

INSTITUTO TECNOLÓGICO Y DE ESTUDIOS
SUPERIORES DE MONTERREY

MONTERREY CAMPUS

GRADUATE PROGRAM IN MECHATRONICS AND
INFORMATION TECHNOLOGIES



TECNOLÓGICO
DE MONTERREY

A VHDL-AMS Transistor Level Model of a UHF RFID Tag for System
Simulation

THESIS

PRESENTED AS A PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF

MASTER OF SCIENCE WITH MAJOR IN ELECTRONIC ENGINEERING
(ELECTRONIC SYSTEMS)

BY

Miguel Angel Rios Gastelum

MONTERREY, N. L., MEXICO, MAY, 2009

**INSTITUTO TECNOLÓGICO Y DE ESTUDIOS
SUPERIORES DE MONTERREY**

MONTERREY CAMPUS

**GRADUATE PROGRAM IN MECHATRONICS AND
INFORMATION TECHNOLOGIES**



**TECNOLÓGICO
DE MONTERREY**

**A VHDL-AMS Transistor Level Model of a UHF RFID Tag for System
Simulation**

THESIS

**PRESENTED AS A PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF**

**MASTER OF SCIENCE WITH MAJOR IN ELECTRONIC ENGINEERING
(ELECTRONIC SYSTEMS)**

BY

Miguel Angel Rios Gastelum

MONTERREY, N. L., MEXICO, MAY, 2009

**INSTITUTO TECNOLÓGICO Y DE ESTUDIOS
SUPERIORES DE MONTERREY**

MONTERREY CAMPUS

**GRADUATE PROGRAM IN MECHATRONICS AND
INFORMATION TECHNOLOGIES**



**TECNOLÓGICO
DE MONTERREY®**

**A VHDL-AMS Transistor Level Model of a UHF RFID Tag for System
Simulation**

THESIS

**PRESENTED AS A PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF**

**MASTER OF SCIENCE WITH MAJOR IN ELECTRONIC ENGINEERING
(ELECTRONIC SYSTEMS)**

BY

Miguel Angel Rios Gastelum

MONTERREY, N.L., MEXICO. MAY, 2009

© Miguel Angel Rios Gastelum, 2009

**A Transistor Level Model of a UHF RFID Tag for System
Simulation**

BY

Miguel Angel Rios Gastelum

THESIS

PRESENTED TO THE GRADUATE PROGRAM IN MECHATRONICS AND
INFORMATION TECHNOLOGIES

THIS THESIS IS A PARTIAL REQUIREMENT FOR THE DEGREE
OF MASTER OF SCIENCE WITH MAJOR IN

**ELECTRONIC ENGINEERING
(ELECTRONIC SYSTEMS)**

**INSTITUTO TECNOLGICO Y DE ESTUDIOS
SUPERIORES DE MONTERREY**

MONTERREY CAMPUS

MAY, 2009

**INSTITUTO TECNOLGICO Y DE ESTUDIOS
SUPERIORES DE MONTERREY**

MONTERREY CÀMPUS

**GRADUATE PROGRAM IN MECHATRONICS AND
INFORMATION TECHNOLOGIES**

The members of the thesis committee hereby approve the thesis of Name as a partial fulfillment of the requirements for the degree of Master of Science with major in

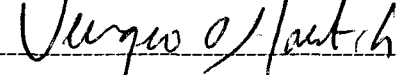
**Electronic Engineering
(Electronic Systems)**

Thesis Committee



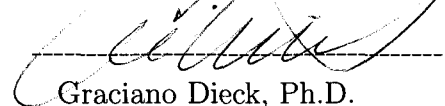
Alfonso Avila, Ph.D.

Thesis Advisor



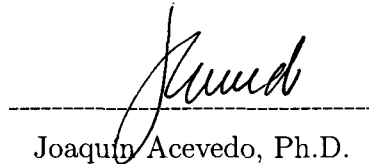
Sergio Omar Martínez, Ph.D.

Thesis Reader



Graciano Dieck, Ph.D.

Thesis Reader



Joaquín Acevedo, Ph.D.

Director of the Graduate Programs in Mechatronics and Information Technologies

May, 2009

To my parents

Acknowledgements

To god, for giving me his blessing and for this opportunity to grow.

To my parents, Maria del Rosario Gastelum and Miguel Angel Rios, for your love, teachings and unconditional support.

To my brothers, Carolina and Omar Rios, for believe in me and for your support.

To my advisor, Alfonso Avila Ortega, PhD., for your teachings, guidance and friendship.

To my friends, for your patience and help.

To Consejo Nacional de Ciencia y Tecnología (CONACYT), for its sponsorship during the realization of this thesis.

And to Instituto Tecnológico de Monterrey (ITESM), for giving me the opportunity to develop myself in this recognized institution.

MIGUEL ANGEL RIOS GASTELUM

Instituto Tecnológico y de Estudios Superiores de Monterrey

MAY, 2009

A VHDL-AMS Transistor Level Model of a UHF RFID Tag for System Simulation

Miguel Angel Rios Gastelum, M.S.

Instituto Tecnológico y de Estudios Superiores de Monterrey, 2009

Thesis Advisor: Alfonso Avila, Ph.D.

Abstract

The fabrication processes of Radio Frequency Identification Systems implies a large and an expensive development effort as well as very long testing periods. System modeling and cosimulation is an important approach for reduce the time to place a product into the market. This thesis presents a transistor level model of a UHF RFID tag, implemented with VHDL-AMS, for both (1)early detection of mixed signal design problems and (2)transistor level estimation of the power consumption in analog and digital subsystems. The transistor model used in the RFID tag model was implemented in VHDL-AMS and it was modeled according the characteristics of a long channel transistor. The power consumption was estimated using analog and digital components. Analog power consumption was calculated from the electric parameters of the transistor model, and digital power consumption from the switching at the output of the gates in the tag model. The functional simulation of the tag was 100% compatible with the implemented communication protocol, and the power consumption had a maximum improvement of 27.15% compared with similar research and development efforts.

Contents

Acknowledgements	i
Abstract	iii
1 Introduction	1
1.1 Problem Description	1
1.2 Objective	1
1.3 Justification	2
1.4 Contribution	2
1.5 Related Work	2
1.6 Thesis Organization	3
2 Theoretical Background	4
2.1 What is RFID?	4
2.1.1 History	5
2.1.2 Development	5
2.2 Parts in a RFID System	6
2.2.1 Tag	6
2.2.1.1 Active vs Passive Tags	7
2.2.2 Reader	7
2.2.3 Middleware	8
2.2.4 Host	8
2.3 RFID Principles	8
2.3.1 Magnetic Field	8
2.3.2 Magnetic Flux and Magnetic Flux Density	9
2.3.3 Inductance L	9

2.3.4	Mutual Inductance M	10
2.3.5	“k” Matching Coefficient	10
2.3.6	Faraday Law	11
2.3.7	The Creation of Electromagnetic Waves	11
2.3.7.1	Transition from Near to Far Field	12
2.3.8	Electromagnetic Waves Reflection	13
2.3.9	Encrypted Data Transfer	14
2.4	RFID Applications	14
2.4.1	Wal-Mart	14
2.4.2	E-ZPass	15
2.4.3	SpeedPass and Contactless Payment Systems	15
3	UHF Passive RFID Tag	17
3.1	Communication Protocol	17
3.1.1	Power-Up Mode	17
3.1.2	Addressing Mode	18
3.1.3	Reading Mode	18
3.2	Transponder Architecture	19
3.3	Simulated Reader Signal	20
3.4	The Rectenna	22
3.4.1	Rectifier Building Blocks	22
3.4.1.1	Clamping Circuit	23
3.4.1.2	Rectifier Circuit	23
3.4.1.3	The Voltage Doubler	23
3.4.1.4	Full-wave Greinacher Rectifier	24
3.4.2	Rectifier	26
3.4.3	Power-on-Reset	26
3.4.4	Detector, Data Slicer and Decoder	27
3.4.5	Shift Register and Logic	29
3.4.6	System Clock	30
3.4.7	Modulator	30
3.4.8	Current Reference	31
3.4.9	Antenna Considerations	31

4	Transistor Model	33
4.1	MOSFET Model	33
4.1.1	Static Model of MOSFET	34
4.1.1.1	Cut-off Region	34
4.1.1.2	Linear Region	34
4.1.1.3	Saturation Region	34
4.1.2	Capacitances	35
4.1.2.1	Overlap Capacitance	35
4.1.2.2	Gate to Channel Capacitance	36
4.1.2.3	Diffusion Capacitance	36
4.1.3	VHDL-AMS Transistor Model	36
4.1.4	Functional Tests	37
5	Results	41
5.1	Module's Signals	41
5.1.1	Incident RF Signal	41
5.1.2	Rectifier	42
5.1.3	Power-on-Reset	46
5.1.4	Detector	48
5.1.5	Comparator	50
5.1.6	Decoder	50
5.1.7	Current Reference	50
5.1.8	System Clock	50
5.1.9	Control Unit and Shift Register	52
5.1.9.1	Right Address Signals	52
5.1.9.2	Wrong Address Signals	52
5.2	Analog Power Consumption	54
5.2.1	Incident RF Power	57
5.2.2	Rectifier	57
5.2.3	Power-on-Reset	58
5.2.4	Detector	58
5.2.5	Comparator	60
5.2.6	Current Reference	60
5.2.7	Total Analog Power Consumption	60

5.3	Digital Power Consumption	62
5.3.1	Dynamic Power	62
5.4	Total Power Consumption	65
5.5	Comparison with Related Works	68
6	Conclusions and Future Work	69
6.1	Conclusions	69
6.2	Future Work	69
A	VHDL-AMS Code	71
A.1	NMOS Transistor Code	71
A.2	Testbench for The NMOS Transistor	73
A.3	PMOS Transistor Code	78
A.4	Testbench for The PMOS Transistor	80
A.5	Ascending DC Source	85
A.6	DC Source	87
A.7	Diode Code	88
A.8	Resistor Code	89
A.9	Capacitor Code	89
A.10	Flip Flop D Code and Test Bench	90
A.11	Analog to Digital Converter Code	93
A.12	System Clock Code	94
A.13	Signal of The Reader	96
A.14	Rectifier Code	99
A.15	Power-on-Reser Code	103
A.16	Detector Code	105
A.17	Comparator Code	106
A.18	Shift Register Logic Code	111
A.19	Current Reference Code	120
A.20	Tag Testbench in The Wrong Address Case	123
A.21	Tag Testbench in The Right Address Case	126
B	Signals of The Blocks	131
C	EPFL EKV 2.6 Transistor Code in Matlab	142

Bibliography

145

Vita

148

List of Figures

2.1	RFID System	4
2.2	E-ZPass Example	15
2.3	SpeedPass Example	16
3.1	Decoder State Chart	19
3.2	Tag Operational Principles [1]	19
3.3	Tag Architecture	20
3.4	Waveform of the Reader's Signal [1]	21
3.5	Typical Rectenna Schematic	22
3.6	Diode Clamp Circuit	23
3.7	Basic Rectifier Circuit	23
3.8	Voltage Doubler Circuit	24
3.9	Full-Wave Voltage Doubler Circuit	24
3.10	Full-wave Modified Greinacher Circuit [2]	25
3.11	Rectifier	25
3.12	Rectifier Equivalent Circuit [1]	26
3.13	Power-on-Reset Circuit [1]	27
3.14	Detector, Data Slicer and Decoder Circuit [1]	28
3.15	Control Unit Circuit [1]	29
3.16	Parallel Modulator Schematic	30
3.17	Current Reference Schematic	31
4.1	MOS Capacitances	35
4.2	VHDL AMS Schematic Model	37
4.3	I_{DS} vs V_{DS} NMOS	37
4.4	I_{DS} vs V_{DS} NMOS	38
4.5	I_{DS} vs V_{DS} PMOS	39

4.6	I_{DS} vs V_{GS} NMOS	39
4.7	I_{DS} vs V_{GS} PMOS	40
5.1	RF Signal Received by the Transponder	42
5.2	Rectifier Implemented in This Simulation	43
5.3	Positive Side Rectifier's Signals	44
5.4	Negative Side Rectifier's Signals	45
5.5	Power-on-Reset Simulation Schematic	46
5.6	Power-on-Reset Analog Signals	47
5.7	Power-on-Reset Digital Signals	48
5.8	Detector Signals	49
5.9	Comparator Signals	51
5.10	Decoder Signals	52
5.11	Current Reference Signals	53
5.12	Current Reference Schematic	54
5.13	Control Unit, Shift Register and Data Enable Signals in the Right Address Case	55
5.14	Control Unit, Shift Register and Data Enable Signals in the Wrong Address Case	56
5.15	Incident RF Power	57
5.16	Power Dissipated By The Rectifier Block	58
5.17	Power Dissipated By The Power-on-Reset Block	59
5.18	Power Dissipated By The Detector Block	59
5.19	Power Dissipated By The Comparator Block	60
5.20	Power Dissipated By The Current Reference Block	61
5.21	Total Analog Power Dissipated	61
5.22	Gates Switching	64
5.23	Right Address Case Power Consumption	66
5.24	Wrong Address Case Power Consumption	67
B.1	Rectifier's Signals 1	131
B.2	Rectifier's Signals 2	132
B.3	Rectifier's Signals 3	133
B.4	Rectifier's Signals 4	134
B.5	Power-on-Reset's Signals	135

B.6	Detector's Signals	136
B.7	Comparator's Signals	137
B.8	Decoder's Signals	138
B.9	Current Reference's Signals	139
B.10	SR and Control's Signals Right Address Case	140
B.11	SR and Control's Signals Wrong Address Case	141

List of Tables

3.1	Characteristics of the Reader's Signal	21
3.2	Characteristics of the System Clock	30
4.1	Channel Capacitance in the Different Regions of Operation	36
5.1	Percentage of Total Power Consumption	62
5.2	Switching Factor and Power Consumed by Each Gate In The Right Address Case	63
5.3	Switching Factor and Power Consumed by Each Gate In The Wrong Address Case	65
5.4	Dynamic Power Consumption Comparison	65
5.5	Comparison of Results with Works [3] and [4]	68
5.6	Comparison of Results with [5]	68

Chapter 1

Introduction

1.1 Problem Description

The development and testing of RFID systems requires some time to integrate digital and analog components. Once the RFID system is fabricated and ready for testing, the detection of design errors slows down the time needed to deliver the system to the market resulting in monetary losses. The development time for RFID systems could be shrunk with adequate system models that allow the designers to have better system simulation methodologies for faster error detection. Most of the models of RFID components describe the behavior at the functional level. These models lack of the required granularity to provide accurate estimations of power consumption.

1.2 Objective

The main purpose of this research is to reduce the development times of RFID systems and detect errors in early stages of the design. The creation of a transistor level model of an RFID tag in VHDL-AMS is proposed to accomplish those goals. The idea is to obtain power consumption estimations from the transistor level model. A modular architecture with mixed signals interconnections between different blocks is intended. The proposed RFID architecture could add extra components in future research projects.

1.3 Justification

Today, the design of Radio Frequency Identification (RFID) systems needs a short development time, to reach the market as soon as possible without absorbing extra costs of NRE (non-recurring engineering) hours. The fast time to market can be achieved with system models design and cosimulations. System models allow the cosimulation between analog and digital parts of the system before assuming the high costs of fabrication of the chip. Cosimulation shows the behavior of the System and important information can be obtained from it, like power consumption and the behavior of the systems signals.

1.4 Contribution

The contributions of this thesis are:

1. A MOSFET transistor model using VHDL-AMS
2. The modeling and cosimulation of a complete mixed signal RFID tag using VHDL-AMS
3. Accurate measurements of power consumption at transistor level for the analog and digital components

1.5 Related Work

An RFID antenna model in VHDL-AMS was developed and cosimulated. This effort focuses on studying the power absorption at the RF link [27]. References [3] and [4], show the implementation of a low power active RFID using both, an FPGA and a SoC. They also measure the power consumption directly from the physical components. A new contactless smartcard with an integrated 8-bit microcontroller is described in [6]. The smartcard was implemented in a SoC and demonstrates that the integrated power reception and management, Radio Frequency communication and signal processing are possible in SoC with its characteristics. Reference [7] presents an RFID system with a fully integrated transponder fabricated in standard CMOS 0.18 μ m process. It uses a near field coupling for enable the on chip tag's antenna. An architecture for a RFID tag

that includes a microprocessor is presented in [8]. Some novel low power technologies for low power microprocessor are adopted, the power consumed by the subsystems is analyzed in the chip implementation of the tag. A novel simulation concept for functional verification based in cosimulation is presented in [5], it combines system level models of the RFID system with hardware level description of components, and with this defines the energy consumption of the tag. This thesis estimates the power consumption of an RFID tag before fabrication using the cosimulation methodology. The near field coupling is considered in the cosimulation for the tag power up and no physical implementation is needed to verify the behavior of the tag.

1.6 Thesis Organization

Chapter 1 presents the problem description, objective, justification, contribution and related work of this Thesis. Chapter 2 presents the necessary Theoretical Background in RFID systems. Chapter 3 covers the Tag's Architecture used in the simulations of this research work. Chapter 4 presents the transistor model implemented in the Architecture. Chapter 5 shows the results obtained from the simulations, and finally Chapter 6 presents conclusions and future research possibilities.

Chapter 2

Theoretical Background

2.1 What is RFID?

All radio transmissions that contain any kind of identification information are considered as Radio Frequency Identification (RFID). RFID involves devices that can exchange data using radio signals. The usual system involves a small tag that identifies a specific object. The specific object is identified by sending a radio signal to the tag, this one interprets it and returns the identification information stored. Sometimes this process can be as complex as using cryptographically encoded data, sending it via satellite and store it in a database[9].

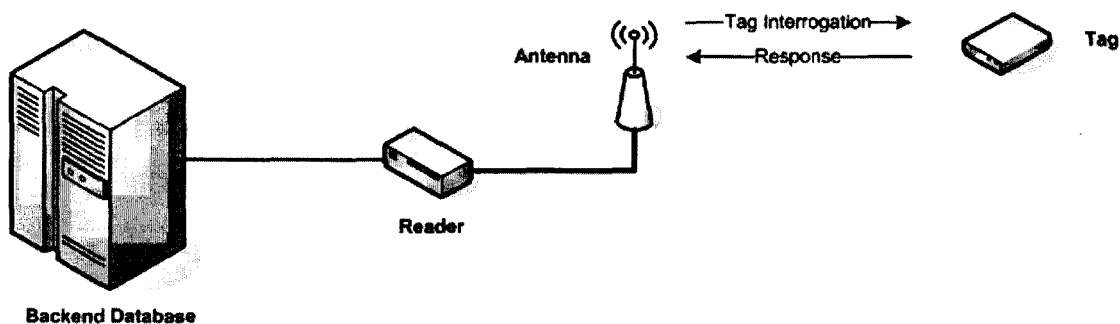


Figure 2.1: RFID System

2.1.1 History

In the year of 1906 Ernst F. Alexanderson demonstrate with his work the first continuous wave radio generation of radio signals, this marked the beginning of the modern radio communications. Also the early 20th century was considered the birth of the RADAR and it developed extensively in the World War II. RFID was conceived as a combination of the previous technologies. One of the first works talking about RFID was the landmark paper “Communications by Means of Reflected Power”, by Harry Stockman, published in 1948. This work talked about the research needed to develop the reflected power communications and that a lot of research was needed before this technology was widespread. In fact, 30 years had to pass before the vision of this man was possible, the development of the transistor, the integrated circuit, the microprocessor, communication networks had to be done before. The 1930’s and 1940’s was the era when the radar had much technical developments and this brought exploring RFID techniques. One of the first explored applications was the “long range transponders systems” for identification of enemy or foe air crafts in the World War II[10].

2.1.2 Development

The 1960’s developments created the bases for the 1970’s RFID explosion. In 1964 R.F. Harrington studied the electromagnetic principles related to the RFID technology and published his work in a paper named ”Theory of Loaded Scatterers”. This decade was also the beginning of RFID in the commercial activities with companies like sensormatics being founded and applications like ”Electronic Article Surveillance” being developed[11].

The 1970’s was a decade of RFID intensive research, everyone was working on it, getting important advances. The use of passive RFID tags was achieved in the work named ”Short-Range Radio Telemetry for Electronic Identification Using Modulated Backscatter” developed by Los Alamos Cientific Laboratory. In that work they talked about completely passive tags with operational ranges of about tens of meters. Large companies like Raytheon’s Raytag and RCA were also involved in the development of RFID technology. RFID as a lot of other electronic technologies got benefits from the development of the low-power CMOS logic circuits since this permitted RFID systems to use smaller tags with less power consumption which was ideal because they were intended to be passive.

The 1980's was an implementation era, there was a lot of interest in develop applications for the close range detection of animals, toll roads and industry automatization. The arrival of the personal computer (PC) was an important aid to manage RFID data, so it could be implemented everywhere a PC was present. By this time tags were built with CMOS technology and the EEPROM became the memory of choice, this lead to important size reductions in the tag and increase the functionality.

The 1990's was an important time for RFID, electronic toll collection was implemented all around the United States, it allowed vehicles to access different areas, like parking lots, highways and stadiums, using a single tag in each vehicle. The first highway with these characteristics was founded in Oklahoma, and then states like Texas and Georgia followed. A single protocol for all these highways was created. A notable application was the one developed by Texas Instruments, named TIRIS, which allowed an automobile user to control the start of the engine[12]. More countries join the use of RFID technologies, like Australia, China, Mexico and Canada. A great number of companies entered the marketplace because of the growing interest on items management and the opportunity of work along bar codes.

In the 21st century the main preoccupations of RFID is to create a standard to unify all the areas of these technologies. At this point two organizations are taking the lead, EPCGlobal and ISO. A lot of interest is also focused in the development of telematics, article tracking, supply chain management and mobile commerce.

The future looks great for RFID but it still needs improvements in other areas as software development, privacy policies and infrastructure in RFID systems. But nothing great was achieved without enthusiasm, so there a lot of that need to be developed before RFID get the best it can [10].

2.2 Parts in a RFID System

2.2.1 Tag

RFID units are in the radio devices class also known as transponders. A transponder is a combination of a transmitter and a receiver It receives a specific radio signal and returns a reply. In the simplest implementations, the transponder just listen to a specific radio signal and its reply consists in just backscattered radio signal. In more complex cases the reply can consists of a string of letters or numbers. Transponders

used in RFID systems are commonly called “tags”, “chips” or “labels”. As a general rule, a RFID tag contains the following items:

- Encoding/Decoding Unit
- Memory
- Antenna
- Power Supply
- Communication Control

Depending on the type of tag and the application, the amount of information carried can change, from a few bytes to a few megabytes[13].

2.2.1.1 Active vs Passive Tags

Tags fall into two categories: “active” and “passive”. Passive RFID tags do not contain a power supply, they get their energy from the electromagnetic signal provided from the reader. To obtain the power from the reader they need to use an electromagnetic property known as “Near Field”. Because this property the transponder can obtain enough power to send a response. Active RFID tags have their own power supply. Since they have power all the time, they can transmit and receive on their own without having to be in the near field. They can reach larger distances due to their power independence [14].

2.2.2 Reader

The second more important component is the reader, that is also a transceiver. His role is to interrogate a tag and receive data from it. It can contain an antenna as an integral or separated part of it, it can also have other elements like: RS-232 serial port, Ethernet jack, cryptographic encoding or decoding, a power supply battery or a communication control circuits. The reader retrieves the information contained in the RFID tag, and can store it locally or send it to a database.

2.2.3 Middleware

The middleware manages the reader and the data received from the tag and has the purpose of store the data in a database. The middleware is in the middle of the communication between the database and readers and it can perform another functions as filter the information.

2.2.4 Host

The host can also be known as the database and it can be a standard commercial database such as SQL, MySQL, Oracle, etc. Depending on the application the host can be a PC or multiple mainframes networked together via global communication systems[14].

2.3 RFID Principles

The major part of the RFID systems operates under the inductive coupling principle, therefore it is necessary to understand the magnetic phenomenon to have a clearer idea on how power and data are transmitted from the reader to the tag in a RFID system.

2.3.1 Magnetic Field

Every moving charge in a conductor is associated to a magnetic field. The intensity of the magnetic field can be described by the equation of magnetic field strength “H”. In a general form it can be say that the sum of the currents in a closed curve is equal to the contour integral of the magnetic field within it.

$$\sum I = \oint \vec{H} \cdot \vec{ds} \quad (2.1)$$

Equation 2.1 can be used to calculates the magnetic field strength. In a straight conductor the field strength “H” along a circular flux line at a distance “r” is constant.

$$H = \frac{1}{2 \cdot \pi \cdot r} \quad (2.2)$$

The path field of strength “H(x)” in conductor loops or “short cylindrical coils” are used to generate the magnetic alternating field in the write/read devices of RFID systems

working under the inductive coupling principle. An in depth analysis shows that when you move the measurement point away from the center of the coil the field strength decrements in proportion with the distance. Equation 2.3 calculates the path field strength along the “x” axis or around the coil:

$$H = \frac{I \cdot N \cdot R}{2 \cdot \sqrt{(R^2 + x^2)^3}} \quad (2.3)$$

where “N” is the number of winding of the coil, “R” is the circle radius “r”, “x” is the distance from the center of the coil in the “x” direction. The following boundaries have to be applied: $d \ll R$ and $x < \frac{\lambda}{2\Phi}$.

2.3.2 Magnetic Flux and Magnetic Flux Density

If a soft iron core is introduced in the center of a coil, leaving everything else equal, the magnetic field strength will keep constant but the flux density, the total number of flux lines, will increase, and this is fundamental for the force generation. The magnetic flux ϕ is the total number of lines of magnetic flux that passes through the inside of a cylindrical coil. Magnetic flux density “B” is a variable related to the area “A”. Magnetic flux is expressed as:

$$\Phi = \int \int_s \vec{B} \cdot d\vec{s} \quad (2.4)$$

In the special case where the surface “S” is a planar surface with area “A”, and the magnetic field is constant with magnitude “B”, the formula simplifies to:

$$\Phi = B \cdot A \cdot \cos \theta \quad (2.5)$$

where θ is the angle between “B” and the surface normal to “S”.

The relationship between the density flux and field strength is given by:

$$B = \mu \cdot H \quad (2.6)$$

Where μ is a material dependent parameter called the permeability.

2.3.3 Inductance L

A magnetic field and a magnetic flux are both generated around a conductor of any shape but it will particularly intense in a loop form conductor. There are conductor

spirals and each of them contributes on the same proportion to the total flux, thus:

$$\psi = \sum_N \Phi_N = N \cdot \Phi \quad (2.7)$$

The ratio of the flux that arises in an area enclosed by a current, to the current in the conductor that encloses it is named inductance L. By definition its formula is:

$$L = \frac{\psi}{I} = \frac{N \cdot \Phi}{I} \quad (2.8)$$

Inductance is one of the variables of conductor loops or coils. It depends of the material properties, the space that the flux flows through and the geometry of the conductor.

2.3.4 Mutual Inductance M

If a second conductor loop is placed near of the first conductor loop and for which a current is flowing, then this will be subject to a proportional of the total magnetic flux flowing through the first conductor loop. The two circuits are coupled by this partial flux. The magnitude of this coupling flux depends on the geometric dimension of both conductor loops, their positions relative to each other and the magnetic properties of the medium. The definition of self inductance is the ratio of the partial flux from the second conductor loop to the first one to the current in the first conductor loop.

$$M_{21} = \frac{\psi_{21} \cdot (I_1)}{I_1} = \oint_{A_2} \frac{B_2 \cdot (I_1)}{I_1} \cdot d \cdot A_2 \quad (2.9)$$

Similarly is a mutual inductance in the inverse way and the next relationship applies:

$$M = M_{12} = M_{21} \quad (2.10)$$

Mutual inductance describes the coupling of two circuits through a magnetic field. Inductively coupled RFID systems are based on this principle.

2.3.5 “k” Matching Coefficient

We talk about a coupling between two conductor loops that are near one another, now the matching coefficient can tell how big is this coupling, and depending on their geometric the following applies:

$$k = \frac{M}{\sqrt{L_1 \cdot L_2}} \quad (2.11)$$

This coefficient always varies between $0 \leq k \leq 1$.

- $k = 0$ for full decoupling
- $k = 1$ for total coupling, both conductor loops are subject to the same magnetic flux

In practice, coupled conductor loops can operate with "k" coefficients that may be as low as 0.01.

