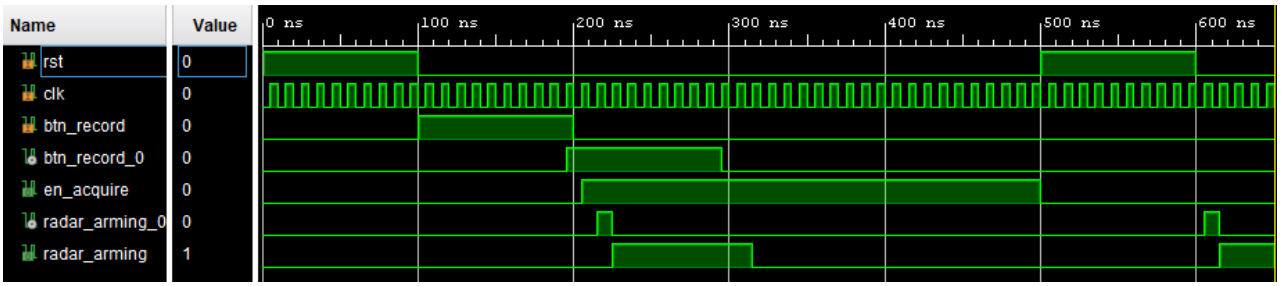


Figure 19: CONTROL MODULE. UNIT TESTING(I). UT. 1-4.

UNIT TESTING(II). UT. 5-7.

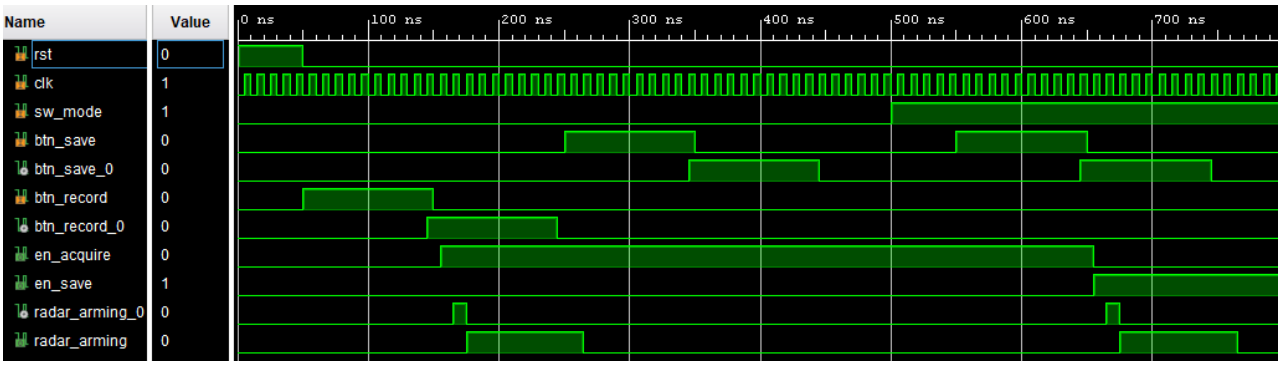


Figure 20: CONTROL MODULE. UNIT TESTING(II). UT. 5-7.

UNIT TESTING(III). UT. 8-10.

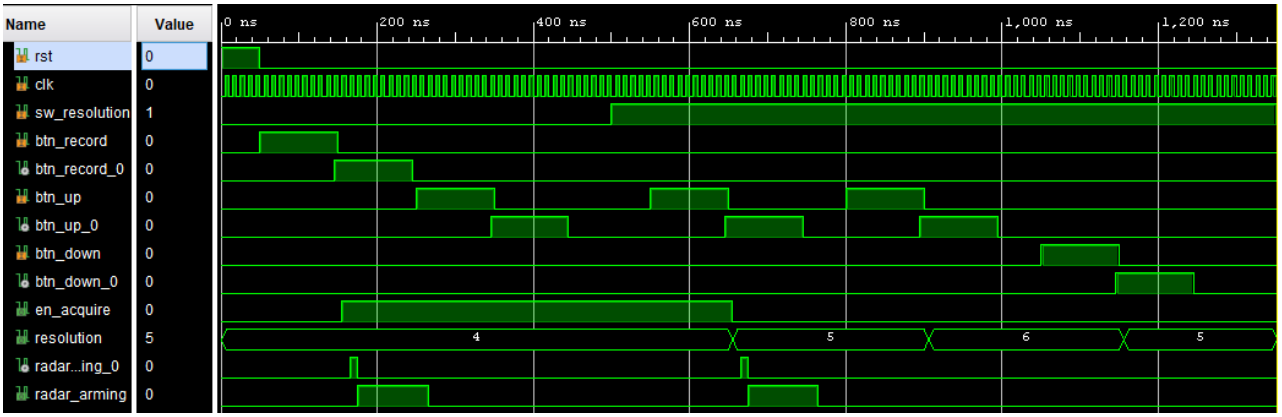


Figure 21: CONTROL MODULE. UNIT TESTING(III). UT. 8-10.

7.2 PROCESSING MODULE.

7.2.1 REGISTER-TRANSFER LEVEL.

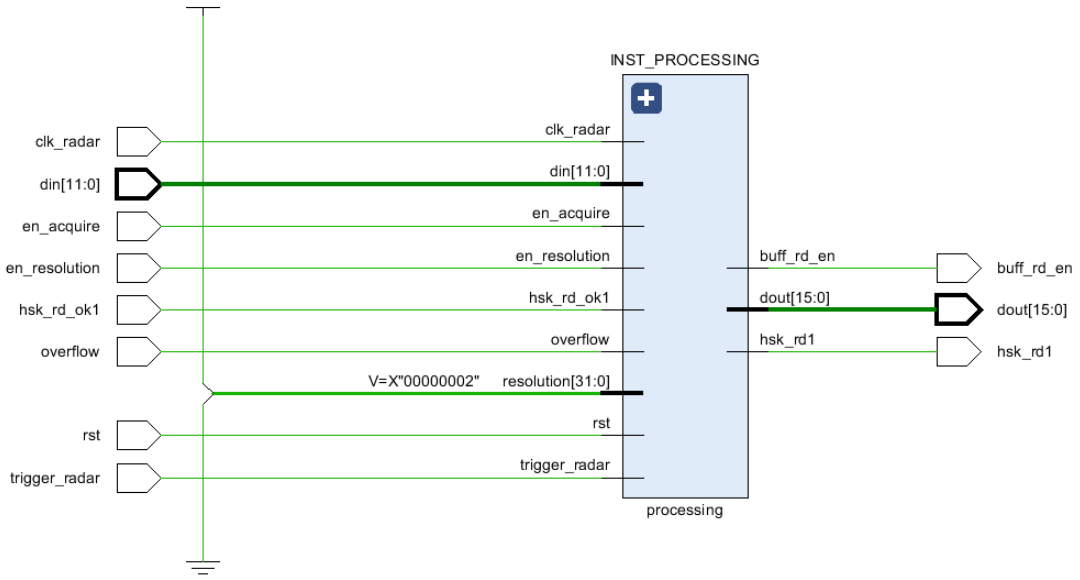


Figure 22: PROCESSING MODULE. REGISTER-TRANSFER LEVEL.

The module to be simulated consists of the following element.

- **Processing:** As mentioned in section [6.1.1], it coordinates the tasks related to the digitization and processing of the radar signals, as well as the external A/D peripheral management as slave controller. In order to achieve this, the signalling is used as the radar to trigger an external clock. Furthermore, it provides a signal processing algorithm based on down-sampling, which is controlled by resolution decimation rate. Its design is based on a complex state machine, see section [E.3].

7.2.2 UNIT TESTING.

TEST			DESCRIPTION	DEPENDENCY
GENERAL	RESET	UT.1	OPERATION: DEFAULT VALUES.	NONE
	BUFFER	UT.2	OPERATION: WRITE BUFFER ENABLE.	NONE
	HANDSHAKE1	UT.3	OPERATION: DRAM HANDSHAKE.	NONE
SPECIFIC	ENABLE	UT.4	IT ENABLES/DISABLES FSM ENABLE. OPERATION: SAMPLING MODE.	UT.1
		UT.5	ONLY AVAILABLE IF ACQUIRE FLAG IS ENABLED.	UT.4
		UT.6	INITIALED WHEN RADAR TRIGGER IS RECEIVED.	UT.4
	OVERFLOW	UT.7	SAMPLED DATA OVERFLOW.	NONE
	BASIC	UT.8	BASIC ACQUIRE ALGORITHM. OPERATION: ACQUIRE PROCESS.	UT.2, UT.3, UT.4
	EXTENDED	UT.9	EXTENDED ACQUIRE ALGORITHM. OPERATION: DOWN-SAMPLING PROCESS.	UT.2, UT.3, UT.4
		UT.10	ONLY AVAILABLE IN RESOLUTION MODE.	UT.9

Tabla 20: PROCESSING MODULE. UNIT TESTING.

Hereafter, the results obtained from the unit testing carried out through simulation analysis are explained, see section [7.2.3]. Also see the TCL commands used, section [E.3]. To test the external A/D peripheral management, an appropriate electronic instrumentation must be used.

Unit Testing(I). UT. 1-8. As previously stated, the reset signal sets the module in idle state restoring the default parameters. Furthermore, the module contains a handshake signalling for interconnection with other modules. These tasks are performed by the handshake and buffer enable signals. The sampling mode is enabled or disabled according to the acquire enable flag. Once the radar trigger is received, a N samples packet is digitized according to the NDATA module parameter. In standalone (ST) mode, for memory reasons, only a limited number of packets can be acquired according to the NPACKETS module parameter. This task is performed by the radar trigger and data input signals, as well as the overflow flag when required. During this interval, the buffer remains enabled, storing the digitized data. Finally, the buffer is disabled and the read handshake triggering indicates that the packet has been successfully buffered. See table [23].

Unit Testing(II). UT. 9-10. Similar to the previous U.T. 1-8, the down-sampling process is enabled or disabled according to the resolution enable flag. In extend mode, a N samples packet is digitized and processed according to the decimate rate. This task is performed by the resolution enable and resolution rate.

- Down-sampling by 2, see table [24].
- Down-sampling by 4, see table [25].

7.2.3 BEHAVIORAL SIMULATION

UNIT TESTING(I). U.T. 1-8.

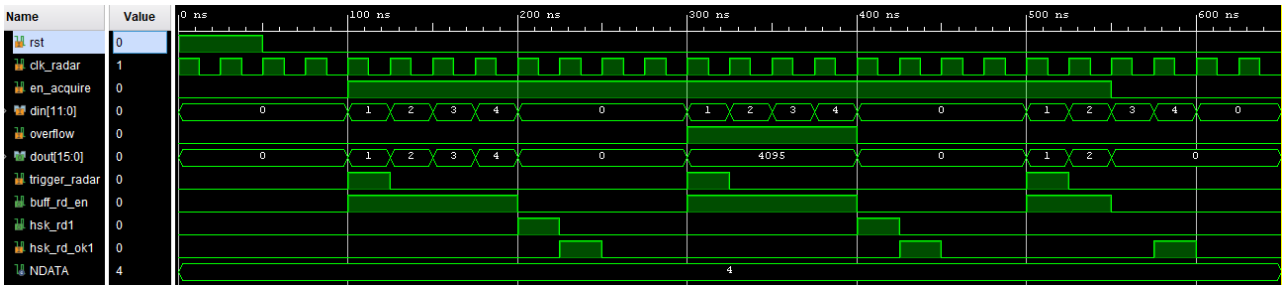


Figure 23: PROCESSING MODULE. UNIT TESTING(I). U.T. 1-8.

UNIT TESTING(II). U.T. 9-10, DOWN-SAMPLING BY 2.

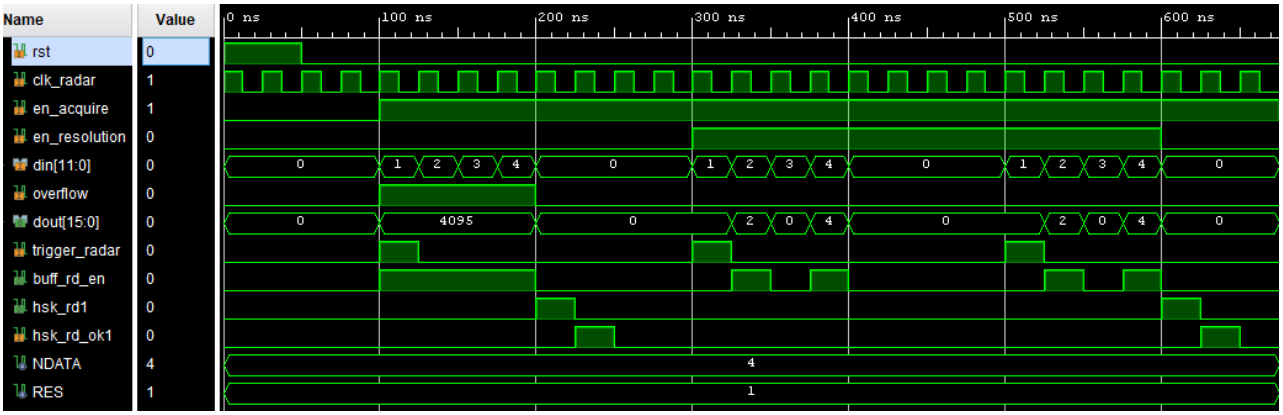


Figure 24: PROCESSING MODULE. UNIT TESTING(II). U.T. 9-10, DOWN-SAMPLING BY 2.

UNIT TESTING(II). U.T. 9-10, DOWN-SAMPLING BY 4.

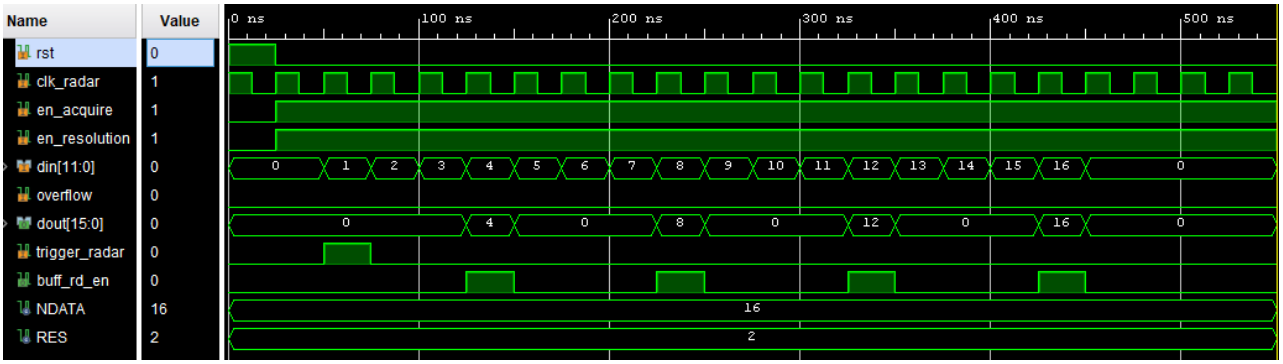


Figure 25: PROCESSING MODULE. UNIT TESTING(II). U.T. 9-10, DOWN-SAMPLING BY 4.

7.3 DDR3 MEMORY MODULE.

7.3.1 REGISTER-TRANSFER LEVEL.

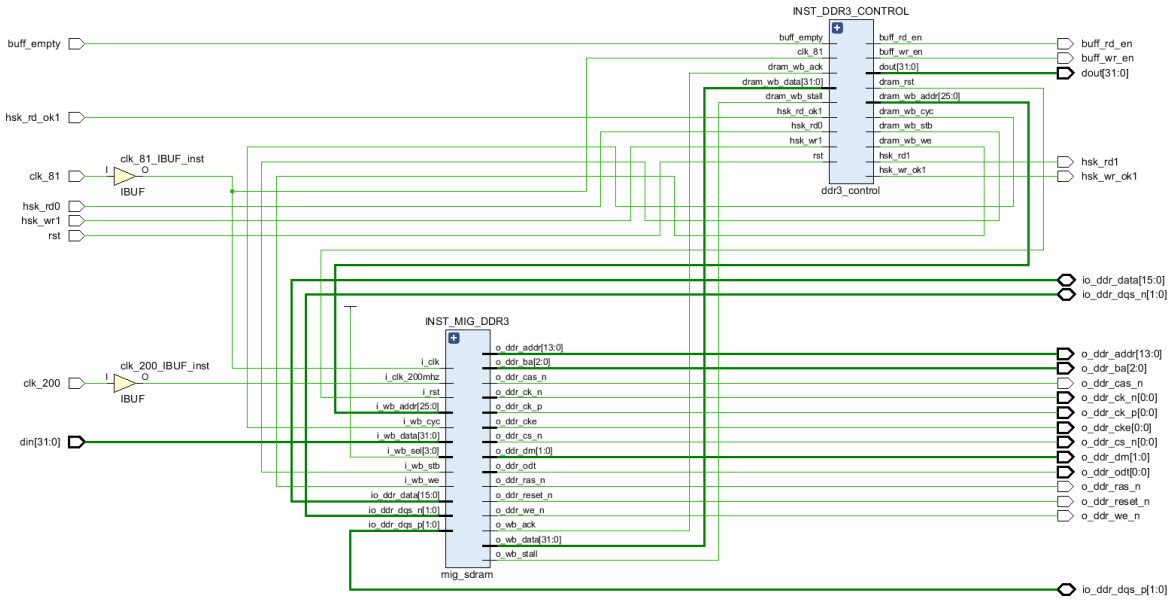


Figure 26: DDR3 MEMORY MODULE. REGISTER-TRANSFER LEVEL.

As mentioned in section [6.1.1], it coordinates the operations needed to control the integrated DDR3L memory on the development board. Its main task is to act as intermediate volatile memory between the acquisition and transfer stages. Due to the fact that the DDR3L memory is external to the FPGA, a suitable interface has been designed as communication interface between them. To achieve this, an AXI interface has been created using Vivado MIG tool. The module to be simulated consists of the following elements. See section [6.3.1].

- **DDR3 control:** It coordinates the hardware controller, as well as the handshake negotiation process between the hardware modules that make up the prototype. Its main task is the management of the signalling and data of the hardware controller. Its design is based on a complex state machine, see section [E.4].
- **MIG DDR3:** As a hardware controller, a combination of two IP blocks has been used by way of solution as a DDR3L memory interface. See section [6.3.1].
  - Vivado MIG IP tool, see reference [11].
  - Wishbone converter, see reference [18].

7.3.2 UNIT TESTING.

TEST			DESCRIPTION	DEPENDENCY
GENERAL	RESET	UT.1	OPERATION: DEFAULT VALUES.	NONE
	BUFFER	UT.2	OPERATION: READ/WRITE BUFFER ENABLE.	NONE
	HANDSHAKE0	UT.3	OPERATION: DRAM HANDSHAKE.	NONE
	HANDSHAKE1	UT.4	OPERATION: INTERFACE/MEMORY HANDSHAKE.	NONE
SPECIFIC	IDLE	UT.4	WAIT FOR HANDSHAKE0. OPERATION: WAITING MODE.	UT.3
	SEND	UT.5	READ BUFFER & HANDSHAKE0 MANAGEMENT. OPERATION: SENDING MODE.	UT.2, UT.3
	RECEIVE	UT.6	WRITE BUFFER & HANDSHAKE1 MANAGEMENT. OPERATION: RECEIVING MODE.	UT.2, UT.4
	WAIT_FOR	UT.7	WAIT FOR HANDSHAKE0 & HANDSHAKE1. OPERATION: WAITING MODE.	UT.3, UT.4

Tabla 21: DDR3 MEMORY MODULE. UNIT TESTING.

In order to test the interface used to enable the communication between the FPGA and the integrated DD3L memory, there is no other option but to monitor the internal signals while the system is running. This is due to the fact that there is no simple way to simulate memory paging. In order to achieve this, ChipScope ILA tool must be used. See reference [10].

7.4 SD MEMORY MODULE.

7.4.1 REGISTER-TRANSFER LEVEL.

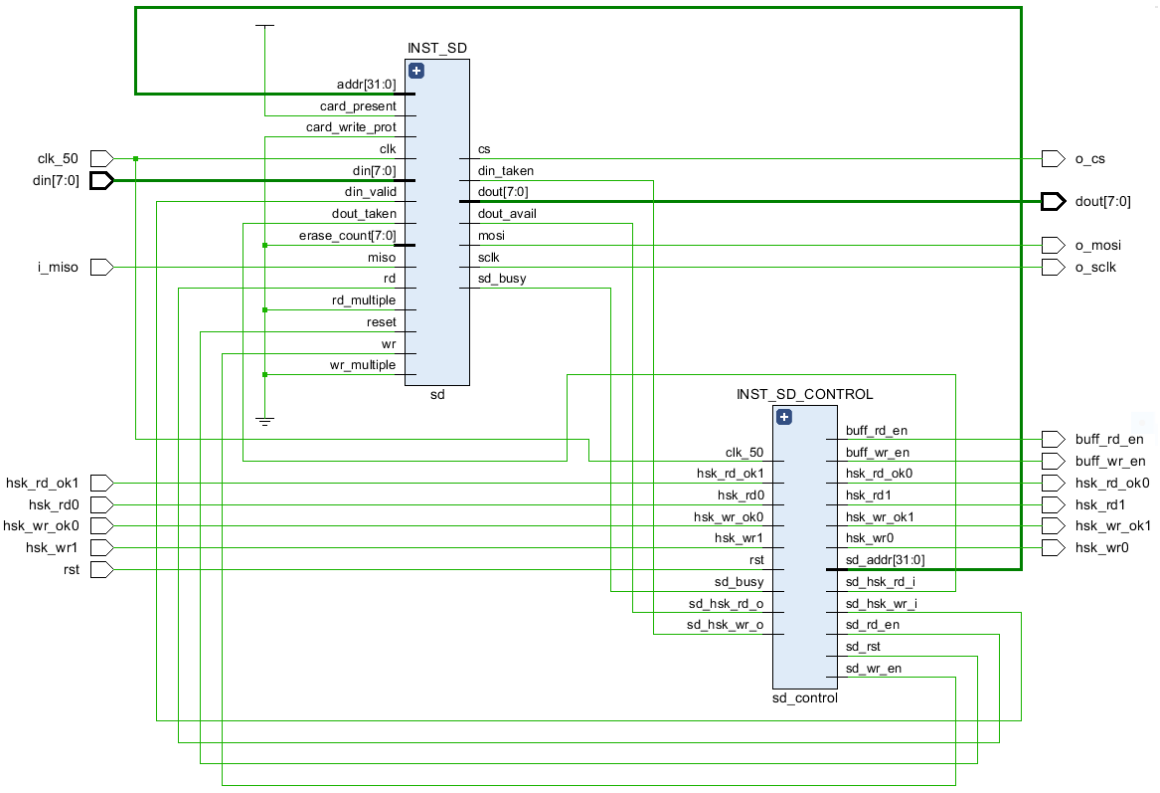


Figure 27: MEMORY MODULE. REGISTER-TRANSFER LEVEL.

As mentioned in section [6.1.1], it coordinates the operations needed to control the external SD memory PMOD peripheral. Its main task is the storage of the processed data into a non-volatile SD memory. Additionally, in standalone mode, it dumps the stored data into the Ethernet interface. Due to the fact that the SD memory is external to the FPGA, a suitable interface has been designed as a communication interface between them. To achieve this, a SPI variant is used as an interface for the UHS-I SDHC standard. The module to be simulated consists of the following elements. See section [6.3.2].

- **SD control:** It coordinates the hardware controller, as well as the handshake negotiation process between the hardware modules that make up the prototype. Its main task is the management of the signalling and data of the hardware controller. Its design is based on a complex state machine, see section [E.5].
- **SD:** As a hardware controller, an open-source IP block has been used by way of solution as a SD interface which supports the UHS-I SDHC standard, see reference [19]. See section [D.3].

7.4.2 UNIT TESTING.

TEST			**DESCRIPTION**	**DEPENDENCY**
GENERAL	RESET	UT.1	OPERATION: DEFAULT VALUES.	NONE
	BUFFER	UT.2	OPERATION: READ/WRITE BUFFER ENABLE.	NONE
	HANDSHAKE0	UT.3	OPERATION: DRAM HANDSHAKE.	NONE
	HANDSHAKE1	UT.4	OPERATION: INTERFACE HANDSHAKE.	NONE
SPECIFIC	IDLE	UT.4	WAIT FOR HANDSHAKE0. OPERATION: WAITING MODE.	UT.3
	WRITE	UT.5	READ BUFFER & HANDSHAKE0 MANAGEMENT. OPERATION: WRITING MODE.	UT.2, UT.3
	READ	UT.6	WRITE BUFFER & HANDSHAKE1 MANAGEMENT. OPERATION: READING MODE.	UT.2, UT.4
	WAIT_FOR	UT.7	WAIT FOR HANDSHAKE0 & HANDSHAKE1. OPERATION: WAITING MODE.	UT.3, UT.4

Tabla 22: MEMORY MODULE. UNIT TESTING.