2.3.6 Faraday Law

A change in the magnetic flux Φ generates an electric field strength " E_i ". This is described by "Faraday's law". The effect of the magnetic field depends on the material surrounding it. A special interest is given to the effect caused in a metal surface. An electric field is induced in the metal surface that causes free charge carriers to flow in the direction of the electric field. Currents flowing in circles are created and are named "Eddy currents", they work against the magnetic flux, which may damp it significantly near metal surfaces. The "Eddy currents" are undesirable in inductively coupled RFID systems, so they are reduced by using ferrite shielding in metallic environments. In its general form, the Faraday's law can be expressed as:

$$u_i = \oint \vec{E}_i \cdot d\vec{s} = -\frac{d\Psi(t)}{dt} \quad (2.12)$$

For a configuration of N windings, Faraday's law can also be expressed as $u_i = N \cdot \frac{d\Psi}{dt}$. As a RFID example of the Faraday's law, let's suppose that we have two conductor loops in a near distance between them and a time variant current is applied through one of them. This current will generate a time variant magnetic flux and a voltage in both conductor loops. In RFID systems, one conductor loop would be the reader antenna while the other the transponder antenna. The reader antenna induces a voltage and also a current in the transponder side due to the resistance of the conductor and the resistance of the memory, this current also generates a magnetic field that opposes the magnetic flux generated by the reader's antenna[15].

2.3.7 The Creation of Electromagnetic Waves

A varying magnetic field in space induces an electric field with closed field lines. The electric field varies in time and surrounds the magnetic field. Due to the variation of the electrical field a magnetic field in space is generated and it also surrounds the electric

field and varies over time. Because of the mutual dependence of the time varying fields there is a number of effects on the magnetic and electric fields.

Radiation can only occur when a finite propagation speed is given for the electromagnetic field, which prevents a change in the voltage of the antenna from being followed immediately by the field in the vicinity of the charge. Because this radiation limitation, even the alternating voltage's zero crossover, the field lines from the previous half wave remaining in space cannot end at the antenna, but close into themselves forming eddies. Eddie currents are formed in the opposite direction of the ones formed in the next half wave propelling them and thus propelling the energy stored in the field away from the emitter at the speed of light. The electromagnetic field is interlinked with the electrical field that propagates at the same time. When a certain distance is reached, the fields are released from the emitter, this point represents the beginning of the electromagnetic radiation. At high frequencies, the radiation is more effective because the magnetic radiation takes place near the emitter, where the strength of the field is high, that means small wavelengths. The wavelengths can be calculated as:

$$\lambda = \frac{c}{f} \quad (2.13)$$

In the far field exists a relationship between the electric and the magnetic field, both fields are linked by the field characteristic impedance:

$$Z_F = \frac{E}{F} = \sqrt{\mu_0 \varepsilon_0} = 376.73 \Omega \quad (2.14)$$

Electromagnetic waves propagates quasi-optically in the far field (free space). The field strength falls as the distance from the emitter increase. Equation 2.14 describes the free space attenuation from a source to a point at a distance d:

$$a_f/dB = -147.6 + 20 \log(d) + 20 \log(f) \quad (2.15)$$

2.3.7.1 Transition from Near to Far Field

The primary magnetic field begins at the antenna and induces electric field lines in space. This area is known as near field. At the distance of $\lambda/2\pi$ the electromagnetic field separates from the antenna and wanders into space as an electromagnetic field. This separated electromagnetic field is named far field. A separated electromagnetic wave have no direct effect in the antenna from which it was generated. For inductively

coupled RFID systems this means that when the far field begins, a transformer-type coupling is no longer possible. The radius $r_f = \lambda/2\pi$ around the antenna represents an insurmountable range limit for inductively coupled systems [16].

2.3.8 Electromagnetic Waves Reflection

An electromagnetic wave emitted by an antenna can encounter various objects. Part of the energy that reaches the object is absorbed by it, the other part is scattered in many directions with different intensities. A small part of the scattered energy finds its way back to the emitter. RADAR technology uses this reflection to measure the distance and position of distant objects. In RFID systems, the reflection of electromagnetic waves is used for the transmission of data from the transponder to the reader. Reflective properties of objects generally increase with higher frequencies. RFID systems are used in frequencies ranges from 915 MHz to 2.45 GHz or higher. Let's consider this specific case, the high frequency generated by the reader is emitted in all directions, so the power density "S" at the transponder is:

$$S = \frac{P \cdot G}{4\pi \cdot R^2} \quad (2.16)$$

where "P" is the transmission power from the reader, "G" is the antenna gain of the transmitter and "R" is the distance between reader and antenna. The power reflected for the transponder is proportional to the power density:

$$P_2 = \sigma \cdot S \quad (2.17)$$

The radar cross section σ is the measure of an object ability to reflect electromagnetic waves. It depends of a range of parameters as object size, shape, material, surface structure but also wavelength and polarization. Equation 2.17 describes the power density that returns to the reader:

$$S_{back} = \frac{P \cdot G \sigma}{(4\pi)^2 \cdot R^4} \quad (2.18)$$

The reception power of the receiver antenna can be determined from its effective absorption area $A_w = \lambda^2 x G / 4\pi$:

$$P_{back} = \frac{P \cdot G \cdot \sigma \cdot A_w}{(4\pi)^2 \cdot R^4} = \frac{P \cdot G^2 \cdot \lambda^2 \cdot \sigma}{(4\pi)^3 \cdot R^4} \quad (2.19)$$

From the previous section it can be concluded that the reading range of a backscattered RFID system is proportional to the fourth root of the transmission power of the reader[17].

2.3.9 Encrypted Data Transfer

In RFID, data transmission can suffer two types of attacks: an attacker can behave passively and try to eavesdrop into the transmission to get confidential information and use it on his benefit. In the other kind of attack, an intruder behaves actively to manipulate the transmission data and alter them to his own benefit. Cryptological procedures are used to protect the data transmission from both active and passive attackers. Modifying the data before the transmission takes place can confuse an intruder to avoid him from getting conclusions. Encrypted data transmissions always follow the same order, first the data to be sent is transformed into a cipher data using a secret key and a secret algorithm. Because the attacker does not know the secret algorithm nor the secret key, he cannot interpret the data and it is not possible to recreate the transmission data from the cipher data. The information is transformed back to its original form in the receiver using the secret algorithm and secret key. RFID systems use the same key to ciphering and deciphering so it is a symmetrical key procedure[15].

2.4 RFID Applications

2.4.1 Wal-Mart

In June 2003 the Wal-Mart CIO and vice president Linda Dillman made an announcement, later called a mandate, requiring Wal-Mart's top 100 suppliers to use RFID on cases and pallets of inventory shipped to the retailer by January 1, 2005. Wal-Mart has 9% of retail sales in the world so this was a big new for the RFID growth. This mandate tells the suppliers to use passive RFID tags in all the shipments sent to three of Wal-Mart's distribution centers in Texas. Once the deadline was met and the pallets with RFID started to arrive, Wal-Mart reported more than five million tag reads. The results on this mandate were:

- A 16% reduction in out-of-stock items because of the use of EPC tags
- Out-of-stock items are replenished three times faster using EPC instead of bar codes
- RFID-equipped stores were 63% more effective at replenishing out-of-stock items

A research firm named Sanford C. Bernstein & Co. estimated that Wal-Mart could save over \$8 billion annually once RFID is fully deployed in all of its locations[18].

2.4.2 E-ZPass

E-ZPass is a RFID-enabled payment system for the highways installed in about 25 states in the United States. It was conceived with the idea of prevent traffic congestion and delays in the roads. E-ZPass Interagency Group was created in 1991. In order to use this payment systems, drivers have to create a prepaid account with a check or a credit card and get in return an active RFID transponder that works at a frequency of 900MHz. Readers are placed in the traffic lanes. As the car passes through the designated E-ZPass lane in the toll place, the system detects the cars with the E-ZPass transponder associating them to their account and charging the toll cost. Despite the complains of the public because of the facility of the government to look at the information of the cars that use E-ZPass, this is one of the more successful deployments of RFID in a consumer facing setting, with more that ten years of commercial use[14].

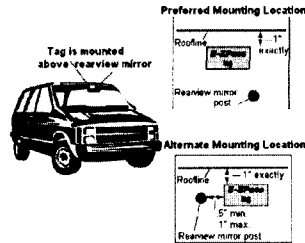


Figure 2.2: E-ZPass Example

2.4.3 SpeedPass and Contactless Payment Systems

This system consists on a card that can achieves electronic payment without having to slice the card, all what it needs is to be near a reader, usually a few inches, and the payment is done automatically. Exxon, MasterCard and Chase bank have some of this payment systems. Other variant of this contactless systems is the Near Field Communication. With this method a mobile phone can be waved in front of a reader

and the payment is completed. Contactless payment systems were introduced in 1997. All contactless payment systems contain passive RFID tags in different devices as credit cards, mobile phones or keys. These devices contain a cryptographically enabled RFID chip and an antenna, the points of sale have a reader installed. When is time to pay, the reader interrogates the tag and check the credit in the database, if the credit is approved the purchase is finished, charging the amount of money to the costumers account. Speedpass is one of the most successful RFID deployments because of the large number of people using it, despite some security problems related to the immunity of the credit card at the time of the transaction[19].



Figure 2.3: SpeedPass Example

Chapter 3

UHF Passive RFID Tag

This simulation is based on an ultra-low power consumption transponder model developed in [1], with some differences. This chapter shows the bases of the communication protocol used to establish the link between the reader and the transponder, the architecture of the transponder and the way each functional block works.

3.1 Communication Protocol

A special protocol is used in this application. It does not implement the available standards, like ISO 18000-6 or EPCGlobal, although it can be easily modified to be compliant with them. Every step of the protocol is controlled by the reader due to the limited intelligence that can be embedded in the tag. The sessions are split into three parts: power-up mode, addressing mode and reading mode.

3.1.1 Power-Up Mode

A reader emits the RF power to a certain area where one or more transponder are located. Any transponder, receiving the RF power, absorbs it using its antenna and converts it to DC using the rectifier. The DC current energizes the tag's subsystems. During this mode, a power-on-reset signal is generated, turning on all the circuitry in the transponders and putting them into the addressing mode.

3.1.2 Addressing Mode

In this mode, all the energized transponders wait for the reader to send data. The reader sends a string of data with a serial number in it to all the transponders. All the transponders receive the serial number and compare it with its own serial number. If the serial number matches the ID of one transponder, the transponder goes to the reading mode, as the others change to quiet mode. The requested transponder now modulates its input impedance to backscatter the reader's signal so it can detect its presence. The transponders in quiet mode wait for a power off or probably a new interrogation session. Under this protocol no collision problem are possible as only one transponder can hold a communication session with the reader at once. To send an address, the reader generates a series of $200ns$ interrupts in the RF signal. The duration of the interrupts should be short to avoid the transponder going off and triggering a new power-on-reset. The start bit is always a logical "1" and corresponds to only one RF interrupt. For the following bits, the reader sends two RF interrupts if the next bit has the same logical value than the preceding one and only one interrupt otherwise, as shown in Figure 3.1. The gap between two successive interrupts is about 200 ns. The time between two successive bits has to be greater than twice gap size. This protocol implementation is very simple and offers low power consumption.

3.1.3 Reading Mode

Only the transponder matches the requested ID jumps into this mode. The shift register is locked by the control logic. The clock turns on and off the modulator to backscatter the signal from the reader and acknowledges its presence. The reader realizes that an active transponder is in the interrogation area and sends short RF interrupts similar to the star bit. The tag receives and decodes the interrupts that work as a clock signal for the shift register whose output is the next bit of information. When the next bit is a logical "1" the transponder keeps the clock on and backscatters the readers signal. If the next bit is a logical "0", the clock is turned off and no backscattered signal exist, so the reader takes it as a "0". The way the decoder work is shown in Figure 3.1.

For each one of the interrupts sent by the reader in the reading mode, the transponder cycles the shift register and the reader can confirms that the serial number sent matches the received one. In this work, the returned data from the transponder to the reader is the transponder's ID, but, it can be modified to send other kind of informa-

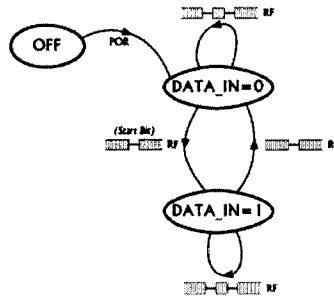


Figure 3.1: Decoder State Chart

tion. The signals shown in Figure 3.2 are a comprehensive example of a session for a three bits transponder, the same used in this work.

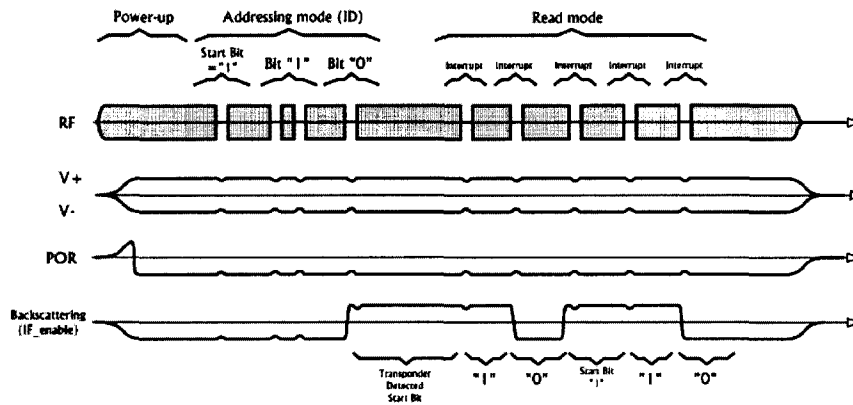


Figure 3.2: Tag Operational Principles [1]

3.2 Transponder Architecture

Figure 3.3 shows all the functional blocks of the transponder simulated in the present work. Passive transponders, as discussed in section 2.2.1.1, get their energy by rectifying the RF input in their antenna. This energy is, then, stored in capacitors to obtain a DC power supply. A power-on-reset signal is generated, with enough energy at the transponder, to wake up the transponder subsystems. Next, the transponder listens

to the data sent by the reader and decodes it. This architecture is simple compared to other communication systems. The only RF parts are the modulator, the envelope detector and the rectifier.

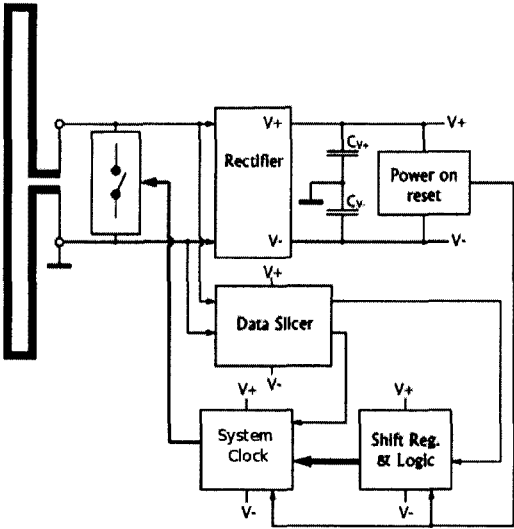


Figure 3.3: Tag Architecture

3.3 Simulated Reader Signal

The function that generates the signal from the reader, multiplies a $2.4GHz$ sinusoidal function and a train of pulses that contains the data to implement the protocol. The EIRP (Equivalent Isotropically Radiated Power) considered in the signal is of $4W$. The resulting signal is equivalent to the signal that a transponder would see in a given application that corresponds to the architecture of this transponder. The characteristics of the simulated reader are shown in Table 3.1.

Figure 3.4 shows the waveform of the signal in the simulation, and the match with the previously described communication protocol can be verified. The simulation of the complete reader is out of the scope of this job.

Table 3.1: Characteristics of the Reader's Signal

Type of Signal	Frequency	Duration of the Pulses	EIRP	Total Time Simulated
Sinusoidal	2.4GHz	200ns	4W	12.4μs

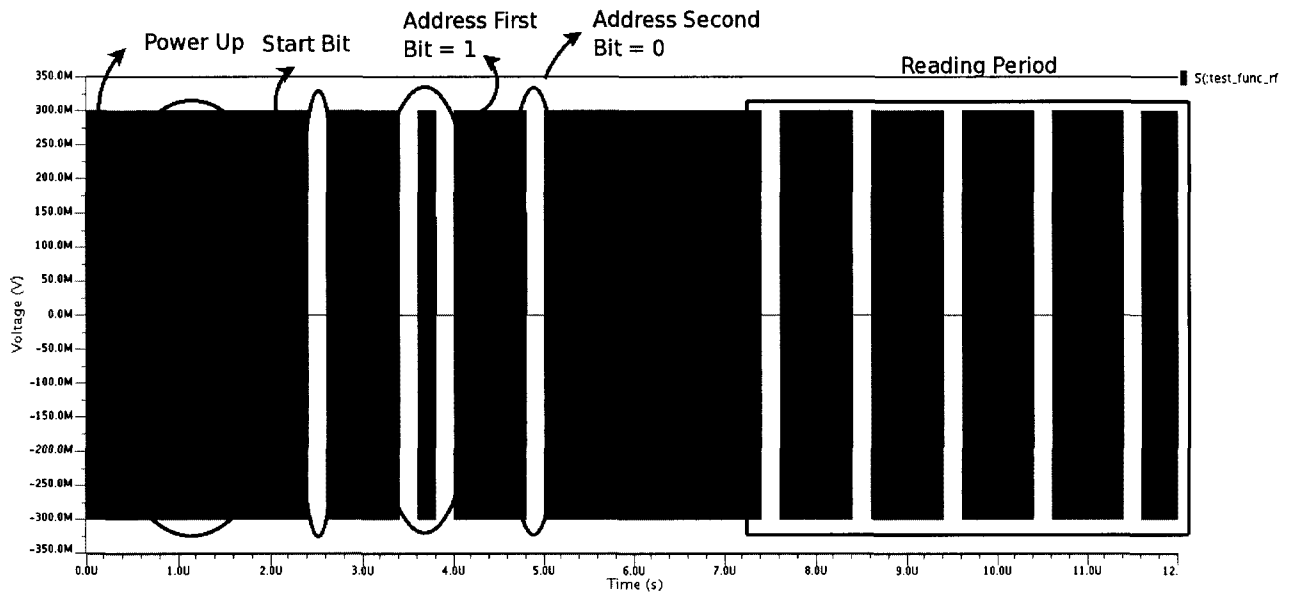


Figure 3.4: Waveform of the Reader's Signal [1]

3.4 The Rectenna

The rectenna is a special antenna that is used to convert microwave energy into DC electricity. A schematic of a typical rectenna is shown in Figure 3.5. The antenna captures the power from the medium and generates a voltage at the diode access. The latter rectifies the voltage to a DC current that charges the capacitor C at the rate RC .

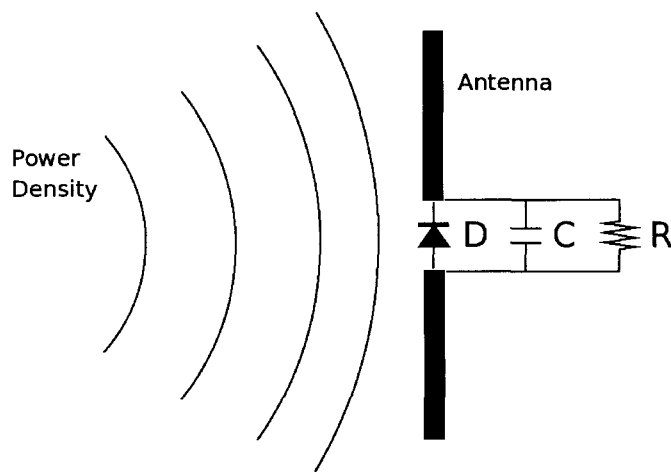


Figure 3.5: Typical Rectenna Schematic

The rectenna was thoroughly studied starting at the second half of the 20th century resulting in a high overall efficiency systems. With the arrival of the integrated circuit and low power technologies, new applications were possible. In the mid 1980's, RFID appeared in which an inductive or electromagnetic antenna was used for power transmission and communication[20]. All these applications used a block named rectifier, similar to the basic rectenna.

3.4.1 Rectifier Building Blocks

The rectifier is built upon two basic electrical circuits, the clamping circuit and the envelope detector circuit.

3.4.1.1 Clamping Circuit

The objective of the clamping circuit is to establish a DC reference for the output voltage by using a diode clamp, as shown in Figure 3.6.

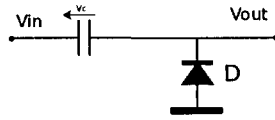


Figure 3.6: Diode Clamp Circuit

Conducting whenever the voltage at the output terminal of the capacitor goes negative, the circuit builds an average charge on the terminal that is sufficient to avoid the output from going negative. Positive charge is effectively trapped.

3.4.1.2 Rectifier Circuit

When a voltage is applied to the input of the circuit, the capacitor is charged until its voltage is equal to the maximum input voltage. If no resistor is connected in parallel with the capacitor, the voltage at the output never reduces in theory. In practice the leakage current of the capacitor induces a voltage drop at the output. If the input voltage is a sinusoidal, the capacitor charges every time its voltage is near the peak value and the mean output voltage is a little smaller than the peak amplitude.

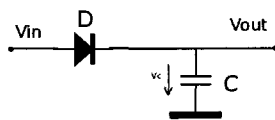


Figure 3.7: Basic Rectifier Circuit

3.4.1.3 The Voltage Doubler

The voltage doubler is obtained by cascading the diode clamping and the rectifier circuit. As seen in Figure 3.8, the circuit outputs a voltage that ideally is twice the amplitude of the input voltage. This circuit is a half wave voltage doubler because only

the positive peaks of the input are rectified. To take advantage of the negative and positive peaks in a sinusoidal signal the use of a full-wave rectifier is needed.

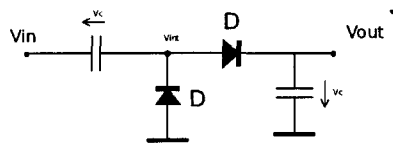


Figure 3.8: Voltage Doubler Circuit

The full-wave voltage doubler is just a mirrored circuit with respect to ground of the voltage doubler. In practice, the effects of the elements, diminishes the voltage multiplication. Thus, one has to cascade the circuit, and it results in a Greinacher cascade.

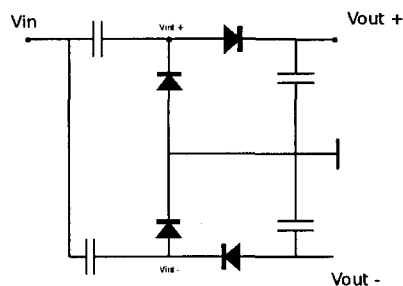


Figure 3.9: Full-Wave Voltage Doubler Circuit

3.4.1.4 Full-wave Greinacher Rectifier

The rectifier offers a good symmetry. It has a low power impedance, low capacitance losses along the RF path to the diodes and reduces the reflected harmonic content. The output voltage is equal to $4NV_{in}$, where N is the number of stages[2]. The Greinacher rectifier is described in detail because is the rectifier used in this design, and it is of great importance due its role as a DC voltage supplier.

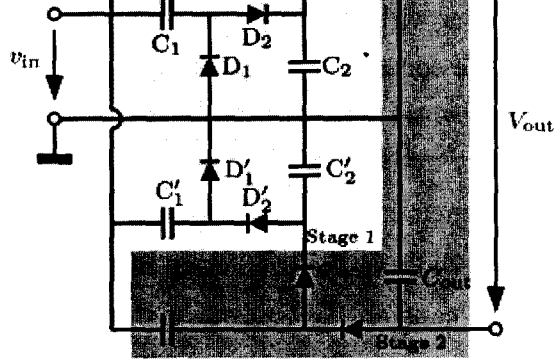


Figure 3.10: Full-wave Modified Greinacher Circuit [2]

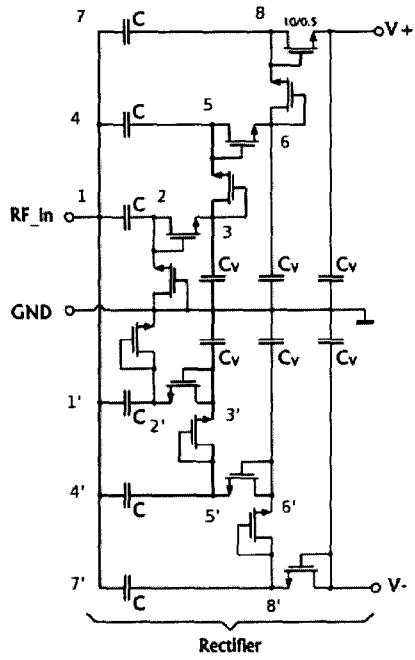


Figure 3.11: Rectifier

3.4.2 Rectifier

Figure 3.11 shows a three stages Greinacher rectifier. In this simulation the rectifier circuit is conformed by diodes with low threshold voltage, low reverse current and low parasitic capacitance. The considered fabrication material was silicon-on-sapphire because of its low parasitics, low threshold voltage and excellent high frequency performance [21]. The rectifier is a very important block in every remotely powered system. A detailed study of the implemented rectifier is given in [2] and [1].

The model in Figure 3.12 allows the prediction of the power needed to supply a given DC output current at a constant DC output voltage.

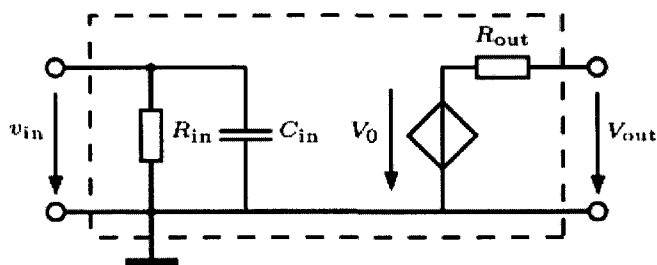


Figure 3.12: Rectifier Equivalent Circuit [1]

The global rectifier efficiency can be defined as:

$$\eta_0 = \frac{DCOutputPower}{IncidentRFPower} \quad (3.1)$$

3.4.3 Power-on-Reset

The power-on-reset circuit, as shown in Figure 3.13, is composed of a cross-coupled pair of transistor and a NOR gate and it is a non stable circuit. When the circuit is energized, one transistor takes advantage from the other. The NOR gate compares the two gate signals ensuring a power-on-reset signal for the logic blocks. The capacitor allows node *A* a slow charging avoiding parasitic oscillation.

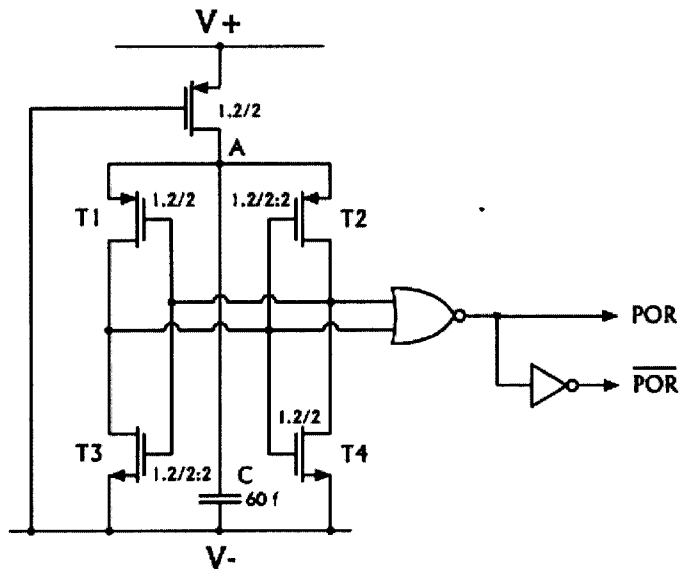


Figure 3.13: Power-on-Reset Circuit [1]

3.4.4 Detector, Data Slicer and Decoder

The detector is formed by an envelope detector and an averaging filter. The outputs are the signals V_{env} and V_{ref} that corresponds to the envelope and the rectified average of the received RF signal. Signal V_{env} is very important because it determines the time to do decoding. When the RF signal increments, the capacitor C_a charges to the RF envelope. When the signal falls, the capacitor discharges through the resistor R_a . Because of the inverse current of the diode, the capacitor C_a discharges sufficiently rapidly. Capacitor C_c and its touching diodes generates the signal V_{ref} . When the RF signal rises, C_c charges down to the half amplitude voltage of the RF envelope signal. When the RF signal falls, the retained voltage of the capacitor slowly reduces due to the inverse current of its parallel diodes that act as a parallel resistance. The data slicer, shown in the Figure 3.14, performs the signal regeneration. The signal G_n is tied to V^- during addressing mode. When V_{env} is higher than V_{ref} , $CK1$ rises to a logic "1" and the D flip-flop toggles. After the power-on-reset, the $Data_{in}$ signal is "0". When one interrupt in the signal is received, or the start bit is received, signal $Data_{in}$ toggles from "0" to "1". The start bit is then registered and the rest of the ID of the tag is received. To avoid the detection of undesired bits in the reading mode,

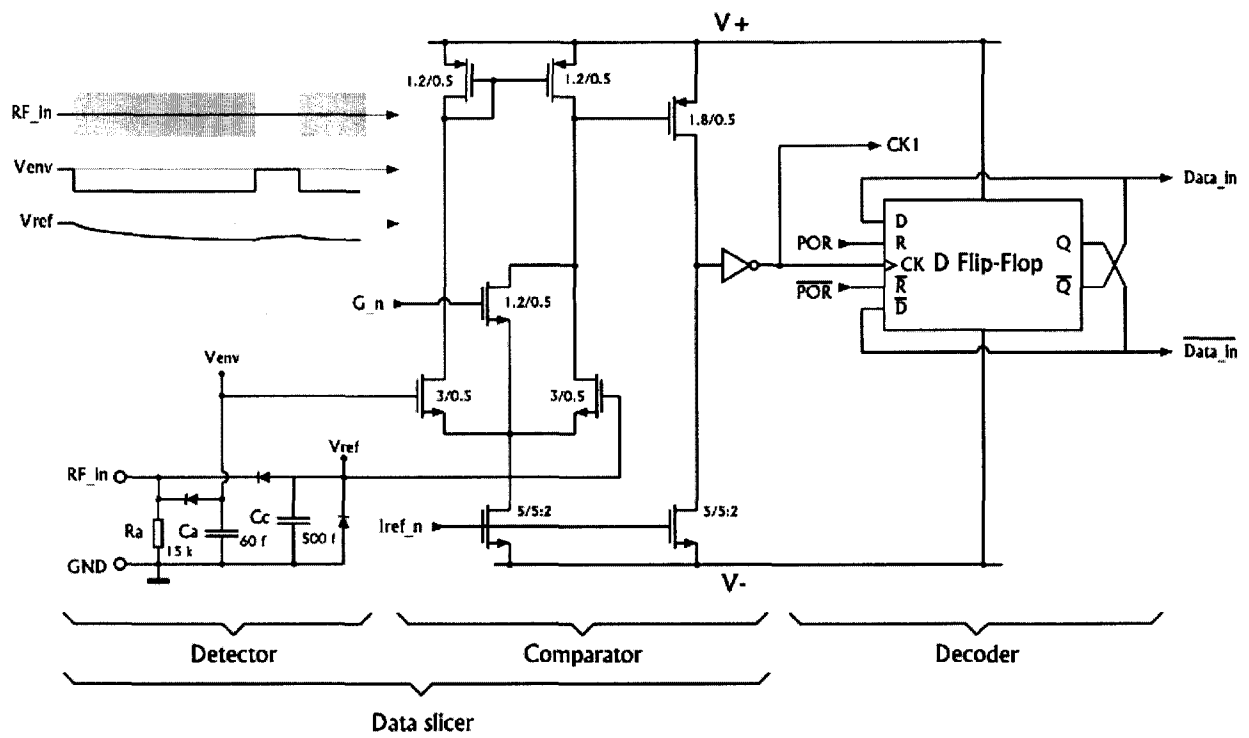


Figure 3.14: Detector, Data Slicer and Decoder Circuit [1]

the data slicer is inhibited when signal G_n goes high, and as a result $CK1$ is always low in reading mode.

3.4.5 Shift Register and Logic

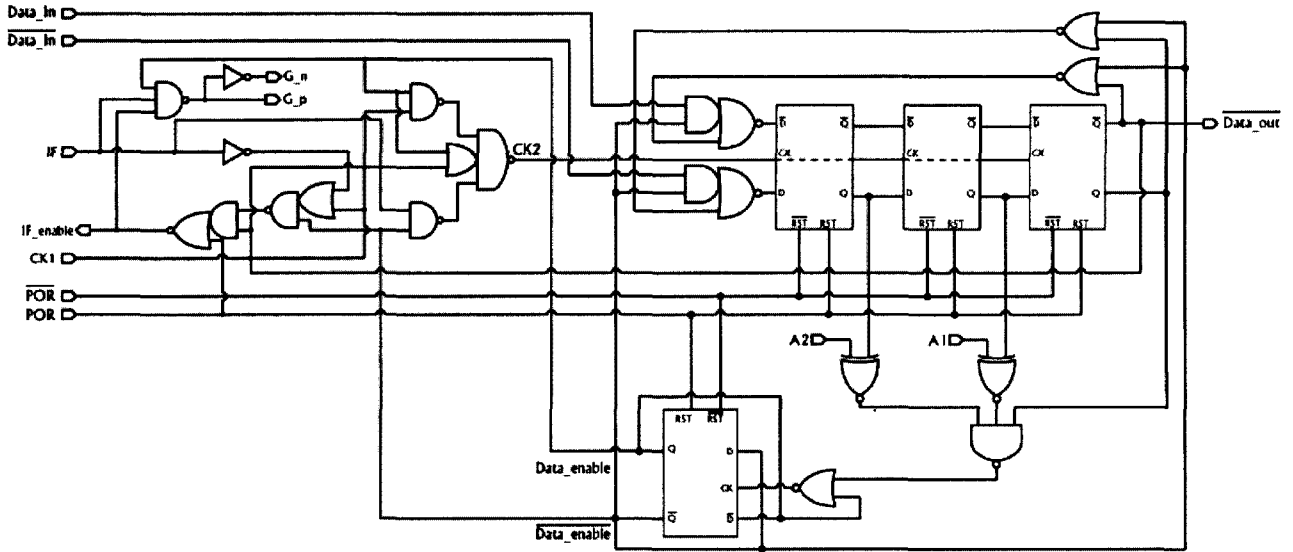


Figure 3.15: Control Unit Circuit [1]

The shift register and logic block is responsible of loading the incoming RF data from the reader, $Data_{in}$. The power-on-reset signal start the shift register to a logic “0”. In this work three bits shift register is used but it can be increased as desired or and EEPROM can be used to avoid the programming pads. In the addressing mode the signals $Data_{enable}$ and $Data_{out}$ are both low. $CK2$ is equal to the clock frequency. The received data, $Data_{in}$, is registered in the shift register on each rising edge of the $CK2$ signal. The address or ID of a transponder is set with the signals $A1$ and $A2$. When the received ID does not match the transponder’s ID, the signal $Data_{enable}$ stays low, $Data_{out}$ goes high and $CK2 = 1$, so no further bits are registered in the shift register. Here the transponder goes to the quiet mode. When a matching ID is received, $Data_{enable}$ goes on and the logic permits the communication with the reader. Signals G_n and G_p toggle the ASK modulator between short and open configurations. When the transponder is in the reading mode $CK2 = \bar{CK1}$ the shift register operates

in closed loop, its output goes directly to its input. At each rising edge of $CK2$, the shift register shifts its bit to the right.

3.4.6 System Clock

The transponder needs a clock to synchronize the control logic and the shift register. In [22] a Intermediate Frequency Oscillator is used to achieve this task. In this work, a digital clock is embedded instead of the complete oscillator model. The digital clock, simplifies the design and provides an accurate operation frequency. After testing the transponder operation with the clock, it can be concluded that the replacement is possible with no functionality penalty.

Table 3.2: Characteristics of the System Clock

Frequency	Duty Cycle
1MHz	0.1

3.4.7 Modulator

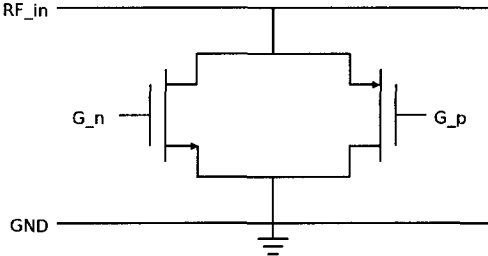


Figure 3.16: Parallel Modulator Schematic

The modulator is the key block for the backscattering communication of the transponder. It is responsible for the tag-reader communication by modulating the impedance presented in the transponder’s antenna to reflect the modulated power. To maximize the operating range, a low duty-cycle pseudo-PSK RF modulation is

implemented[1]. Because of this, the modulator can be realized by using just a simple switch. When the switch is open, the reflected power is equal to the difference between the available power from the antenna and the absorbed power. It is dependent of the transponder power consumption. The switch is realized using two MOS transistors in parallel, both controlled by the signals G_n and G_p .

3.4.8 Current Reference

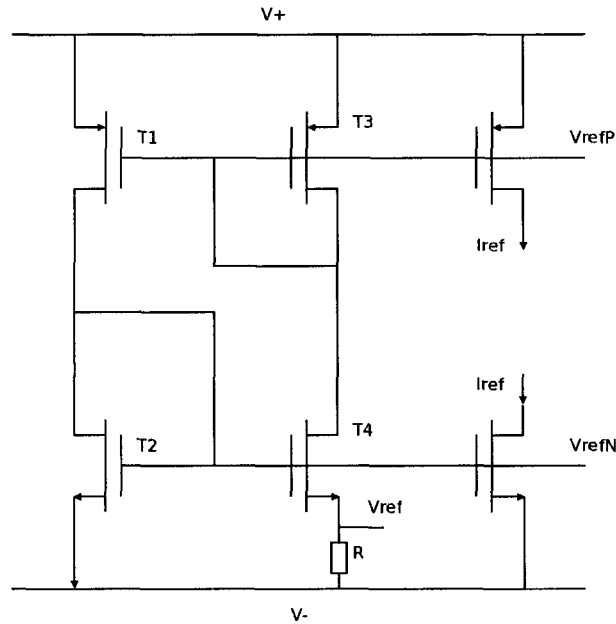


Figure 3.17: Current Reference Schematic

The current reference block is used to bias the current used in the different blocks in the circuit. It consists of two PMOS and two NMOS transistors. The transistors $T2$ and $T4$ have to work in weak inversion mode. This structure is specially suited for ultra low power applications because a very low current can be obtained by choosing a high value R .

3.4.9 Antenna Considerations

The input impedance of the transponder is mainly given by the rectifier. The input impedance is composed by an imaginary part caused by of the parasitic capacitance and

by a real part given by the output current consumption. In this work an optimum power transmission from the reader to the transponder is considered, which means that the antenna has a high radiation resistance [2], [23] and an optimum power matching with the transponder circuit. The antenna considered is the 3-wire folded dipole antenna because it helps to raise the value of the radiation resistance. The antenna and power matching strategies goes further from the scope of this work and it could be proposed as a future work.

Chapter 4

Transistor Model

Modeling and simulation are an important part in the integrated circuits design flow. Modeling the systems on integrated circuits provides a better understanding of the behavior of the chip and the models can be used to create simulations and detect errors even before the chip is fabricated, avoiding the costs of fabrication. During system simulations, the equations that govern the behavior of the system's elements and their interactions are solved by a simulator. A system can be modeled at a transistor level for simulation with more real results. The transistor models cover a large number of linear and non-linear equations that demand large computational times. In order to reduce the time and complexity of the simulations, the compact models are suited for transistor level simulations. The compact models use analytical equations that describe the behavior of the transistors and takes less time to simulate but at the cost of accuracy[24].

4.1 MOSFET Model

Metal Oxide Semiconductor Field Effect transistor is a four-terminal device consisting of source, drain, gate and bulk. The voltage applied to the gate terminal determines the quantity of current that flows between the drain and source terminals. There are two types of MOSFET devices: NMOS consists of n^+ drain and source regions embedded in a p-type substrate, the current flow between source and drain is formed by electrons, and PMOS is formed for p^+ drain and source regions in a n-type substrate, the current is carried by holes. This model considers the physical effects of a long channel transistor. For a submicron technology model the mobility effects due to vertical and lateral fields,

velocity saturation, short-channel effects as channel-length modulation effects should be considered.

4.1.1 Static Model of MOSFET

The characteristic of a MOS transistor can be divided in three main regions of operation: Cut-off region, Linear region and Saturation region.

4.1.1.1 Cut-off Region

When no gate voltage is applied and drain and source are connected to ground, the drain and source act as a back to back pn-junction which results in an extremely high resistance between source and drain, the current flowing across them is mainly zero.

4.1.1.2 Linear Region

When an applied gate voltage is greater than the threshold voltage and a voltage is applied between drain and source, a current starts to flow between drain and source. The threshold voltage represents the value of the gate voltage where the strong inversion occurs. The threshold voltage is given by

$$V_T = V_{TO} + \gamma \cdot (\sqrt{(-2\Phi_F) + V_{SB}} - \sqrt{2\Phi_F}) \quad (4.1)$$

γ is called the body effect coefficient and Φ_F is the Fermi Potential and V_{TO} represents the threshold voltage for a voltage from source to bulk equal to zero. The current in this region is given by

$$I_D = \mu_n \cdot C_{ox} \cdot \left(\frac{W}{L}\right) \cdot [(V_{GS} - V_T) \cdot V_{DS} - \frac{V_{DS}^2}{2}] \quad (4.2)$$

μ is the mobility, C_{OX} is the capacitance of the gate oxide and W, L are the width and length of the transistor.

4.1.1.3 Saturation Region

When the drain to source voltage is increased and $V_{DS} \geq V_{GS} - V_T$, the channel thickness is reduced gradually from source to drain, until pinch-off occurs at the drain terminal. The transistor is in the saturation region and the current between drain and source remains constant. The current in this region is given by

$$I_D = \frac{\mu_n C_{OX}}{2} \cdot \frac{W}{L} \cdot (V_{GS} - V_T)^2 \quad (4.3)$$

The last equation suggests that the current remains constant and independent of the voltage applied to the terminals. This is not completely true. The effective channel length is modulated by the voltage applied between drain and source, when this voltage increases the depletion region grows and the effective channel reduces. The next equation describes better the current behavior

$$I_D = I'_D \cdot (1 + \lambda \cdot V_{DS}) \quad (4.4)$$

I'_D is the current expression derived earlier and λ is the empirical parameter called the channel length modulation [25].

4.1.2 Capacitances

The MOS transistor have some parasitic capacitances between its terminals due the fabrication process. The capacitances can be seen in Figure 4.1. The capacitances can be divided in three major components.

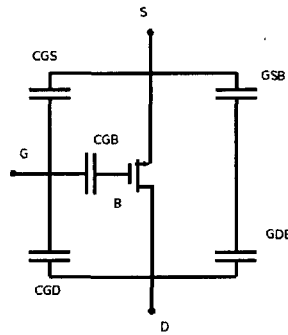


Figure 4.1: MOS Capacitances

4.1.2.1 Overlap Capacitance

The overlap capacitance is generated by the tendency at the source and drain terminals to extend below the oxide of the gate terminal in an amount called lateral diffusion or X_d .

$$C_{GSO} = C_{GDO} = C_{OX} \cdot X_D \cdot W \quad (4.5)$$

4.1.2.2 Gate to Channel Capacitance

The major component is given by the gate to channel capacitance. These components are given by the region of operation of the transistor. It has three components: C_{GCS} , C_{GCD} and C_{GCB} . The next table will show how they behave in the different regions of operation.

Table 4.1: Channel Capacitance in the Different Regions of Operation

Operation Region	C_{GCB}	C_{GCS}	C_{GCD}	C_{GC}
Cutoff	$C_{OX} \cdot W \cdot L$	0	0	$C_{OX} \cdot W \cdot L$
Linear	0	$C_{OX} \cdot W \cdot L/2$	$C_{OX} \cdot W \cdot L/2$	$C_{OX} \cdot W \cdot L$
Saturation	0	$(2/3) \cdot C_{OX} \cdot W \cdot L$	0	$(2/3) \cdot C_{OX} \cdot W \cdot L$

4.1.2.3 Diffusion Capacitance

The last capacitive component is the diffusion capacitance. This component is caused by the reverse biased current caused by the pn-junctions. It has two components, bottom plate and side wall.

$$C_{diff} = C_{bottom} + C_{side-wall} = C_j \cdot Area + C_{jw} \cdot Perimeter \quad (4.6)$$

The parameters used in the transistor of the simulation were taken from [26] and do not correspond to a particular technology but have reasonable values for standard 0.5um CMOS.

4.1.3 VHDL-AMS Transistor Model

The model included in the simulations was captured by mapping the terminals of the transistor as terminals in the entity of the model. The voltage across the terminals is taken as an across quantity and the currents as a through quantity. The current from drain to source in the different regions is modeled according to the equations 4.1 to 4.4. The parasitic capacitances are also included. Figure 4.2 shows the main quantities of the model and how they are mapped.

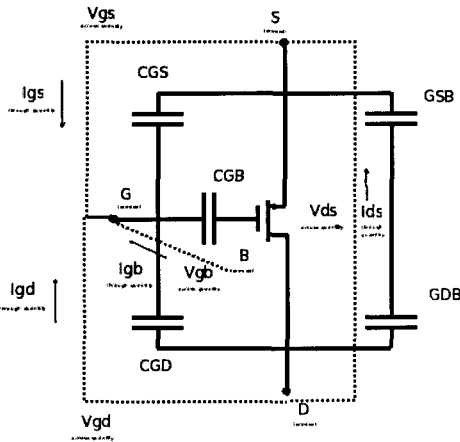


Figure 4.2: VHDL AMS Schematic Model

4.1.4 Functional Tests

The next circuits were implemented to verify the behavior of the transistor and for testing its functionality.

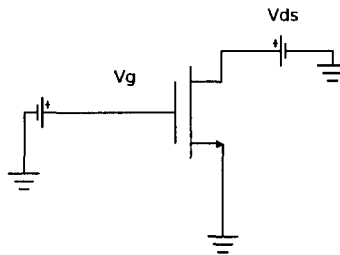


Figure 4.3: I_{DS} vs V_{DS} NMOS

The tests are based on the schematic circuit shown Figure 4.3. First, for test the $V_{ds} - I_{ds}$ characteristics, a constant voltage at the gate was kept, $V_g = 1.25V$, and then a variation in steps of 10_{mV} in the voltage between drain and source was induced, V_{ds} . The curves in the Figure 4.4 and 4.5 shows the characteristic curves for the NMOS and PMOS transistors used in this work with the given set of parameters. The circuit was simulated in Mentor's Graphics AdvanceMS and the testbench used can be found in Appendix A.

In Figures 4.4 and 4.5 the "X" axis shows the V_{ds} in the transistor and the "Y"

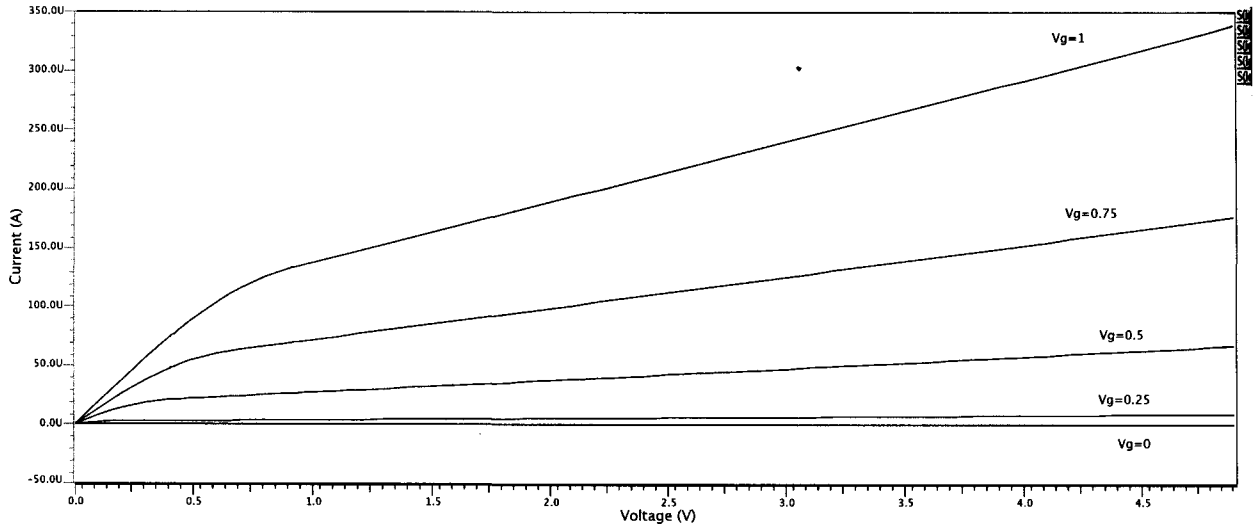


Figure 4.4: I_{DS} vs V_{DS} NMOS

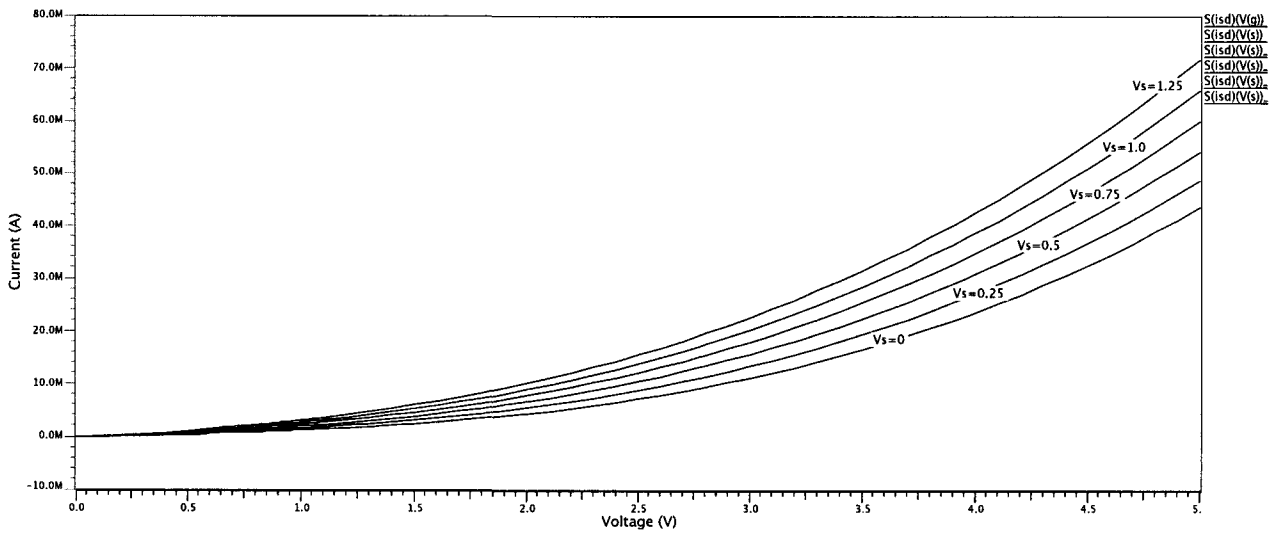


Figure 4.5: I_{DS} vs V_{DS} PMOS

axis shows the current I_{ds} . The circuit was simulated for $5mS$.

For test the $V_{gs} - I_{ds}$ characteristics for the circuit, the voltage V_{ds} was kept constant and a variation in steps of $10mV$ in the V_g voltage was induced. The simulation result graphics have the same characteristics than the ones in Figures 4.4 and 4.5.

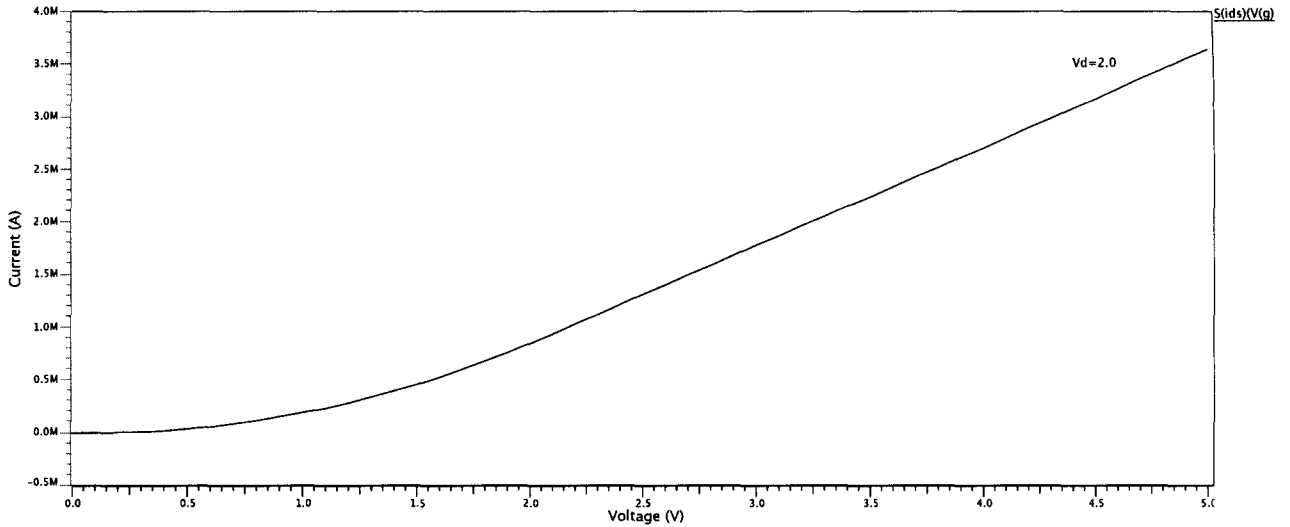


Figure 4.6: I_{DS} vs V_{GS} NMOS

In Figures 4.6 and 4.7 that the “Y” axis represents the current I_{ds} and in the “X” axis the voltage V_g is represented. It can be observed that for a major V_g a greater I_{DS} current is flowing from drain to source. The testbench used for this simulation is available in Appendix A.

Based on the tests made to the transistor, it is concluded that the transistor has an acceptable performance for the simulations.

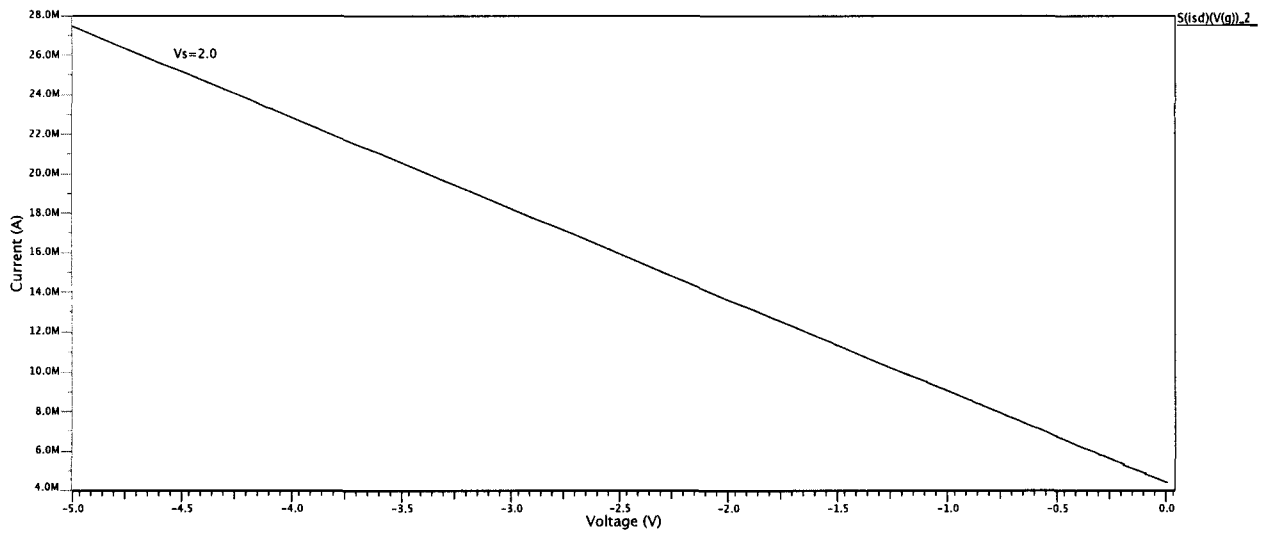


Figure 4.7: I_{DS} vs V_{GS} PMOS

Chapter 5

Results

This chapter presents the simulations result of the tag architecture described in section 3.2. The presented results are conformed by the signals that confirm the correct functional behavior of the tag model. It is shown how the signals fulfill the communication protocol described in section 3.1. The total power consumption, obtained by adding the individual power consumption of the subsystems in the tag, and its comparison with other efforts is presented too.

5.1 Module's Signals

The signals of each module are presented under two operation conditions: the address sent to the tag matches the own tag's address, and the sent address does not match the tag's address. The difference between the two cases can be appreciated in the signals of the control unit block.