Similarly to the previous section, in order to test the interface used for the communication between the FPGA and the external SD memory PMOD peripheral, there is no other option but to monitor the internal signals while the system is running. This is due to the fact that there is no simple way to simulate memory paging. See reference [10]. In order to test the standard SPI interface, the use of an appropriate electronic instrumentation is enough.

7.5 ETHERNET INTERFACE MODULE.

7.5.1 REGISTER-TRANSFER LEVEL.

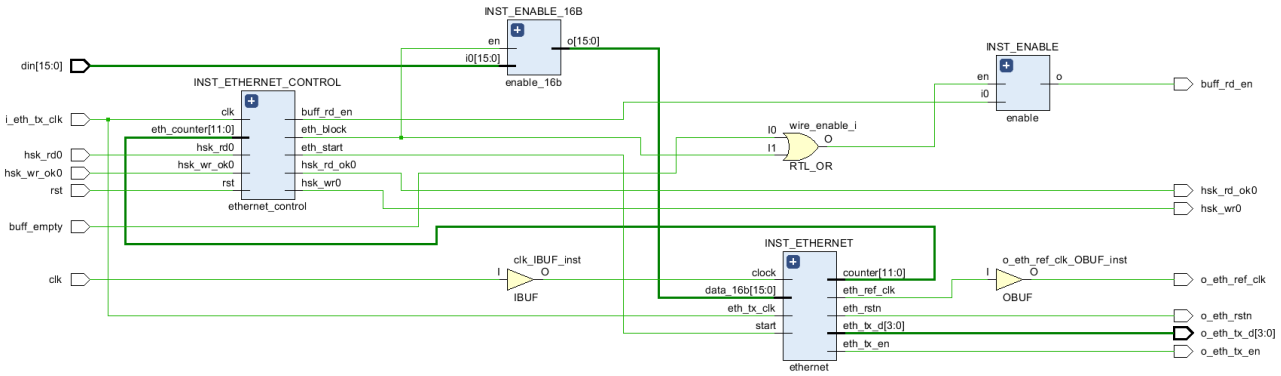


Figure 28: INTERFACE ETHERNET MODULE. REGISTER-TRANSFER LEVEL.

It coordinates the operations needed to control the integrate Ethernet interface on the development board. Its main task is the download of the processed data on a PC. Due to the fact that the Ethernet memory is external to the FPGA, a suitable interface has been designed as a communication interface between them. To achieve this, the 100BASE-TX standard is used as a communication protocol. The module to be simulated consists of the following elements. See section [6.3.3].

- **Ethernet control:** It coordinates the hardware controller, as well as the handshake negotiation process between the hardware modules that make up the prototype. Its main task is the management of the signalling and data of the hardware controller. Its design is based on a complex state machine, see section [E.6].
- **Ethernet:** As a hardware controller, an open-source IP block has been used by way of solution to send raw data from the development board through the Ethernet interface, see reference [20]. See sections [D.5] & [D.4].
- **Enable:** It is a circuit used to enable or disable an input signal according to a control signal.

7.5.2 UNIT TESTING.

TEST			DESCRIPTION	DEPENDENCY
GENERAL	RESET	UT.1	OPERATION: DEFAULT VALUES.	NONE
	BUFFER	UT.2	OPERATION: READ BUFFER ENABLE.	NONE
	HANDSHAKE0	UT.3	OPERATION: DRAM/MEMORY HANDSHAKE.	NONE
SPECIFIC	IDLE	UT.4	WAIT FOR HANDSHAKE0 OPERATION: WAITING MODE.	UT.3
	SEND	UT.5	READ BUFFER & HANDSHAKE0 MANAGEMENT. OPERATION: SENDING MODE.	UT.2, UT.3
	PACKET	UT.6	OPERATION: BUILD AN UDP PACKET ACCORDING TO TCP/IP MODEL.	NONE

Tabla 23: ETHERNET INTERFACE MODULE. UNIT TESTING.

Hereafter, the results obtained from the unit testing carried out through a simulation analysis are explained, see section [7.5.3]. Also see the TCL used commands, section [E.6].

Unit Testing(I). UT. 1-5. As previously stated, the reset signal sets the module in idle state restoring the default parameters. Furthermore, the module includes a handshake signalling for interconnection with other modules. These tasks are performed by the handshake and buffer enable signals. The send state is started according to the read handshake triggering. During this interval, a TCP/IP packet is sent through the Ethernet interface. This task is performed by the Ethernet signalling. Obviously, the buffer remains enabled during the payload reading. Finally, the buffer is disabled and the write handshake triggering indicates that the packet has been successfully sent. See table [29].

Unit Testing(II). UT. 6, TCP/IP PACKET. It is shown how an UDP packet is built according to a TCP/IP model. In order to achieve the necessary visibility, the simulation delves into the nibble module signalling of the processing unit. Thus, it is possible to see the raw data which make up an UDP packet. It should be noted that the interface uses a big-endian architecture whose data are divided into 4-bit flipped pieces. To perform this task, the nibble module uses the data and the counter signals.

- MAC address. See table [30].
  - ETH_SRC_MAC: MAC address of the sender of the packet. Note: DE:AD:BE:EF:01:23.
  - ETH_DST_MAC: MAC address of the receiver of the packet. Note: PC MAC ADDRESS.
- IP address: See table [31].
  - IP_SRC_ADDR: IP address of the sender of the packet. Note: 10.10.10.10.
  - IP_DST_ADDR: IP address of the receiver of the packet. Note: 10.10.10.1.
- UDP ports address: See table [32].
  - UPD_SRC_PORT: UDP port of the sender of the packet. Note: 4096.
  - UDP_DST_PORT: UDP port of the sender of the packet. Note: 4096.
- Payload: See table [33].
  - DATA_IN: 1001, 1002, 1003 and 1004.

7.5.3 BEHAVIORAL SIMULATION

UNIT TESTING(I). U.T. 1-5.

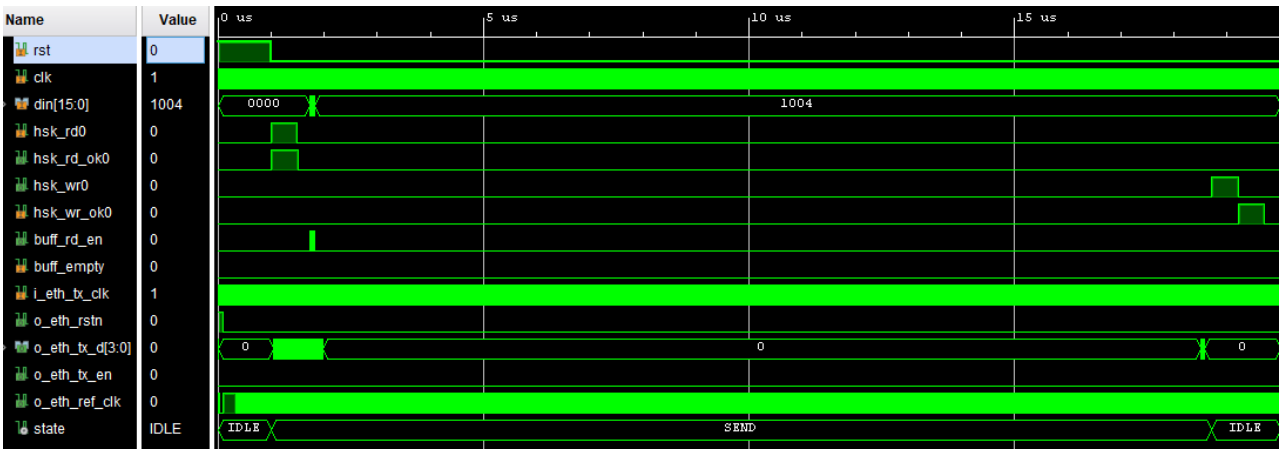


Figure 29: ETHERNET INTERFACE MODULE. UNIT TESTING(I). U.T. 1-5.

UNIT TESTING(II). U.T. 6, TCP/IP PACKET - MAC ADDRESS.

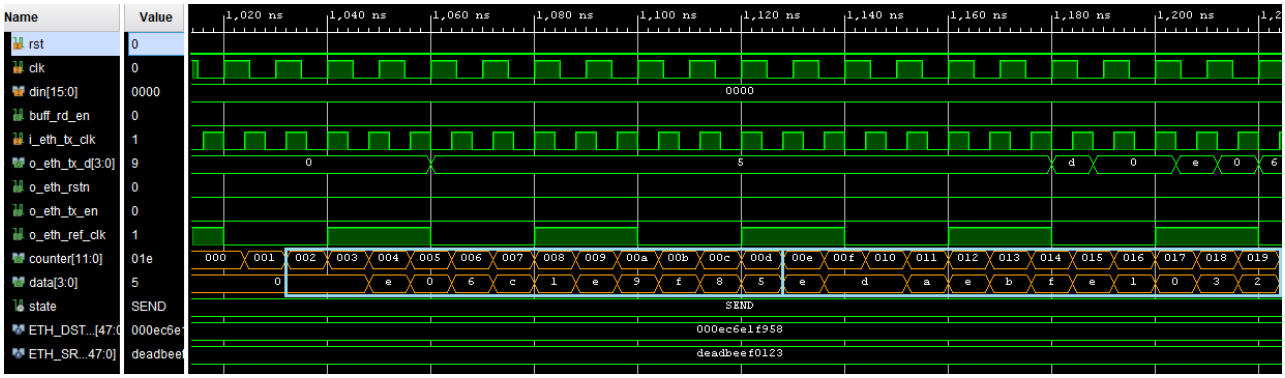


Figure 30: ETHERNET INTERFACE MODULE. UNIT TESTING(II). U.T. 6, TCP/IP PACKET - MAC ADDRESS.

UNIT TESTING(II). U.T. 6, TCP/IP PACKET - IP ADDRESS.

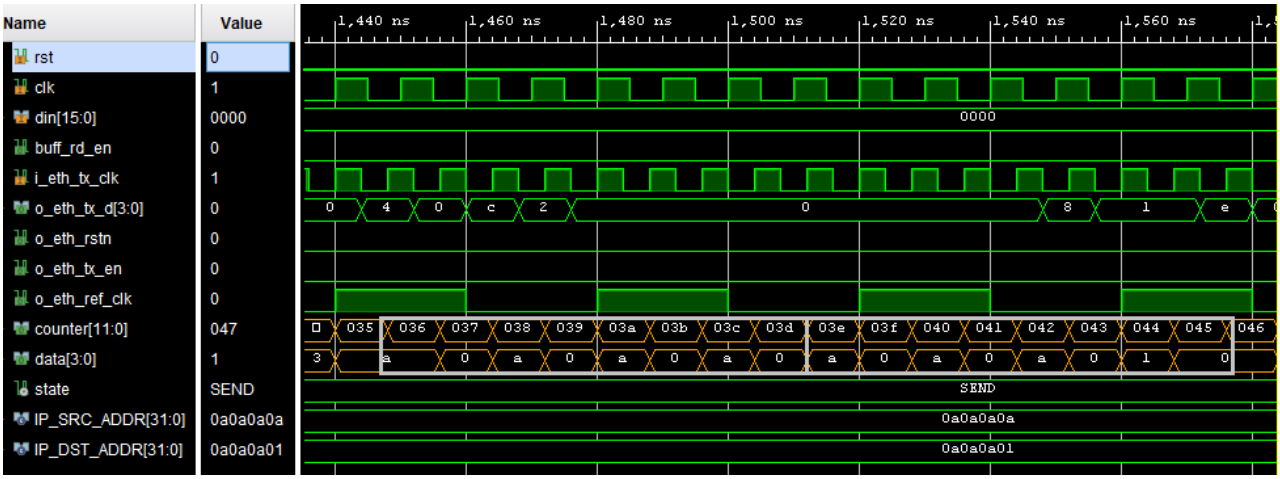


Figure 31: ETHERNET INTERFACE MODULE. UNIT TESTING(II). U.T. 6, TCP/IP PACKET - IP ADDRESS.

UNIT TESTING(II). U.T. 6, TCP/IP PACKET - UDP PORTS.

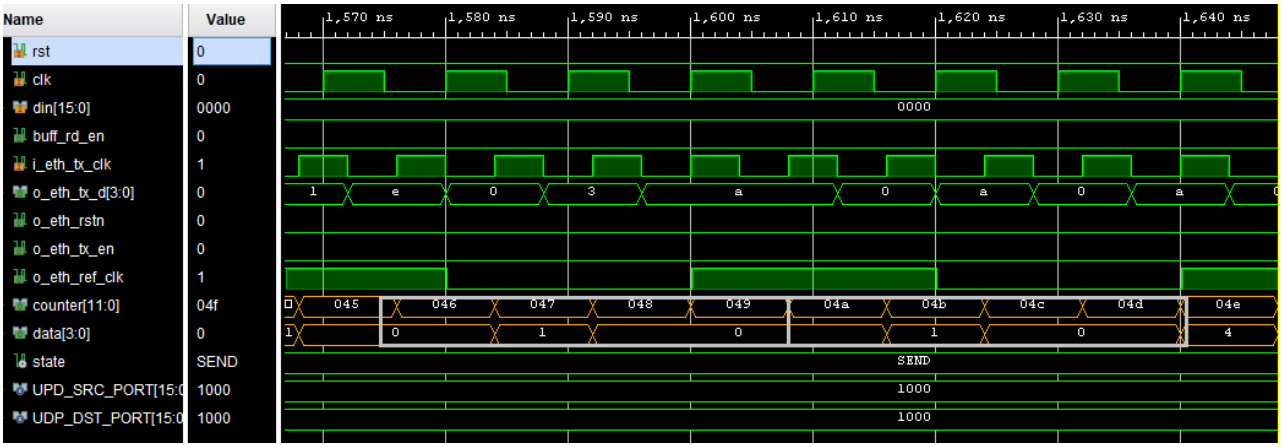


Figure 32: ETHERNET INTERFACE MODULE. UNIT TESTING(II). U.T. 6, TCP/IP PACKET - UDP PORTS.

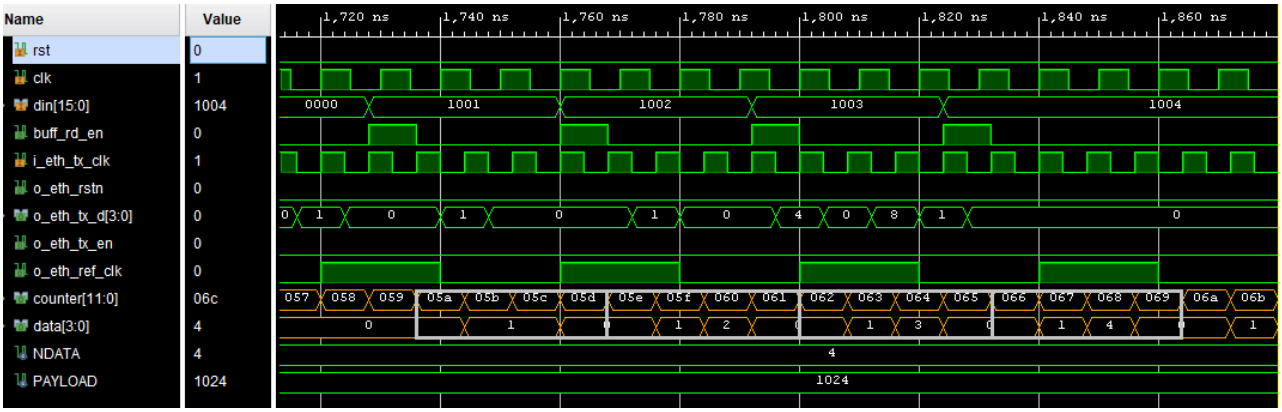


Figure 33: ETHERNET INTERFACE MODULE. UNIT TESTING(II). U.T. 6, TCP/IP PACKET - PAYLOAD.

8 CONCLUSIONS.

This master's dissertation has been an excellent challenge to establish myself as an embedded engineer in the telecommunications field. In addition, I had the opportunity to improve my knowledge of certain technologies, such as radar systems and FPGA; as well as particular tasks, such as problem statement, methodology analysis, and architecture design, among others. This project has laid the foundations of the product engineering necessary to reach, over time, the rank of senior engineer.

8.1 RESULTS.

Once the prototype has been verified, the results obtained must be similar to the following, see tables [34] & [35] . This approximation task is performed by a MATLAB software development, see appendix [F].

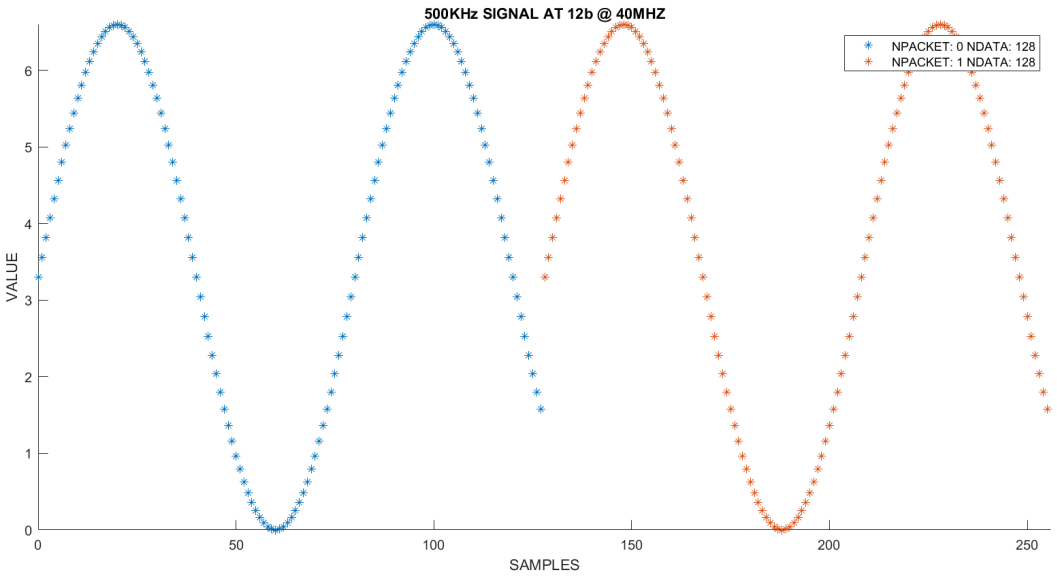


Figure 34: RESULTS(I). 500KHz SIGNAL AT 12b @ 40MHz.

A NDATA samples packet is digitized and processed according to the radar signalling. Obviously, the packet size is determined by the prototype configuration. In standalone (ST) mode, for memory reasons, only a limited number of packets can be acquired.

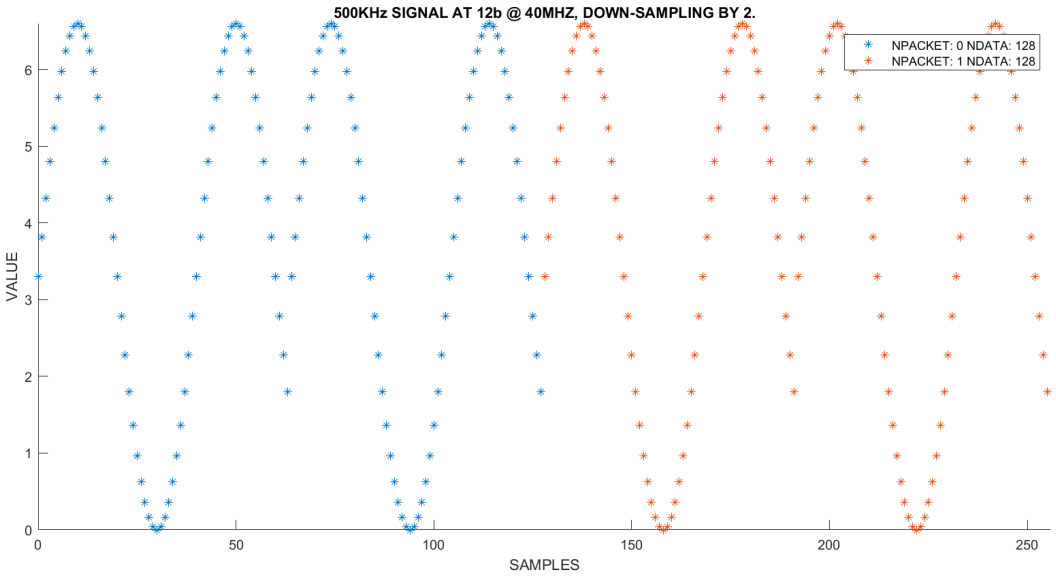


Figure 35: RESULTS(II). 500KHz SIGNAL AT 12b @ 40MHz, DOWN-SAMPLING BY 2.

Similar to the previous table, NDATA samples packets must be digitized and processed according to the radar signalling and decimate rate. In this case, the signal is down-sampled by 2, consequently reducing throughput in half. It is important to remember that the decimate rate must be in accordance with the Nyquist–Shannon sampling theorem in order to avoid aliasing problems. See reference [5].

8.2 REVIEW.

Starting from a poor-defined and non-functional prototype, a tailored hardware solution has been formalized. Moreover, a suitable hardware architecture for the modules that make up the prototype has been successfully designed. Afterwards, the system has been developed according to the software engineering standards. The design guidelines, in accordance with FPGA hardware technologies, to coordinate the hardware/software development cycle. As implementation tools, VHDL and GitHub have been used as a hardware description language and a software repository, respectively, see references [12] & [14]. Engineering problems have been properly documented through flow charts. Finally, the prototype has been correctly verified according to the defined unit testings. Unfortunately, due to the state of emergency caused by COVID-19, it has not been possible to carry out the hardware unit testing needed in order to verify the development kit.

8.2.1 ACHIEVED GOALS.

Hereafter, the main goals achieved through this project are described.

1. The level of development reached in the different technologies that could be used to establish an appropriate solution has been analyzed.
2. A formal proposal for a suitable solution has been established; in it, the restrictions that the prototype designed should fulfill are laid out.

3. The theoretical concepts of the body of methods and principles associated with the project have been established. In addition, the components used for its development were analyzed.

4. A formal description and representation of the hardware and software design have been established, organized in a way that supports the reasoning behind the structure and behaviors of the system. In addition, the system architecture, handshake architecture and architecture of the modules that make up the prototype were analyzed.

5. The design guidelines needed in order to coordinate the hardware and software engineering cycle for the prototype development have been established. In addition, the source-code has been refactored and the designed modules have been properly documented through flow charts.

6. The procedures carried out in order to confirm that the prototype meets the requirements of the intended purpose have been established. To that effect, unit testings have been resorted to in order to verify the designed modules.

7. An user guide document which provides practical information about the use of the unit has been drafted.

8.2.2 PENDING WORK.

Unfortunately, due to the state of emergency caused by COVID-19, it was not possible to have access to the laboratory. Hereafter, the pending tasks needed to finalize the project development are exposed.

1. The verification of the interfaces used for the communication between the FPGA and the memory modules. There is no other option but to monitor the internal signals while the system is running. This is due to the fact that there is no simple way to simulate memory paging. To achieve this, the ChipScope ILA tool must be used. This problem mainly affects the SD and DDR3 memory modules.

2. The modules that make up the development kit must be assembled. Finally, it is necessary to verify that the prototype functions properly according to the user guide.

8.3 RECOMMENDATIONS.

Once the prototype has been developed, it must be presented to the customer for testing. At this point, the client could suggest the necessary modifications in order to better meet their needs. Based on customer comments, a new prototype will be developed, opening the door to the development of a final product.
Personally, I would like to propose the following improvements to the current solution.

- Microblaze: A soft-core processor implemented entirely in the general-purpose memory and logic fabric of Xilinx FPGAs. Powered by the GNU toolchain, as a GNU compiler/debugger collection, the Vivado SDK enables programmers to write, compile and debug C/C++ applications. See reference [21].

- Upgrade Ethernet standard: Using the 1000BASE-T standard, it would be possible to reach rates up to 1000 Mbps. The protocol differs considerably from its predecessor, especially in the number of pair/lines. In order to achieve this improvement, the Ethernet IP controller must be updated. See section [3.3.1].

- Upgrade standard SD: Through the UHS-II standard, it could reach rates up to 50 - 104 MB/s depending on the transfer mode (half/full-duplex). The protocol is similar to that of a High-Speed bus, it makes use of one command channel and four lines as data channel for faster transfer. See sections [3.4.2] and [5.2.2].

- Display: It could considerably improve the user experience. In this regard, a wide variety of solutions are available in the market, see reference [22].
- Product integration: Once prototype improvements have been integrated, the solution must be formalized. At this point, a PCB and a suitable chassis must be developed in order to obtain a final product.

References

[1] Adrià Arroyo de Fàbregas. *Universitat Politècnica de Catalunya.* SISTEMA DE DIGITALIZACIÓN CONTROLADO POR FGPA PARA UN RADAR EMBARCADO EN DRON.

[2] Arturo Martínez Cervera. *Universitat Politècnica de Catalunya.* DISEÑO Y CARACTERIZACIÓN DEL MODO DE ADQUISICIÓN ON THE FLY EN UN SAR DE 94GHZ.