5.1.1 Incident RF Signal

Figure 5.1 shows the RF signal at the antenna of the transponder. As commented in section 3.4.9, an optimum power transmission is considered between the reader and the transponder. The signal match the communication protocol described in section 3.1. In the wave form of the signal a power-up period, that is necessary to energize the circuit, can be seen. An addressing period that sends the start bit and the address of the tag that the reader wants to access. In this simulation the bits sent by the reader are 1-1-0. At the end is the reading period where short interrupts clock the shift register of

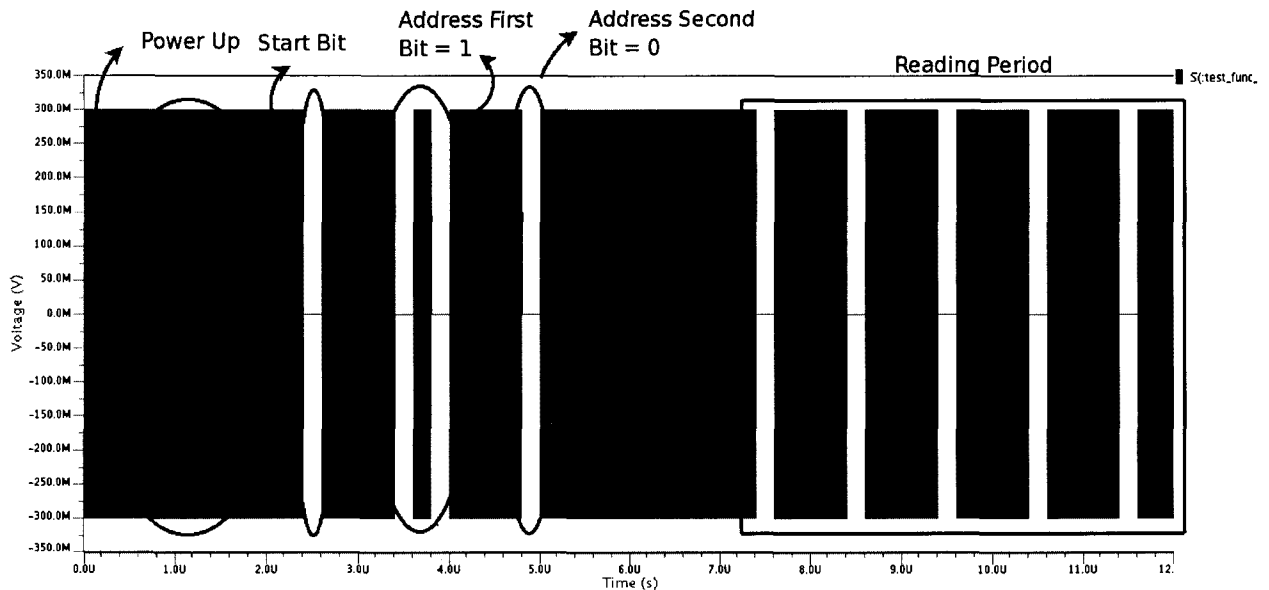


Figure 5.1: RF Signal Received by the Transponder

the transponder. Five interrupts are sent in the simulation. The transponder receives the signal, converts it to DC current, stores the power in its capacitors and decodes the information in it. The signal was obtained at the output of the simulated reader module explained in section 3.3.

5.1.2 Rectifier

Figure 5.2 shows how the rectifier was implemented in this simulation. The difference between the rectifier showed in Figure 5.2 and the explained in section 3.4.2 is that the transistor model was replaced with an ideal diode model in order to simplify the circuit and to reduce the simulation time. The threshold voltage of the diodes is $0.1V$ and the capacitance of the capacitors is equal to $100pF$, this value allows to the capacitor to charge quickly enough and have a good peak voltage.

Figure 5.3 shows the node voltages of the positive side of the full-wave rectifier incorporated to the simulation. The voltages in Figure 5.3a) shows how the voltage increments its DC reference on each stage of the rectifier. The increment is due to the effect of the clamping circuit. The clamping circuit conducts every time the voltages at nodes 2, 5 and 8 are negative and builds and average charge that prevents the output

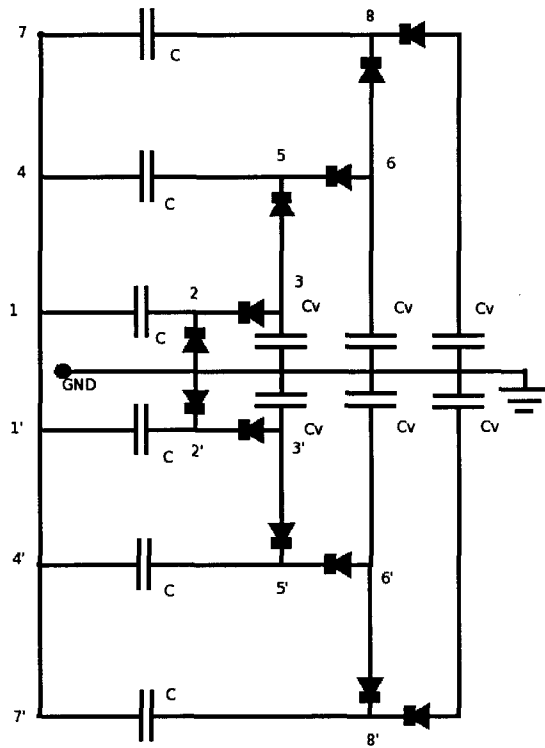
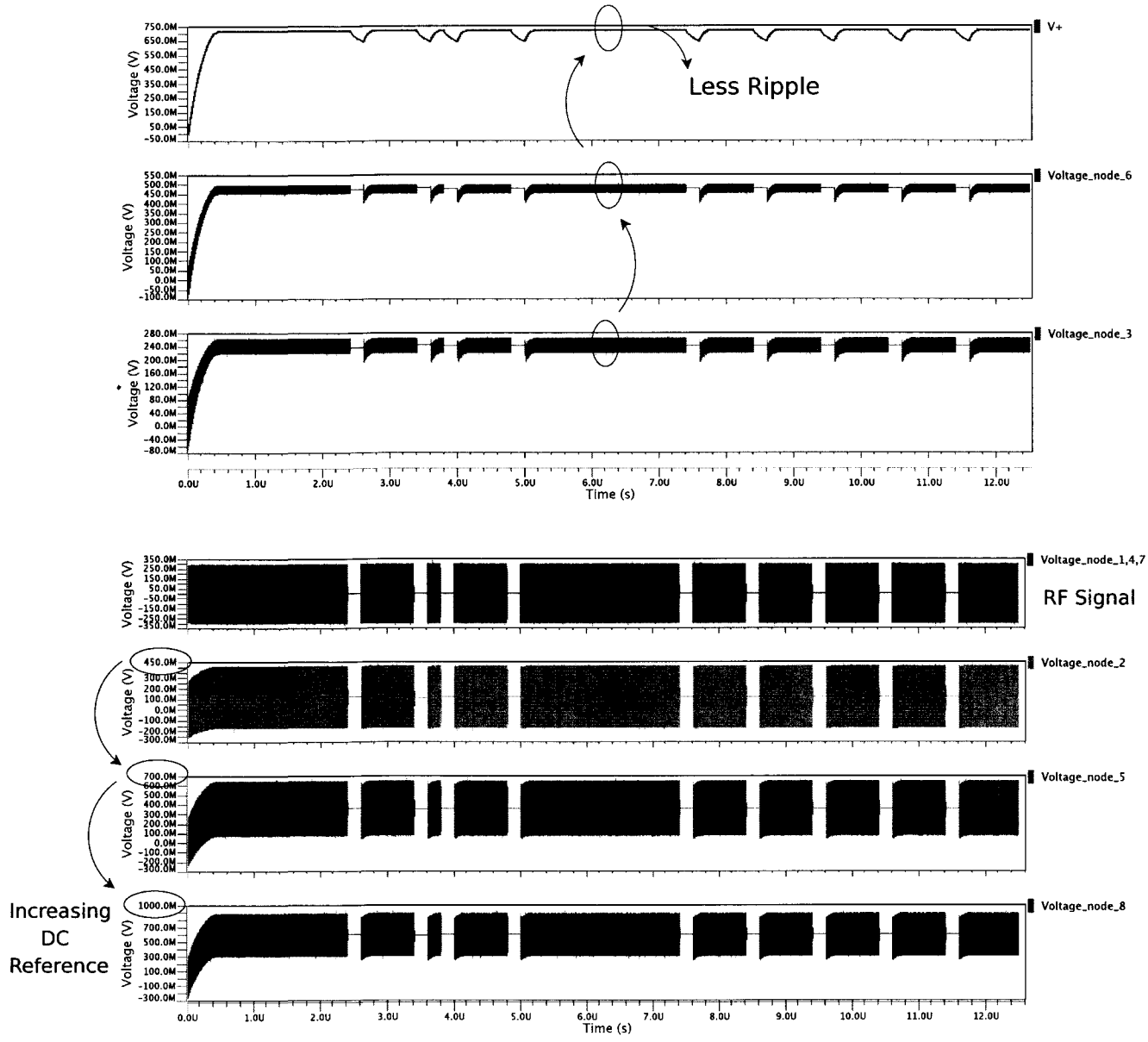
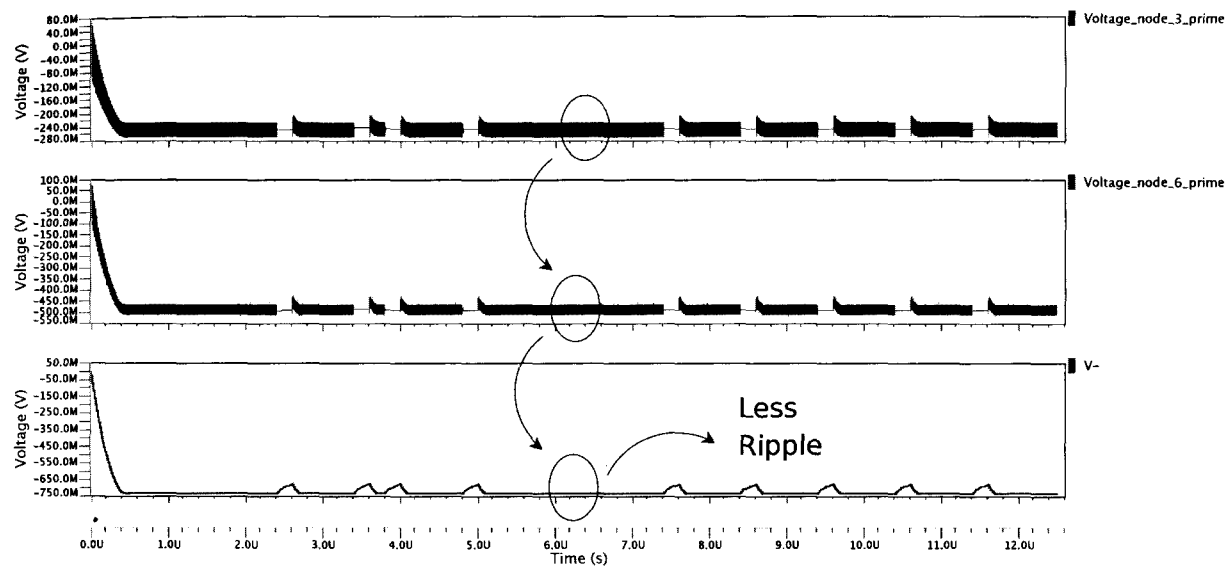


Figure 5.2: Rectifier Implemented in This Simulation

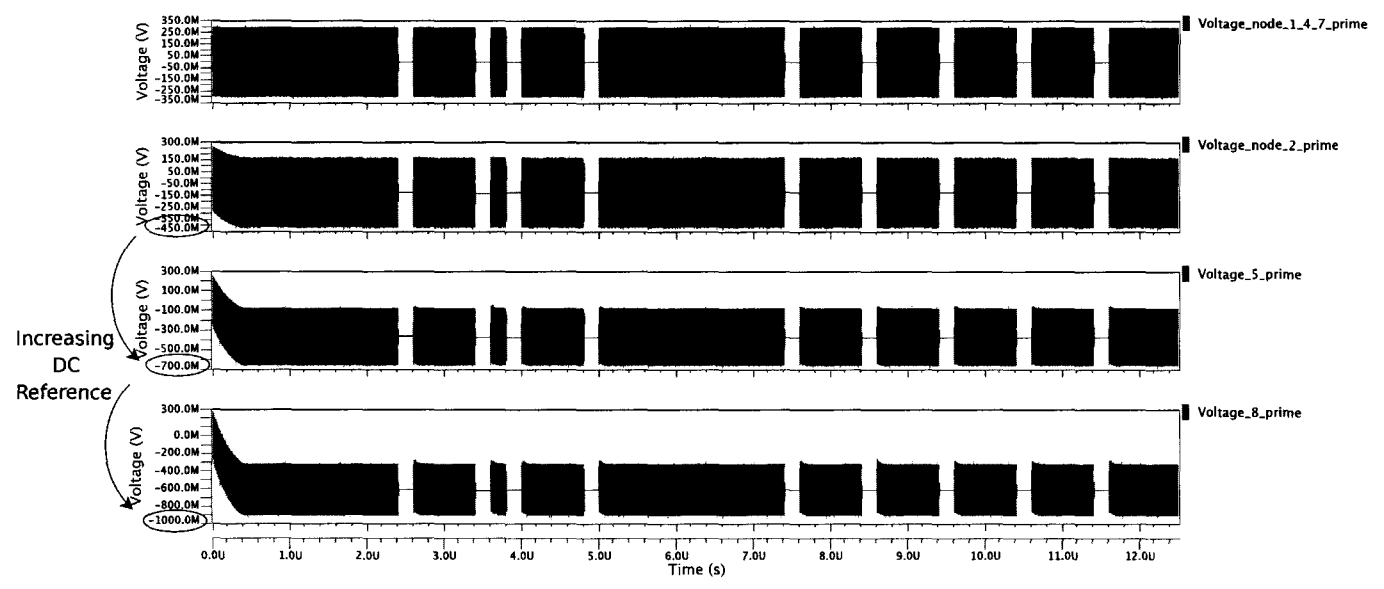


(a) Increasing Positive DC Reference (b) Rectifying the Positive Voltage

Figure 5.3: Positive Side Rectifier's Signals



(b) Rectifying the Negative Voltage



(a) Increasing Negative DC Reference

Figure 5.4: Negative Side Rectifier's Signals

from going negative. In Figure 5.3b), it can be seen how the voltage is being rectified for the rectifier circuit because of the diminishing magnitude of the ripple at each stage. The voltage at node 3 is the output of the first stage of the rectifier. This voltage is added to the output of the clamping circuit in second stage, node 5, to increment its DC reference voltage. The same happens from second to third stage, voltage at node 6 is added to voltage at node 8. The output voltage of the third positive stage is the rectifier's output, $V+$. The same principles are observed in Figure 5.4, the increment of the negative DC reference is observed in Figure 5.4 a), the outputs of each negative stage of the rectifier are shown in Figure 5.4 b). $V-$ is the negative output of the rectifier.

5.1.3 Power-on-Reset

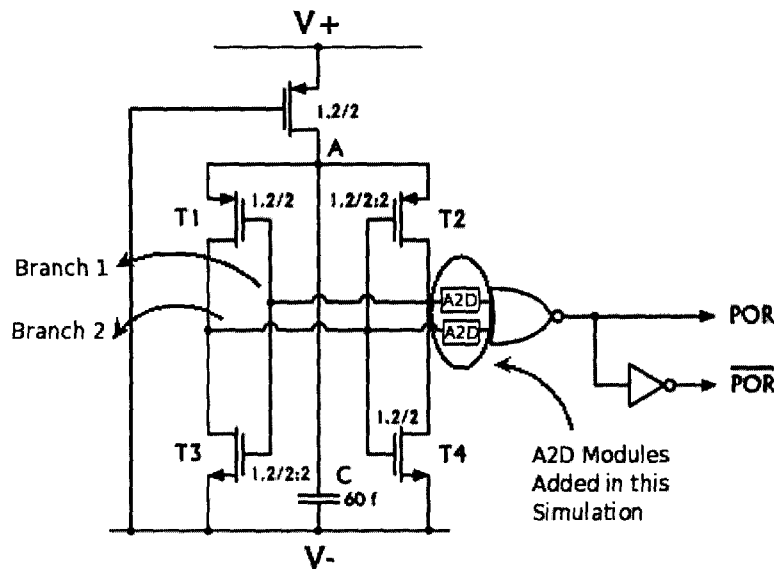
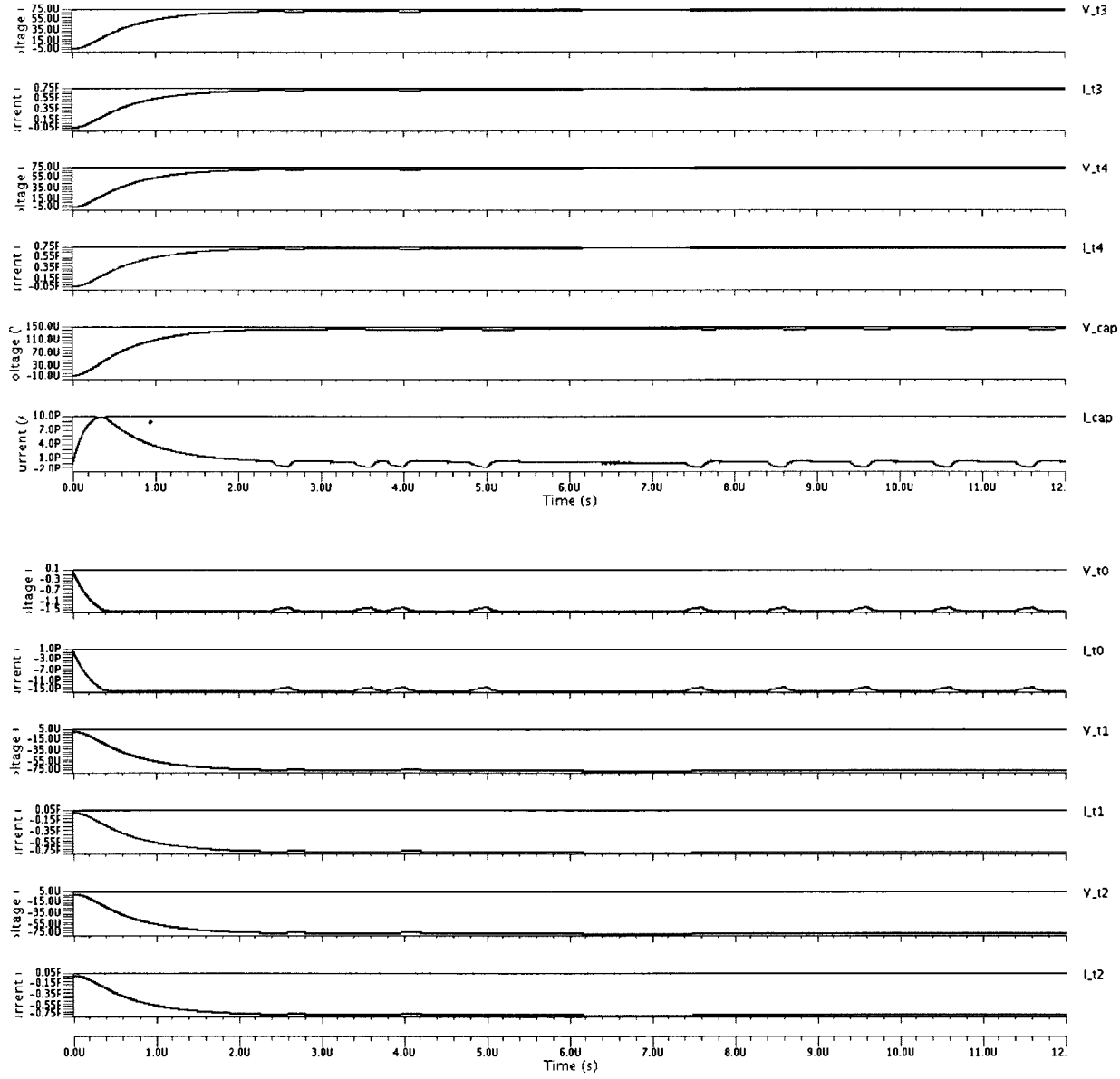


Figure 5.5: Power-on-Reset Simulation Schematic

The schematic of the power-on-reset block (PoRB) is observed in Figure 5.5. The PoRB is responsible to trigger all the blocks in the transponder. In the original design is a non-stable circuit but in this simulation some modifications were done. In the real analog design, the PoRB is a non-stable circuit; however, one branch of the system



(a) Signals Transistors 0,1,2

(b) Signals Transistors 3,4 and Capacitor Signals

Figure 5.6: Power-on-Reset Analog Signals

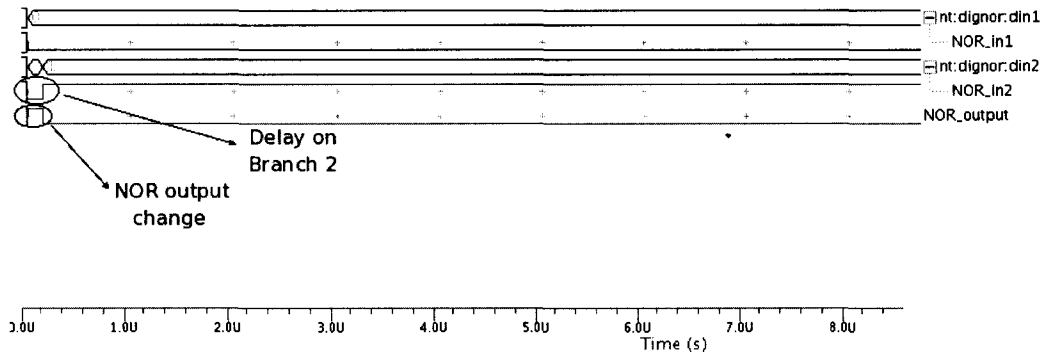
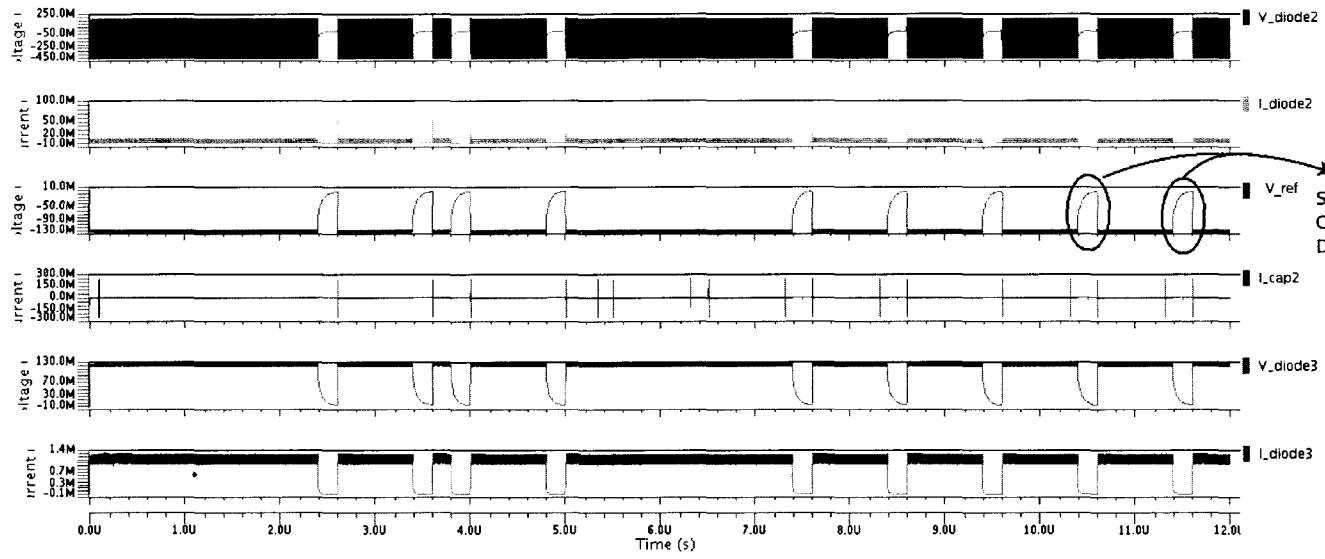


Figure 5.7: Power-on-Reset Digital Signals

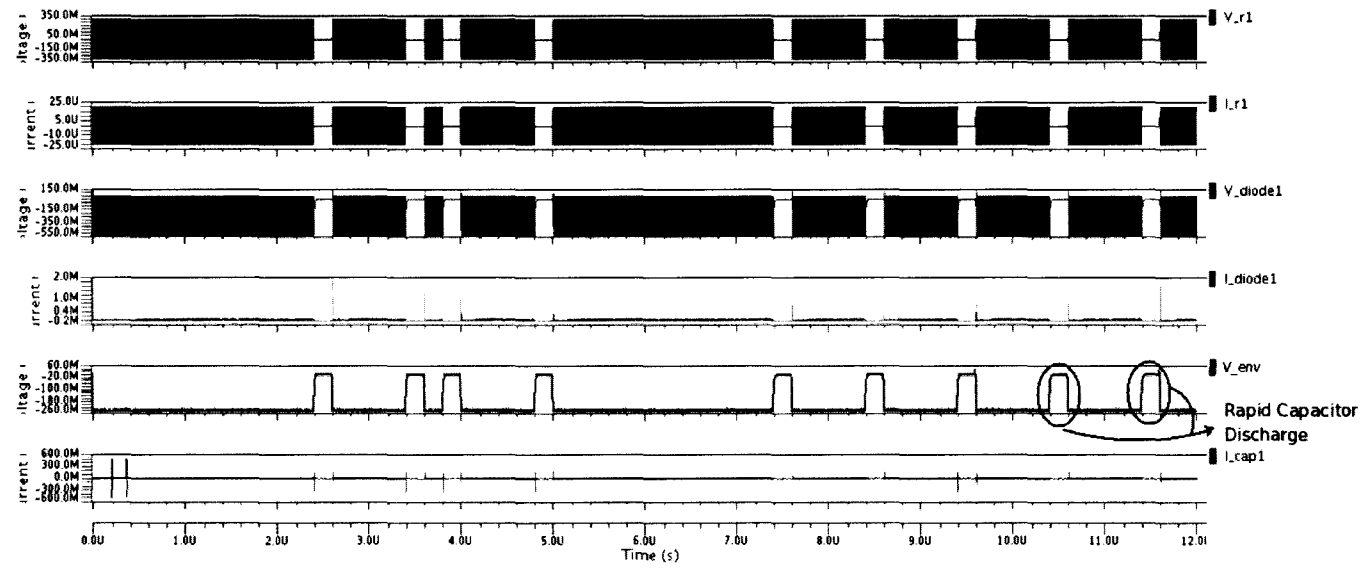
takes advantage over the other when both branches start to conduct in the real analog design. In this simulation both branches conduct at the same time due to the nature of the ideal simulated models. Thus, the NOR gate did not change its state properly. The solution was to add a delay in one of the branches to permit the branch to take advantage over the other and switch the output of the NOR gate. Before the signals of the branches reach the NOR gate they are converted to digital signals by an analog to digital converter. The output signal of the NOR gate is the Power-on-Reset signal. As shown in Figure 5.6, there is a moment, at the time both transistors start to conduct, when the NOR gate change its output. This allow the subsystems to have reset signal. The signal NOR_{output} in Figure 5.7 illustrates this behavior.

5.1.4 Detector

The signals V_{env} and V_{ref} can be obtained from Figure 5.8. V_{env} and V_{ref} are compared to achieve the demodulation of the signals sent by the reader. How the capacitor C_a charges to RF envelope can be seen in Figure 5.8a). When the RF signal falls, C_a is discharged rapidly through resistor R_a because of the inverse current of the diode. The components values were chosen to get a time constant of a few ns in order to follow the RF signal. The signal V_{cap1} in the Figure 5.8a) demonstrates this. Figure 5.8b) shows how the voltage of C_c charges to the half the amplitude of the RF signal. When the RF signal falls, the retained voltage of C_c discharges slow enough to give a reference voltage to compare to V_{env} .



(b) V_{ref} Signals



(a) V_{env} Signals

Figure 5.8: Detector Signals

5.1.5 Comparator

With the combination of this circuit and the detector, the regeneration of the data sent by the reader can be done. This circuit compares the signals V_{env} and V_{ref} . When V_{env} is higher than V_{ref} the comparator gives a drop of voltage in node C, showed in Figure 5.9. In this node an analog to digital converter was connected, so when the voltage drops, CK1 toggle to a logical “1”. This process can be seen in Figure 5.9 b) where V_{t6} is the voltage at node C. The voltage drop can be appreciated when the RF signal has an interruption.