[3] Antonio Artés Rodríguez. *Universidad Carlos III de Madrid.* COMUNICACIONES DIGITALES. http://www.tsc.uc3m.es/ antonio/libro_comunicaciones/El_libro.html

[4] B. Murmann. *Stanford University.* ADC PERFORMANCE SURVEY 1997 - 2019. https://web.stanford.edu/ murmann/adcsurvey.html

[5] *ScienceDirect.* NYQUIST-SHANNON SAMPLING THEOREM. https://www.sciencedirect.com/topics/computer-science/shannon-sampling-theorem

[6] *Analog Devices.* AD9224: 12-BIT 40 MSPS MONOLITHIC A/D CONVERTER. https://www.analog.com/en/products/ad9224.html

[7] *Digilent.* PMOD MODULES AND CONNECTORS. https://store.digilentinc.com/pmod-modules-connectors/

[8] *Digilent.* ARTIX-7 FPGA DEVELOPMENT BOARD. https://reference.digilentinc.com/reference/programmable-logic/arty-a7/start

[9] *Xilinx, Inc.* VIVADO DESIGN HUB - LOGIC SIMULATION. https://www.xilinx.com/support/documentation-navigation/design-hubs/dh0010-vivado-simulation-hub.html

[10] *Xilinx, Inc.* CHIPSCOPE INTEGRATED LOGIC ANALYZER (ILA) REFERENCE GUIDE. https://www.xilinx.com/products/intellectual-property/chipscope_ila.html

[11] *Xilinx, Inc.* MEMORY INTERFACE GENERATOR (MIG) REFERENCE GUIDE. https://www.xilinx.com/products/intellectual-property/mig.html

[12] *Tutorials Point.* VLSI DESIGN TUTORIAL. https://www.tutorialspoint.com/vlsi_design/index.html

[13] *Xilinx, Inc.* TLC COMMANDS REFERENCE GUIDE. https://www.xilinx.com/support/documentation/sw_manuals/xilinx2013_4/ug835-vivado-tcl-commands.pdf

[14] *GitHub, Inc.* GITHUB REPOSITORY REFERENCE GUIDE. https://guides.github.com/

[15] *Tutorials Point.* MATLAB TUTORIAL. https://www.tutorialspoint.com/matlab/index.htm

[16] *Corelis, Inc.* SPI TUTORIAL WHITEPAPER. https://www.corelis.com/whitepapers/request-whitepaper-spi-tutorial/

[17] *Xilinx, Inc.* ADVANCED EXTENSIBLE INTERFACE (AXI) REFERENCE GUIDE. http://www.verien.com/axi-reference-guide.html

[18] Dan Gisselquist. *Gisselquist Technology, LLC.* PIPELINED WISHBONE TO AXI CONVERTER. https://github.com/ZipCPU/wb2axip/blob/master/rtl/migsdram.v

[19] Lawrence Wilkinson. A BASIC SD CARD SPI INTERFACE IN VHDL, SUPPORTS SD V1, V2 AND SDHC. https://github.com/ibm2030/SimpleSDHC

[20] Mike Field. TRANSMIT UDP PACKETS VIA THE ARTY ETHERNET PHY. https://github.com/hamsternz/ArtyEtherentTX

[21] *Xilinx, Inc.* MICROBLAZE SOFT PROCESSOR CORE. https://www.xilinx.com/products/design-tools/microblaze.html

[22] *Adafruit.* LCDS & DISPLAYS. https://www.adafruit.com/category/63

A USER GUIDE.

A.1 OPERATION MANUAL.

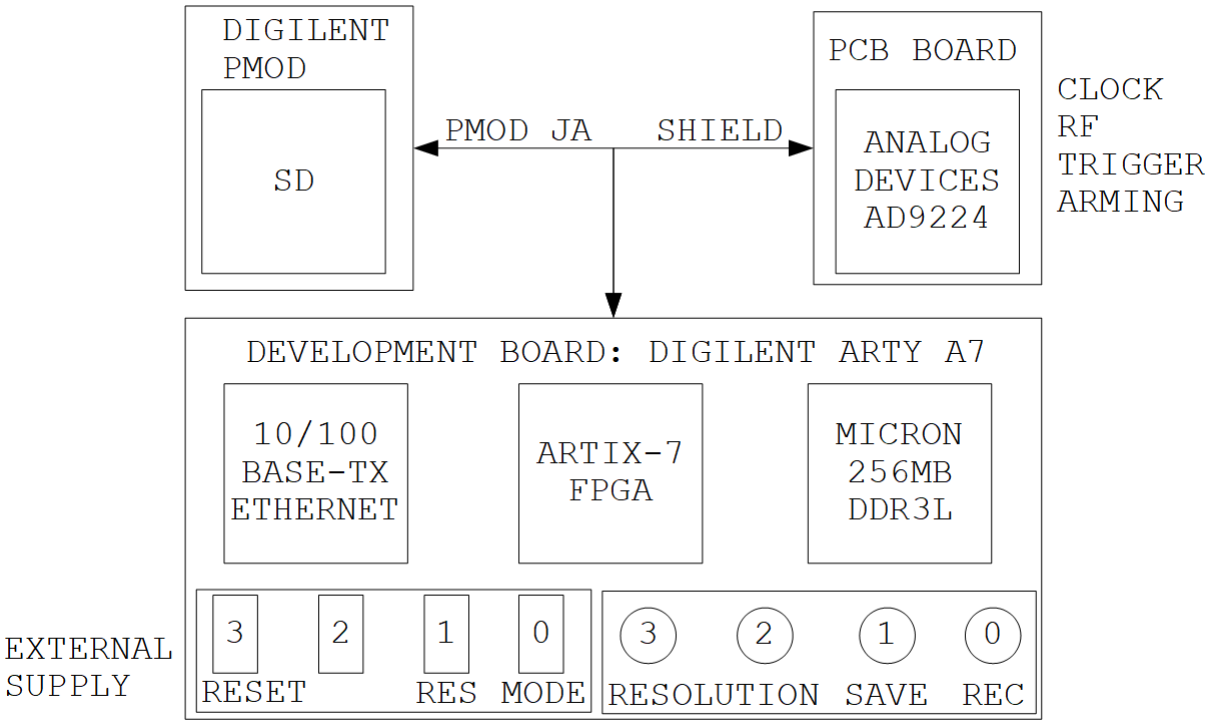


Figure 36: USER GUIDE. OPERATION MANUAL. REFERENCE MODEL.

ID.		ABOUT	
RECORD	BTN0	INITIALIZATION OF THE ACQUISITION SYSTEM.	
SAVE	BTN1	IN STANDALONE MODE, IT DUMPS THE SD MEMORY INTO A COMPUTER THROUGH THE ETHERNET INTERFACE.	
RESOLUTION RATE	BTN2	INCREASE	DECIMATION RATE OF THE DOWN-SAMPLING ALGORITHM.
	BTN3	DECREASE	
MODE	SW0	OFF, PC MODE	DIGITIZE THE RADAR SIGNAL CONTINUOUSLY THROUGH THE ETHERNET INTERFACE.
		ON, STANDALONE	DIGITIZE THE RADAR SIGNAL UNTIL THE SD MEMORY GOES FULL.
RESOLUTION ENABLE	SW1	OFF, DISABLE	SIGNAL PROCESSING ALGORITHM BASED ON DOWN-SAMPLING TO MANAGE THE ACQUISITION RESOLUTION.
		ON, ENABLE	
RESET	SW3	OFF, DISABLE	INITIAL STATE OF THE UNIT.
		ON, ENABLE	

Tabla 24: USER GUIDE. OPERATION MANUAL. MENU MANAGEMENT.

A.2 INSTALLATION MANUAL.

ID.	ABOUT	DEFAULT
NDATA	NUMBER OF SAMPLES PER PACKETS.	4096
NPACKETS	TOTAL NUMBER OF PACKETS.	32768
RES	DECIMATION RATE OF THE DOWN-SAMPLING ALGORITHM.	4
ETH_SRC_MAC	MAC ADDRESS OF THE SENDER OF THE PACKET.	DE:AD:BE:EF:01:23
ETH_DST_MAC	MAC ADDRESS OF THE RECEIVER OF THE PACKET.	PC MAC ADDRESS
IP_SRC_ADDR	IP ADDRESS OF THE SENDER OF THE PACKET.	10.10.10.10
IP_DST_ADDR	IP ADDRESS OF THE RECEIVER OF THE PACKET.	10.10.10.1
UPD_SRC_PORT	UDP PORT OF THE SENDER OF THE PACKET.	4096
UDP_DST_PORT	UDP PORT OF THE RECEIVER OF THE PACKET.	4096

NDATA

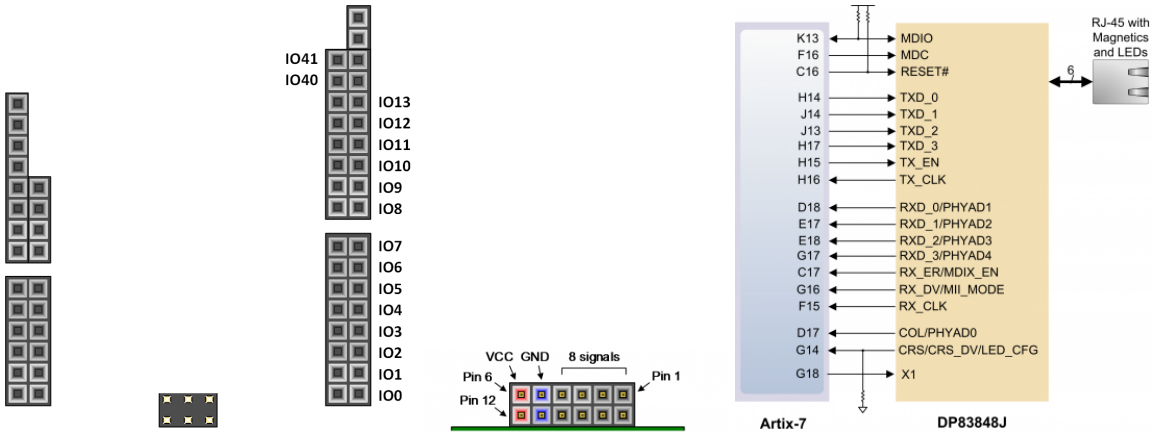
Tabla 25: USER GUIDE. INSTALLATION MANUAL. CONFIGURATION PARAMETERS.

1. Installation. The assembly and connections of the modules are shown in sections [A.1], [5.2.1], [5.2.2] and [5.2.3]. Once the prototype is assembled, carefully check the interconnections. See section [A.3]

2. Configuration. Export the project in Vivado Design Suite, open the main file and modify the configuration parameters as needed. See table [25].

3. Compilation. Compile the project using Vivado Design Suite tools. Finally, dump the binary file into the development board.

A.3 INTERCONNECTIONS.



(a) (b) (c)

Figure 37: USER GUIDE. INTERCONNECTIONS. ARTY A7; (A) SHIELD, (B) PMOD, (C) ETHERNET.

A.3.1 SHIELD AND PMOD.

PORT		PIN	PORT		PIN
SWITCH	SW_MODE	A8/SW0	RADAR A/D	RADAR_IN[3]	R12/IO4
	SW_RESOLUTION	C11/SW1		RADAR_IN[4]	T14/IO5
	RST	A10/SW3		RADAR_IN[5]	T15/IO6
BUTTON	BTN_RECORD	D9/BTN0		RADAR_IN[6]	T16/IO7
	BTN_SAVE	C9/BTN1		RADAR_IN[7]	N15/IO8
	BTN_UP	B9/BTN2		RADAR_IN[8]	M16/IO9
	BTN_DOWN	B8/BTN3		RADAR_IN[9]	V17/IO10
RADAR SIGNALS	RADAR_CLK	F4/JD3		RADAR_IN[10]	U18/IO11
	RADAR_TRIGGER	N17/IO41		RADAR_IN[11]	R17/IO12
	RADAR_ARMING	P18/IO40		RADAR_OVERFLOW	P17/IO13
	CLK_OUT	V15/IO0	SD	O_CS	G13/JA1
RADAR A/D	RADAR_IN[0]	U16/IO1		O_MOSI	B11/JA2
	RADAR_IN[1]	P14/IO2		I_MISO	A11/JA3
	RADAR_IN[2]	T11/IO3		O_CLK	D12/JA4

Tabla 26: USER GUIDE. INTERCONNECTIONS(I). ARTY A7; SHIELD(IO) AND PMOD(JA, JD).

A.3.2 ETHERNET AND DDR CONNECTIONS.

PORT	PIN	PORT	PIN	PORT	PIN	PORT	PIN
I_ETH_COL	D17	O_DDR_RESET_N	K6	O_DDR_ADDR[4]	N6	IO_DDR_DATA[0]	K5
I_ETH_CRS	G14	O_DDR_RAS_N	P3	O_DDR_ADDR[5]	R7	IO_DDR_DATA[1]	L3
I_ETH_RX_CLK	F15	O_DDR_CAS_N	M4	O_DDR_ADDR[6]	V6	IO_DDR_DATA[2]	K3
I_ETH_RX_DV	G16	O_DDR_WE_ N	P5	O_DDR_ADDR[7]	U7	IO_DDR_DATA[3]	L6
I_ETH_RX_D[0]	D18	O_DDR_CK_P[0]	U9	O_DDR_ADDR[8]	R8	IO_DDR_DATA[4]	M3
I_ETH_RX_D[1]	E17	O_DDR_CK_N[0]	V9	O_DDR_ADDR[9]	V7	IO_DDR_DATA[5]	M1
I_ETH_RX_D[2]	E18	O_DDR_CKE[0]	N5	O_DDR_ADDR[10]	R6	IO_DDR_DATA[6]	L4
I_ETH_RX_D[3]	G17	O_DDR_CS_N[0]	U8	O_DDR_ADDR[11]	U6	IO_DDR_DATA[7]	M2
I_ETH_RX_ERR	C17	O_DDR_ODT[0]	R5	O_DDR_ADDR[12]	T6	IO_DDR_DATA[8]	V4
I_ETH_TX_CLK	H16	O_DDR_BA[0]	R1	O_DDR_ADDR[13]	T8	IO_DDR_DATA[9]	T5
O_ETH_MDC	F16	O_DDR_BA[1]	P4	O_DDR_DM[0]	L1	IO_DDR_DATA[10]	U4
O_ETH_REF_CLK	G18	O_DDR_BA[2]	P2	O_DDR_DM[1]	U1	IO_DDR_DATA[11]	V5
O_ETH_RSTN	C16	O_DDR_ADDR[0]	R2	IO_DDR_DQS_P[0]	N2	IO_DDR_DATA[12]	V1
O_ETH_TX_EN	H15	O_DDR_ADDR[1]	M6	IO_DDR_DQS_P[1]	U2	IO_DDR_DATA[13]	T3
O_ETH_TX[0]	H14	O_DDR_ADDR[2]	N4	IO_DDR_DQS_N[0]	N1	IO_DDR_DATA[14]	U3
O_ETH_TX[1]	J14	O_DDR_ADDR[3]	T1	IO_DDR_DQS_N[1]	V2	IO_DDR_DATA[15]	R3
O_ETH_TX[2]	J13						
O_ETH_TX[3]	H17						
IO_ETH_MDIO	K13						

Tabla 27: USER GUIDE. INTERCONNECTIONS(II). ARTY A7; ETHERNET AND DRAM.

B TECHNICAL REQUIREMENTS.

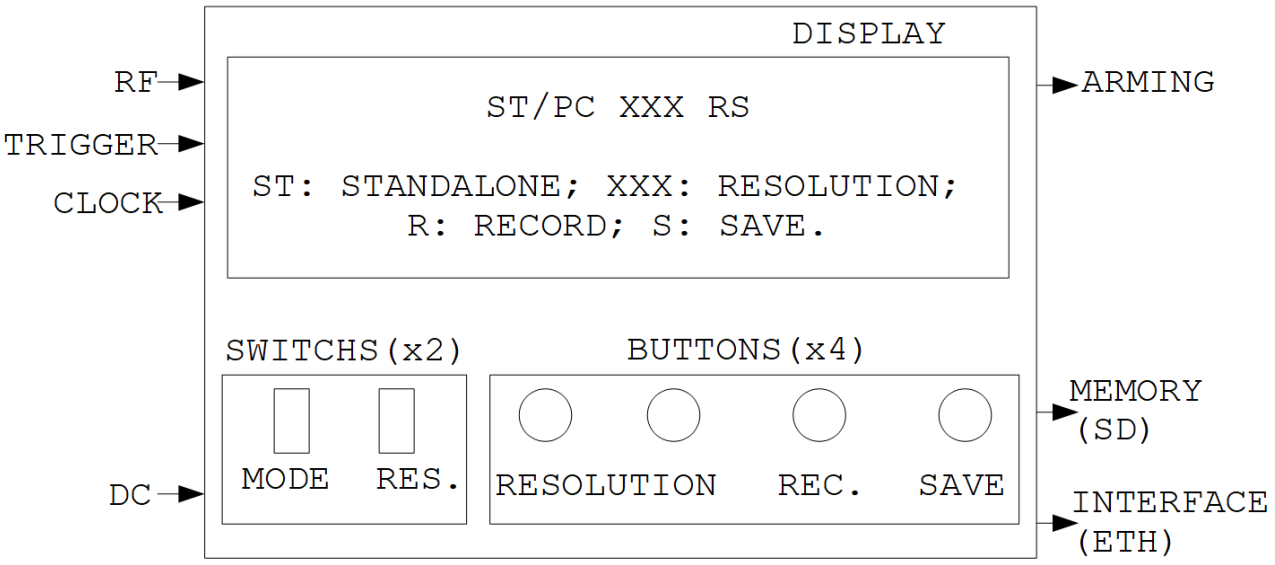


Figure 38: TECHNICAL REQUIREMENTS. SYSTEM DESCRIPTION.

ID.	DESCRIPTION	ABOUT
MECH.1	INPUTS: DC, RF, TRIGGER, CLOCK, (x4)BUTTONS & (x2)SWITCHES.	REQUIRED
MECH.2	OUTPUTS: ARMING, MEMORY, INTERFACE & DISPLAY(DESIRABLE).	REQUIRED
MECH.3	LIGHT CHASSIS RESISTANT TO ATMOSPHERIC CONDITIONS.	DESIRABLE
HW.1	DC: 9V STANDARD PLUG.	DESIRABLE
HW.2	RF: A/D CONVERTER AT LEAST 12b AT 40MHz.	REQUIRED
HW.3	RADAR SIGNALLING: TRIGGER, CLOCK, & ARMING.	REQUIRED
HW.4	BUTTONS(x4): (x2)RESOLUTION ,(x1)RECORD & (x1)SAVE.	REQUIRED
HW.5	SWITCHES(x2): PC/STANDALONE(ST) MODE & RESOLUTION ON/OFF.	REQUIRED
HW.6	DISPLAY: ST/PC MODE, RESOLUTION AND (R)RECORD/(S)SAVE INDICATIONS. E.G. ST 10 RS.	DESIRABLE
HW.7	MEMORY: SECURE DIGITAL (SD).	REQUIRED
HW.8	INTERFACE: ETHERNET (ETH).	REQUIRED
SW.1	STANDALONE(ST) MODE: ACQUISITION UNTIL MEMORY(SD) GOES FULL.	REQUIRED
SW.2	PC MODE: ACQUISITION OVER INTERFACE(ETH).NOTE: MEMORY(SD) IS NOT NEEDED.	REQUIRED
SW.3	RESOLUTION: SIGNAL PROCESSING ALGORITHM BASED ON DOWN-SAMPLING.	REQUIRED
SW.4	RECORD(R): IT INITIALIZES THE ACQUISITION SYSTEM.	REQUIRED
SW.5	SAVE(S): IT DUMPS MEMORY(SD) DATA INTO THE INTERFACE(ETH). NOTE: ONLY AVAILABLE IN STANDALONE MODE.	REQUIRED
SW.6	MENU FOR THE MANAGEMENT OF THE SYSTEM FUNCTIONALITIES USING (x4)BUTTONS, (x2)SWITCHES AND DISPLAY(DESIRABLE).	DESIRABLE

Tabla 28: TECHNICAL REQUIREMENTS. REQUIREMENTS DESCRIPTION.

C FREQUENCY-MODULATED CONTINUOUS-WAVE RADAR.

FMCW radar is a special type of radar sensor which radiates continuous transmission power, like a simple continuous wave (CW) radar. In contrast, FMCW radar can change its operating frequency during the measurement: i.e., the transmission signal is modulated in frequency or in phase. Possibilities of radar measurements through run-time measurements are only technically possible through these changes in frequency or phase. Simple CW radars without frequency modulation have the disadvantage that they cannot determine target range because they lack the timing mark necessary to allow the system to accurately time the transmit and receive cycle and to convert this into range. Such a time reference for measuring the distance of stationary objects, but can be generated through the frequency modulation of the transmitted signal.

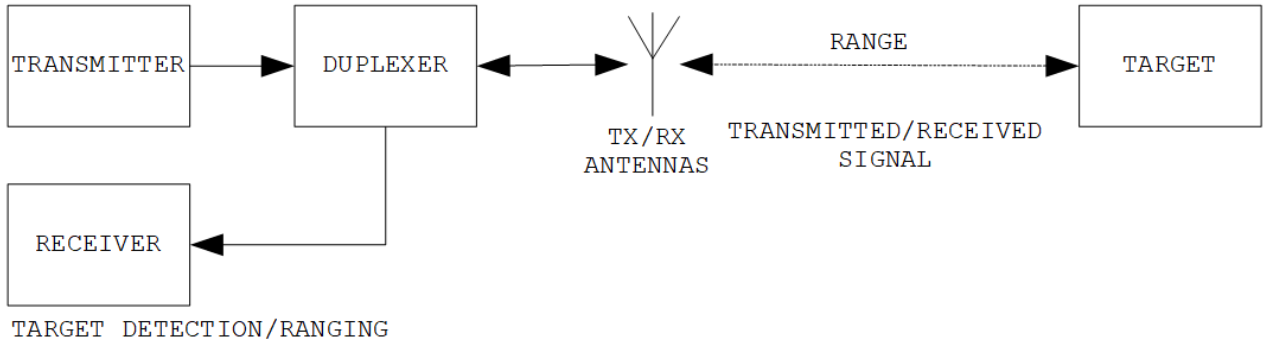


Figure 39: FMCW RADAR. REFERENCE MODEL.

In a FMCW radar, a signal is transmitted, which increases or decreases in frequency periodically. When an echo signal is received, that change or frequency gains a delay like the one obtained using the pulse radar technique. In pulse radar, however, the run-time must be measured directly. In FMCW radars, the differences in frequency or phase between the actually transmitted and the received signed are measured instead.

The main characteristics of a FMCW radar are:

- The distance measurement is accomplished through the comparison of the frequency of the received signal and a reference, usually the transmission signal directly.
- The duration of the transmitted waveform T is substantially higher than the required receiving time for the installed distance measuring range.

The basic features of a FMCW radar are:

- Ability to measure a very small range to the target with very high accuracy.
- Ability to measure simultaneously the target range and its relative velocity.
- The mixing is performed at a low frequency range, simplifying the development of the signal processing algorithms.

D ENGINEERING FUNDAMENTALS.

D.1 DOWN-SAMPLING.

Down-sampling, also known as decimation, is the process of re-sampling a digital signal in order to reduce its bandwidth and sample-rate.

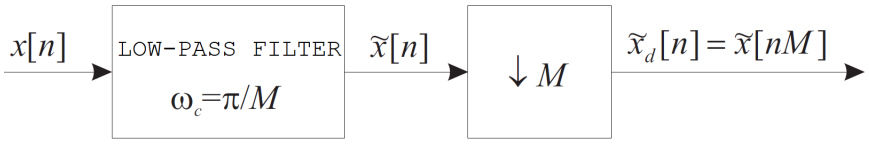


Figure 40: DOWN-SAMPLING PROCESSING. DECIMATION BY M.

A system that performs a down-sampling process for a generic sequence $x[n]$ is shown in figure [40]. Aliasing is avoided using a low-pass filter with π/M cutoff frequency before down-sampling process, labelled as $\downarrow M$. If $x[n]$ is a low-pass signal with π/M cutoff frequency (at least), $\tilde{x}[n]$ will be equal to $x[n]$.

D.2 ADVANCED EXTENSIBLE INTERFACE (AXI).

The AXI specifications describe an interface between a single AXI master and a single AXI slave, representing IP cores that exchange information between each other. Memory mapped AXI masters and slaves can be connected together using a structure called Interconnect block. The Xilinx AXI Interconnect IP contains an AXI-compliant master and slave interfaces, and can be used to route transactions between one or more AXI masters and slaves. Data can move in both directions between the master and the slave simultaneously, and data transfer sizes can vary. The limit in AXI4 is a burst transaction of up to 256 data transfers. See reference [17].

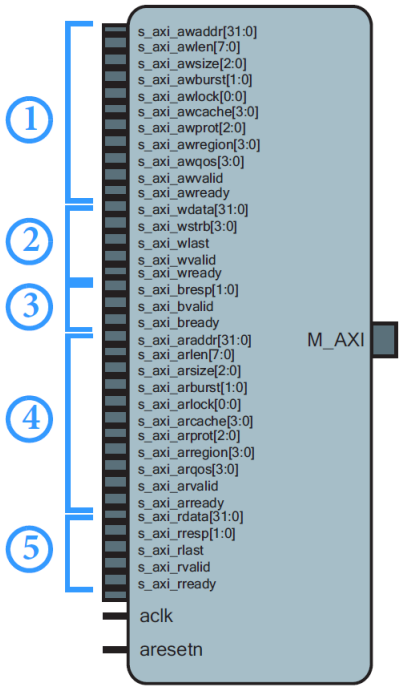


Figure 41: ADVANCED EXTENSIBLE INTERFACE (AXI). REPRESENTATIVE AXI4 DATA FIFO.

The AXI4 interface consists of five different channels: Write Address Channel(1), Write Data Channel(2), Write Response Channel(3), Read Address Channel(4) and Read Data Channel(5).

D.2.1 WRITE TRANSACTION.

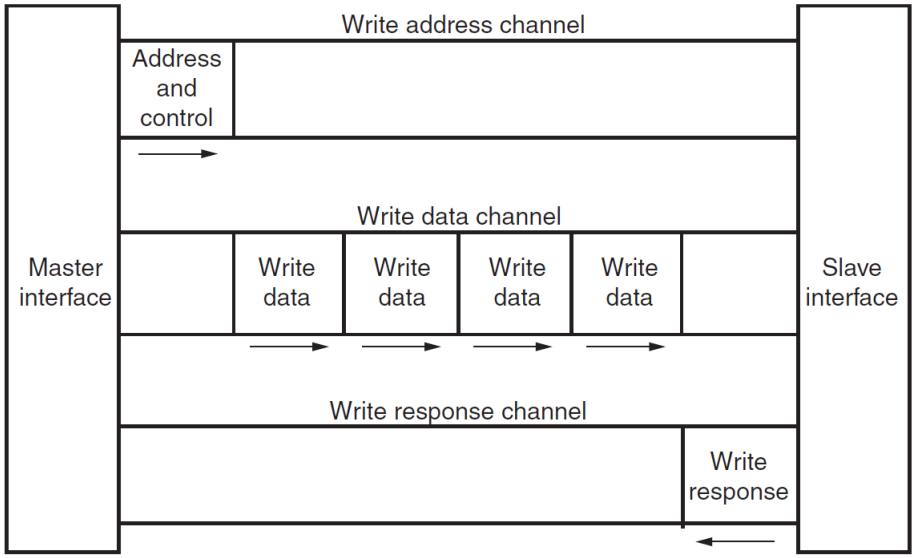


Figure 42: ADVANCED EXTENSIBLE INTERFACE (AXI). CHANNEL ARCHITECTURE OF WRITES.

Figure [42] shows how a write transaction uses the Write Address, the Write Data, and the Write Response Channels.

D.2.2 READ TRANSACTION.

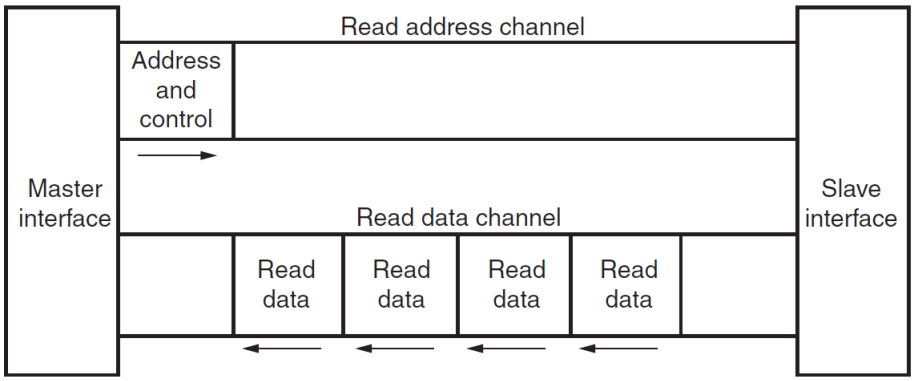


Figure 43: ADVANCED EXTENSIBLE INTERFACE (AXI). CHANNEL ARCHITECTURE OF READS.

Figure [43] shows how an AXI4 Read transaction uses the Read Address and the Read Data Channels.

AXI4 provides separate data and address connections for reads and writes, which allows simultaneous, bidirectional data transfer. AXI4 requires a single address and then bursts up to 256 words of data. The AXI4 protocol describes a variety of options that allow AXI4-compliant systems to achieve very high data throughput. Some of these features, in addition to bursting, are: data upsizing and downsizing, multiple outstanding addresses, and

out-of-order transaction processing. At a hardware level, AXI4 allows a different clock for each AXI master-slave pair. In addition, the AXI protocol enables the insertion of register slices (often called pipeline stages) to aid in timing closure.

D.3 SERIAL PERIPHERAL INTERFACE (SPI).

SPI is a full-duplex synchronous serial interface used by embedded systems for communicating with one or more peripheral devices quickly over short distances. See reference [16].

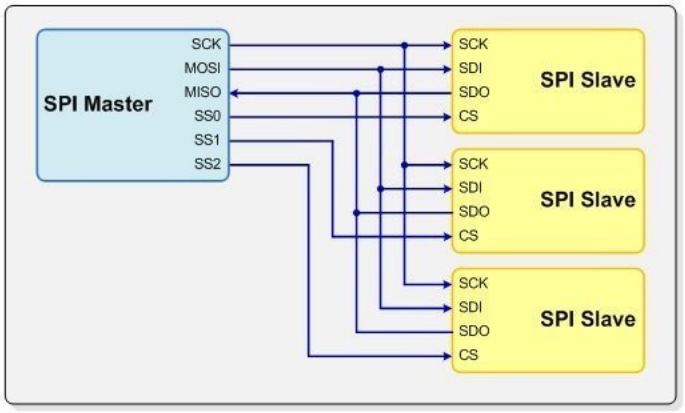


Figure 44: SERIAL PERIPHERAL INTERFACE (SPI). 4-WIRE SPI BUS CONFIGURATION WITH MULTIPLE SLAVES.

A standard SPI connection involves a master connected to slaves using the serial clock (SCK), Master Out Slave In (MOSI), Master In Slave Out (MISO), and Slave Select (SS) lines. The SCK, MOSI, and MISO signals can be shared by slaves while each slave has a unique SS line.

D.3.1 WRITE TRANSACTION.

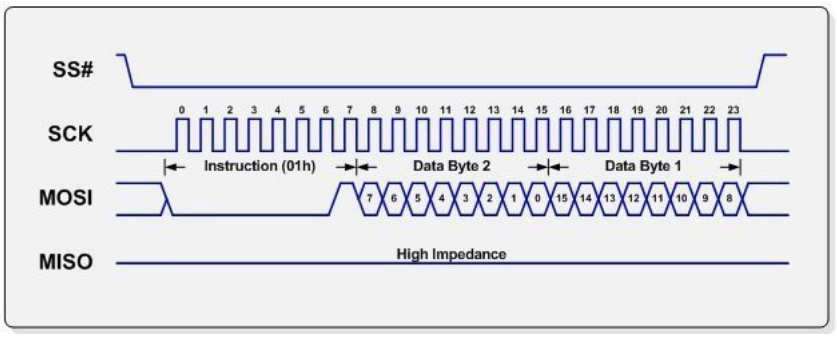


Figure 45: SERIAL PERIPHERAL INTERFACE (SPI). WRITE COMMAND USING SINGLE-BYTE INSTRUCTIONS AND A TWO-BYTE DATA WORD.

Most SPI flash memories include a write status register command that writes one or two byte of data. To write to the status register, the SPI host first enables the slave select (SS) line for the current dive. The master then outputs the appropriate instruction followed by two data byte that define the intended status register contents. Since the transactions do not need to return any data, the slave device keeps the MISO line in a high impedance state and the master masks any incoming data. Finally, the slave select (SS) line is de-asserted to complete the transaction.

D.3.2 READ TRANSACTION.

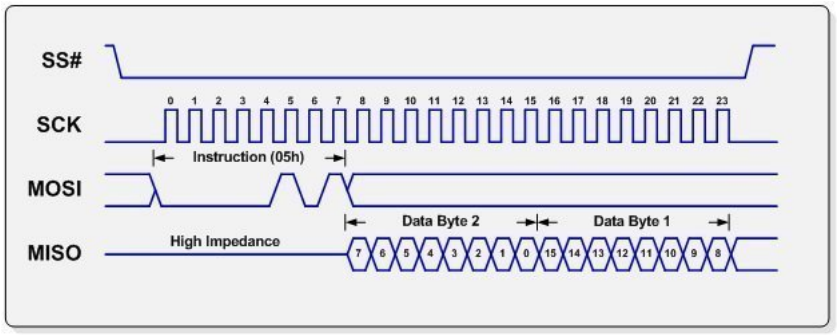


Figure 46: SERIAL PERIPHERAL INTERFACE (SPI). READ COMMAND USING SINGLE-BYTE INSTRUCTIONS AND A TWO-BYTE DATA WORD.

A status register read transaction would be similar to the write transaction, but it now takes advantage of data returned from the slave. After sending the read status register instruction, the slave begins transmitting data on the MISO line at a rate of one byte per eight clock cycles. The host receives the bitstream and completes the transaction by de-asserting the slave select (SS) line.

D.4 100BASE-TX STANDARD.

Fast Ethernet physical layers carry traffic at a rate of 100 Mbps, while the speed of prior standards was 10 Mbps. Out of the Fast Ethernet physical layers, 100BASE-TX standard is by far the most common.

NAME	SPEED	PAIRS	LINES/ DIRECTION	BITS/Hz	LINE CODE	SYMBOL RATE	BW	DISTANCE	CABLE
10BASE-TX	10 Mbps	2	1	1	PE	10 MBd	10MHz	100m	CAT. 3
100BASE-TX	100 Mbps			3.2	4B5B MLT-3 NRZ-I	125 MBd	31.25 MHz		CAT. 5 (5e)

Tabla 29: FAST ETHERNET. 10/100BASE-TX STANDARD.

Some of the specified characteristics are attenuation, impedance, jitter, delay, noise and cross-talk. These characteristics are expected to be met by 100 meters of unshielded twisted-pair cable. However, high quality cable can reach 150 meters or longer. A 10BASE-TX standard uses two differential voltages $(+2.5V, -2.5V)$, while 100BASE-TX uses three differential voltages $(+1V, 0V, -1V)$. The 10/100BASE-TX share the same wiring patterns, yet the wire qualities are more restrictive in 100BASE-TX due to the higher bit rates.

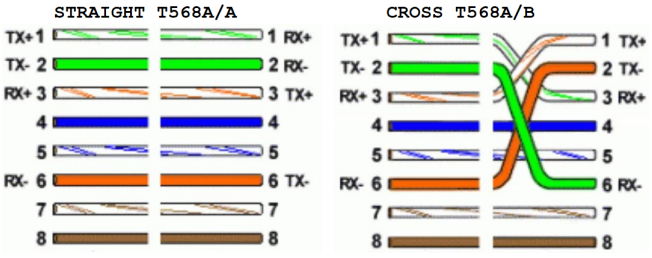


Figure 47: FAST ETHERNET. T568A/B STANDARD.

D.4.1 T5681A/B.

It is the most used termination in Ethernet cables. This standard differs in the cross pairs used for transmitting and receiving, a cable with a T568A/B pair at its respective terminations results in a crossover cable.

D.4.2 MEDIUM DEPENDENT INTERFACE (MDI).

A host uses MDI wiring, transmitting on pins 1/2 and receiving on pins 3/6 to a network device. Up-link ports, e.g.: routers, servers and hosts. Accordingly, an infrastructure node uses a connection wiring called MDI-X, transmitting on pins 3/6 and receiving on pins 1/2. Regular ports, e.g.: hubs and switches. When two nodes having the same type of ports need to be connected, a crossover cable may be required, especially for old equipment. Connecting nodes presenting different type of ports require straight-through cable. Modern host adapters can automatically detect another computer connected with straight-through and then automatically introduce the required crossover, if needed.

D.5 TCP/IP MODEL.

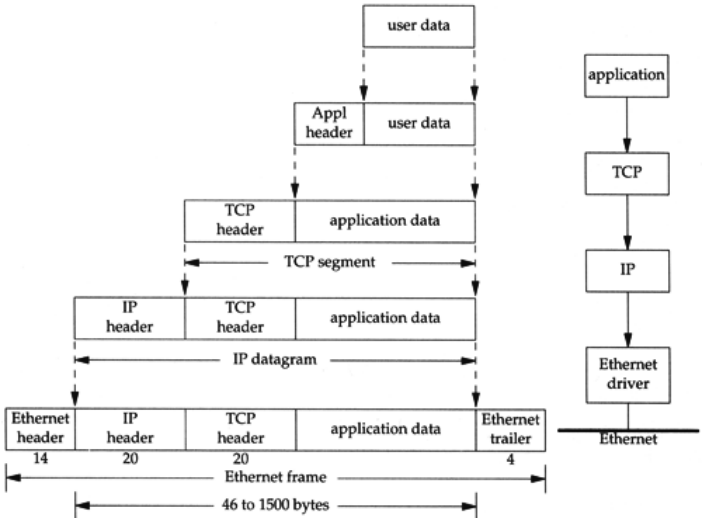


Figure 48: TCP/IP PROTOCOL STACK(I). REFERENCE MODEL.

The TCP/IP protocol stack is the heart of the Internet, it contains four layers as described below.

- **Application**: This layer is responsible for host-to-host communication and controls user-interface specifications.
- **Transport**: This layer is responsible for end-to-end communication and error-free delivery of data. It shields the upper-layer applications from the complexities of data.
- **Internet**: This layer defines the protocols which are responsible for logical transmission of data over the entire network.
- **Network access**: This layer looks out for hardware addressing, the protocols present in this layer allow the physical transmission of data.

D.5.1 TRANSPORT: USER DATAGRAM PROTOCOL (UDP).

UDP is an Internet protocol used for message exchange between hosts in a computer network which is formally defined in RFC 768. It uses a simple connection-less communication model with a minimum of protocol mechanisms. UDP provides port numbers for addressing different functionalities at the source and destination sides. It does not require a three-way handshake; consequently, there is no guarantee of delivery, ordering or duplicate protection.

OCTET	0	1	2	3
0	SOURCE PORT		DESTINATION PORT	
4	LENGTH		CHECKSUM	

Tabla 30: TCP/IP PROTOCOL STACK(II). UDP HEADER FORMAT.

- **Source port**: This field is the UDP port of the sender of the packet. If the source is a server, the port should be a well-known number.
- **Destination port**: This field is the UDP port of the receiver of the packet. It is similar to the source port.
- **Length**: This field defines the entire packet size in bytes, including header and data. The minimum size is 8 bytes (header without data) and the maximum is 65535 bytes.
- **Checksum**: This field contains the error-checking code of the header and data.

D.5.2 NETWORK: INTERNET PROTOCOL (IP).

IP is an Internet protocol for relaying packets across boundaries, which is formally defined in RFC 791. Its routing function enables the packet delivery solely based on the IP addressing.

OCTET	0	1	2	3
0	VERSION(4), IHL(4)	DSCP(6), ECN(2)	TOTAL LENGTH	
4	IDENTIFICATION		FLAGS(3), FRAGMENT OFFSET(15)	
8	TIME TO LIVE	PROTOCOL	HEADER CHECKSUM	
12	SOURCE IP ADDRESS			
16	DESTINATION IP ADDRESS			
20	OPTIONS (IF IHL >5)			
24				
28				
32				

Tabla 31: TCP/IP PROTOCOL STACK(III). IP HEADER FORMAT.

- **Total length**: This field defines the entire packet size in bytes, including header and data. The minimum size is 20 bytes (header without data) and the maximum is 65535 bytes.
- **Header checksum**: This field contains the error-checking code of the header.
- **Source address**: This field is the IP address of the sender of the packet. Note that this address may be changed in transit by a network address translation (NAT) device.
- **Destination address**: This field is the IP address of the receiver of the packet. As in the case of the source address, this one may be changed in transit by a network address translation (NAT) device.

D.5.3 NETWORK ACCESS: ETHERNET.

Ethernet is an Internet protocol that acts as a data communication network connecting various hosts within a limited area, which is formally defined in IEEE 802.3.

PREAMBLE	SFD	DESTINATION ADDRESS	SOURCE ADDRESS	LENGTH	DATA	CRC
7 BYTES	1 BYTE	6 BYTES	6 BYTES	2 BYTES	46-1500 BYTES	4 BYTES

Tabla 32: TCP/IP PROTOCOL STACK(IV). ETHERNET HEADER FORMAT.

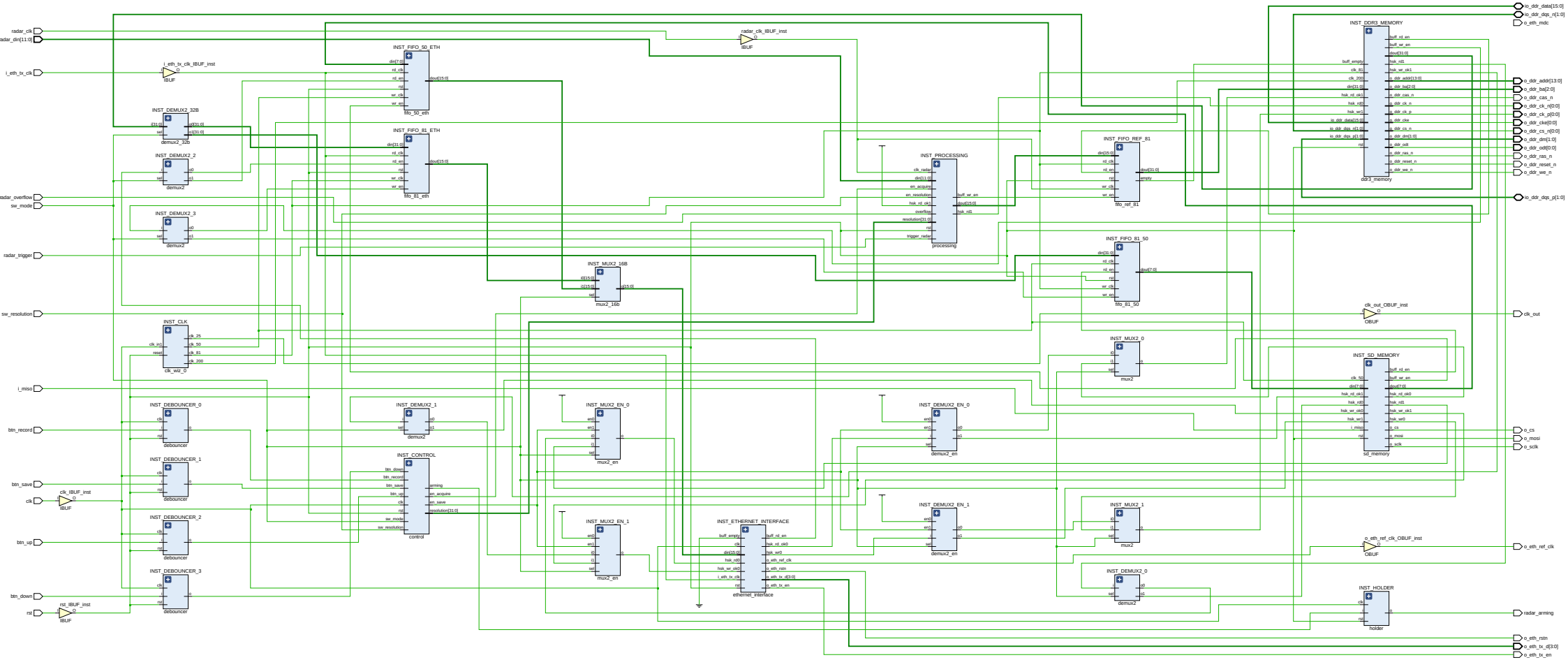
- **Preamble**: This field indicates the start of the frame and allows sender and receiver to establish bit synchronization.
- **Start of frame delimiter (SFD)**: This field indicates that upcoming bits are starting the frame, which is the destination address.
- **Destination Address**: This field refers to the MAC address of the sender of the packet.
- Source Address: This field refers to the MAC address of the receiver of the packet.
- **Data**: This is the place where data are inserted, also known as payload. The minimum size is 46 bytes and the maximum is 1500 bytes.
- **Cyclic Redundancy Check (CRC)**: This field contains the error-checking code of the header (only address and length) and data.

E DEVELOPMENT AND IMPLEMENTATION.

E.1 MAIN MODULE.

ID		**ABOUT**	
	RST	IN	GLOBAL RESET SYSTEM.
	CLK	IN	GENERAL CLOCK.
	SWITCHES	IN	MENU MANAGEMENT.
	BUTTONS	IN	MENU MANAGEMENT.
	RADAR	IN/OUT	RADAR SYSTEM MANAGEMENT.
	DDR3_MEMORY	IN/OUT	DDR3 MEMORY INTERFACE.
	SD_MEMORY	IN/OUT	SD MEMORY INTERFACE.
	ETH_INTERFACE	IN/OUT	ETHERNET INTERFACE
CONF.	NDATA	NUMBER OF SAMPLES PER PACKETS.	
	NPACKETS	TOTAL NUMBER OF PACKETS.	
	RES	DECIMATION RATE OF THE DOWN-SAMPLING ALGORITHM.	
	ETH_SRC_MAC	MAC ADDRESS OF THE SENDER OF THE PACKET.	
	ETH_DST_MAC	MAC ADDRESS OF THE RECEIVER OF THE PACKET.	
	IP_SRC_ADDR	IP ADDRESS OF THE SENDER OF THE PACKET.	
	IP_DST_ADDR	IP ADDRESS OF THE RECEIVER OF THE PACKET.	
	UPD_SRC_PORT	UDP PORT OF THE SENDER OF THE PACKET.	
	UDP_DST_PORT	UDP PORT OF THE RECEIVER OF THE PACKET.	

Tabla 33: VHDL DESIGN(I). MAIN MODULE.


radar_clk
radar_din[11:0]
i_eth_tx_clk
radar_overflow
sw_mode
radar_trigger
sw_resolution
i_miso
btn_record
btn_save
clk
btn_up
btn_down
rst
i_eth_tx_clk_IBUF_inst
IBUF
INST_FIFO_50_ETH
fifo_50_eth
INST_DEMUX2_32B
demux2_32b
INST_FIFO_81_ETH
fifo_81_eth
INST_DEMUX2_2
demux2
INST_DEMUX2_3
INST_CLK
clk_wiz_0
INST_DEBOUNCER_0
debouncer
INST_DEBOUNCER_1
clk_IBUF_inst
INST_DEBOUNCER_2
INST_DEBOUNCER_3
rst_IBUF_inst
INST_DEMUX2_1
INST_CONTROL
control
INST_MUX2_16B
mux2_16b
INST_MUX2_EN_0
mux2_en
INST_MUX2_EN_1
radar_clk_IBUF_inst
INST_PROCESSING
processing
INST_ETHERNET_INTERFACE
ethernet_interface
INST_DEMUX2_EN_0
demux2_en
INST_DEMUX2_EN_1
INST_FIFO_REF_81
fifo_ref_81
INST_FIFO_81_50
fifo_81_50
INST_MUX2_0
INST_MUX2_1
INST_DEMUX2_0
INST_DDR3_MEMORY
ddr3_memory
clk_out_OBUF_inst
OBUF
INST_SD_MEMORY
sd_memory
o_eth_ref_clk_OBUF_inst
INST_HOLDER
holder
io_ddr_data[15:0]
io_ddr_dqs_n[1:0]
o_eth_mdc
o_ddr_addr[13:0]
o_ddr_ba[2:0]
o_ddr_cas_n
o_ddr_ck_n[0:0]
o_ddr_ck_p[0:0]
o_ddr_cke[0:0]
o_ddr_cs_n[0:0]
o_ddr_dm[1:0]
o_ddr_odt[0:0]
o_ddr_ras_n
o_ddr_reset_n
o_ddr_we_n
io_ddr_dqs_p[1:0]
clk_out
o_cs
o_mosi
o_sclk
o_eth_ref_clk
radar_arming
o_eth_rstn
o_eth_tx_d[3:0]
o_eth_tx_en

E.2 CONTROL MODULE.