5.1.6 Decoder

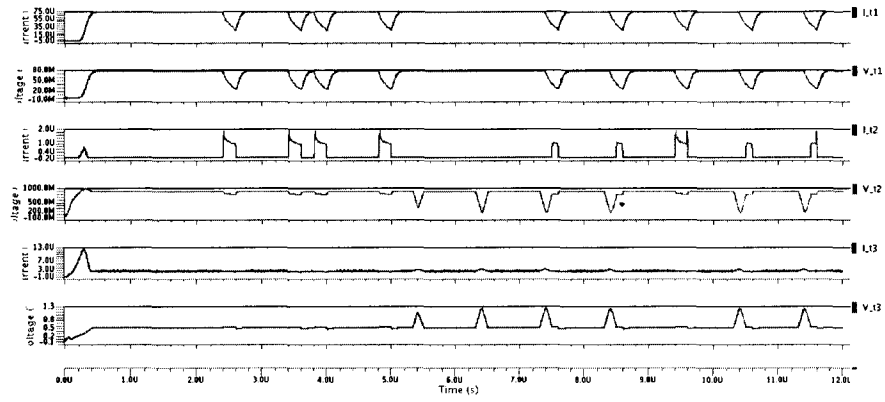
The decoder signals are shown in Figure 5.10. After the power-on-reset the $Data_{in}$ signal or nQ signal in Figure 5.10 is 0. When the start bit is received the clock of the flip flop toggles, clk signal in Figure 5.10, and $Data_{in}$ signal changes to 1. The communication process of the decoder is presented in the Figure 5.10. How data is entering to the logic of the circuit can be seen in Figure 5.10.

5.1.7 Current Reference

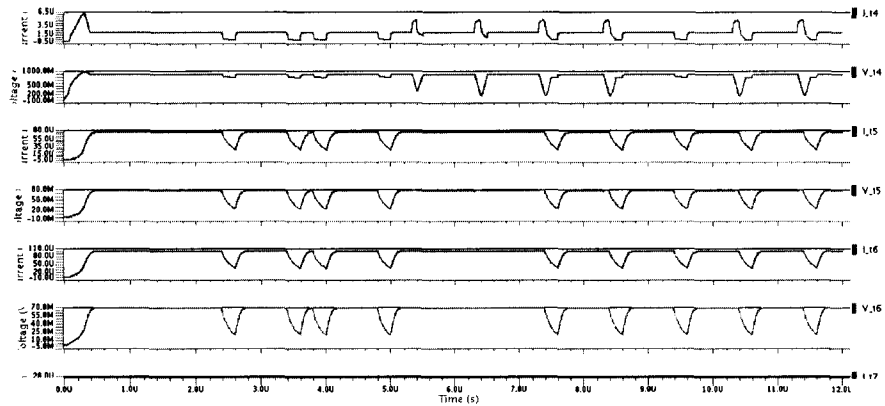
The circuit shown in figure 5.12 is just to have a reference current for all the circuits in the tag. It is especially suited for ultra-low power applications since a very low reference current can be obtained for a high value of R . In this simulation a value of $100M\Omega$ is given to R in order to get a low current reference and a low power consumption.

5.1.8 System Clock

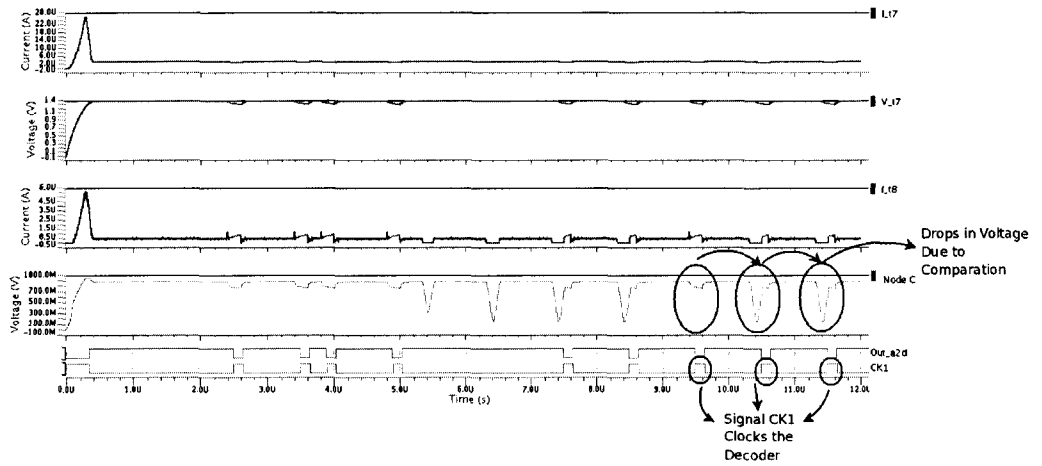
This block is responsible of the clocking of the control logic and the shift register. It turns on when the start bit is received. The signal CK1 enables it. When $Data_{enable}$ signal is turned on, the shift register shifts its data on each rising edge of the CK1 signal. It is possible to see how the shift happens in Figure 5.13 b). The output of this clock is an input to the logic and is one of the signals that determines the output of CK2 that is the main clock signal for the shift register.



(a) Signals of Transistors 1,2,3



(b) Signals of Transistors 4,5,6



(c) Signal of Transistors 7,8 and the CK1 Signal Generation

Figure 5.9: Comparator Signals

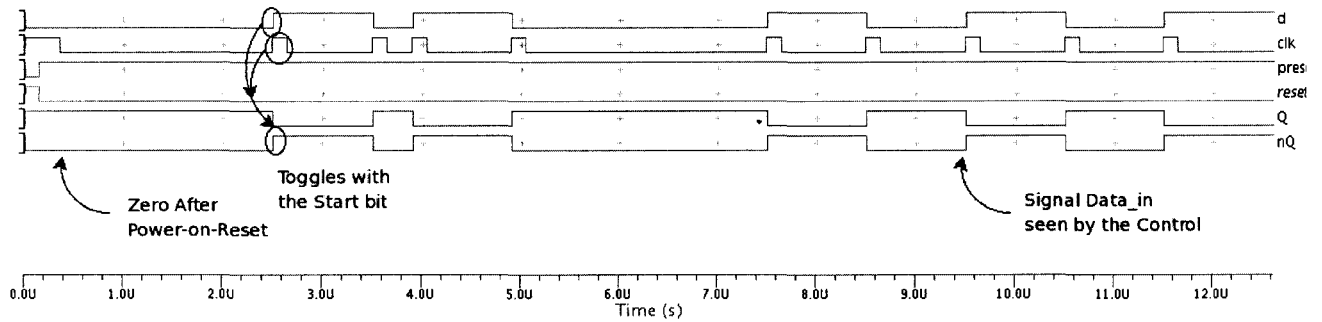


Figure 5.10: Decoder Signals

5.1.9 Control Unit and Shift Register

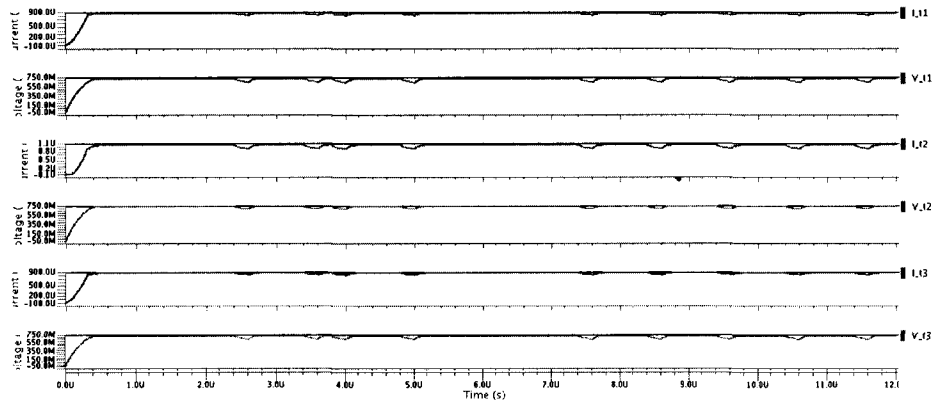
The diagram of this block is shown in Figure 3.15. This block is responsible for loading the information that the decoder is generating from the reader's signal. In this simulation, a 3-bit shift register is used. In the addressing mode, $Data_{enable}$ and $Data_{out}$ are both 0. $CK2$ is equal to the clock signal. These signals can be seen in figure 5.13 b) and c). The received data is registered in each rising edge of the system clock. The address of a transponder is programmed with the signals A_0 and A_1 . In figure 5.13 c) it can be seen that this values are $A_0 = 0$ and $A_1 = 1$. The behavior of the control logic when the received address matches the address of the transponder and when it does not is explained in the next section.

5.1.9.1 Right Address Signals

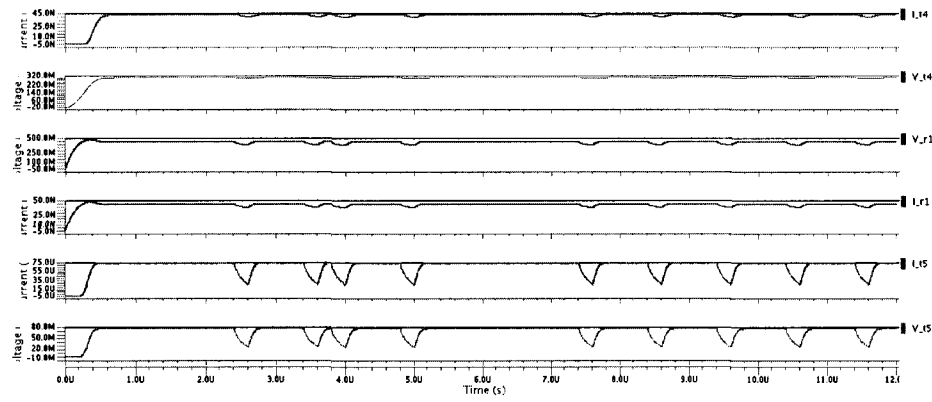
When the received address match the address in the transponder, the $Data_{enable}$ signals force the shift register to operate in closed loop. Its output goes directly to its input. The shift register start to shift its bit to the right. The value of $Data_{enable}$ determines if the transponder backscatters the signal. If $Data_{enable}$ is 0, there is no backscattered signal, and if it is 1, the signal is reflected by the transponder's antenna.

5.1.9.2 Wrong Address Signals

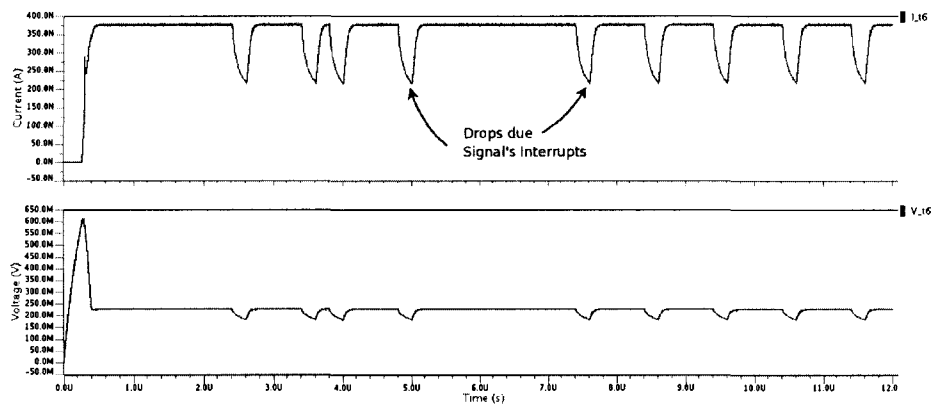
If the received address does not match the transponder's address, $Data_{enable}$ stays low, $Data_{out}$ goes high, and $CK2 = 1$, and no more bits are registered in the shift



(a) Signals Transistors 1,2,3



(b) Signals Transistors 4,5 and Resistor



(c) Signals Transistor 6

Figure 5.11: Current Reference Signals

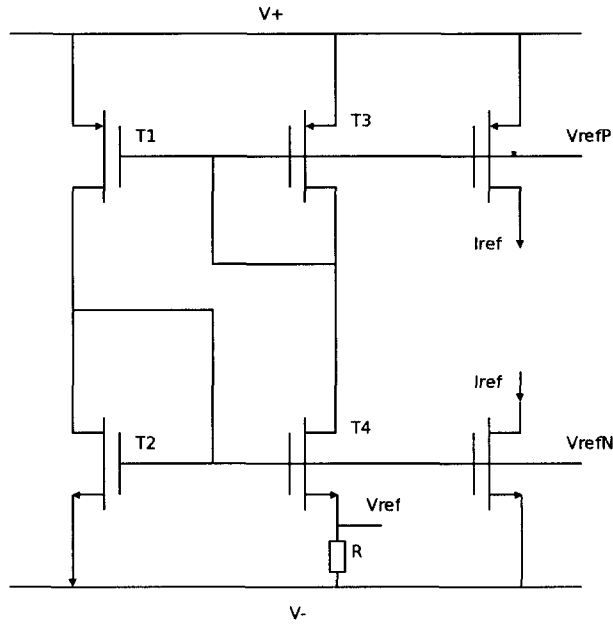


Figure 5.12: Current Reference Schematic

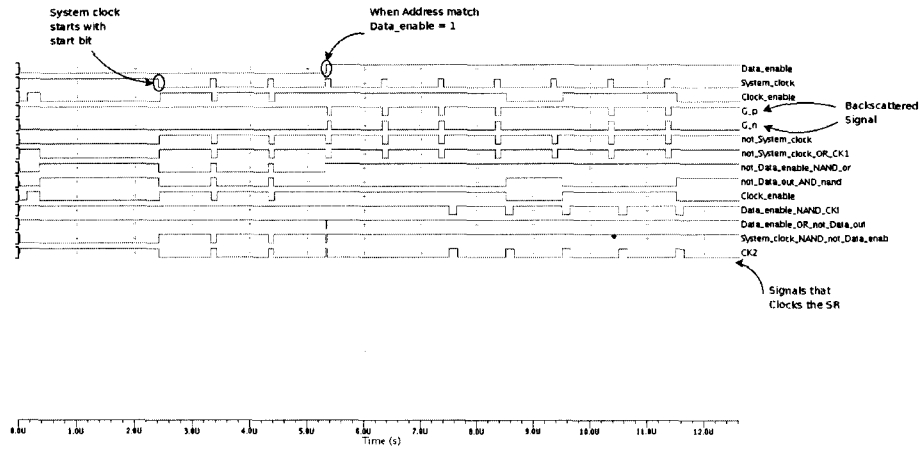
register. Here, the tag is in quiet mode, and have no further communication with the reader. The transponder is waiting for the RF signal to vanish or for the start of a new communication session. The behavior of this signals can be seen in figure 5.14. It can be seen how signals match the description above.

5.2 Analog Power Consumption

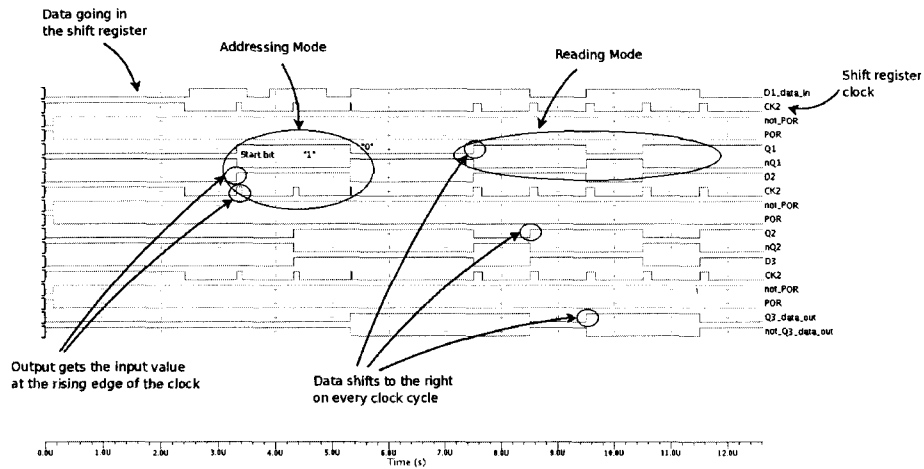
The power consumption of each module is a function of the number of transistors required by each module. In the next section the total instantaneous power of each module is calculated. It was obtained by adding the instantaneous power of each transistor on each one of the modules. The way the power consumption of each transistor is calculated is by the instantaneous power consumption formula:

$$P_{analog_subsystem} = \sum_{i=1}^n V_{ds} \cdot I_{ds} \quad (5.1)$$

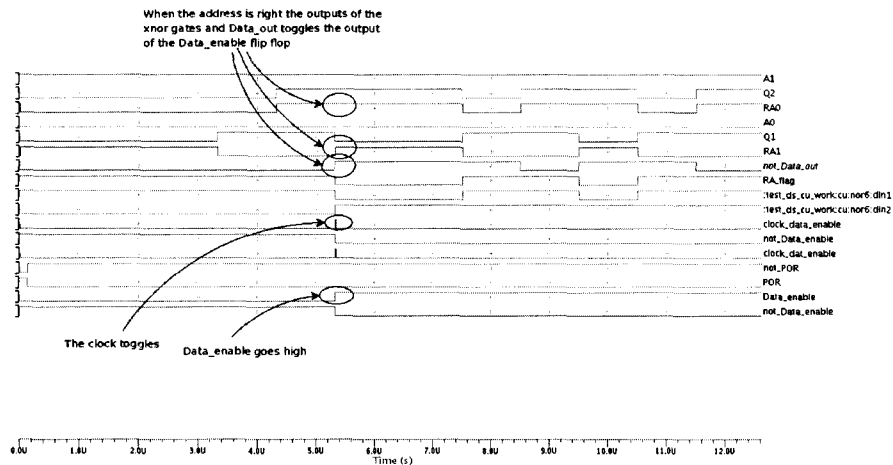
where n is the number of transistors, V_{ds} is the voltage across the drain and source terminals of the transistor, and I_{ds} is the current through drain and source terminals.



(a) Control Logic Right Address Signals

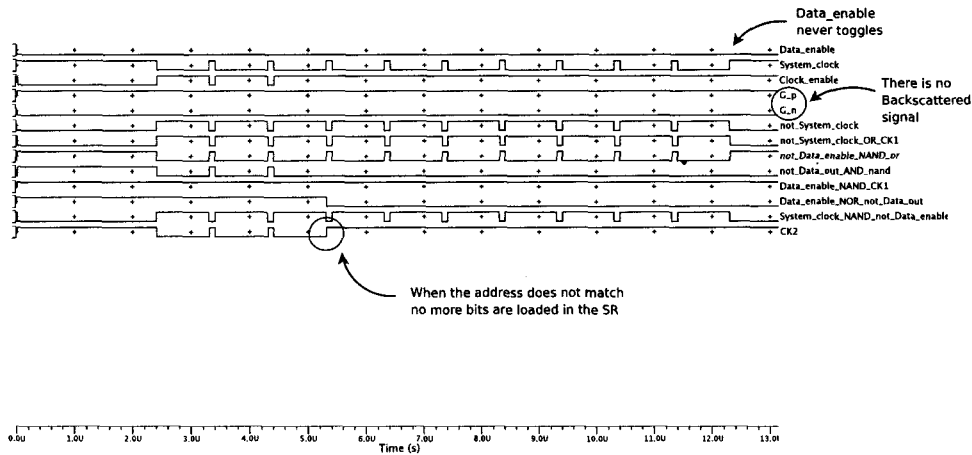


(b) Shift Register Right Address Signals

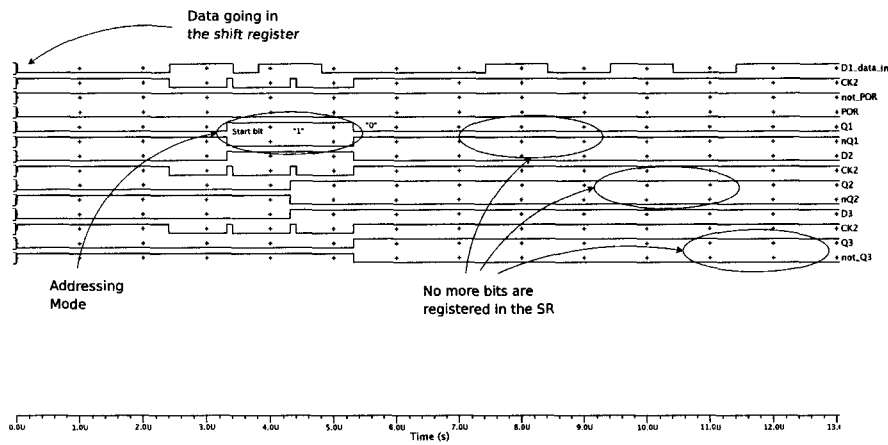


(c) Data Enable Right Address Signals

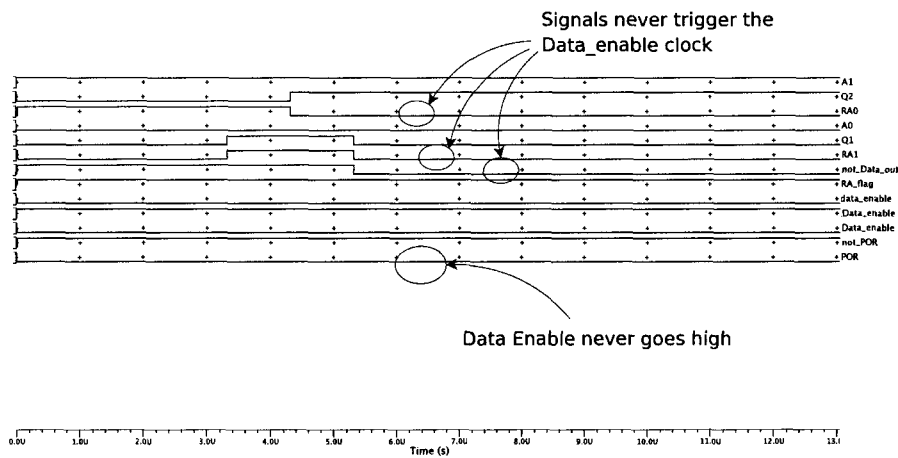
Figure 5.13: Control Unit, Shift Register and Data Enable Signals in the Right Address Case



(a) Control Logic Wrong Address Signals



(b) Shift Register Wrong Address Signals



(c) Data Enable Right Address Signals

Figure 5.14: Control Unit, Shift Register and Data Enable Signals in the Wrong Address Case

5.2.1 Incident RF Power

This is the total power received by the antenna, the total power at the rectifiers output is the total DC power. Figure 5.15 shows the total RF incident power. These graphics were obtained by multiplying the total voltage and the total current received by the antenna. The conclusion from Figure 5.15 is that the power consumption is elevated. This is due the lack of details in our implemented models. This lead us to a model that is somehow far from the real values but it is very useful for system verification models [5] [27].

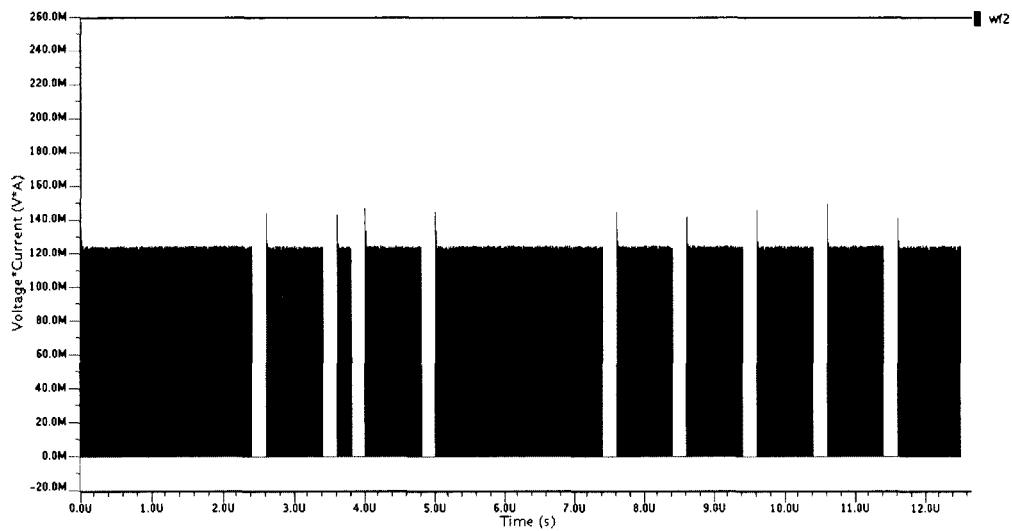


Figure 5.15: Incident RF Power

5.2.2 Rectifier

As it can be seen in Figure 5.16, this is the block that have the highest power consumption. The high power consumption is due to the constant switching of the diodes and the charge and discharge of the capacitors because of the high frequency of operation. Based on the last graphic, the global rectifiers efficiency is [1]:

$$\eta_o = \frac{DCOutputPower}{IncidentRFPower} \quad (5.2)$$

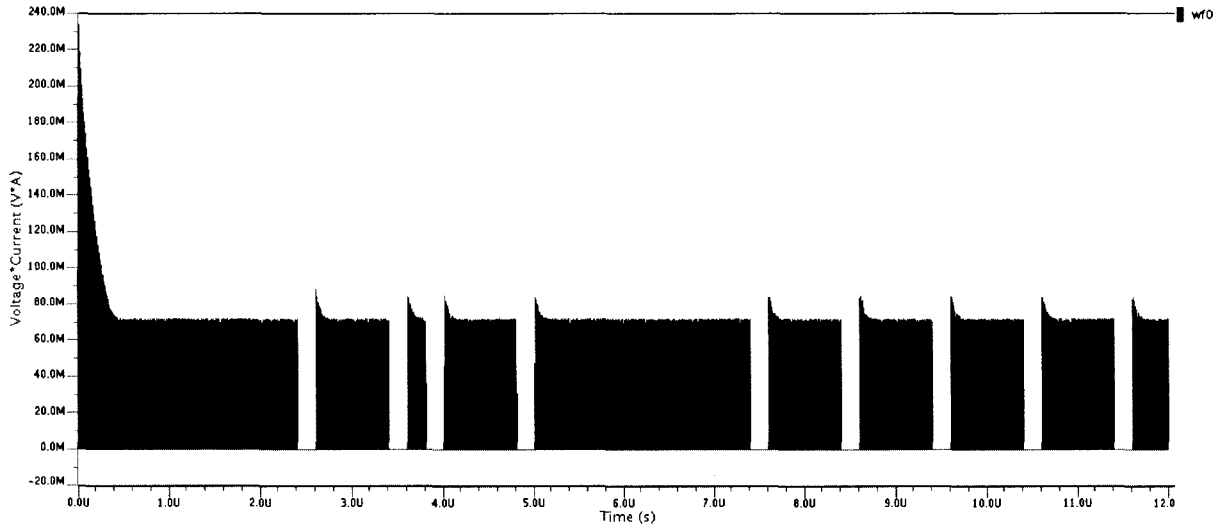


Figure 5.16: Power Dissipated By The Rectifier Block

The detailed graphics of each element in the rectifier can be seen in appendix b. From the total RF incident power, the rectifier’s efficiency is:

$$\eta_o = \frac{68.33_{mW}}{124.5_{mW}} = 0.548 = 54.8\% \quad (5.3)$$

5.2.3 Power-on-Reset

As explained in section 3.4.3, this circuit is just to create the signal that triggers all the digital circuits of the tag. The switching is almost null and because of the low leakage current, the consumption is minimum and caused for the static power consumption. The detailed graphics of the voltages and currents of the Power-on-Reset module are presented in Appendix B.

5.2.4 Detector

The detector is the circuit that creates the signals that are compared between them in order to determine the value of the data received. It needs to give a constant output, but because of the use of low threshold voltage diodes and the reduced quantity of them the power consumption is very low.