ID		ABOUT	
	RST	IN	GLOBAL RESET SYSTEM.
	CLK	IN	GENERAL CLOCK.
	SW_MODE	IN	SELECT PC/STANDALONE(ST) MODE.
	SW_RESOLUTION	IN	SELECT RESOLUTION ON/OFF. SIGNAL PRE-PROCESSING BASED ON SAMPLE WEIGHTING.
	BTN_RECORD	IN	INITIALIZE ACQUISITION
	BTN_SAVE	IN	DUMPS MEMORY(SD) DATA INTO THE INTERFACE. NOTE: ONLY AVAILABLE IN STANDALONE(ST) MODE.
	BTN_UP	IN	RESOLUTION HIGHER WEIGHTED SAMPLING.
	BTN_DOWN	IN	RESOLUTION LOWER WEIGHTED SAMPLING.
	EN_ACQUIRE	OUT	ENABLE ACQUISITION SUBSYSTEM.
	EN_SAVE	OUT	ENABLE SAVE SUBSYSTEM.
	RESOLUTION	OUT	RESOLUTION VALUE.
	ARMING	OUT	RADAR RESET/ENABLE.
CONF.	RES	DEFAULT RESOLUTION VALUE.	
STATE	FSM	RESET, ARM, REC, SAVE, UP & DOWN.	

Tabla 34: VHDL DESIGN(II). CONTROL MODULE.

TCL COMMANDS. UNIT TESTING(I).

```
add_force rst {1} {0 100ns} {1 500ns} {0 600ns}
add_force clk {0 0ns} {1 5000ps} -repeat_every 10000ps
add_force btn_record {0} {1 100ns} {0 200ns}
```

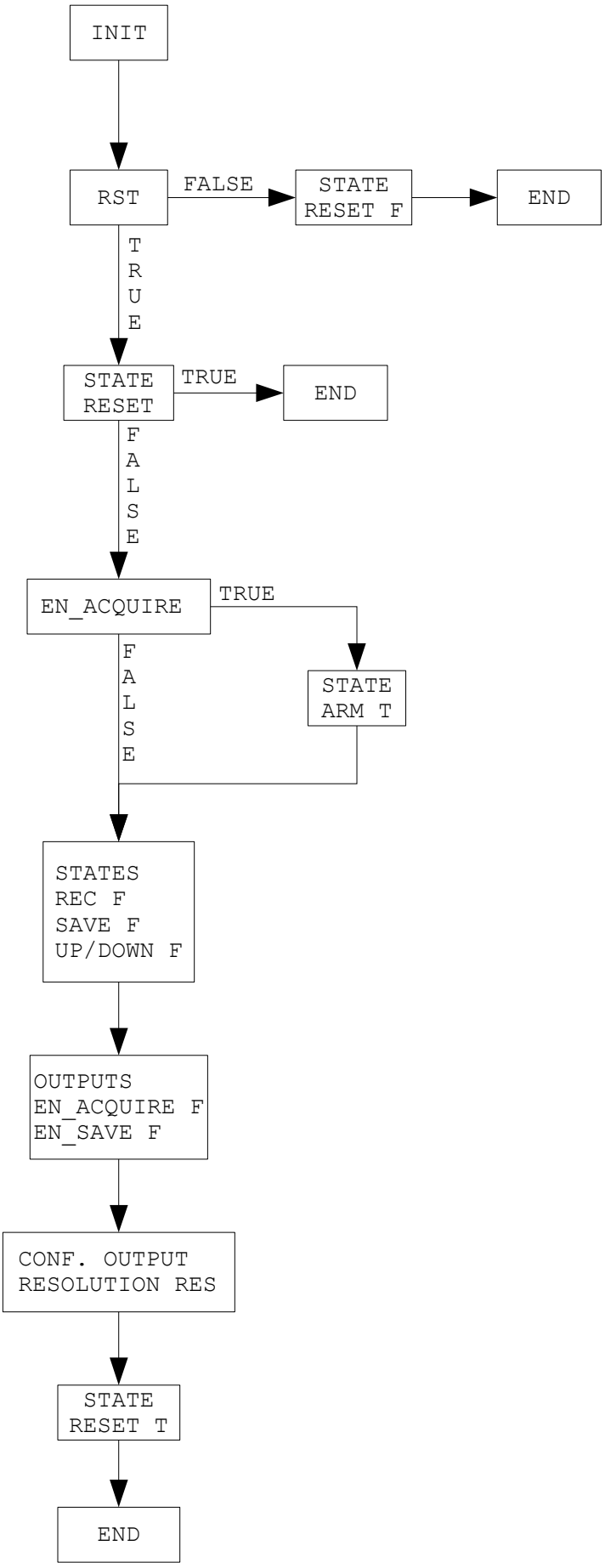
TCL COMMANDS. UNIT TESTING(II).

```
add_force rst {1} {0 50ns }
add_force clk {0 0ns} {1 5000ps} -repeat_every 10000ps
add_force sw_mode {0} {1 500ns }
add_force btn_save {0} {1 250ns} {0 350ns} {1 550ns} {0 650ns}
add_force btn_record {0} {1 50ns} {0 150ns}
```

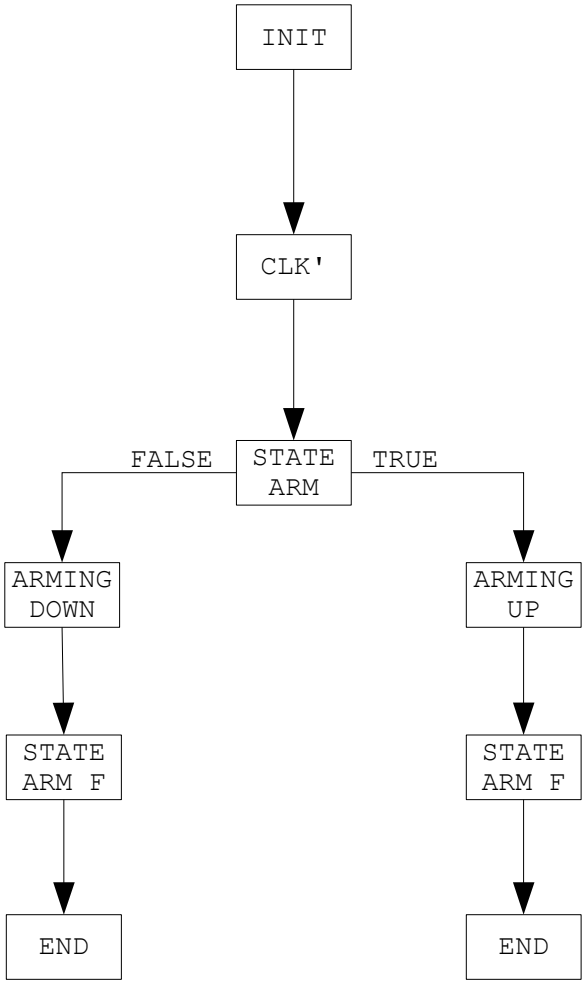
TCL COMMANDS. UNIT TESTING(III).

```
add_force rst {1} {0 50ns }
add_force clk {0 0ns} {1 5000ps} -repeat_every 10000ps
add_force sw_resolution {0} {1 500ns }
add_force btn_record {0} {1 50ns } {0 150ns}
add_force btn_up {0} {1 250ns} {0 350ns} {1 550ns} {0 650ns} {1 800ns} {0 900ns}
add_force btn_down {0} {1 1050ns} {0 1150ns}
```

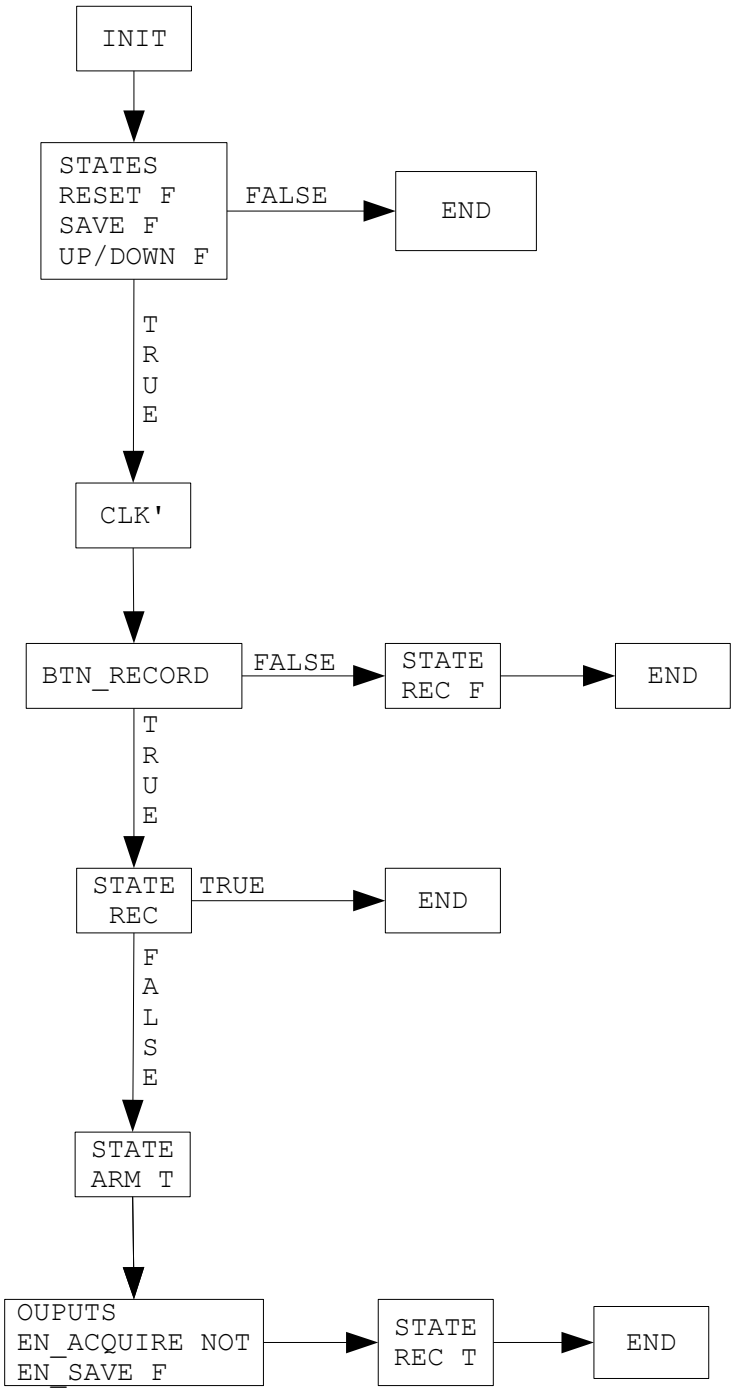
@AUTHOR: BLANCO CAAMANO, RAMON. <ramonblancocaamano@gmail.com>.
@ABOUT: MAIN CONTROL - RESET VERSION 0.2.



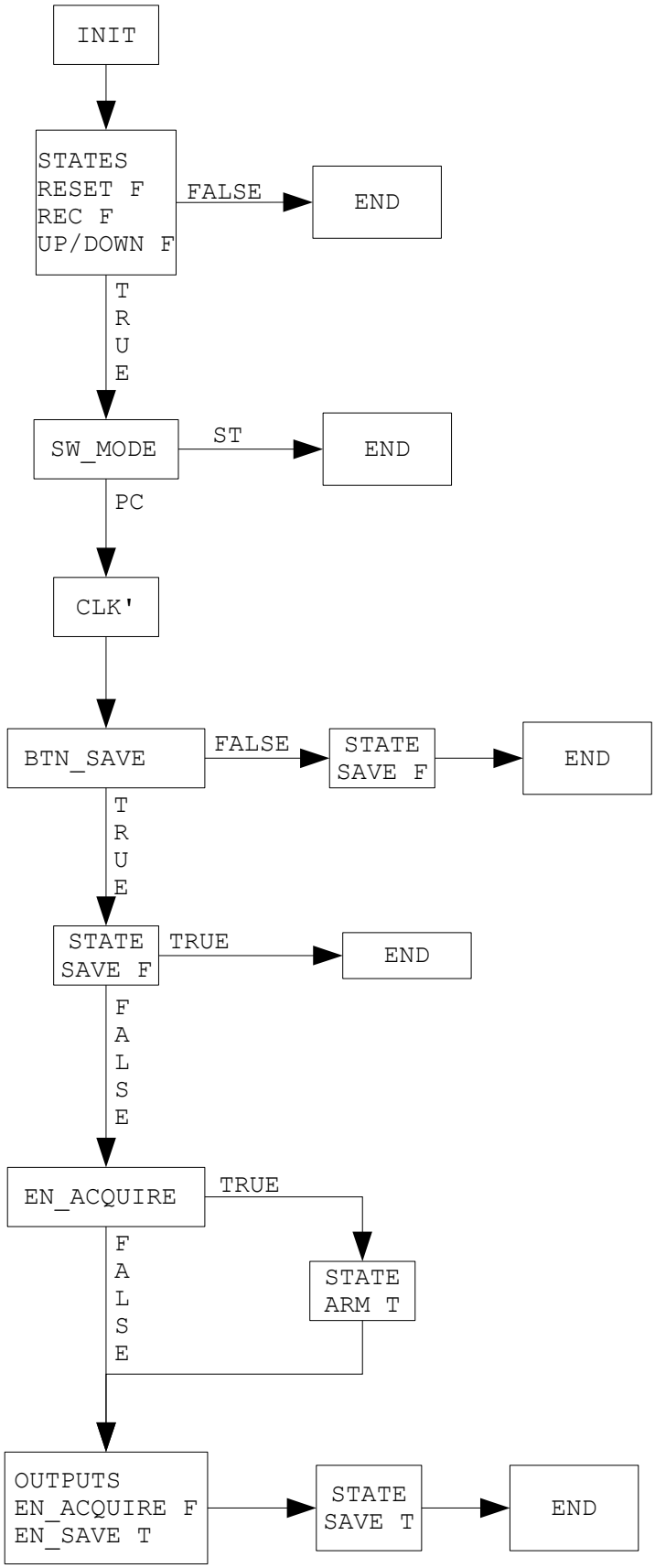
@AUTHOR: BLANCO CAAMANO, RAMON. <ramonblancocaamano@gmail.com>.
@ABOUT: MAIN CONTROL - ARMING VERSION 0.2.



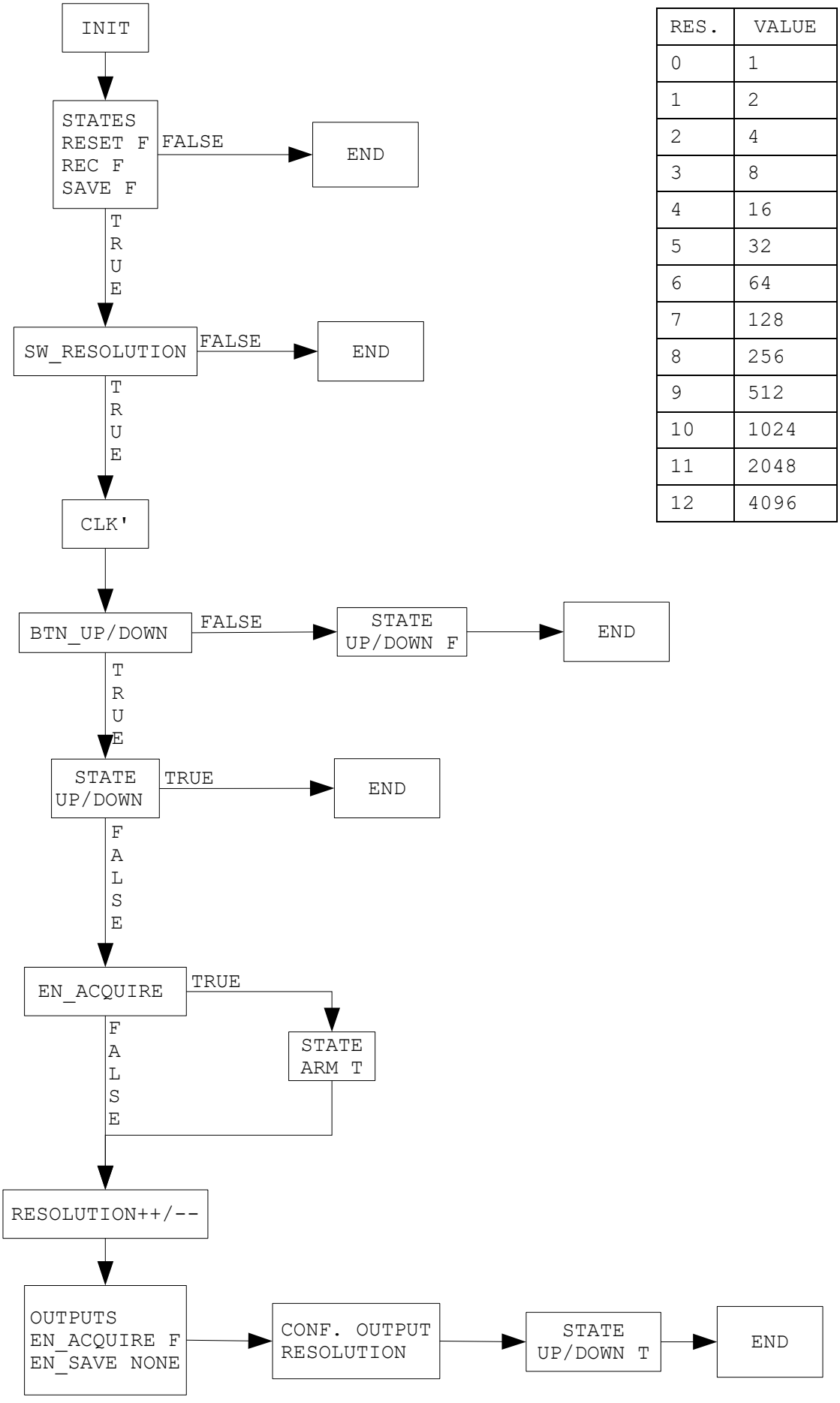
@AUTHOR: BLANCO CAAMANO, RAMON. <ramonblancocaamano@gmail.com>.
@ABOUT: MAIN CONTROL - RECORD VERSION 0.2.



@AUTHOR: BLANCO CAAMANO, RAMON. <ramonblancocaamano@gmail.com>.
@ABOUT: MAIN CONTROL - SAVE VERSION 0.2.



@AUTHOR: BLANCO CAAMANO, RAMON. <ramonblancocaamano@gmail.com>.
@ABOUT: MAIN CONTROL – UP/DOWN RESOLUTION VERSION 0.2.



RES.	VALUE
0	1
1	2
2	4
3	8
4	16
5	32
6	64
7	128
8	256
9	512
10	1024
11	2048
12	4096

E.3 PROCESSING MODULE.

ID		ABOUT	
	RST	IN	GLOBAL RESET SYSTEM.
	CLK_RADAR	IN	RADAR CLOCK.
	TRIGGER_RADAR	IN	RADAR TRIGGER.
	EN_ACQUIRE	IN	ENABLE ACQUISITION SUBSYSTEM.
	EN_RESOLUTION	IN	SELECT RESOLUTION ON/OFF. SIGNAL PREPROCESSING BASED ON SAMPLE WEIGHTING.
	RESOLUTION	IN	RESOLUTION VALUE.
	DIN	IN	SAMPLED DATA. NOTE: 12b LENGTH.
	OVERFLOW	IN	OVERFLOW DATA.
	DOUT	OUT	PROCESSED DATA. NOTE: 16b LENGTH
BUFF	BUFF_WR_EN	OUT	WRITE BUFFER ENABLE.
HSK 1	HSK_RD1	OUT	READ HANDSHAKE: INTERFACE/MEMORY TRIGGER.
	HSK_RD_EN1	IN	READ HANDSHAKE:INTERFACE/MEMORY ACK.
CONF.	NDATA	NUMBER OF SAMPLES PER PACKETS. NOTE: 4096.	
STATE	FSM	ENABLE.	
VAR.	COUNTER0	SAMPLING COUNTER MANAGED BY NDATA CONFIGURATION.	
	COUNTER1	DOWN-SAMPLING COUNTER MANAGED BY NDATA CONFIGURATION.	
	COUNTER_RES	RESOLUTION COUNTER MANAGED BY RESOLUTION PARAMETER.	

Tabla 35: VHDL DESIGN(III). PROCESSING MODULE.

TCL COMMANDS. UNIT TESTING(I).

```
add_force rst {1} {0 50ns }
add_force clk_radar  {1 0ns} {0 12500ps} -repeat_every 25000ps
add_force en_acquire {0} {1 100ns } {0 550ns }
add_force din -radix dec {0} {1 100ns } {2 125ns} {3 150ns} {4 175ns} {0 200ns} {1 300ns }
    {2 325ns} {3 350ns} {4 375ns} {0 400ns} {1 500ns } {2 525ns} {3 550ns} {4 575ns} {0 600ns}
add_force overflow {0} {1 300ns } {0 400ns }
add_force trigger_radar {0} {1 100ns} {0 125ns} {1 300ns} {0 325ns}  {1 500ns} {0 525ns}
add_force hsk_rd_ok1 {0} {1 225ns} {0 250ns} {1 425ns} {0 450ns}  {1 575ns} {0 600ns}
```

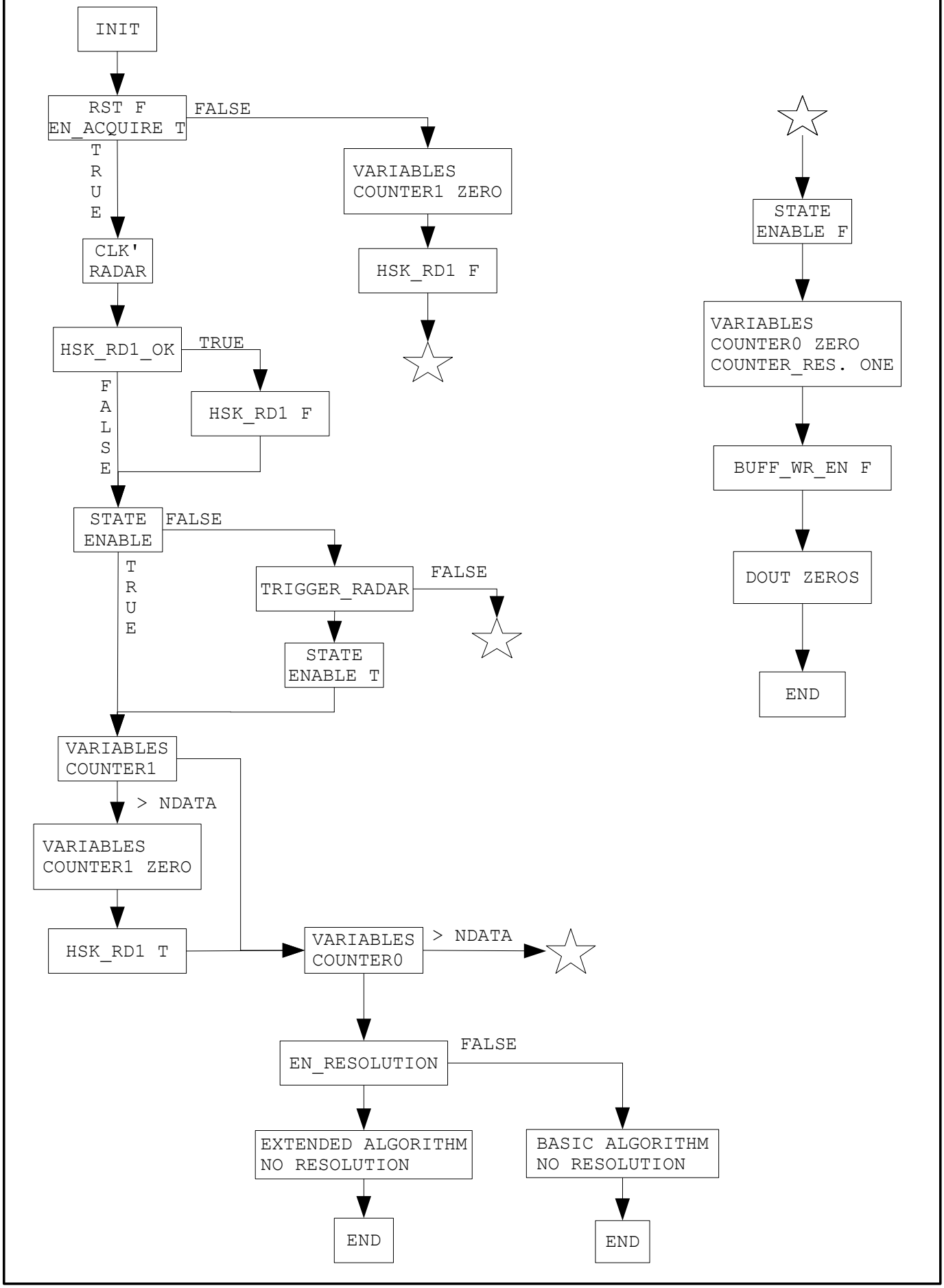
TCL COMMANDS. UNIT TESTING(II). DOWN-SAMPLING BY 2.

```
add_force rst {1} {0 50ns }
add_force clk_radar  {1 0ns} {0 12500ps} -repeat_every 25000ps
add_force en_acquire {0} {1 100ns }
add_force en_resolution {0} {1 300ns } {0 600ns }
add_force din -radix dec {0} {1 100ns } {2 125ns} {3 150ns} {4 175ns} {0 200ns} {1 300ns }
    {2 325ns} {3 350ns} {4 375ns} {0 400ns} {1 500ns } {2 525ns} {3 550ns} {4 575ns} {0 600ns}
add_force overflow {0} {1 100ns } {0 200ns }
add_force trigger_radar {0} {1 100ns} {0 125ns} {1 300ns} {0 325ns}  {1 500ns} {0 525ns}
add_force hsk_rd_ok1 {0} {1 225ns} {0 250ns} {1 625ns} {0 650ns}
```

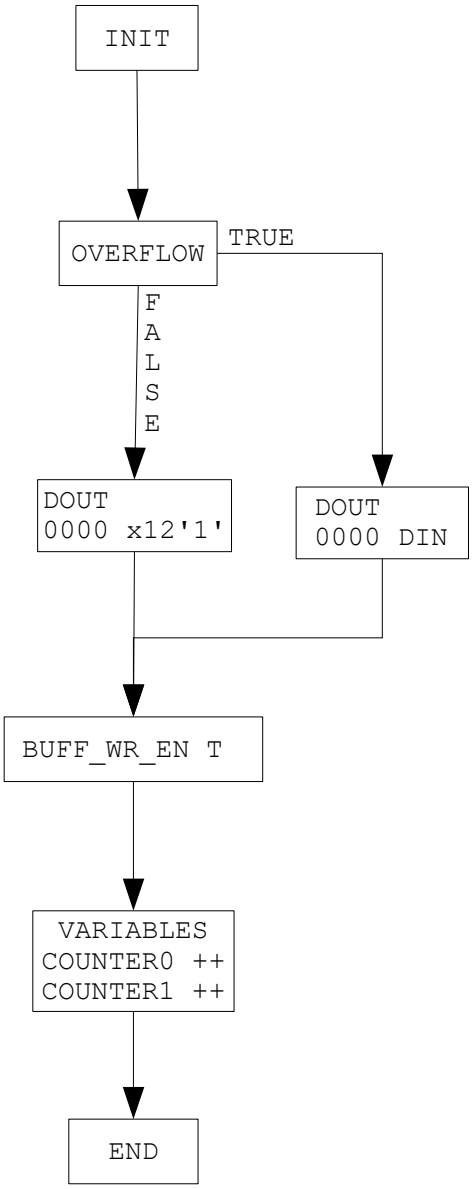
TCL COMMANDS. UNIT TESTING(II). DOWN-SAMPLING BY 4.

```
add_force rst {1} {0 25ns }
add_force clk_radar {1 0ns} {0 12500ps} -repeat_every 25000ps
add_force en_acquire {0} {1 25ns }
add_force en_resolution {0} {1 25ns }
add_force din -radix dec {0} {1 50ns } {2 75ns} {3 100ns} {4 125ns} {5 150ns } {6 175ns}
    {7 200ns} {8 225ns} {9 250ns } {10 275ns} {11 300ns} {12 325ns} {13 350ns } {14 375ns}
    {15 400ns} {16 425ns} {0 450ns }
add_force overflow {0}
add_force trigger_radar {0} {1 50ns} {0 75ns}
```

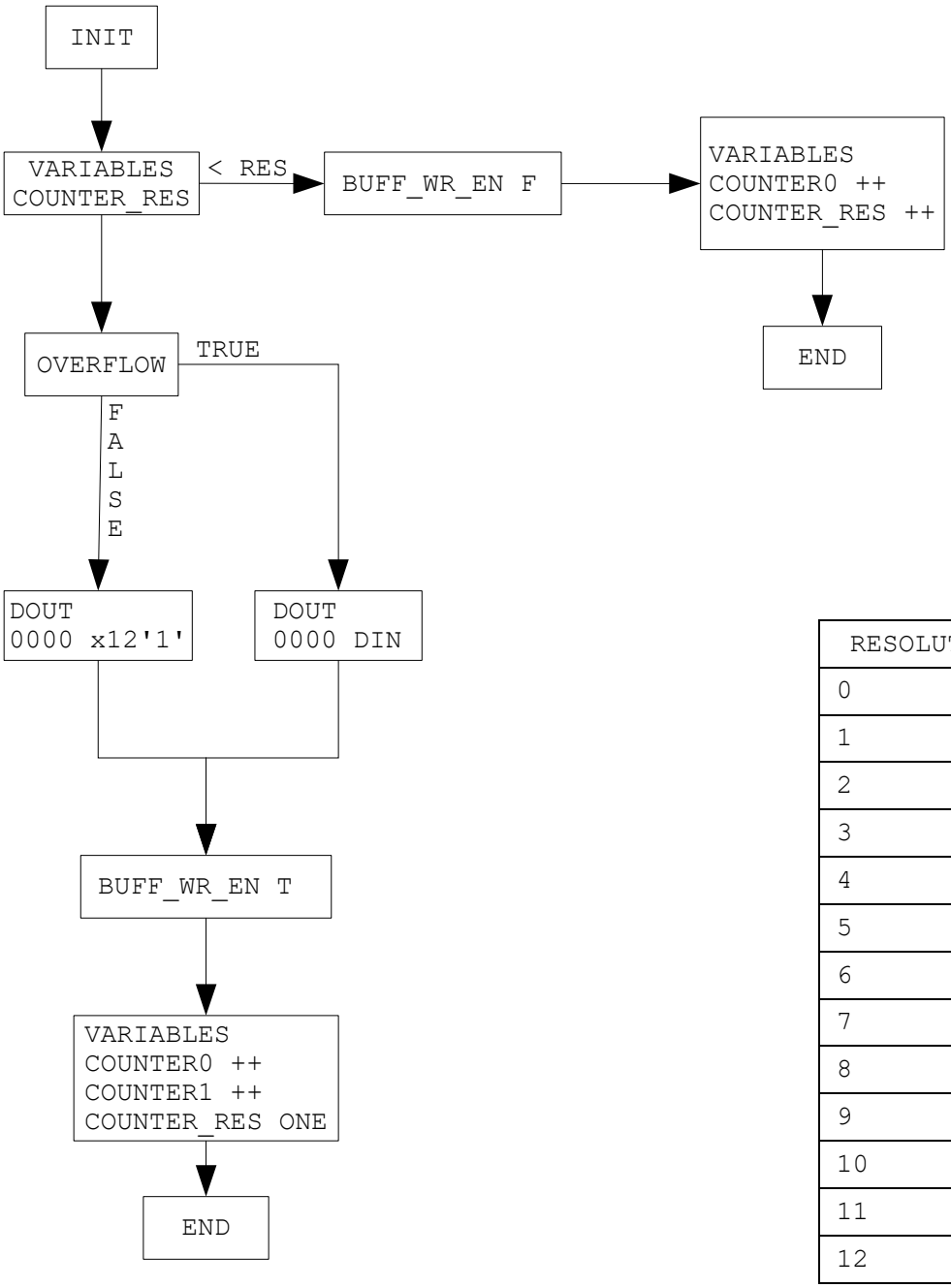
@AUTHOR: BLANCO CAAMANO, RAMON. <ramonblancocaamano@gmail.com>.
@ABOUT: PROCESSING - MAIN VERSION 0.4.



@AUTHOR: BLANCO CAAMANO, RAMON. <ramonblancocaamano@gmail.com>.
@ABOUT: PROCESSING – BASIC ALGORITHM, NO RESOLUTION VERSION 0.4.



@AUTHOR: BLANCO CAAMANO, RAMON. <ramonblancocaamano@gmail.com>.
@ABOUT: PROCESSING – EXTENDED ALGORITHM, RESOLUTION ENABLED VERSION 0.4.



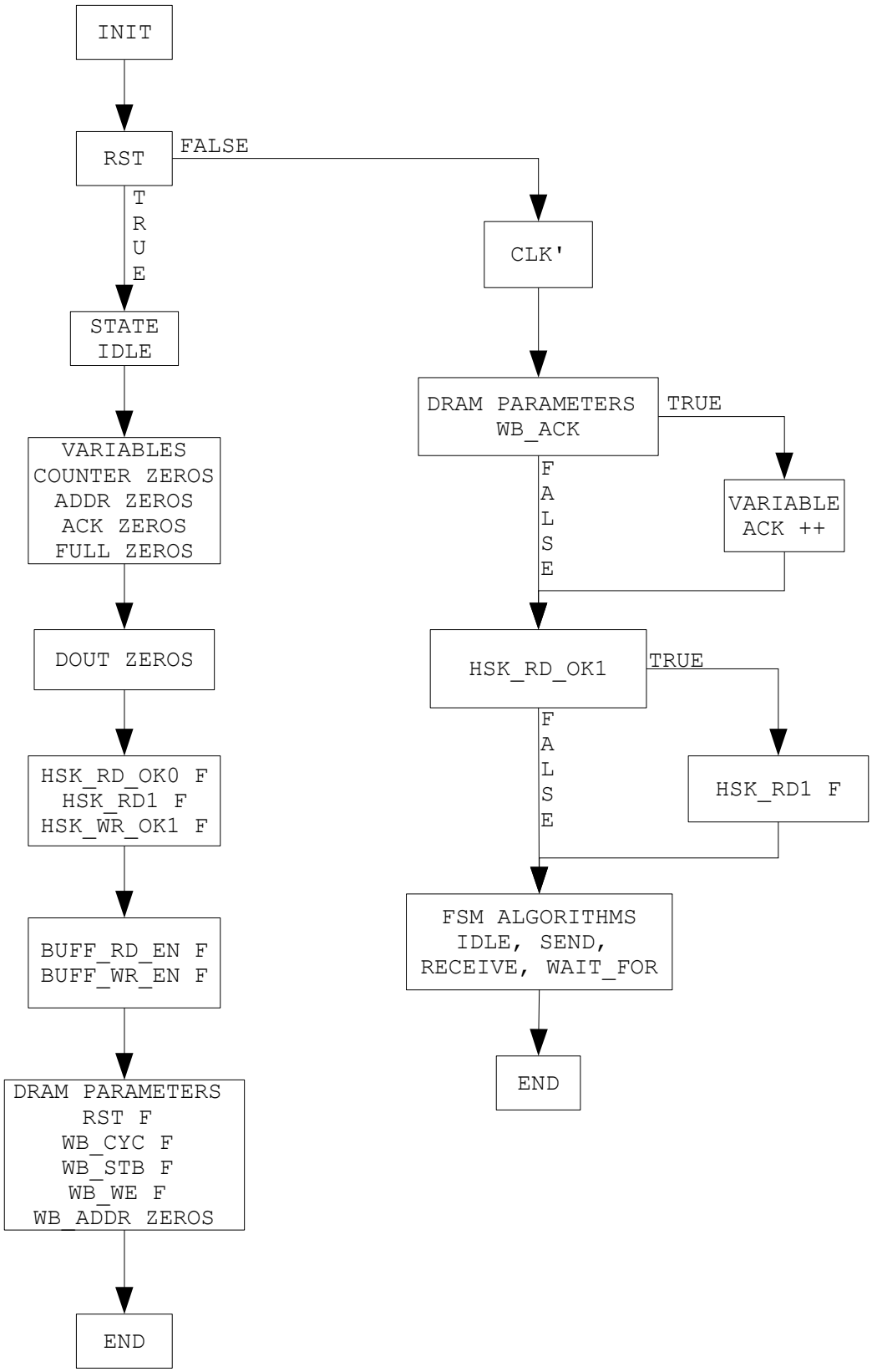
RESOLUTION	COUNTER_RES
0	1
1	2
2	4
3	8
4	16
5	32
6	64
7	128
8	256
9	512
10	1024
11	2048
12	4096

E.4 DDR3 CONTROL.

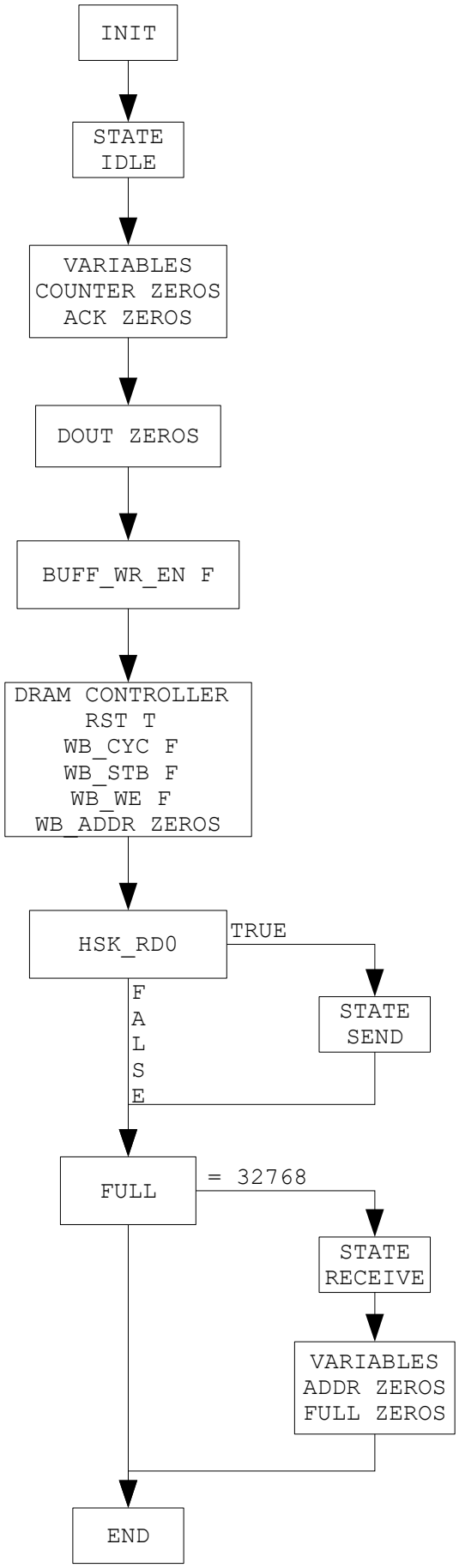
ID		ABOUT	
	RST	IN	GLOBAL RESET SYSTEM.
	CLK_81	IN	DRAM CLOCK.
	DRAM_PARAMETERS	IN	WB_DATA, WB_ACK & WB_STALL.
	DRAM_PARAMETERS	OUT	RST, WB_CYC, WB_STB, WB_WE & WB_ADDR
BUFF	BUFF_RD_EN	IN	READ BUFFER ENABLE.
	BUFF_EMPTY	IN	BUFFER EMPTY.
	BUFF_WR_EN	OUT	WRITE BUFFER ENABLE.
HSK 0	HSK_RD0	IN	READ HANDSHAKE: PROCESSING TRIGGER.
	HSK_RD_EN0	OUT	READ HANDSHAKE: PROCESSING ACK.
HSK 1	HSK_RD1	OUT	READ HANDSHAKE: INTERFACE/MEMORY TRIGGER.
	HSK_RD_EN1	IN	READ HANDSHAKE:INTERFACE/MEMORY ACK.
	HSK_WR1	IN	WRITE HANDSHAKE:INTERFACE/MEMORY TRIGGER.
	HSK_WR_EN1	OUT	WRITE HANDSHAKE:INTERFACE/MEMORY ACK.
CONF.	NDATA	NUMBER OF SAMPLES PER PACKETS.	
	MEMORY PAGING	NACK & AW.	
STATE	FSM	IDLE, SEND, RECEIVE & WAIT_FOR.	
VAR.	COUNTERS	COUNTER, ADDR, ACK & FULL.	

Tabla 36: VHDL DESIGN(IV). DDR3 CONTROL MODULE.

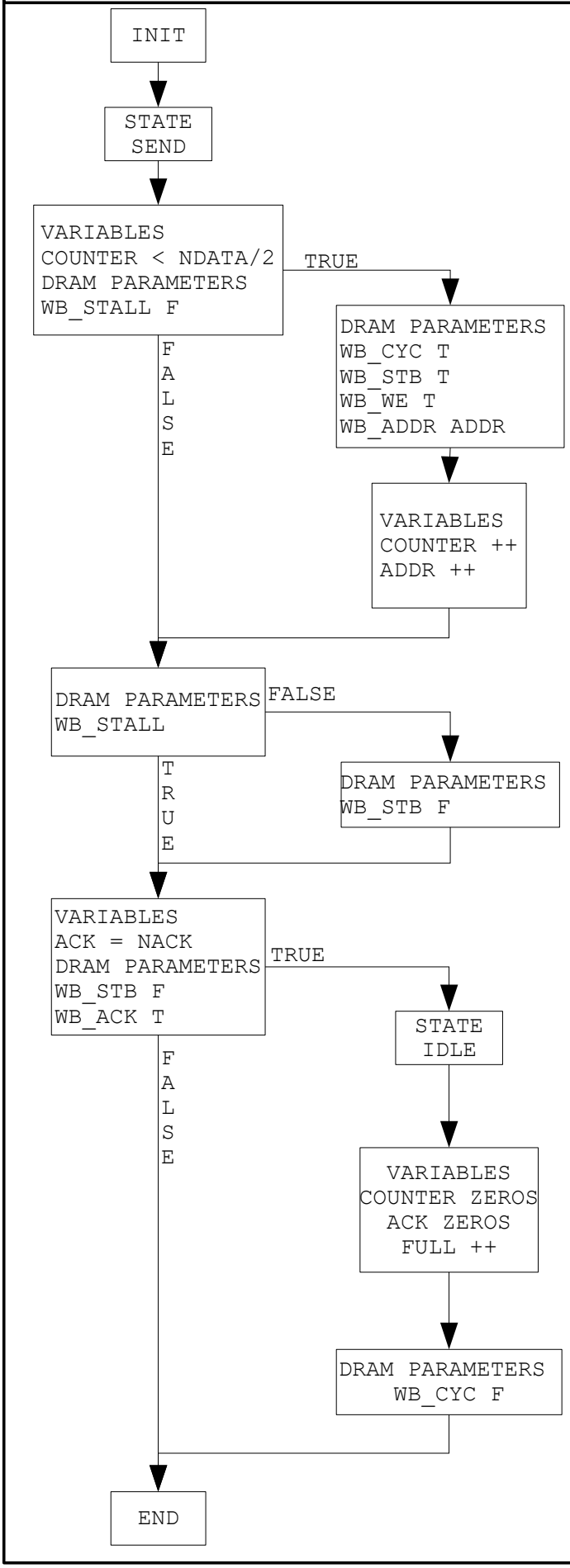
@AUTHOR: BLANCO CAAMANO, RAMON. <ramonblancocaamano@gmail.com>.
@ABOUT: DRAM CONTROL VERSION 0.3.



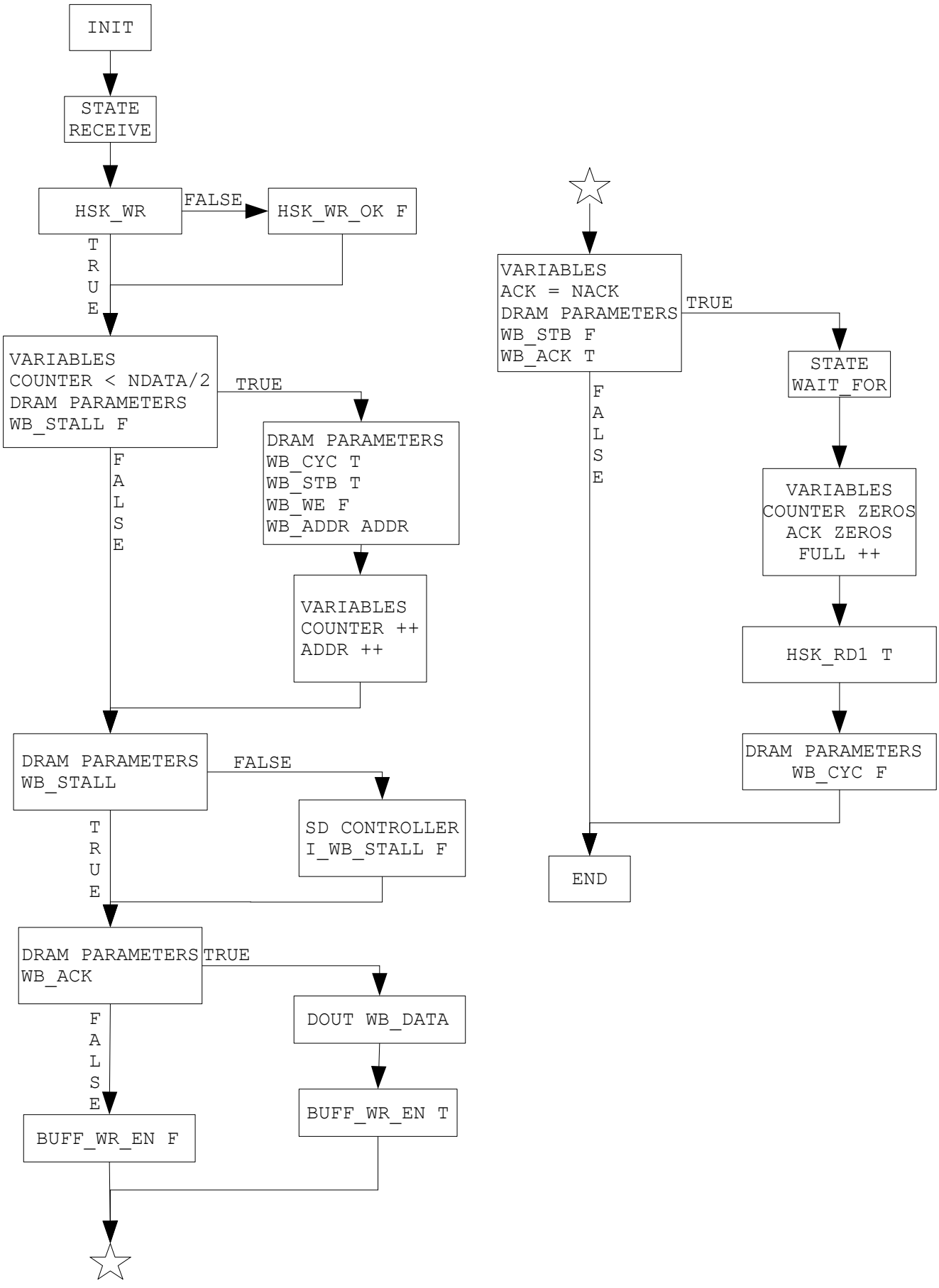
@AUTHOR: BLANCO CAAMANO, RAMON. <ramonblancocaamano@gmail.com>.
@ABOUT: DRAM CONTROL – IDLE ALGORITHM VERSION 0.3.



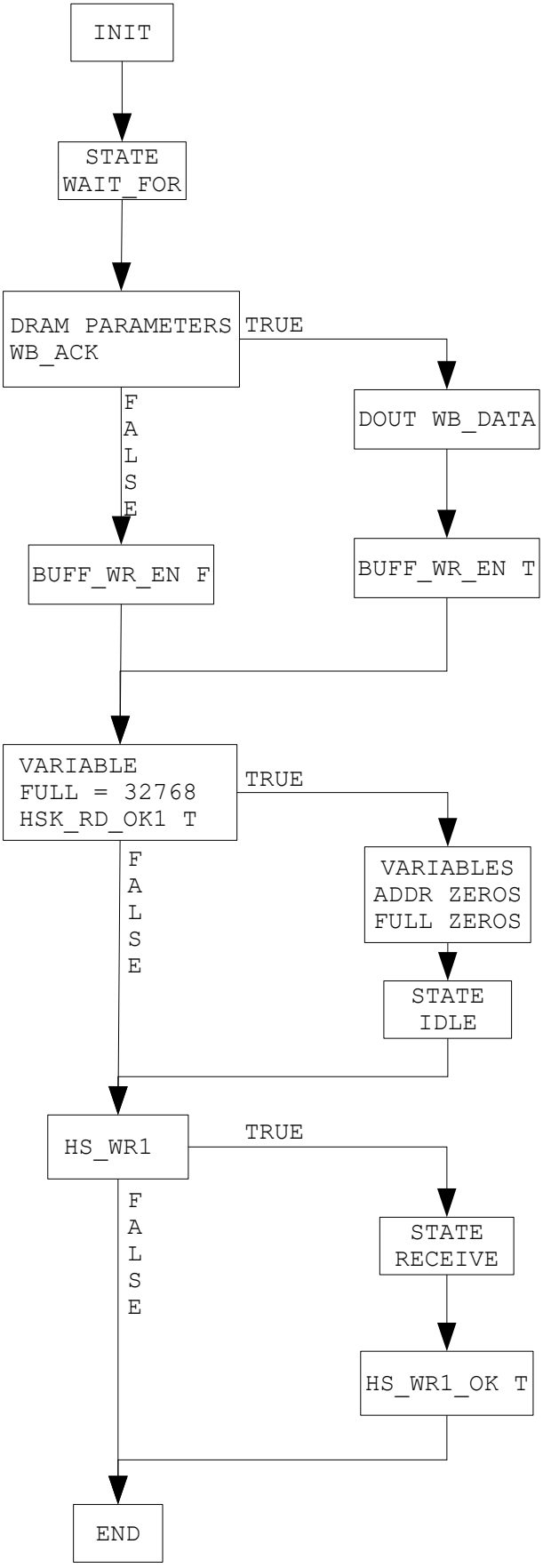
@AUTHOR: BLANCO CAAMANO, RAMON. <ramonblancocaamano@gmail.com>.
@ABOUT: DRAM CONTROL – SEND ALGORITHM VERSION 0.3.