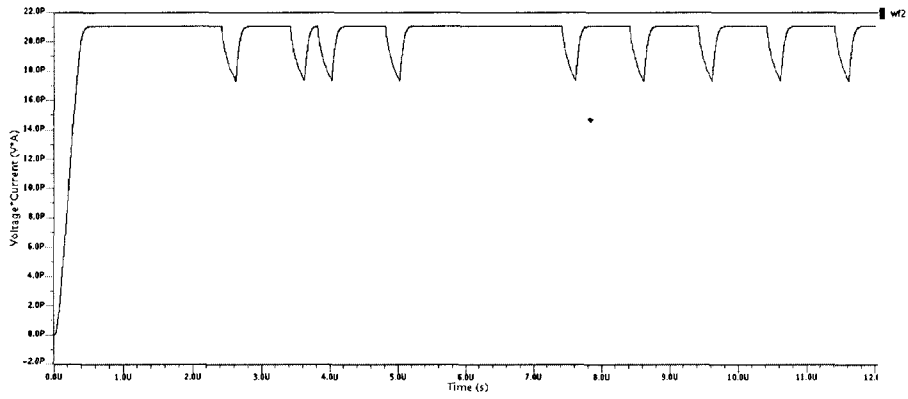


Figure 5.17: Power Dissipated By The Power-on-Reset Block

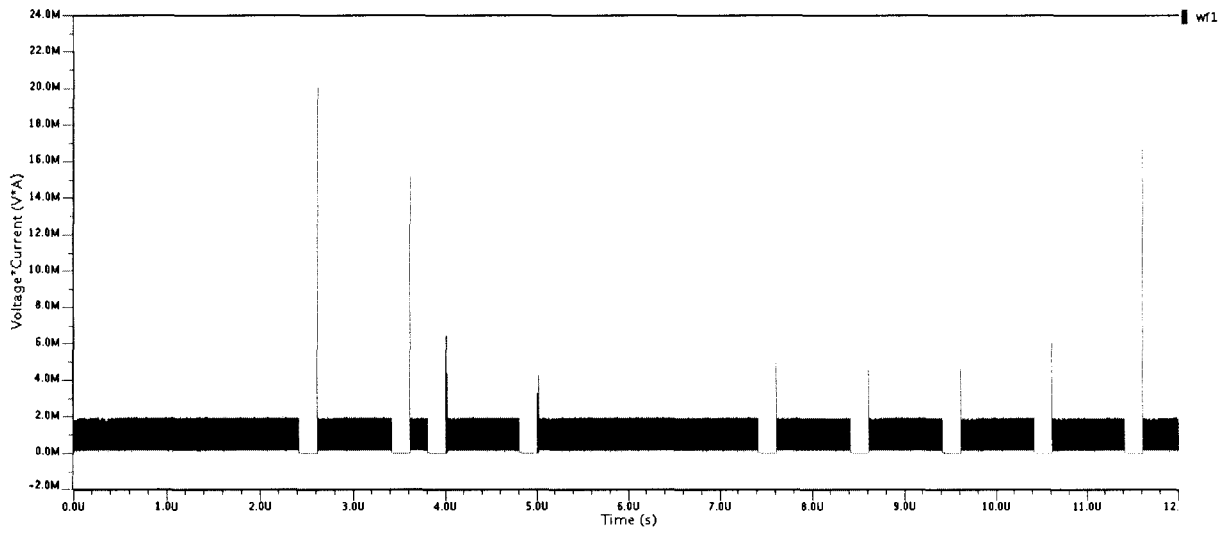


Figure 5.18: Power Dissipated By The Detector Block

5.2.5 Comparator

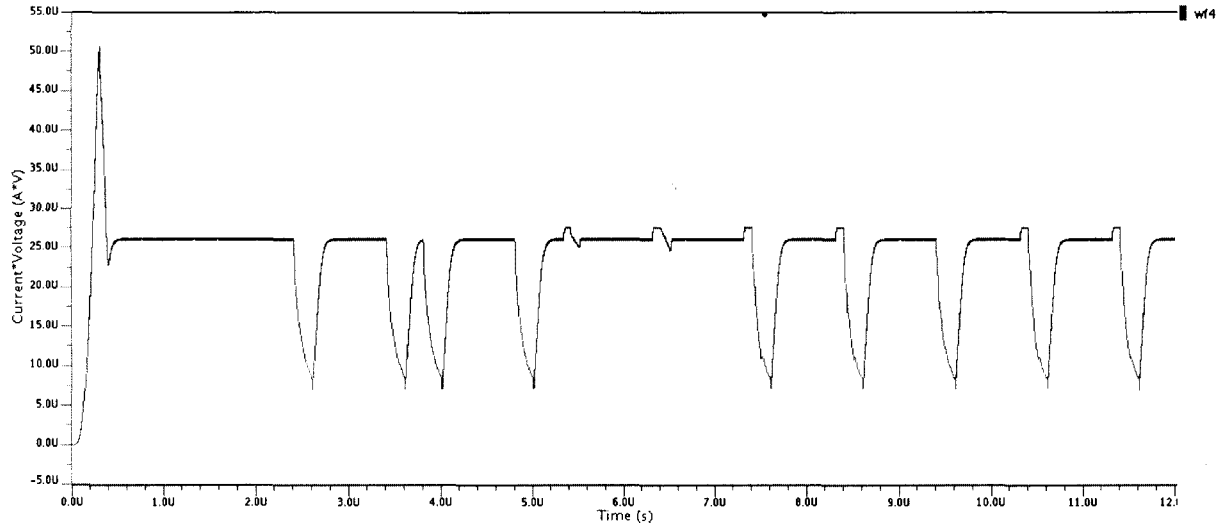


Figure 5.19: Power Dissipated By The Comparator Block

This block ensures the regeneration of the signals from which the data is regenerated. Because the transistors are regularly off and only switches on at the moment of the comparison, the power consumption is low.

5.2.6 Current Reference

The purpose of this block is to provide a biased current to the rest of the blocks, but there is no frequent switching. The consumption is a small percentage of the total consumption of the tag and in its major part due to the power dissipated by the resistance.

5.2.7 Total Analog Power Consumption

The total consumption is shown in Figure 5.21. It was obtained by adding the consumption of each block mentioned earlier in this section. It can be concluded that the rectifier contributes with the major part of the power consumption. In Table 5.1 the percentage of the power used by each one of the components are compared. The

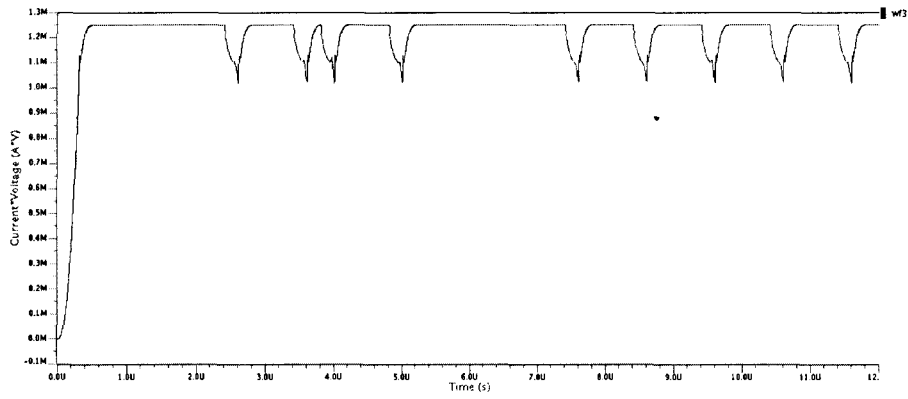


Figure 5.20: Power Dissipated By The Current Reference Block

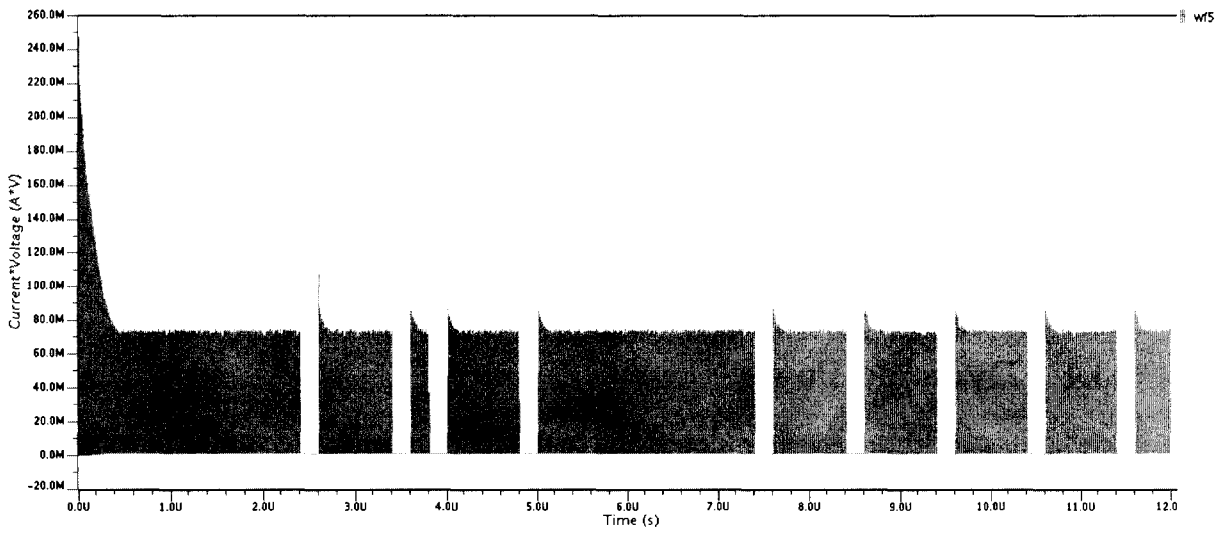


Figure 5.21: Total Analog Power Dissipated

percentage was calculated adding the analog power consumption of each block in the circuit, considering this result as the 100% of the power in the circuit, and with this reference the percentage of each block was calculated.

Table 5.1: Percentage of Total Power Consumption

Tag's Block	Power Consumption	Percentage of the Total Power Consumption
Rectifier and Limiter	65_{mW}	95.12%
Power-on-Reset	21_{pW}	$3.07 \cdot 10^{-5}\%$
Detector	2_{mW}	2.92%
Comparator	$35_{\mu W}$	0.05%
Current Reference	1.3_{mW}	1.90%
Total	68.35_{mW}	

5.3 Digital Power Consumption

The sources of power in a digital CMOS circuit can be categorized in two groups: Dynamic power and Static power. Dynamic power comes from three different sources: power due to capacitance switching, short-circuit power due to “crowbar” current flowing from V_{DD} to ground during switching and power due to glitches in the output waveforms. Capacitance switching and short-circuit power are the ones considered in this work. Static power is due to leakage currents and DC standby currents [28]. The static power analysis is not included in this work as the transistor model does not model the leakage current needed to calculate it.

5.3.1 Dynamic Power

Most of the gate power consumption is due to the charging and discharging of capacitors at the output of the circuits because of logic switching events. When a switching event occurs, a capacitor charges at one half of the cycle and discharges in the other half. A current flows from V_{DD} to ground once this happens and it leads to power dissipation. The switching frequency determines how much power is consumed. The next formula explains the total power consumed due to switching:

$$P_{switching} = C_L \cdot V_{DD}^2 \cdot f_{avg} \quad (5.4)$$

where C_L is the value of the capacitor at the output of the gate, V_{DD} is the source voltage and f_{avg} is the frequency of the clock.

A factor that should be taken into account is the average switching frequency f_{avg} . The clock frequency is normally taken to be f_{clk} and the clock switches every cycle. If a toggle is referred as a transition from low to high or high to low, two toggles are needed to have power dissipation. If the gates outputs are observed closely, they do not switch on every clock cycle. The average frequency of operation can be specified using an activity factor calculated as:

$$\alpha_{0 \rightarrow 1} = \frac{\text{toggles}/2}{\text{clockcycles}} \quad (5.5)$$

$\alpha_{0 \rightarrow 1}$ is multiplied by the clock frequency. Adding this changes, the power equation can be modified and the resulting is:

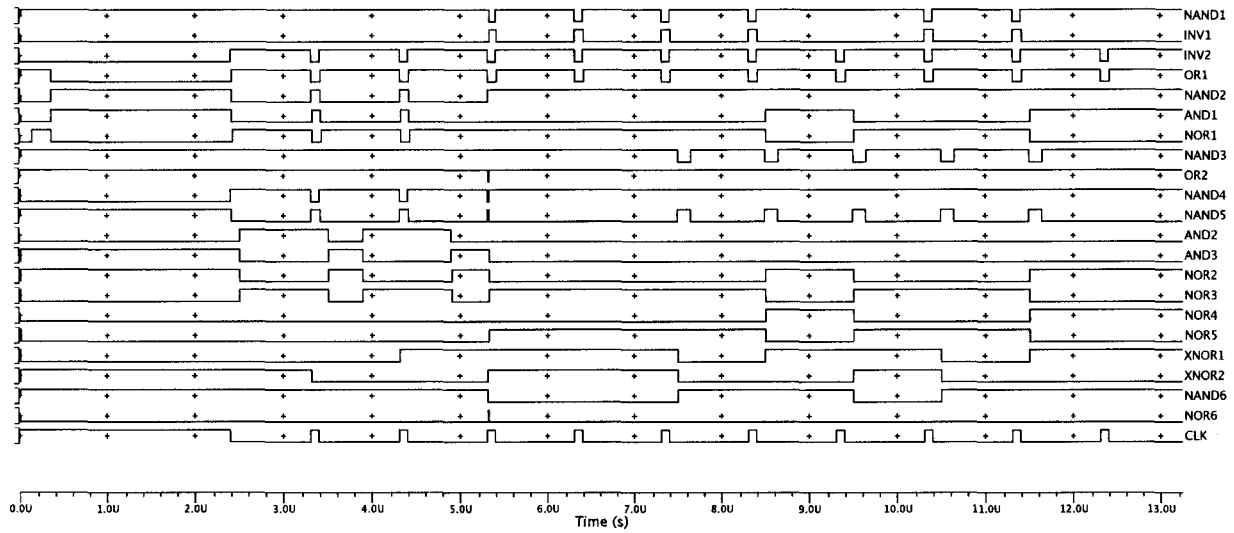
$$P_{switching} = \alpha_{0 \rightarrow 1} C_L \cdot V_{DD}^2 \cdot f_{clk}. \quad (5.6)$$

Tables 5.2 and 5.3 shows the considered magnitudes for each one of the elements of the Equation 5.6 and the switching factor for each gate of the control unit, according to Figure 5.22. The two possible cases of operation are considered, one on each Table. The value of C_L was chosen from [29].

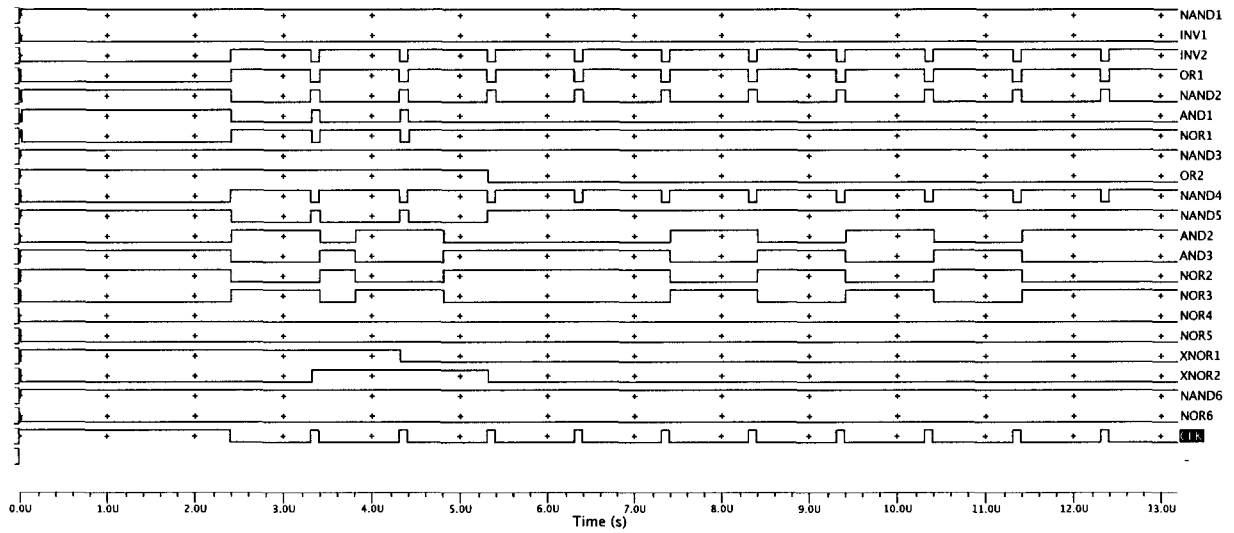
Table 5.2: Switching Factor and Power Consumed by Each Gate In The Right Address Case

@ $f_{clk} = 1MHz$, $C_L = 50pF$, $V_{DD} = 750mV$ [29]					
Gate	$\alpha_{0 \rightarrow 1}$	Power Consumed	Gate	$\alpha_{0 \rightarrow 1}$	Power Consumed
NAND ₁	0.333	9.365 _{μW}	NAND ₂	0.25	7.031 _{μW}
NAND ₃	0.416	11.7 _{μW}	NAND ₄	0.333	9.365 _{μW}
NAND ₅	0.75	21.09 _{μW}	NAND ₆	0.166	4.668 _{μW}
INV ₁	0.5	14.062 _{μW}	INV ₂	0.916	25.762 _{μW}
OR ₁	0.916	25.762 _{μW}	OR ₂	0.083	2.334 _{μW}
AND ₁	0.333	9.365 _{μW}	AND ₂	0.166	4.668 _{μW}
AND ₃	0.25	7.031 _{μW}	NOR ₁	0.333	9.365 _{μW}
NOR ₂	0.333	9.365 _{μW}	NOR ₃	0.333	9.365 _{μW}
NOR ₄	0.166	4.668 _{μW}	NOR ₅	0.166	4.668 _{μW}
NOR ₆	0.083	2.334 _{μW}	XNOR ₁	0.25	7.031 _{μW}
XNOR ₂	0.25	7.031 _{μW}	CLOCK	1	37.5 _{μW}

The total power consumption and a comparison on each case is shown Table 5.4. From Table 5.4 can be concluded that an extra power consumption is needed in the case



(a) Gates Switching for the Right Address Case



(b) Gates Switching for the Wrong Address case

Figure 5.22: Gates Switching

Table 5.3: Switching Factor and Power Consumed by Each Gate In The Wrong Address Case

@ $f_{clk} = 1_{MHz}$, $C_L = 50_{pF}$, $V_{DD} = 750_{mV}$ [29]					
Gate	$\alpha_{0 \rightarrow 1}$	Power Consumed	Gate	$\alpha_{0 \rightarrow 1}$	Power Consumed
NAND ₁	0	0	NAND ₂	0.916	25.762 _{μW}
NAND ₃	0	0	NAND ₄	0.916	25.762 _{μW}
NAND ₅	0.25	7.031 _{μW}	NAND ₆	0	0
INV ₁	0	0	INV ₂	0.916	25.762 _{μW}
OR ₁	0.916	25.762 _{μW}	OR ₂	0.083	2.334 _{μW}
AND ₁	0.25	7.031 _{μW}	AND ₂	0.416	11.7 _{μW}
AND ₃	0.416	11.7 _{μW}	NOR ₁	0.25	7.031 _{μW}
NOR ₂	0.416	11.7 _{μW}	NOR ₃	0.416	11.7 _{μW}
NOR ₄	0	0	NOR ₅	0	0
NOR ₆	0	0	XNOR ₁	0.083	2.334 _{μW}
XNOR ₂	0.083	2.334 _{μW}	CLOCK	1	37.5 _{μW}

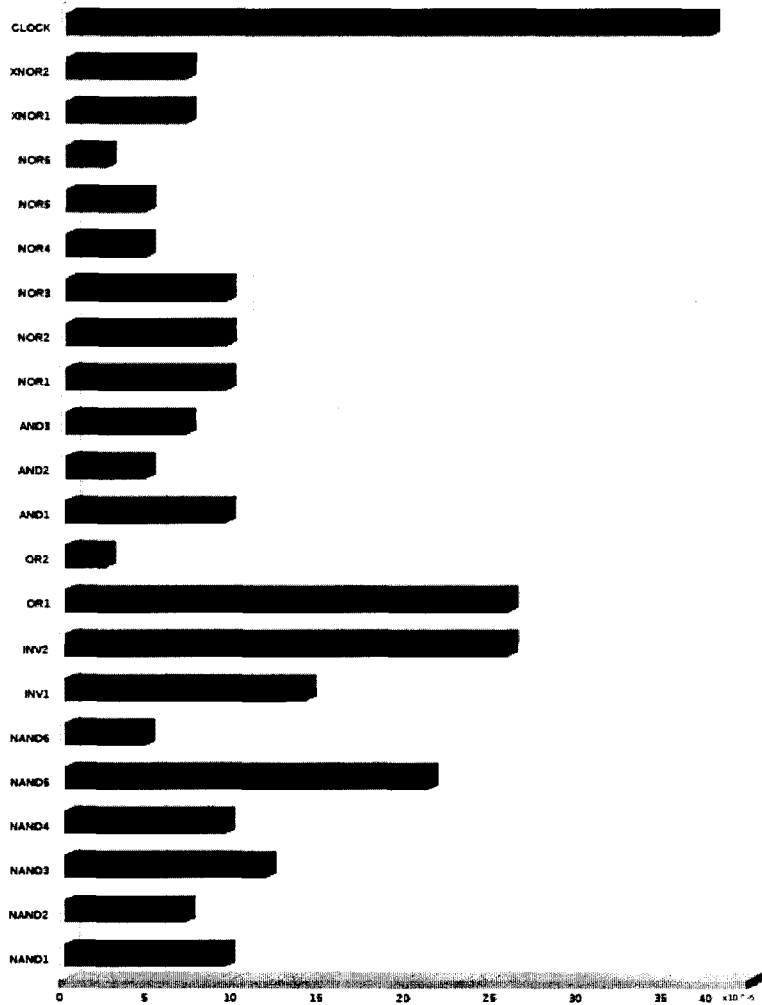
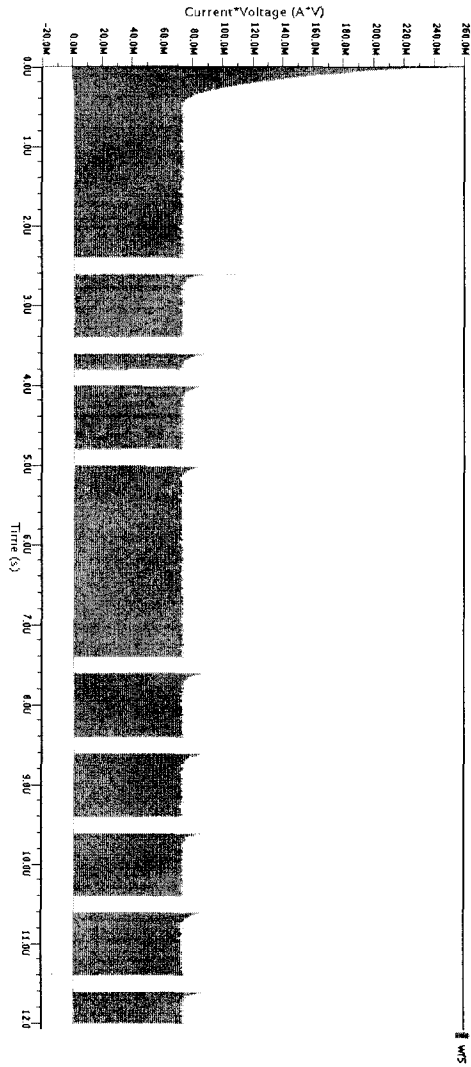
of a right address is received. The extra power consumed is due to the extra switching in the gates needed to backscatter the reader's signal and retransmit its ID to it.

Table 5.4: Dynamic Power Consumption Comparison

Case	Total Power Consumption
Right Address	243.53 _{μW}
Wrong Address	205.109 _{μW}

5.4 Total Power Consumption

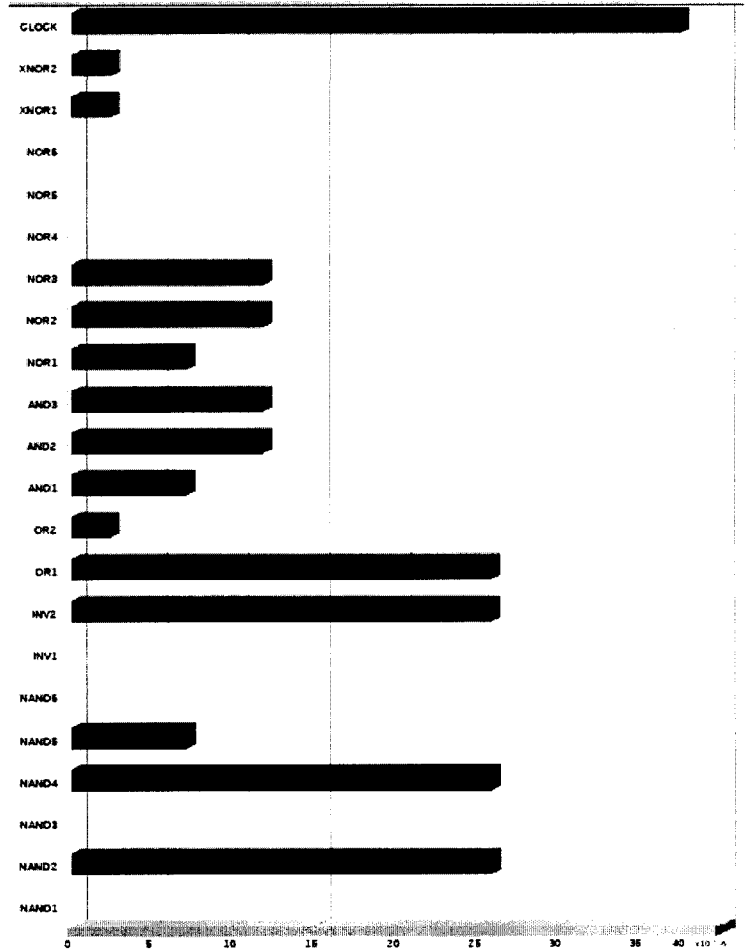
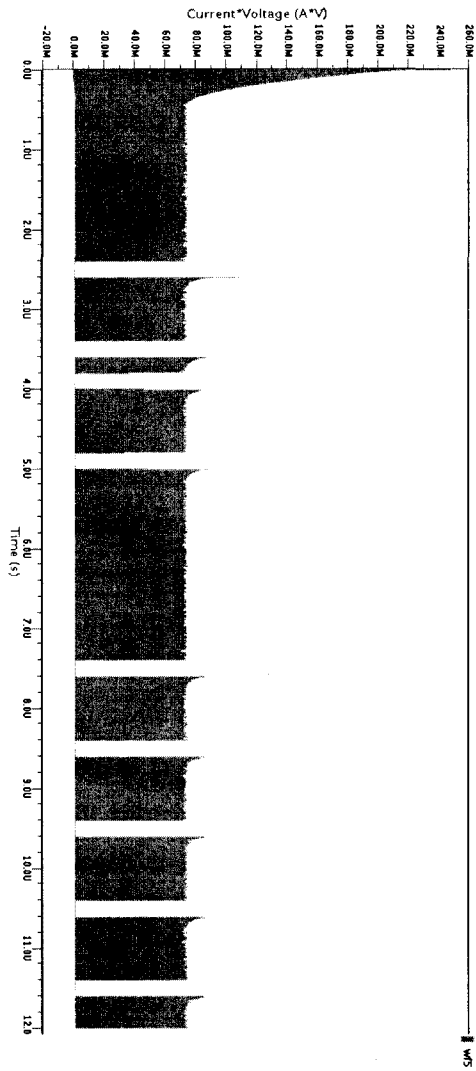
The total power consumption is presented in this section, considering the results of the analog and digital power consumption. As it can be appreciated in Figure 5.23 and 5.24, the digital power consumption contributes with a very small percentage of the total power consumption. Figure 5.23 shows how the gates have a more frequent switching in the right address case due to the extra bits loaded in the shift register. A lower switching frequency is observed in wrong address case, shown in Figure 5.24, because the tag gets in the quiet mode and no more bits are loaded after the addressing mode.



(a) Right Address Analog Power Consumption

(b) Right Address Digital Power Consumption

Figure 5.23: Right Address Case Power Consumption



(a) Wrong Address Analog Power Consumption

(b) Wrong Address Digital Power Consumption

Figure 5.24: Wrong Address Case Power Consumption

5.5 Comparison with Related Works

The power consumption obtained in this work is compared with the power consumption of works that have similar blocks in their architectures, like [3] and [4]. Table 5.5 shows a comparison between the blocks that have similar tasks in each architecture, the ones that have no match are taken into account for a total power consumption comparison. Table 5.6 shows the power consumption of the subsystems in [5] and the subsystems of this work, although they do not have similar tasks, to compare the total power consumption of both.