@AUTHOR: BLANCO CAAMANO, RAMON. <ramonblancocaamano@gmail.com>.
@ABOUT: DRAM CONTROL – RECEIVE ALGORITHM VERSION 0.1.



@AUTHOR: BLANCO CAAMANO, RAMON. <ramonblancocaamano@gmail.com>.
@ABOUT: DRAM CONTROL – WAIT_FOR ALGORITHM VERSION 0.1.

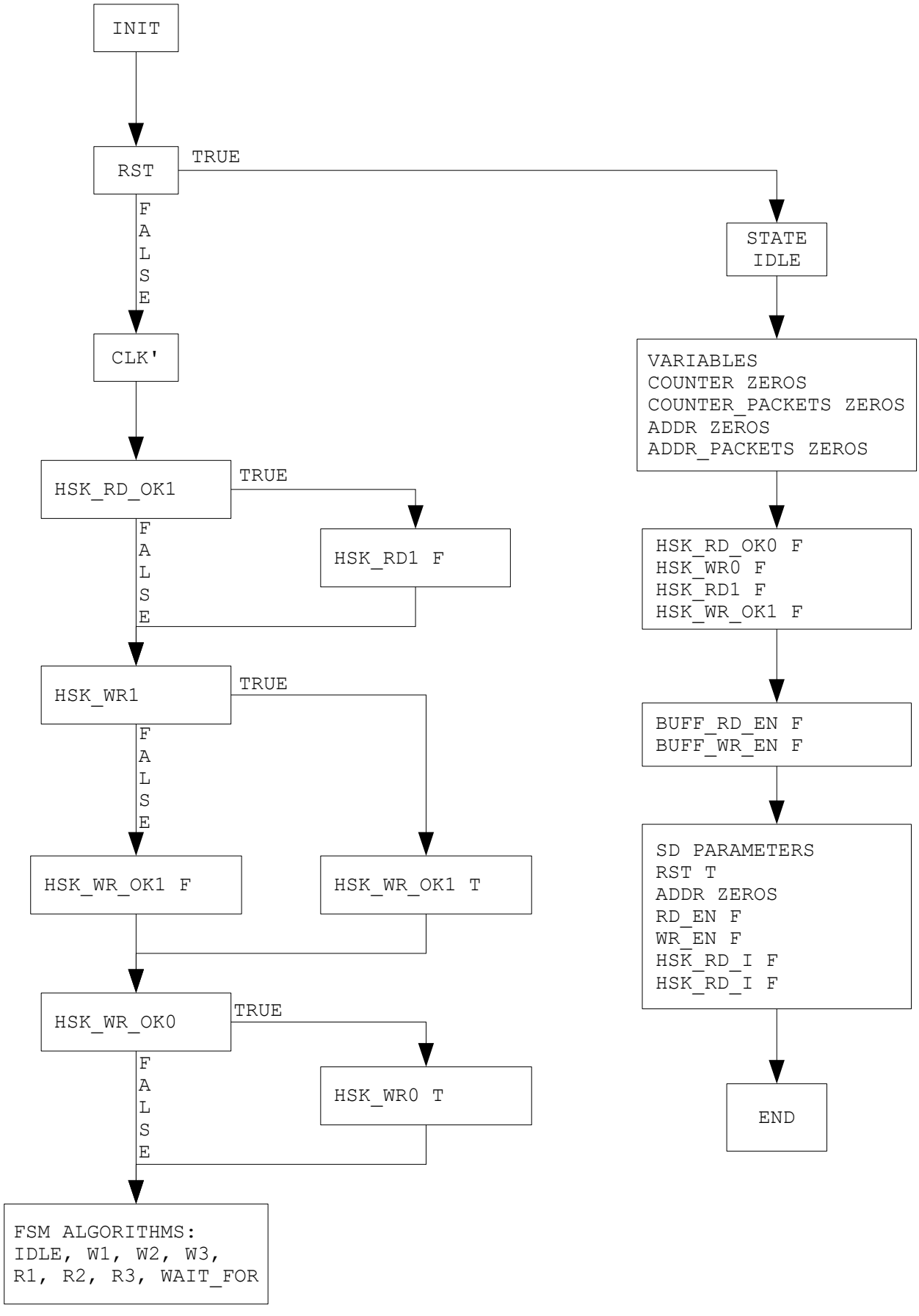


E.5 SD CONTROL.

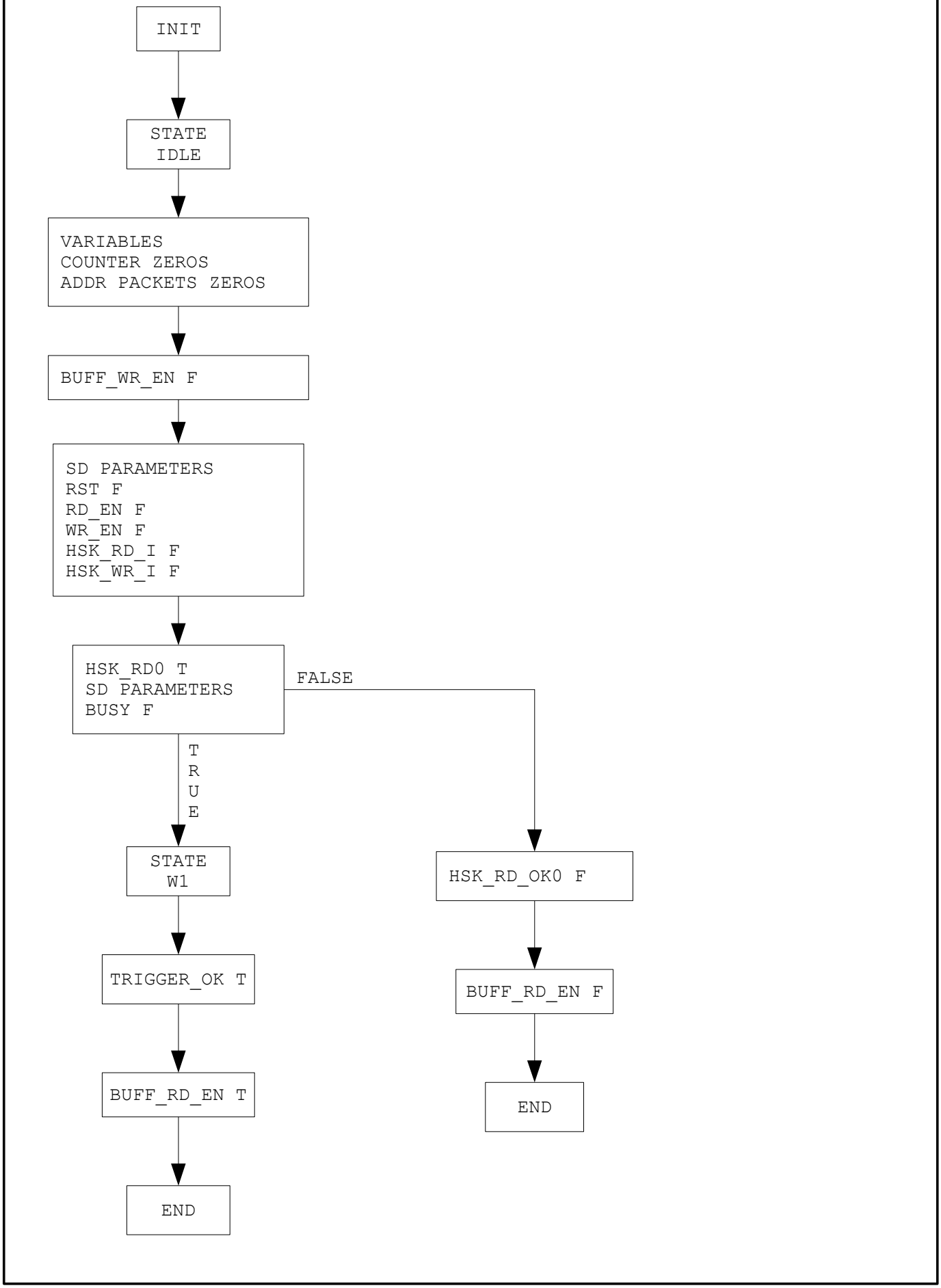
ID		ABOUT	
	RST	IN	GLOBAL RESET SYSTEM.
	CLK_50	IN	SD CLOCK.
	SD_PARAMETERS	IN	HSK_RD_O, HSK_WR_O & BUSY.
	SD_PARAMETERS	OUT	RST, ADDR, RD_EN, WR_EN, HSK_RD_I & HSK_WR_I.
BUFF	BUFF_RD_EN	OUT	READ BUFFER ENABLE.
	BUFF_WR_EN	OUT	WRITE BUFFER ENABLE.
HSK 0	HSK_RD0	IN	READ HANDSHAKE. DRAM TRIGGER.
	HSK_RD_OK0	OUT	READ HANDSHAKE: DRAM ACK.
	HSK_WR0	OUT	WRITE HANDSHAKE: DRAM TRIGGER.
	HSK_ WR_OK0	IN	WRITE HANDSHAKE: DRAM ACK.
HSK 1	HSK_RD1	OUT	READ HANDSHAKE. INTERFACE TRIGGER.
	HSK_RD_OK1	IN	READ HANDSHAKE. INTERFACE ACK.
	HSK_WR1	IN	WRITE HANDSHAKE. INTERFACE TRIGGER.
	HSK_WR_OK1	OUT	WRITE HANDSHAKE. INTERFACE ACK.
CONF.	DATA	NUMBER OF SAMPLES PER PACKETS. NOTE: 4096.	
	PACKETS	TOTAL NUMBER OF PACKETS. NOTE: 32768	
STATE	FSM	IDLE, W1, W2, W3, R1, R2, R3 & WAIT_FOR.	
VAR.	COUNTER	SAMPLING COUNTER MANAGED BY DATA CONFIGURATION.	
	COUNTER_PACKETS	PACKETS COUNTER MANAGED BY PACKETS CONFIGURATION.	
	ADDR	DATA SD ADDRESSING.	
	ADDR_PACKETS	PACKETS SD ADDRESSING.	

Tabla 37: VHDL DESIGN(V). SD CONTROL MODULE.

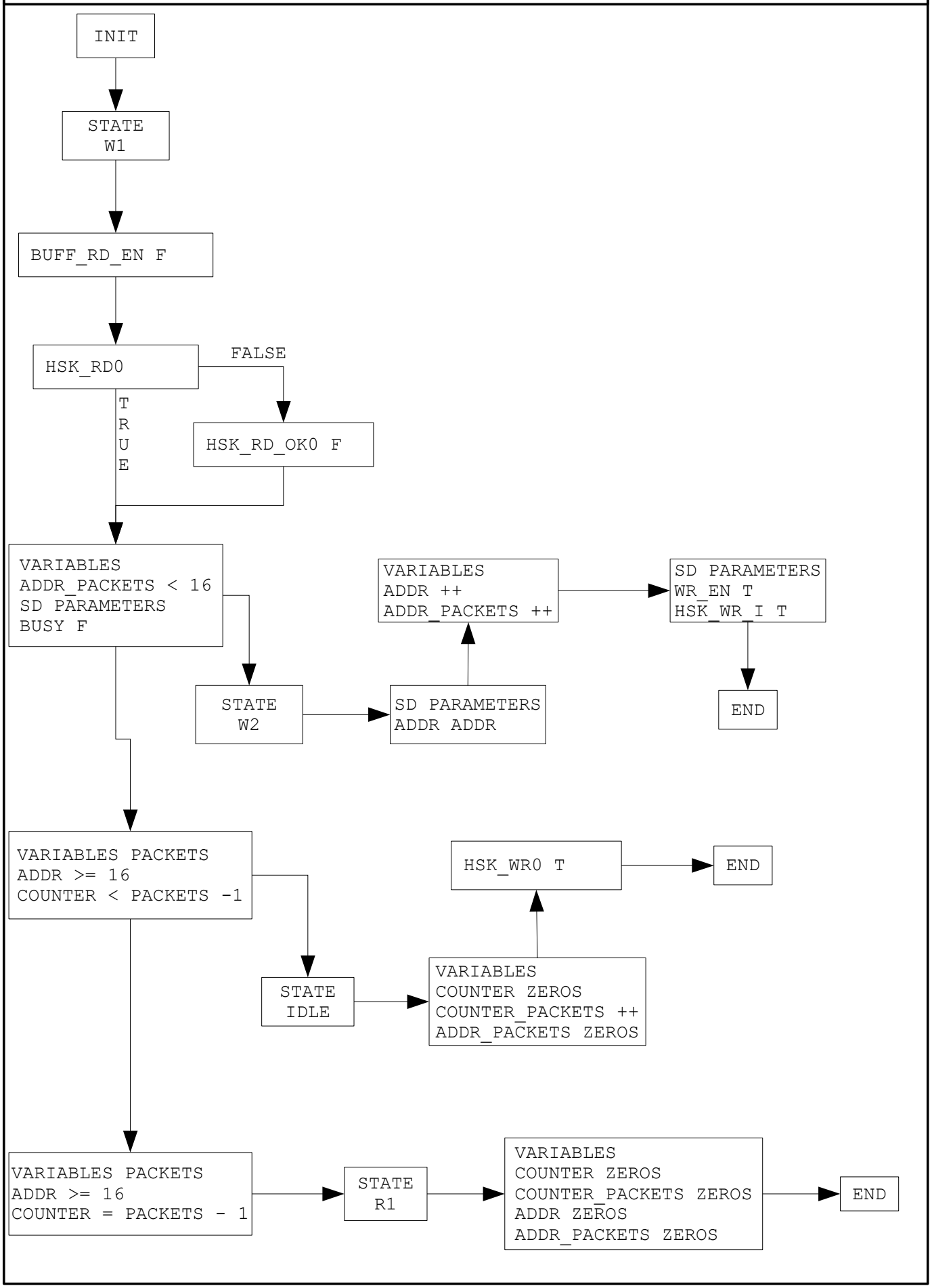
@AUTHOR: BLANCO CAAMANO, RAMON. <ramonblancocaamano@gmail.com>.
@ABOUT: SD CONTROL VERSION 0.3.



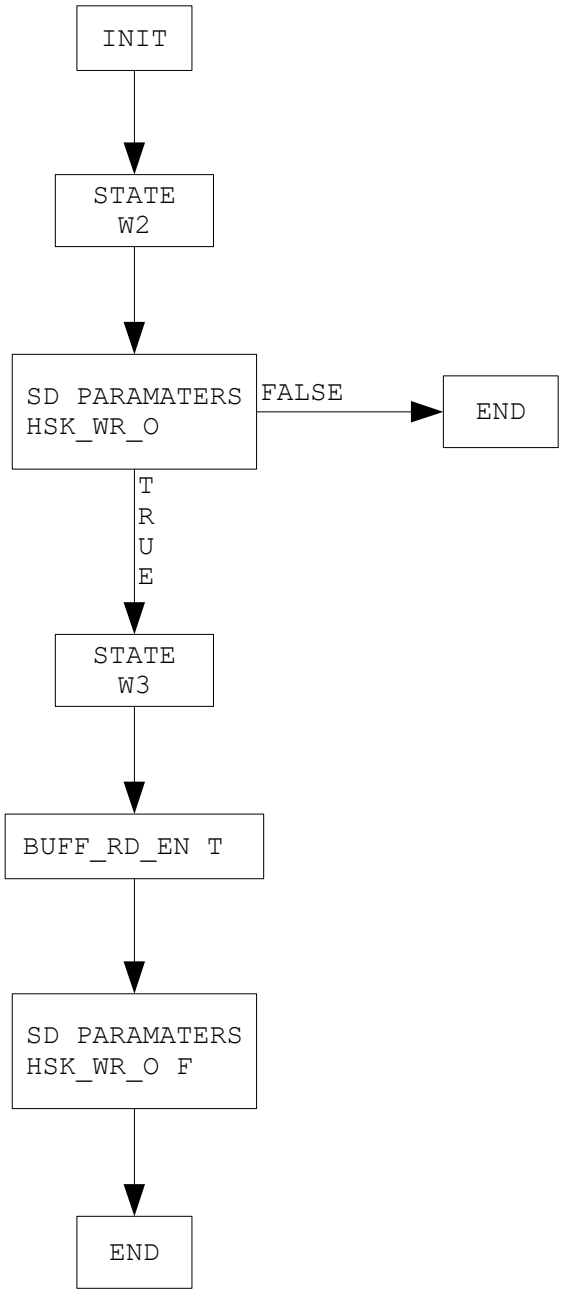
@AUTHOR: BLANCO CAAMANO, RAMON. <ramonblancocaamano@gmail.com>.
@ABOUT: SD CONTROL - IDLE ALGORITHM VERSION 0.3.



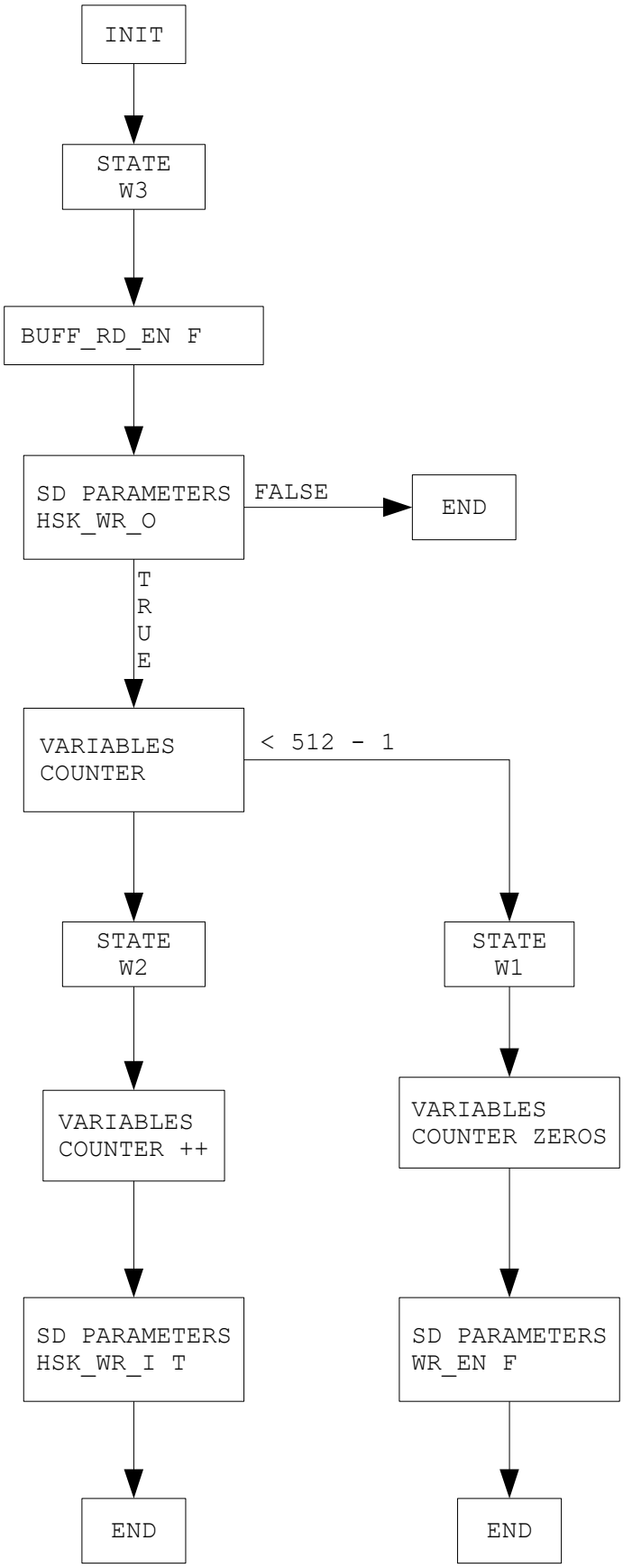
@AUTHOR: BLANCO CAAMANO, RAMON. <ramonblancocaamano@gmail.com>.
@ABOUT: SD CONTROL – W1 ALGORITHM VERSION 0.3.



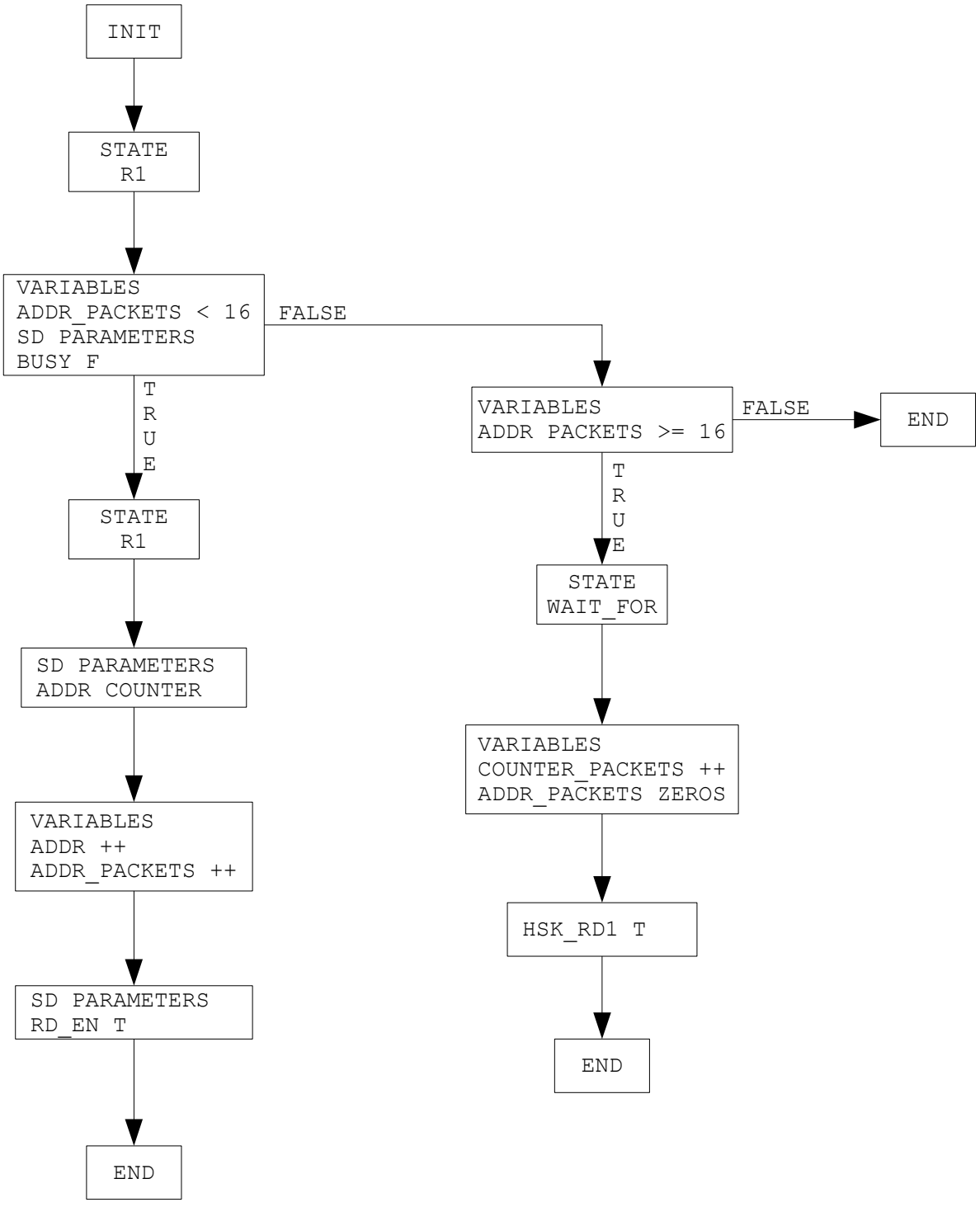
@AUTHOR: BLANCO CAAMANO, RAMON. <ramonblancocaamano@gmail.com>.
@ABOUT: SD CONTROL - W2 ALGORITHM VERSION 0.3.