Table 5.5: Comparison of Results with Works [3] and [4]

Block in the System	Power Consumption	
	This Work	Work in [3] and [4]
Rectifier and Limiter	65_{mW}	
Power-on-Reset	21_{pW}	0.01_{mW}
Detector	2_{mW}	0.87_{mW}
Comparator	35_{uW}	0.95_{mW}
Current Reference	1.3_{mW}	
Quiescent Power		92_{mW}
Total	68.35_{mW}	93.83_{mW}

Table 5.6: Comparison of Results with [5]

This Work		Work [5]	
Rectifier	65_{mW}	Standby energy dissipation	10.25_{mW}
Power-on-Reset	21_{pW}	Temperature log dissipation	2.575_{mW}
Detector	2_{mW}	Reader interrogation energy dissipation	2.325_{mW}
Comparator	35_{uW}	Memory read energy dissipation	1.362_{mW}
Current Reference	1.3_{mW}		
Total	68.35_{mW}	Total	16.51_{mW}

Chapter 6

Conclusions and Future Work

6.1 Conclusions

This thesis presented a model of a UHF RFID tag based on transistor level components to increase the granularity of the model and to allow more accurate power consumption measurements. The cosimulation of the RFID tag also resulted in summarizing power measurements from both, digital and analog components. A MOSFET transistor model was used to describe the architecture of the UHF RFID tag. The complete tag cosimulation was executed with satisfactory communication among blocks. The analog and digital power consumptions of the tag were compared with the work in [4] and [6] with an improvement of 27.15%. The communication signals among blocks agreed in 100% with the communication protocol.

6.2 Future Work

The improvement of the transistor model, adding more physical effects like mobility effects due to vertical and lateral fields, velocity saturation, short-channel effects as channel-length modulation, etc., to have better results. Incorporate a microprocessor to the tag design to have the sufficient processing power to execute cryptographic functions and increase the security of the information in the tag. Modify the communication protocol to adequate it to the RFID standards like ISO-18000 and EPC-Global. The fabrication of the tag in a SOC to calculate the real power consumption and compare it with the power consumption obtained from the cosimulation. Elaborate a model

of the tag in a simulation suite like Mentor Graphic's ICstudio to compare its power consumption with the power consumed by the SoC of the tag and the power estimated in this thesis.

Appendix A

VHDL-AMS Code

A.1 NMOS Transistor Code

```
library IEEE;
use IEEE.math_real.all;
use ieee.math_complex.all;
use ieee.std_logic_1164.all;
use ieee.electrical_systems.all;
entity nmos is generic (cgso:real:=2.0e-010;
                        cgdo:real:=2.0e-010;
                        cgbo:real:=1.0e-009;
                        mu0:real:=442.2; --Surface mobility of the channel for n or p channel
                        cbd:real:=5.0e-014;
                        cbs:real:=5.0e-014;
                        Iss:real:=1.0e-015;--Specific current
                        cox:real:=0.08e-6;--Capacitance per unit area of the gate oxide
                        l:real:=5.0e-007;--Effective channel length
                        vt0:real:=0.646;--Threshold voltage
                        T:real:=298.0;--Reference temperature
                        w:real:=5.0e-007;-- Effective channel width
                        twophi:real:=0.65;--Strong inversion surface potential
                        gamma:real:=0.25;--Bulk threshold parameter
                        lambda:real:=0.1);--Channel length modulation
port (terminal g :electrical;
      terminal s : electrical;
      terminal d : electrical;
      terminal b : electrical);
end entity nmos;
```

```

architecture arch_nmos of nmos is
  quantity Vt : real;
  quantity cgb : real;
  quantity cgs : real;
  quantity cgd : real;
  quantity Qgs : real;
  quantity Qgd : real;
  quantity Qbs : real;
  quantity Qbd : real;
  quantity Qgb : real;
  quantity Igd through g to d;
  quantity Vgd across g to d;
  quantity Igs through g to s;
  quantity Vgs across g to s;
  quantity Ibd through b to d;
  quantity Vbd across b to d;
  quantity IbdCS through b to d;
  quantity VbdCS across b to d;
  quantity Ibs through b to s;
  quantity Vbs across b to s;
  quantity Ids through d to s;
  quantity Vds across d to s;
  quantity Igb through g to b;
  quantity Vgb across g to b;
  quantity IbsCS through b to s;
  quantity VbsCS across b to s;
begin
  Vt==(vt0+(gamma*(sqrt(abs(twophi-Vbs))-sqrt(twophi))));
  if not Vgs'above(Vt) use
    cgb==(((cox*w)*1)+((2.0*cgbo)*1));
    cgs==(cgso*w);
    cgd==(cgdo*w);
  elsif Vgs'above(Vt) and not Vds'above(Vgs-Vt) use
    cgb==(cgbo*1);
    cgs==(w*(cgso+((0.5*cox)*1)));
    cgd==(w*(cgdo+((0.5*cox)*1)));
  else
    cgb==(cgbo*1);
    cgs==((cgso*w)+(((0.67*cox)*w)*1));
    cgd==(cgdo*w);
  end use;

```

```

break on Vds'above(Vgs-Vt),Vgs'above(Vt);
Qgs==(cgs*Vgs);
Qgd==(cgd*Vgd);
Qbs==(cbs*Vbs);
Qbd==(cbd*Vbd);
Qgb==(cgb*Vgb);
Igd==Qgd'dot;
Igs==Qgs'dot;
Ibd==Qbd'dot;
IbdCS==(Iss*(exp(((1.60217733e-19*Vbd)/(1.380658e-23*T)))-1.0));
Ibs==Qbs'dot;
if not Vgs'above(Vt) use      --Cut-off region
    Ids==0.0;
elseif Vgs'above(Vt) and not Vds'above(Vgs-Vt) use      --Linear region
    Ids====((((mu0*cox)*w)/l)*((Vgs-Vt)-(Vds/2.0))*Vds*(1.0+(lambda*Vds)));
else
    Ids====((((mu0*cox)*w)/(2.0*l))*(Vgs-Vt))*(Vgs-Vt)*(1.0+(lambda*Vds));
    --Saturation region
end use;
break on Vds'above(Vgs-Vt),Vgs'above(Vt);
Igb==Qgb'dot;
IbsCS==(Iss*(exp(((1.60217733e-19*Vbs)/(1.380658e-23*T)))-1.0));
end architecture ;

```

A.2 Testbench for The NMOS Transistor

```

library ieee;
    use ieee.math_real.all;
    use ieee.electrical_systems.all;
library ieee_proposed;
    use ieee_proposed.energy_systems.all;
    use ieee_proposed.thermal_systems.all;

entity test_curves_nmos is

end entity test_curves_nmos;
architecture ideal of test_curves_nmos is
terminal a,b,c,d,e,f,g,h:electrical;
begin

```

```
amp1: entity vdc(simple)
generic map(
  ampl => 5.0,
  trise => 0.1 ns,
  tfall => 0.1 ns,
  n => 20
)
port map(elec_p=>a,elec_n =>electrical_ref);
```

```
dc2:entity dc(equ)
generic map(
  amplitude => 0.0
)
port map(tlp=>b,tlm=>electrical_ref);
```

```
dc3:entity dc(equ)
generic map(
  amplitude => 0.25
)
port map(tlp=>c,tlm=>electrical_ref);
```

```
dc4:entity dc(equ)
generic map(
  amplitude => 0.5
)
port map(tlp=>d,tlm=>electrical_ref);
```

```
dc5:entity dc(equ)
generic map(
  amplitude => 0.75
)
port map(tlp=>e,tlm=>electrical_ref);
```

```
dc6:entity dc(equ)
generic map(
  amplitude => 1.0
)
port map(tlp=>f,tlm=>electrical_ref);
```

```
dc7:entity dc(equ)
generic map(
```

```

    amplitude => 1.25
)
port map(tlp=>g, tlm=>electrical_ref);

dc8:entity dc(equ)
generic map(
    amplitude => 2.0
)
port map(tlp=>h, tlm=>electrical_ref);

trans1:entity work.nmos(arch_nmos)
generic map(cgso=>0.5e-010,
            cgdo=>0.5e-010,
            cgbo=>2.0e-009,
            mu0=>442.2,
            cbd=>5.0e-014,
            cbs=>5.0e-014,
            Iss=>1.0e-015,
            cox=>0.08e-6,
            l=>0.5e-6,
            vt0=>0.1,
            T=>298.0,
            w=>3.0e-6,
            twophi=>0.65,
            gamma=>0.2,
            lambda=>0.6)
port map(g=>a, d=>h, s=>electrical_ref, b=>electrical_ref);

trans2:entity work.nmos(arch_nmos)
generic map(cgso=>0.5e-010,
            cgdo=>0.5e-010,
            cgbo=>2.0e-009,
            mu0=>442.2,
            cbd=>5.0e-014,
            cbs=>5.0e-014,
            Iss=>1.0e-015,
            cox=>0.08e-6,
            l=>0.5e-6,
            vt0=>0.1,
            T=>298.0,
            w=>3.0e-6,

```



```

                twophi=>0.65,
                gamma=>0.2,
                lambda=>0.6)
port map(g=>a, d=>h, s=>electrical_ref, b=>electrical_ref);

```

```

trans3:entity work.nmos(arch_nmos)

```

```

generic map(cgso=>0.5e-010,
            cgdo=>0.5e-010,
            cgbo=>2.0e-009,
            mu0=>442.2,
            cbd=>5.0e-014,
            cbs=>5.0e-014,
            Iss=>1.0e-015,
            cox=>0.08e-6,
            l=>0.5e-6,
            vt0=>0.1,
            T=>298.0,
            w=>3.0e-6,
            twophi=>0.65,
            gamma=>0.2,
            lambda=>0.6)
port map(g=>a, d=>h, s=>electrical_ref, b=>electrical_ref);

```

```

trans4:entity work.nmos(arch_nmos)

```

```

generic map(cgso=>0.5e-010,
            cgdo=>0.5e-010,
            cgbo=>2.0e-009,
            mu0=>442.2,
            cbd=>5.0e-014,
            cbs=>5.0e-014,
            Iss=>1.0e-015,
            cox=>0.08e-6,
            l=>0.5e-6,
            vt0=>0.1,
            T=>298.0,
            w=>3.0e-6,
            twophi=>0.65,
            gamma=>0.2,
            lambda=>0.6)
port map(g=>a, d=>h, s=>electrical_ref, b=>electrical_ref);

```

```

trans5:entity work.nmos(arch_nmos)
generic map(cgso=>0.5e-010,
            cgdo=>0.5e-010,
            cgbo=>2.0e-009,
            mu0=>442.2,
            cbd=>5.0e-014,
            cbs=>5.0e-014,
            Iss=>1.0e-015,
            cox=>0.08e-6,
            l=>0.5e-6,
            vt0=>0.1,
            T=>298.0,
            w=>3.0e-6,
            twophi=>0.65,
            gamma=>0.2,
            lambda=>0.6)
port map(g=>a, d=>h, s=>electrical_ref, b=>electrical_ref);

```

```

trans6:entity work.nmos(arch_nmos)
generic map(cgso=>0.5e-010,
            cgdo=>0.5e-010,
            cgbo=>2.0e-009,
            mu0=>442.2,
            cbd=>5.0e-014,
            cbs=>5.0e-014,
            Iss=>1.0e-015,
            cox=>0.08e-6,
            l=>0.5e-6,
            vt0=>0.1,
            T=>298.0,
            w=>3.0e-6,
            twophi=>0.65,
            gamma=>0.2,
            lambda=>0.6)
port map(g=>a, d=>h, s=>electrical_ref, b=>electrical_ref);

```

```

trans7:entity work.nmos(arch_nmos)
generic map(cgso=>0.5e-010,
            cgdo=>0.5e-010,
            cgbo=>2.0e-009,
            mu0=>442.2,

```

```

        cbd=>5.0e-014,
        cbs=>5.0e-014,
        Iss=>1.0e-015,
        cox=>0.08e-6,
        l=>0.5e-6,
        vt0=>0.1,
        T=>298.0,
        w=>3.0e-6,
        twophi=>0.65,
        gamma=>0.2,
        lambda=>0.6)
port map(g=>a, d=>h, s=>electrical_ref, b=>electrical_ref);
end architecture;

```

A.3 PMOS Transistor Code

```

library IEEE;
use IEEE.math_real.all;
use ieee.math_complex.all;
use ieee.std_logic_1164.all;
use ieee.electrical_systems.all;
entity pmos is generic (cgso:real:=2.62e-010;
        cgdo:real:=2.62e-010;
        mu0:real:=209.0;
        cbd:real:=5.0e-014;
        cbs:real:=5.0e-014;
        Iss:real:=1.0e-015;
        cox:real:=0.08e-6;
        l:real:=5.0e-007;
        vt0:real:=-0.92;
        cgbo:real:=1.0e-009;
        T:real:=298.0;
        w:real:=5.0e-007;
        twophi:real:=0.65;
        gamma:real:=0.55;
        lambda:real:=0.01);
port (terminal g :electrical;
        terminal s : electrical;
        terminal d : electrical;
        terminal b : electrical);

```

```

end entity pmos;
architecture arch_pmos of pmos is
    quantity Vt : real;
    quantity cgb : real;
    quantity cgs : real;
    quantity cgd : real;
    quantity Qsg : real;
    quantity Qdg : real;
    quantity Qsb : real;
    quantity Qdb : real;
    quantity Qbg : real;
    quantity Idg through d to g;
    quantity Vdg across d to g;
    quantity Isg through s to g;
    quantity Vsg across s to g;
    quantity Idb through d to b;
    quantity Vdb across d to b;
    quantity IdbCS through d to b;
    quantity VdbCS across d to b;
    quantity Isb through s to b;
    quantity Vsb across s to b;
    quantity Isd through s to d;
    quantity Vsd across s to d;
    quantity Ibg through b to g;
    quantity Vbg across b to g;
    quantity IsbCS through s to b;
    quantity VsbCS across s to b;
begin
    Vt==(vt0+(gamma*(sqrt(abs(twophi-Vsb))-sqrt(twophi))));
    --Vt==vt0;
    if Vsg'above(Vt) use
        cgb==(((cox*w)*1)+((2.0*cgbo)*1));
        cgs==(cgso*w);
        cgd==(cgdo*w);
    elsif Vsg'above(Vt) and not Vsd'above(Vsg-Vt) use
        cgb==(cgbo*1);
        cgs==(w*(cgso+((0.5*cox)*1)));
        cgd==(w*(cgdo+((0.5*cox)*1)));
    else
        cgb==(cgbo*1);
        cgs==((cgso*w)+(((0.67*cox)*w)*1));
    end if;
end architecture arch_pmos;

```

```

    cgd==(cgdo*w);
end use;
break on Vsd'above(Vt-Vsg),Vsg'above(-Vt);
Qsg==(cgs*Vsg);
Qdg==(cgd*Vdg);
Qsb==(cbs*Vsb);
Qdb==(cbd*Vdb);
Qbg==(cgb*Vbg);
Idg==Qdg'dot;
Isg==Qsg'dot;
Idb==Qdb'dot;
IdbCS==(Iss*(exp(((1.60217733e-19*Vdb)/(1.380658e-23*T)))-1.0));
Isb==Qsb'dot;
if not Vsg'above(Vt) use
    Isd==0.0;
elseif Vsg'above(Vt) and not Vsd'above(Vsg-Vt) use
    Isd====((((mu0*cox)*w)/l)*((Vsg-Vt)-(Vsd/2.0))*Vsd*(1.0+(lambda*Vsd)));
else
    Isd====((((mu0*cox)*w)/(2.0*l))*((Vsg-Vt))*((Vsg-Vt))*((1.0+(lambda*Vsd)));
end use;
break on Vsd'above(Vt-Vsg),Vsg'above(-Vt);
Ibg==Qbg'dot;
IsbCS==(Iss*(exp(((1.60217733e-19*Vsb)/(1.380658e-23*T)))-1.0));
end architecture ;

```

A.4 Testbench for The PMOS Transistor

```

library ieee;
use ieee.math_real.all;
use ieee.electrical_systems.all;
library ieee_proposed;
use ieee_proposed.energy_systems.all;
use ieee_proposed.thermal_systems.all;

entity test_curves_pmos is

end entity test_curves_pmos;
architecture ideal of test_curves_pmos is

```

```

terminal a,b,c,d,e,f,g,h:electrical;
begin
amp1: entity vdc(simple)
generic map(
  ampl => 5.0,
  trise => 0.1 ns,
  tfall => 0.1 ns,
  n => 20
)
port map(elec_p=>a,elec_n =>electrical_ref);

dc2:entity dc(equ)
generic map(
  amplitude => 0.0
)
port map(tlp=>b,tlm=>electrical_ref);

dc3:entity dc(equ)
generic map(
  amplitude => -0.25
)
port map(tlp=>c,tlm=>electrical_ref);

dc4:entity dc(equ)
generic map(
  amplitude => -0.5
)
port map(tlp=>d,tlm=>electrical_ref);

dc5:entity dc(equ)
generic map(
  amplitude => -0.75
)
port map(tlp=>e,tlm=>electrical_ref);

dc6:entity dc(equ)
generic map(
  amplitude => -1.0
)
port map(tlp=>f,tlm=>electrical_ref);

```

```

dc7:entity dc(equ)
generic map(
  amplitude => -1.25
)
port map(tlp=>g,tlm=>electrical_ref);

dc8:entity dc(equ)
generic map(
  amplitude =>2.0
)
port map(tlp=>h,tlm=>electrical_ref);

trans1: entity work.pmos(arch_pmos)
generic map(cgso=>1.5e-010,
           cgdo=>1.5e-010,
           mu0=>209.0,
           cbd=>5.0e-014,
           cbs=>5.0e-014,
           Iss=>1.0e-015,
           cox=>3.45e-6,
           l=>1.0e-6,
           vt0=>-0.2,
           cgbo=>4.0e-010,
           T=>298.0,
           w=>1.0e-6,
           twophi=>0.65,
           gamma=>0.69,
           lambda=>1.1)

port map(g=>a, d=>electrical_ref, s=>h, b=>electrical_ref);

trans2: entity work.pmos(arch_pmos)
generic map(cgso=>1.5e-010,
           cgdo=>1.5e-010,
           mu0=>209.0,
           cbd=>5.0e-014,
           cbs=>5.0e-014,
           Iss=>1.0e-015,
           cox=>3.45e-6,
           l=>1.0e-6,

```

```

        vt0=>-0.2,
        cgbo=>4.0e-010,
        T=>298.0,
        w=>1.0e-6,
        twophi=>0.65,
        gamma=>0.69,
        lambda=>1.1)

port map(g=>c, d=>electrical_ref, s=>a, b=>electrical_ref);

```

```

trans3: entity work.pmos(arch_pmos)
generic map(cgso=>1.5e-010,
            cgdo=>1.5e-010,
            mu0=>209.0,
            cbd=>5.0e-014,
            cbs=>5.0e-014,
            Iss=>1.0e-015,
            cox=>3.45e-6,
            l=>1.0e-6,
            vt0=>-0.2,
            cgbo=>4.0e-010,
            T=>298.0,
            w=>1.0e-6,
            twophi=>0.65,
            gamma=>0.69,
            lambda=>1.1)

```

```

port map(g=>d, d=>electrical_ref, s=>a, b=>electrical_ref);

```

```

trans4: entity work.pmos(arch_pmos)
generic map(cgso=>1.5e-010,
            cgdo=>1.5e-010,
            mu0=>209.0,
            cbd=>5.0e-014,
            cbs=>5.0e-014,
            Iss=>1.0e-015,
            cox=>3.45e-6,
            l=>1.0e-6,
            vt0=>-0.2,
            cgbo=>4.0e-010,
            T=>298.0,

```



```

        w=>1.0e-6,
        twophi=>0.65,
        gamma=>0.69,
        lambda=>1.1)

port map(g=>e, d=>electrical_ref, s=>a, b=>electrical_ref);

trans5: entity work.pmos(arch_pmos)
generic map(cgso=>1.5e-010,
            cgdo=>1.5e-010,
            mu0=>209.0,
            cbd=>5.0e-014,
            cbs=>5.0e-014,
            Iss=>1.0e-015,
            cox=>3.45e-6,
            l=>1.0e-6,
            vt0=>-0.2,
            cgbo=>4.0e-010,
            T=>298.0,
            w=>1.0e-6,
            twophi=>0.65,
            gamma=>0.69,
            lambda=>1.1)

port map(g=>f, d=>electrical_ref, s=>a, b=>electrical_ref);

trans6: entity work.pmos(arch_pmos)
generic map(cgso=>1.5e-010,
            cgdo=>1.5e-010,
            mu0=>209.0,
            cbd=>5.0e-014,
            cbs=>5.0e-014,
            Iss=>1.0e-015,
            cox=>3.45e-6,
            l=>1.0e-6,
            vt0=>-0.2,
            cgbo=>4.0e-010,
            T=>298.0,
            w=>1.0e-6,
            twophi=>0.65,
            gamma=>0.69,

```

```

        lambda=>1.1)

port map(g=>g, d=>electrical_ref, s=>a, b=>electrical_ref);

trans7: entity work.pmos(arch_pmos)
generic map(cgso=>1.5e-010,
            cgdo=>1.5e-010,
            mu0=>209.0,
            cbd=>5.0e-014,
            cbs=>5.0e-014,
            Iss=>1.0e-015,
            cox=>3.45e-6,
            l=>1.0e-6,
            vt0=>-0.2,
            cgbo=>4.0e-010,
            T=>298.0,
            w=>1.0e-6,
            twophi=>0.65,
            gamma=>0.69,
            lambda=>1.1)

port map(g=>h, d=>electrical_ref, s=>a, b=>electrical_ref);
end architecture;

```

A.5 Ascending DC Source

```

library MGC_AMS;
use MGC_AMS.conversion.all;
library IEEE;
use IEEE.electrical_systems.all;
entity vdc is
Generic(ampl: voltage :=5.0;
        trise : time    := 0.1ns;      -- Output rise time
        tfall : time    := 0.1ns;
        n     : integer := 20);      -- Output fall time);
Port(terminal elec_p, elec_n: electrical);
end entity vdc;

architecture simple of vdc is

```



```

        wait for 0.1ms;
        dout <= dout+a;
        wait for 0.1ms;
        dout <= dout+a;
        wait for 0.1ms;
        dout <= dout+a;
        wait for 0.1ms;
        dout <= dout+a;

    end loop;

end process multi;

v==dout'ramp(time2real(trise),time2real(tfall));
end architecture simple;

library MGC_AMS;
use MGC_AMS.conversion.all;
library IEEE;
use IEEE.electrical_systems.all;
entity test_vdc_as is

end entity test_vdc_as;
architecture ideal of test_vdc_as is
terminal w1:electrical;
begin
source: entity vdc(simple)
Generic map(ampl=>5.0,
            trise => 0.1ns,          -- Output rise time
            tfall => 0.1ns,
            n => 20)                -- Output fall time);
Port map(elec_p=>w1, elec_n=>electrical_ref);
end architecture;

```

A.6 DC Source

```

library IEEE;
use IEEE.math_real.all;
use IEEE.electrical_systems.all;

```

```

entity v_constant is
    generic(level: voltage);

    port(terminal pos,neg:electrical);

end entity v_constant;

architecture ideal of v_constant is
    quantity v across i through pos to neg;
begin
    v == level;
end architecture;

```

A.7 Diode Code

```

library IEEE;
    use IEEE.electrical_systems.all;
entity diode is

    generic (
        rD    : resistance := 1.0;    -- ON resistance of diode
        rDOFF : resistance := 1.0e8; -- OFF resistance of diode
        Vg    : voltage     := 0.6); -- Cut-in voltage of diode
    port (
        terminal p : electrical; -- anode
        terminal n : electrical); -- cathode

begin
    assert Vg >= 0.0
        report "The cut-in voltage of the diode should be >= 0.0";
    assert rD > 1.0e-6
        report "The diode forward resistance should be > 1.0e-6 ohms"
        severity warning;
    assert rDOFF > rD
        report "The diode off resistance should be greater than the on resistance";

end entity diode;
-----
architecture a1 of diode is

```

```

quantity Vd across Id through p to n;

begin
  if Vd'above(Vg) use
    Id == (Vd - Vg)/rD + Vg/rDOFF;
  else
    Id == Vd/rDOFF;
  end use;
  break on Vd'above(Vg);
end architecture a1;

```

A.8 Resistor Code

```

Library IEEE;--IEEE_proposed;
Use IEEE.std_logic_1164.all;
Use IEEE.math_real.all;
Use IEEE.electrical_systems.all;
entity resistor is
  generic (
    res : real);          -- resistance (no initial value)
  port (
    terminal p1, p2 : electrical);
end entity resistor;
architecture ideal of resistor is
  quantity v across i through p1 to p2;
begin
  v == i*res;
end architecture ideal;

```

A.9 Capacitor Code

```

library IEEE;
use ieee.electrical_systems.all;
entity capacitor is
  generic(c:real:=1.0e-6;
    r_leak:real:=10.0e6);
  port(terminal node1, node2:electrical);
end entity capacitor;

```

```

architecture leakage of capacitor is
quantity v_cap across i_cap, i_leak through node1 to node2;
begin
i_cap==c*v_cap'dot;
i_leak==v_cap/r_leak;
end architecture leakage;

```

A.10 Flip Flop D Code and Test Bench

```

library ieee;
use ieee.std_logic_1164.all;

entity dig_dff3 is

generic (
    tPLH  : time      := 0 ps;      -- Propagation delay from clock to out (L->H)
    tPHL  : time      := 0 ps;      -- Propagation delay from clock to out (H->L)
    edge  : real := 1.0;-- Triggering edge of the flip-flop neg for negative edge,
    -- pos for positive
    Q_init : std_logic := '0');    -- Initial value of output Qq

port (
    D      : in std_logic;          -- input port
    clk    : in std_logic;          -- input clock
    preset : in std_logic;          -- input preset
    clear  : in std_logic;          -- input clear
    Qq     : out std_logic;         -- output port
    nQ     : out std_logic);        -- inverted output port

begin

    assert (tPLH >= 0 ps)
        report "tPLH must be >= 0"
        severity error;
    assert (tPHL >= 0 ps)
        report "tPHL must be >= 0"
        severity error;
    assert (edge = 1.0 or edge = 0.0)

```

```

        report "'edge' defines either positive edge triggered or negative edge triggered"
        severity error;

end entity dig_dff3;
-----
architecture QuickStart of dig_dff3 is

    function edge2level ( edge : real)
        return std_logic is
    begin
        if edge = 1.0 then
            return '1';
        else
            return '0';
        end if;
    end edge2level;

    constant tPO : time := (tPLH + tPHL)/2; -- delay for others
    constant level : std_logic := edge2level(edge);
    signal Sig_nQ : std_logic := not Q_init;
    signal PresetClear : std_logic_vector(1 to 2);

begin

nQ <= Sig_nQ;

PresetClear(1) <= not(not preset);
PresetClear(2) <= not(not clear);

p1: process

begin

if Preset='0' and Clear='1' then
    Qq <= '1';
    Sig_nQ <= '0';
elsif Preset='1' and Clear='0' then
    Qq <= '1';
    Sig_nQ <= '0';
elsif Preset='0' and Clear='0' then

```



```

    Qq <= '1';
    Sig_nQ <= '1';
else
    Qq <= not(not Q_init);
end if;

loop
    wait on PresetClear, clk;

    if PresetClear="00" then
        Qq <= transport '1' after tPLH;
        Sig_nQ <= transport '1' after tPLH;
    elsif PresetClear="01" then
        Qq <= transport '1' after tPLH;
        Sig_nQ <= transport '0' after tPHL;
    elsif PresetClear="11" then
        Qq <= transport '0' after tPHL;
        Sig_nQ <= transport '1' after tPLH;
    elsif PresetClear="10" then
        --Qq <= transport '1' after tPLH;
        --Sig_nQ <= transport '0' after tPHL;
        -- DFF normal operation
        if clk'event and not(not clk)=level then
            case not(not D) is
                when '0'    => Qq <= transport '0' after tPHL;
                when '1'    => Qq <= transport '1' after tPLH;
                when 'U'    => Qq <= 'U';
                when 'X'    => Qq <= transport 'X' after tP0;
            end case;
            case (not D) is
                when '0'    => Sig_nQ <= transport '0' after tPHL;
                when '1'    => Sig_nQ <= transport '1' after tPLH;
                when 'U'    => Sig_nQ <= 'U';
                when 'X'    => Sig_nQ <= transport 'X' after tP0;
            end case;
        end if;
    else
        Qq <= transport 'X' after tP0;
        Sig_nQ <= transport 'X' after tP0;
    end if;
end if;

```

```

        end loop;
    end process;
end architecture QuickStart;

library ieee;
    use ieee.std_logic_1164.all;
entity test_ff3 is
end entity test_ff3;

architecture ideal of test_ff3 is
    signal data_in_u,clk1,preset1,clear1,data_in:std_logic;
begin
    ff1:entity dig_dff3(QuickStart)
    generic map(
        tPLH=> 0 ps,          -- Propagation delay from clock to out (L->H)
        tPHL=>0 ps,          -- Propagation delay from clock to out (H->L)
        edge=>1.0,
        -- Triggering edge of the flip-flop neg for negative edge, pos for positive edge
        Q_init=>'0')        -- Initial value of output Qq
    port map(
        D      =>data_in,          -- input port
        clk    =>clk1,            -- input clock
        preset =>preset1,         -- input preset
        clear  =>clear1,          -- input clear
        Qq     =>data_in_u,       -- output port
        nQ     =>data_in);        -- inverted output port
end architecture;

```

A.11 Analog to Digital Converter Code

```

library ieee;
use ieee.electrical_systems.all;
use IEEE.math_real.all;
use ieee.std_logic_1164.all;
entity a2d is
port(terminal v:electrical;

```

```

    signal data: out std_logic := '0');
end entity a2d;
architecture ideal of a2d is
constant v_il:real:=1.0e-13;
constant v_ih:real:=1.0;
quantity v_in across v;
begin
analog2std: process(v_in'above(v_il), v_in'above(v_ih)) is
begin
if not v_in'above(v_il) then
data<='0';
elsif v_in'above(v_ih) then
data<='1';
else
data<='X';
end if;
end process analog2std;
end architecture;

```

A.12 System Clock Code

```

library DISCIPLINES;
use DISCIPLINES.ELECTROMAGNETIC_SYSTEM.all;
library MGC_AMS;
    use MGC_AMS.conversion.all;
library ieee;
    use ieee.std_logic_1164.all;

entity clk_gen is

    generic (
        fclk          : real := 1.0e6;           -- Clock frequency.
        duty_cycle    : real := 0.5;           -- Duty cycle.
        tD_on         : time := 1 fs;          -- Clock on delay.
        tD_off        : time := 1 fs);         -- Clock off delay.

    port (
        signal clk     : inout std_logic := '0'; -- Clock output.
        signal enable  : in    std_logic := '0'); -- Clock enable.