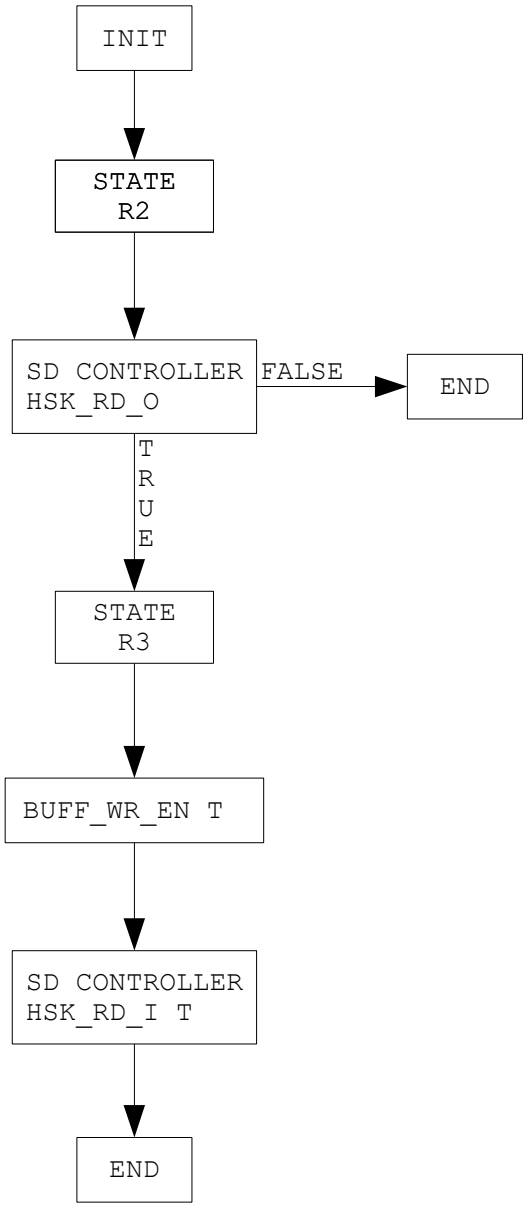
@AUTHOR: BLANCO CAAMANO, RAMON. <ramonblancocaamano@gmail.com>.
@ABOUT: SD CONTROL – W3 ALGORITHM VERSION 0.3.



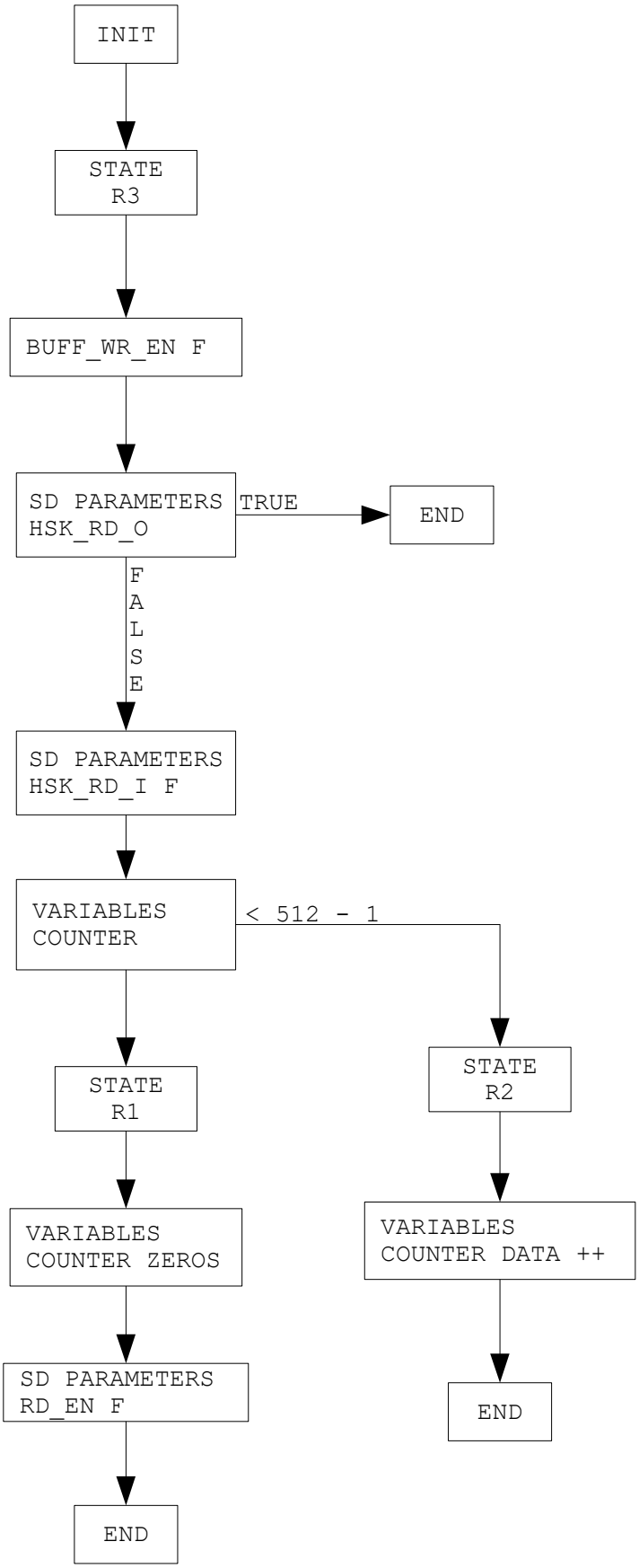
@AUTHOR: BLANCO CAAMANO, RAMON. <ramonblancocaamano@gmail.com>.
@ABOUT: SD CONTROL – R1 ALGORITHM VERSION 0.3.



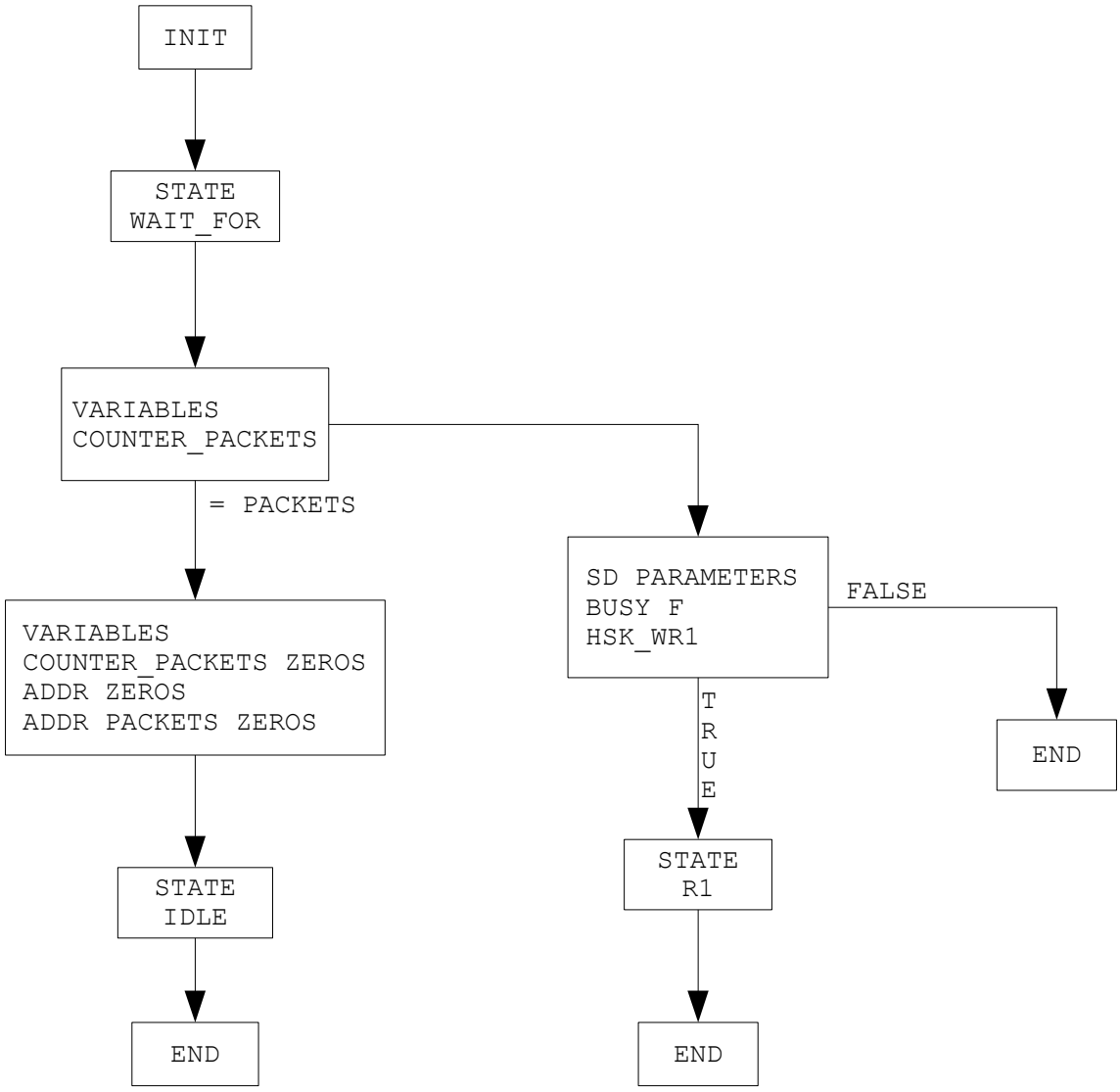
@AUTHOR: BLANCO CAAMANO, RAMON. <ramonblancocaamano@gmail.com>.
@ABOUT: SD CONTROL - R2 ALGORITHM VERSION 0.3.



@AUTHOR: BLANCO CAAMANO, RAMON. <ramonblancocaamano@gmail.com>.
@ABOUT: SD CONTROL – R3 ALGORITHM VERSION 0.3.



@AUTHOR: BLANCO CAAMANO, RAMON. <ramonblancocaamano@gmail.com>.
@ABOUT: SD CONTROL – WAIT_FOR ALGORITHM VERSION 0.3.



E.6 ETHERNET CONTROL.

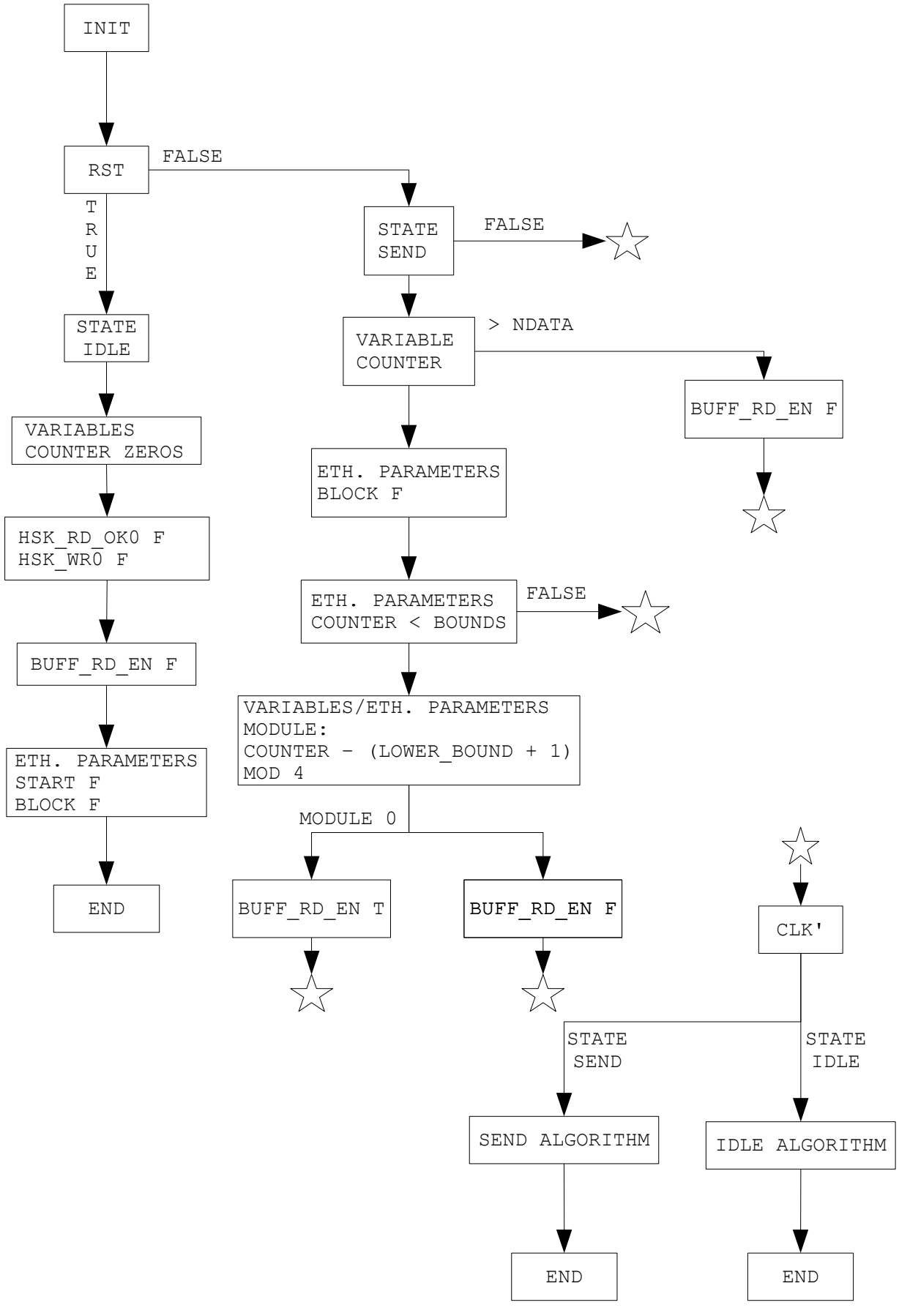
ID		ABOUT	
	RST	IN	GLOBAL RESET SYSTEM.
	CLK	IN	ETHERNET CLOCK.
	ETH_PARAMETERS	IN	COUNTER.
	ETH_PARAMETERS	OUT	START, BLOCK.
BUFF	BUFF_RD_EN	OUT	READ BUFFER ENABLE.
HSK 0	HSK_RD0	IN	READ HANDSHAKE. DRAM/MEMORY TRIGGER.
	HSK_RD_OK0	OUT	READ HANDSHAKE: DRAM/MEMORY ACK.
	HSK_WR0	OUT	WRITE HANDSHAKE: DRAM/MEMORY TRIGGER.
	HSK_WR_OK0	IN	WRITE HANDSHAKE: DRAM/MEMORY ACK.
CONF.	NDATA	NUMBER OF SAMPLES PER PACKETS. NOTE: 4096.	
STATE	FSM	IDLE & SEND.	
VAR.	COUNTER	SAMPLING COUNTER MANAGED BY DATA CONFIGURATION.	
CONST.	BOUNDS	SET PAYLOAD DIMENSIONS IN A TCP/IP PACKET.	

Tabla 38: VHDL DESIGN(VI). ETHERNET CONTROL MODULE.

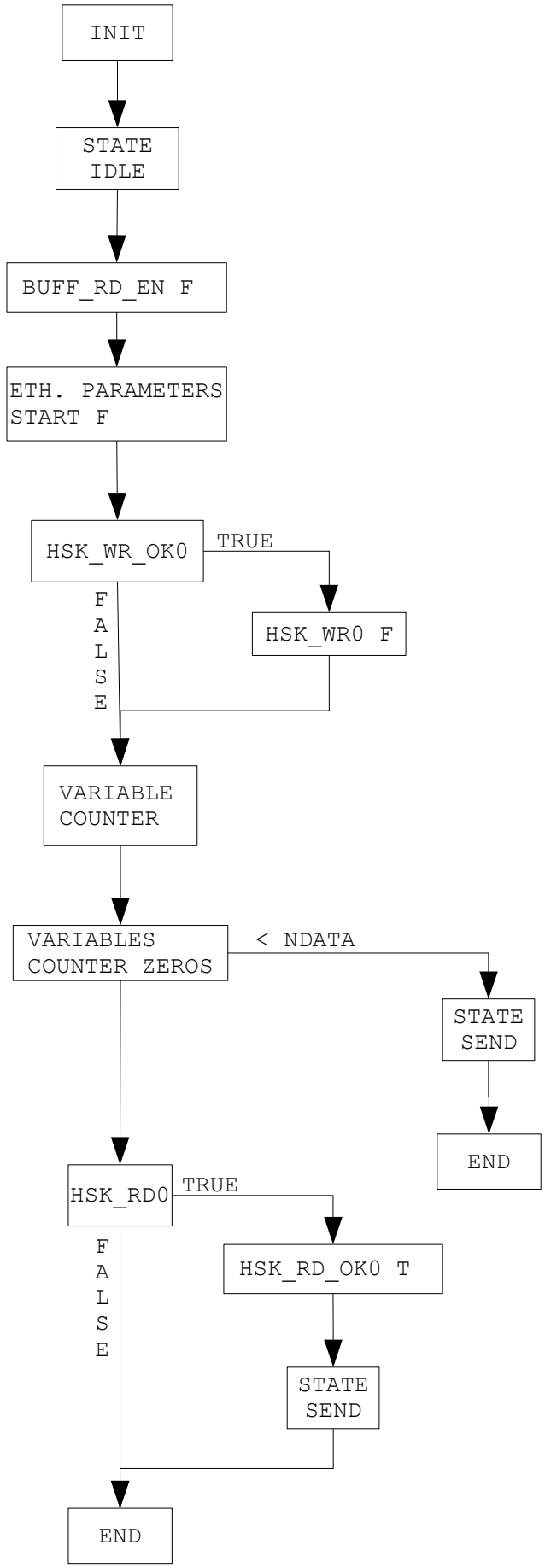
TCL COMMANDS. UNIT TESTING(I)(II).

```
add_force rst {1} {0 1000ns }
add_force clk  {1 0ns} {0 5000ps} -repeat_every 10000ps
add_force hsk_rd0 {0} {1 1000ns} {0 1500ns}
add_force hsk_wr_ok0 {0} {1 19212ns} {0 19712ns}
add_force buff_empty {0 0ns}
add_force i_eth_tx_clk  {1 0ns} {0 4000ps} -repeat_every 8000ps
add_force din -radix hex {0} {1001 1728ns} {1002 1760ns}  {1003 1792ns}  {1004 1824ns}
```

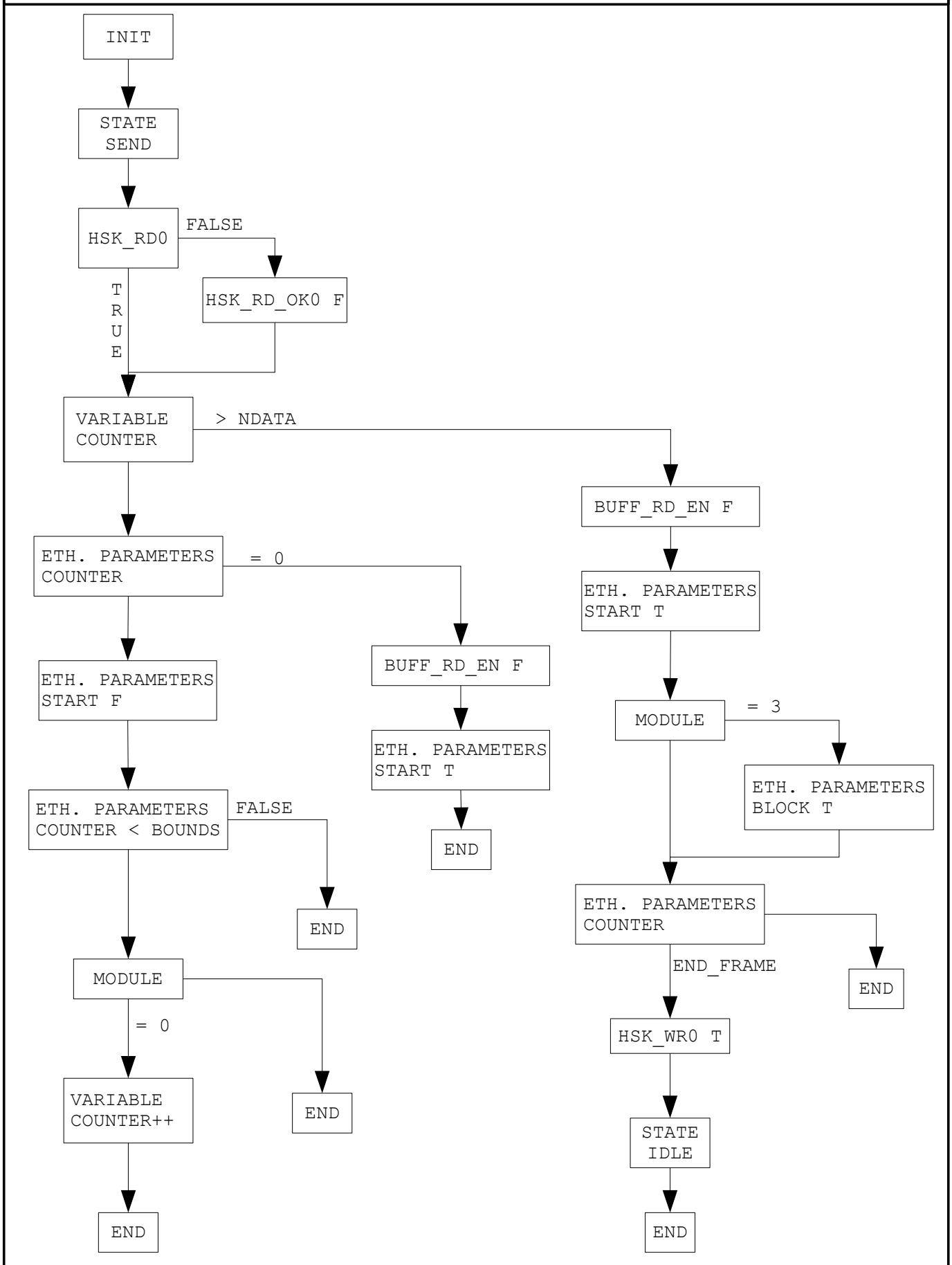
@AUTHOR: BLANCO CAAMANO, RAMON. <ramonblancocaamano@gmail.com>.
@ABOUT: ETHERNET CONTROL VERSION 0.3.



@AUTHOR: BLANCO CAAMANO, RAMON. <ramonblancocaamano@gmail.com>.
@ABOUT: ETHERNET CONTROL – IDLE ALGORITHM VERSION 0.3.



@AUTHOR: BLANCO CAAMANO, RAMON. <ramonblancocaamano@gmail.com>.
@ABOUT: ETHERNET CONTROL - SEND ALGORITHM VERSION 0.3.



F MATLAB CODE.

F.1 DOWN-SAMPLING

```
%%
% @file: down_sampling.m
% @author: BLANCO CAAMANO, RAMON.
% @about: DOWN-SAMPLING ALGORITHM.
%%
clearvars;

%% DEFINE PARAMATERS.
NDATA = 128;
NPACKETS = 2;
FREQUENCY = 0.5;
AMPLITUDE = 3.3;
SAMPLING_RATE = 40;
NBITS = 12;
PERIOD = 1/SAMPLING_RATE;
TIME = 0: PERIOD : PERIOD*(NDATA - 1);
N = 0:((NDATA*NPACKETS -1));
x = zeros(1,NDATA);
x_n = zeros(NPACKETS,NDATA*NPACKETS);
x_nd = zeros(NPACKETS,NDATA*NPACKETS);


%% RADAR SIGNAL
x = AMPLITUDE * sin(2*pi*FREQUENCY*TIME)+ AMPLITUDE;

%% QUANTIZATION.
x_q = quantization(x, AMPLITUDE, NBITS);


%% DOWN-SAMPLING.
x_d = downsample(x_q,2);


%% RESULTS.
for i = 1 : NPACKETS
    x_n(i, 1+NDATA*(i-1) : NDATA*i) = x;
    x_nd(i, 1+NDATA*(i-1) : NDATA*i) = [x_d x_d];
end

%% FIGURES.
figure(1)
hold on;
for i = 1 : NPACKETS
```

```
    plot(N, x_n(i,:),'*');
    ylim([0 2*AMPLITUDE]);
    xlim([0 NDATA*NPACKETS]);
end
legend('NPACKET: 0 NDATA: 128','NPACKET: 1 NDATA: 128')
hold off;
xlabel('SAMPLES');
ylabel('VALUE');
title('500KHz SIGNAL AT 12b @ 40MHZ');

figure(2)
hold on;
for i = 1 : NPACKETS
    plot(N, x_nd(i,:),'*');
    ylim([0 2*AMPLITUDE]);
    xlim([0 NDATA*NPACKETS]);
end
legend('NPACKET: 0 NDATA: 128','NPACKET: 1 NDATA: 128')
hold off;
xlabel('SAMPLES');
ylabel('VALUE');
title('500KHz SIGNAL AT 12b @ 40MHZ, DOWN-SAMPLING BY 2.');
```

F.2 QUANTIZATION.

```
%%
% @file: quantization.m
% @author: BLANCO CAAMANO, RAMON.
% @about: QUANTIZATION ALGORITHM.
%%
function [x_q] = quantization(x, amplitude, nbits)

    x_q = zeros(1,length(x));

    % DETERMINING THE STEP SIZE.
    step = 2*amplitude/(2^nbits);
    low = 0;
    high= 2*amplitude;

    % ITERATING FROM LOWEST TO THE HIGHEST LEVELS.
    for i = low : step : high
        for j=1:length(x)
            if(((i-step/2)<x(j))&&(x(j)<(i+step/2)))%
                x_q(j)=i;
            end
        end
    end
```

```
end
```