```

```

begin
  assert fclk >= 0.0
    report "Clock frequency must be greater than zero!";
  assert 0.0<duty_cycle and duty_cycle<1.0
    report "Duty cycle must be between 0.0 and 1.0";
end entity clk_gen;
-----
architecture QuickStart of clk_gen is

  constant ton : time := real2time(duty_cycle/fclk);
  constant toff : time := real2time(1.0 - duty_cycle)/fclk;
  signal clk_tmp:std_logic:='1';
  signal enable_delay : std_logic;

begin -- QuickStart

  enable_delay <= enable when now=0ps
    else enable after tD_on when enable = '1'
    else enable after tD_off;

  clk <= clk_tmp;

process
begin

  loop

    wait on clk_tmp, enable_delay;

    if enable_delay = '1' and enable_delay'event then
      clk_tmp <= '0';
    elsif enable = '0' and enable_delay'event then
      clk_tmp <= '1';
    end if;

    if clk_tmp'event and enable_delay = '1' then
      if clk_tmp = '1' then
        clk_tmp <= '0' after ton;
      else
        clk_tmp <= '1' after toff;
      end if;
    end if;
  end if;
end if;

```

```

    end loop;
end process;
end architecture QuickStart;

```

A.13 Signal of The Reader

```

library MGC_AMS;
  use MGC_AMS.conversion.all;
library ieee;
use IEEE.std_logic_1164.all;
use IEEE.math_real.all;
use ieee.electrical_systems.all;
library ieee_proposed;
  use ieee_proposed.energy_systems.all;
  use ieee_proposed.thermal_systems.all;
entity func_rf is
generic( f:real:=2.4e9;
        v:real:=1.0;
        Vhi  : voltage := 1.2;  -- Voltage level corresponding to high level
        Vlo  : voltage := 0.0;  -- Voltage level corresponding to low level
        level : real    := 1.0;  -- 1: increasing pulses are Vhi
                                -- 0: increasing pulses are Vlo
        toff  : time    := 40ns; -- Duration time of OFF time
        ton   : time    := 1ns;  -- Duration time of ON time (increasing pulses)
        trise : time    := 0.1ns; -- Output rise time
        tfall : time    := 0.1ns; -- Output fall time
        n     : integer := 20);  -- Number of pulses per cycle);
port(terminal salida:electrical);
end entity func_rf;
architecture ideal of func_rf is
constant von      : real := vhi*level + vlo*(1.0-level);
  constant voff   : real := vlo*level + vhi*(1.0-level);
  constant dout_init : real := vlo*level + vhi*(1.0-level);
quantity vsin:real;
signal dout:real:=Vhi;
quantity Vsal across  salida;
quantity Isal through salida;
begin
Vsin==v*sin(2.0*math_pi*f*now);
multi: process is

```

```

begin -- process
  wait for 1ns;
  for i in 1 to n loop
    dout <= von;
    wait for 2.4us;
    dout <= voff;
    wait for 200ns;
    dout<=von;
    wait for 800ns;
    dout<=voff;
    wait for 200ns;
    dout<=von;
    wait for 200ns;
    dout<=voff;
    wait for 200ns;
    dout<=von;
    wait for 800ns;
    dout<=voff;
    wait for 200ns;
    dout<=von;
    wait for 2.4us;
    dout<=voff;
    wait for 200ns;
    dout<=von;
    wait for 800ns;
    dout<=voff;
    wait for 200ns;
    dout<=von;
    wait for 800ns;
    dout<=voff;
    wait for 200ns;
    dout<=von;
    wait for 800ns;
    dout<=voff;
    wait for 200ns;
    dout<=von;
    wait for 800ns;
    dout<=voff;
    wait for 200ns;
    dout<=von;
    wait for 800ns;
    dout<=voff;
    wait for 200ns;
    dout<=von;
    wait for 3us;
  end loop
end process

```

```

        dout<=voff;

    end loop;

end process multi;

Vsal==vsin*dout'ramp(time2real(trise),time2real(tfall));
end architecture;

library MGC_AMS;
    use MGC_AMS.conversion.all;
library ieee;
    use IEEE.std_logic_1164.all;
    use IEEE.math_real.all;
        use ieee.electrical_systems.all;

entity test_func_rf is
end entity test_func_rf;
architecture ideal of test_func_rf is
terminal senal:electrical;
begin
source: entity work.func_rf(ideal)
generic map(f=>2.4e9,
            v=>1.0,
            Vhi=> 0.3,           -- Voltage level corresponding to high level
            Vlo=>0.0,           -- Voltage level corresponding to low level
            level=>1.0,         -- 1: increasing pulses are Vhi
                                -- 0: increasing pulses are Vlo
            toff=>200ns,        -- Duration time of OFF time
            ton =>200ns,        -- Duration time of ON time (increasing pulses)
            trise=> 0.1ns,      -- Output rise time
            tfall=> 0.1ns,     -- Output fall time
            n => 20)            -- Number of pulses per cycle);
    port map(salida=>senal);
resis: entity resistor(ideal)
generic map(res=>1.0e6)
port map(p1=>senal, p2=>electrical_ref);
end architecture;

```

A.14 Rectifier Code

```
Library ieee;
Use ieee.std_logic_1164.all;
Use ieee.math_real.all;
use IEEE.ELECTRICAL_SYSTEMS.all;
entity Rect_Diodes is
port(terminal vmas,vmenos,rf:electrical);
end entity Rect_Diodes;
architecture ideal of Rect_Diodes is
terminal w1,w2,w3,w4,w5,w6,w7,w8,w9,w10,w11,w12,w13,w14:electrical;
begin

cap1: entity work.capacitor(leakage)
generic map(c=>100.0e-12,
r_leak=>10.0e6)
port map(node1=>rf, node2=>w1);

cap2: entity work.capacitor(leakage)
generic map(c=>100.0e-12,
r_leak=>10.0e6)
port map(node1=>rf, node2=>w2);

cap3: entity work.capacitor(leakage)
generic map(c=>100.0e-12,
r_leak=>10.0e6)
port map(node1=>rf, node2=>w3);

diode1: entity work.diode(a1)
generic map(
    rD    => 1.0,    -- ON resistance of diode
    rDOFF => 1.0e10, -- OFF resistance of diode
    Vg    => 0.1)   -- Cut-in voltage of diode
port map(p =>w1,n =>Vmas); -- cathode

diode2: entity work.diode(a1)
generic map(
    rD    => 1.0,    -- ON resistance of diode
    rDOFF => 1.0e8,  -- OFF resistance of diode
```



```

    Vg    => 0.1) -- Cut-in voltage of diode
port map(p =>w4,n =>w1); -- cathod

diode3: entity work.diode(a1)
generic map(
    rD    => 1.0,  -- ON resistance of diode
    rDOFF => 1.0e8, -- OFF resistance of diode
    Vg    => 0.1) -- Cut-in voltage of diode
port map(p =>w2,n =>w4); -- cathod

diode4: entity work.diode(a1)
generic map(
    rD    => 1.0,  -- ON resistance of diode
    rDOFF => 1.0e8, -- OFF resistance of diode
    Vg    => 0.1) -- Cut-in voltage of diode
port map(p =>w5,n =>w2); -- cathod

diode5: entity work.diode(a1)
generic map(
    rD    => 1.0,  -- ON resistance of diode
    rDOFF => 1.0e8, -- OFF resistance of diode
    Vg    => 0.1) -- Cut-in voltage of diode
port map(p =>w3,n =>w5); -- cathod

diode6: entity work.diode(a1)
generic map(
    rD    => 1.0,  -- ON resistance of diode
    rDOFF => 1.0e8, -- OFF resistance of diode
    Vg    => 0.1) -- Cut-in voltage of diode
port map(p =>electrical_ref,n =>w3); -- cathod

cap4: entity work.capacitor(leakage)
generic map(c=>100.0e-12,
    r_leak=>10.0e6)
port map(node1=>vmas, node2=>electrical_ref);

cap5: entity work.capacitor(leakage)
generic map(c=>100.0e-12,
    r_leak=>10.0e6)
port map(node1=>w4, node2=>electrical_ref);

```

```

cap6: entity work.capacitor(leakage)
generic map(c=>100.0e-12,
           r_leak=>10.0e6)
port map(node1=>w5, node2=>electrical_ref);
-----
-- parte negativa
cap7: entity work.capacitor(leakage)
generic map(c=>100.0e-12,
           r_leak=>10.0e6)
port map(node1=>rf, node2=>w6);

cap8: entity work.capacitor(leakage)
generic map(c=>100.0e-12,
           r_leak=>10.0e6)
port map(node1=>rf, node2=>w7);

cap9: entity work.capacitor(leakage)
generic map(c=>100.0e-12,
           r_leak=>10.0e6)
port map(node1=>rf, node2=>w8);

diode7: entity work.diode(a1)
generic map(
    rD    => 1.0,    -- ON resistance of diode
    rDOFF => 1.0e8, -- OFF resistance of diode
    Vg    => 0.1) -- Cut-in voltage of diode
port map(p =>w6,n =>electrical_ref); -- cathod

diode8: entity work.diode(a1)
generic map(
    rD    => 1.0,    -- ON resistance of diode
    rDOFF => 1.0e8, -- OFF resistance of diode
    Vg    => 0.1) -- Cut-in voltage of diode
port map(p =>w9,n =>w6); -- cathod

diode9: entity work.diode(a1)
generic map(
    rD    => 1.0,    -- ON resistance of diode
    rDOFF => 1.0e8, -- OFF resistance of diode
    Vg    => 0.1) -- Cut-in voltage of diode
port map(p =>w7,n =>w9); -- cathod

```

```

diode10: entity work.diode(a1)

generic map(
    rD    => 1.0,    -- ON resistance of diode
    rDOFF => 1.0e8, -- OFF resistance of diode
    Vg    => 0.1) -- Cut-in voltage of diode
port map(p =>w10,n =>w7); -- cathod

diode11: entity work.diode(a1)
generic map(
    rD    => 1.0,    -- ON resistance of diode
    rDOFF => 1.0e8, -- OFF resistance of diode
    Vg    => 0.1) -- Cut-in voltage of diode
port map(p =>w8,n =>w10); -- cathod

diode12: entity work.diode(a1)
generic map(
    rD    => 1.0,    -- ON resistance of diode
    rDOFF => 1.0e8, -- OFF resistance of diode
    Vg    => 0.1) -- Cut-in voltage of diode
port map(p =>vmenos,n =>w8); -- cathod

cap10: entity work.capacitor(leakage)
generic map(c=>100.0e-12,
    r_leak=>10.0e6)
port map(node1=>w9, node2=>electrical_ref);

cap11: entity work.capacitor(leakage)
generic map(c=>100.0e-12,
    r_leak=>10.0e6)
port map(node1=>w10, node2=>electrical_ref);

cap12: entity work.capacitor(leakage)
generic map(c=>100.0e-12,
    r_leak=>10.0e6)
port map(node1=>vmenos, node2=>electrical_ref);
end architecture;

```

A.15 Power-on-Reser Code

```
library ieee;
use ieee.electrical_systems.all;
use ieee.std_logic_1164.all;
entity por is
port (terminal vmas, vmenos, a1, b1:electrical;
signal por0:out std_logic;
signal por_u:out std_logic);
end entity por;

architecture ideal of por is
terminal w1:electrical;
signal uf:std_logic:='1';
signal ovf:std_logic;
signal a11:std_logic_vector(0 downto 0);
signal b11:std_logic_vector(0 downto 0);
signal encode1:std_logic:='0';
signal encode2:std_logic:='0';
signal por1:std_logic;
quantity v1 across b1;
quantity vm1 across vmenos;
begin
encode1<='1' after 0.1ns;
encode2<='1' after 0.1ns;
t0: entity mos_book(pmos)
generic map(
w=>1.2,
l=>2.0,
threshold=>-0.15,
r=>10.0e10,
k=>120.0e-6)
port map(g=>vmenos,d=>w1,s=>vmas,b=>electrical_ref);
t1: entity mos_book(pmos)
generic map(
w=>1.2,
l=>2.0,
threshold=>-0.15,
r=>10.0e10,
k=>120.0e-6)
port map(g=>a1,d=>b1,s=>w1,b=>electrical_ref);
```

```

t2: entity mos_book(pmos)
generic map(
    w=>1.2,
    l=>2.2,
    threshold=>-0.15,
    r=>10.0e10,
    k=>120.0e-6)
port map(g=>b1,d=>a1,s=>w1,b=>electrical_ref);
t3: entity mos_book(nmos)
generic map(
    w=>1.2,
    l=>2.2,
    threshold=>0.15,
    r=>10.0e10,
    k=>120.0e-6)
port map(g=>a1,d=>b1,s=>vmenos,b=>electrical_ref);
t4: entity mos_book(nmos)
generic map(
    w=>1.2,
    l=>2.0,
    threshold=>0.15,
    r=>10.0e10,
    k=>120.0e-6)
port map(g=>b1,d=>a1,s=>vmenos,b=>electrical_ref);
cap1: entity work.capacitor(leakage)
generic map(c=>60.0e-15,
    r_leak=>10.0e6)
port map(node1=>w1, node2=>vmenos);
a2d1: entity a2d_1(QuickStart)
    generic map(
        n      => 1,          -- Converter resolution.
        Rref   => 10.0e3,     -- Reference input resistance.
        tD_out => 0 ns)     -- Output delay.
    port map(
        dout=>a11, -- Converter digital output.
        underflow=>uf, -- Underflow output.
        overflow=>ovf, -- Overflow output.
        encode=>encode1, -- Encode input
        ref_plus=>vmas, -- Positive Reference.
        ref_minus=>vmenos, -- Negative Reference.
        ain=>a1); -- Converter analog input.

```

```

b11(0)<='1' when (not v1'above(-0.389)) else '0';
dignor:entity dig_nor21(QuickStart)
  generic map(
    tPLH => 0 ns,          -- Propagation delay from low to high
    tPHL => 0 ns)         -- Propagation delay from high to low
  port map(
    din1 => a11,          -- first input port
    din2 => b11,          -- second input port
    dout => por1);        -- output port
por0<=por1;
por_u<=not por1;
end architecture;

```

A.16 Detector Code

```

library ieee;
use ieee.electrical_systems.all;
use IEEE.math_real.all;
use IEEE.FUNDAMENTAL_CONSTANTS.all;
entity detector is
port(terminal rf_in,Venv,Vref:electrical);
end entity detector;
architecture ideal of detector is
begin
res1: entity work.resistor(ideal)
generic map(res=>15.0e3)
port map(p1=>rf_in,p2=>electrical_ref);
d1: entity diode1(ideal)
  generic map(Isat=> 1.0e-5) -- Saturation current [Amps]
  port map( p=>Venv, n=>rf_in);
  cap1: entity work.capacitor(leakage)
generic map(c=>600.0e-15,
r_leak=>1.0e6)
port map(node1=>Venv, node2=>electrical_ref);
d2: entity diode1(ideal)
  generic map(Isat=> 1.0e-5) -- Saturation current [Amps]
  port map(p =>Vref,n =>rf_in); -- cathode
  cap2: entity work.capacitor(leakage)
generic map(c=>500.0e-13,

```

```

r_leak=>10000.0e6)
port map(node1=>Vref, node2=>electrical_ref);
d3: entity diode1(ideal)
    generic map(Isat=> 1.0e-5) -- Saturation current [Amps]
    port map(p =>electrical_ref,n =>Vref); -- cathode
    end architecture;
library ieee;
use ieee.electrical_systems.all;
entity test_det is
end entity test_det;
architecture ideal of test_det is
terminal senal,env,ref:electrical;
begin
source: entity work.func_rf(ideal)
generic map(f=>2.4e9,
    v=>1.0,
    Vhi=> 0.3,          -- Voltage level corresponding to high level
    Vlo=>0.0,          -- Voltage level corresponding to low level
    level=>1.0,        -- 1: increasing pulses are Vhi
                       -- 0: increasing pulses are Vlo
    toff=>200ns,       -- Duration time of OFF time
    ton =>200ns,       -- Duration time of ON time (increasing pulses)
    trise=> 5ns,       -- Output rise time
    tfall=> 5ns,      -- Output fall time
    n => 20)          -- Number of pulses per cycle);
port map(salida=>senal);
det1: entity detector(ideal)
port map(rf_in=>senal,Venv=>env,Vref=>ref);

end architecture;

```

A.17 Comparator Code

```

Library ieee;
Use ieee.std_logic_1164.all;
Use ieee.math_real.all;
use IEEE.ELECTRICAL_SYSTEMS.all;
library IEEE_PROPOSED;
use IEEE_PROPOSED.thermal_systems.all;
use IEEE_PROPOSED.energy_systems.all;

```

```

entity comparator_nmos is
port(terminal Venv, G_n, Vref, Vmas, Vmenos, Iref_n, clk: electrical;
signal clk_out: inout std_logic;
signal ck1:out std_logic);
end entity;
architecture ideal of comparator_nmos is
terminal w1,w2,w3,w4:electrical;
quantity vclk across clk;
quantity vref1 across Vref;
quantity venv1 across Venv;
quantity vmas1 across Vmas;
begin
trans1: entity work.pmos(arch_pmos)
generic map(cgso=>1.5e-010,
            cgdo=>1.5e-010,
            mu0=>209.0,
            cbd=>5.0e-014,
            cbs=>5.0e-014,
            Iss=>1.0e-015,
            cox=>3.45e-6,
            l=>0.5e-6,
            vt0=>-0.1,
            cgbo=>4.0e-010,
            T=>298.0,
            w=>1.20e-6,
            twophi=>0.65,
            gamma=>0.69,
            lambda=>1.1)

port map(g=>w1, d=>w1, s=>Vmas, b=>electrical_ref);
trans2:entity work.nmos(arch_nmos)
generic map(cgso=>0.5e-010,
            cgdo=>0.5e-010,
            cgbo=>2.0e-009,
            mu0=>442.2,
            cbd=>5.0e-014,
            cbs=>5.0e-014,
            Iss=>1.0e-015,
            cox=>0.08e-6,
            l=>0.5e-6,

```



```

        vt0=>0.1,
        T=>298.0,
        w=>3.0e-6,
        twophi=>0.65,
        gamma=>0.2,
        lambda=>0.8)
port map(g=>Venv, d=>w1, s=>w4, b=>electrical_ref);
trans3:entity work.nmos(arch_nmos)
generic map(cgso=>0.5e-010,
            cgdo=>0.5e-010,
            cgbo=>2.0e-009,
            mu0=>442.2,
            cbd=>5.0e-014,
            cbs=>5.0e-014,
            Iss=>1.0e-015,
            cox=>0.08e-6,
            l=>2.5e-6,
            vt0=>0.1,
            T=>298.0,
            w=>5.0e-6,
            twophi=>0.65,
            gamma=>0.2,
            lambda=>0.6)
port map(g=>Iref_n, d=>w4, s=>Vmenos, b=>electrical_ref);
trans4:entity work.nmos(arch_nmos)
generic map(cgso=>0.5e-010,
            cgdo=>0.5e-010,
            cgbo=>2.0e-009,
            mu0=>442.2,
            cbd=>5.0e-014,
            cbs=>5.0e-014,
            Iss=>1.0e-015,
            cox=>0.08e-6,
            l=>0.5e-6,
            vt0=>0.1,
            T=>298.0,
            w=>1.2e-6,
            twophi=>0.65,
            gamma=>0.2,
            lambda=>0.6)
port map(g=>G_n, d=>w3, s=>w4, b=>electrical_ref);

```

```

trans5: entity work.pmos(arch_pmos)
generic map(cgso=>1.5e-010,
            cgdo=>1.5e-010,
            mu0=>209.0,
            cbd=>5.0e-014,
            cbs=>5.0e-014,
            Iss=>1.0e-015,
            cox=>3.45e-6,
            l=>0.5e-6,
            vt0=>-0.1,
            cgbo=>4.0e-010,
            T=>298.0,
            w=>1.20e-6,
            twophi=>0.65,
            gamma=>0.69,
            lambda=>1.1)

port map(g=>w1, d=>w3, s=>Vmas, b=>electrical_ref);
trans6: entity work.pmos(arch_pmos)
generic map(cgso=>1.5e-010,
            cgdo=>1.5e-010,
            mu0=>209.0,
            cbd=>5.0e-014,
            cbs=>5.0e-014,
            Iss=>1.0e-015,
            cox=>3.45e-6,
            l=>0.5e-6,
            vt0=>-0.1,
            cgbo=>4.0e-010,
            T=>298.0,
            w=>1.80e-6,
            twophi=>0.65,
            gamma=>0.69,
            lambda=>1.1)

port map(g=>w3, d=>clk, s=>Vmas, b=>electrical_ref);
trans7: entity work.nmos(arch_nmos)
generic map(cgso=>0.5e-010,
            cgdo=>0.5e-010,
            cgbo=>2.0e-009,
            mu0=>442.2,

```

```

        cbd=>5.0e-014,
        cbs=>5.0e-014,
        Iss=>1.0e-015,
        cox=>0.08e-6,
        l=>2.5e-6,
        vt0=>0.1,
        T=>298.0,
        w=>5.0e-6,
        twophi=>0.65,
        gamma=>0.2,
        lambda=>0.6)
port map(g=>Iref_n, d=>clk, s=>Vmenos, b=>electrical_ref);
trans8:entity work.nmos(arch_nmos)
generic map(cgso=>0.5e-010,
            cgdo=>0.5e-010,
            cgbo=>2.0e-009,
            mu0=>442.2,
            cbd=>5.0e-014,
            cbs=>5.0e-014,
            Iss=>1.0e-015,
            cox=>0.08e-6,
            l=>0.5e-6,
            vt0=>0.1,
            T=>298.0,
            w=>3.0e-6,
            twophi=>0.65,
            gamma=>0.2,
            lambda=>0.6)
port map(g=>Vref, d=>w3, s=>w4, b=>electrical_ref);
clk_out<='0' when ((not vclk'above(635.0e-3))) else '1';

inv1: entity dig_inv(QuickStart)
generic map(
    tplh => 3 ns,           -- Propagation delay from low to high
    tphl => 3 ns)         -- Propagation delay from high to low
port map(
    din =>clk_out,         -- input port
    dout =>ck1);          -- output port
end;
```

A.18 Shift Register Logic Code

```
library ieee;
  use ieee.std_logic_1164.all;
  use ieee.electrical_systems.all;
entity ctrl_log is

port(terminal Gn_a,Gp_a,if_en_a:electrical;
     signal Data_in: in std_logic;
     signal Data_in_u: in std_logic;
     signal if1: in std_logic;
     signal if1_en: out std_logic;
     signal Gn: out std_logic;
     signal Gp: out std_logic;
     signal ck1: in std_logic;
     signal por: in std_logic;
     signal por_u: in std_logic;
     signal Data_out_u: out std_logic;
     signal a1:in std_logic;           --direccion
     signal a2:in std_logic);        --direccion

end entity ctrl_log;

architecture ideal of ctrl_log is

quantity Gn_analogic_v across Gn_analogic_i through Gn_a;
quantity Gp_analogic_v across Gp_analogic_i through Gp_a;
quantity if_en_analogic_v across if_en_analogic_i through if_en_a;

constant tr:real:=0.1e-6;
constant tf:real:=0.1e-6;
signal Data_enable: std_logic;
signal Data_enable_u: std_logic;
signal in_or1: std_logic;
signal in_nand1: std_logic;
signal in_and1: std_logic;
signal in_nor1: std_logic;
signal in_nand2: std_logic;
signal in_nand22: std_logic;
signal in_nand23: std_logic;
signal ck2:std_logic;
```

```

signal in_nor2: std_logic;
signal in_nor22: std_logic;
signal in_nor3: std_logic;
signal in_nor33: std_logic;
signal d1: std_logic;
signal d_u1: std_logic;
signal q1:std_logic;
signal q_u1:std_logic;
signal q2:std_logic;
signal q_u2:std_logic;
signal q3:std_logic;
signal in_nand6:std_logic;
signal in_nand62:std_logic;
signal out_nand6:std_logic;
signal q31: std_logic;
signal ck4: std_logic;
signal if1_en1:std_logic:= '0';
signal Gn1:std_logic;
signal Gp1:std_logic;
signal Data_out_u_1:std_logic;
signal Gn_i:real:=0.0;
signal Gp_i:real:=0.0;
signal if_en_i:real:=0.0;
signal vh:real:=1.0;
signal vl:real:=0.0;
signal in_nor1_n:std_logic;
signal in_nor22_n:std_logic;
signal Data_out_u_11:std_logic;

begin

nand1: entity dig_nand3(QuickStart)

generic map(
    tPLH => 3 ns,           -- Propagation delay from low to high
    tPHL => 3 ns)          -- Propagation delay from high to low

port map(
    din1=>Data_enable,     -- first input port
    din2=>if1,             -- second input port
    din3 =>if1_en1,       -- third input port

```

```

dout =>Gp1);          -- output port

    Gp<=Gp1;
inv1: entity dig_inv(QuickStart)

generic map(
    tplh => 3 ns,          -- Propagation delay from low to high
    tphl => 3 ns)        -- Propagation delay from high to low

port map(
    din =>Gp1,            -- input port
    dout =>Gn1);         -- output port

Gn<=Gn1;

inv2: entity dig_inv(QuickStart)

generic map(
    tplh => 3 ns,          -- Propagation delay from low to high
    tphl => 3 ns)        -- Propagation delay from high to low

port map(
    din =>if1,            -- input port
    dout =>in_or1);      -- output port

or1: entity dig_or2(QuickStart)

generic map(
    tPLH => 3 ns,          -- Propagation delay from low to high
    tPHL => 3 ns)        -- Propagation delay from high to low

port map(
    din1 =>in_or1,       -- first input port
    din2 =>ck1,         -- second input port
    dout =>in_nand1);    -- output port

nand2: entity dig_nand2(QuickStart)

generic map(
    tPLH => 3 ns,          -- Propagation delay from low to high
    tPHL => 3 ns)        -- Propagation delay from high to low

```

```

port map(
  din1 =>in_nand1,    -- first input port
  din2 =>Data_enable_u,  -- second input port
  dout =>in_and1);    -- output port .

and1: entity dig_and2(QuickStart)

generic map(
  tPLH => 3 ns,      -- Propagation delay from low to high
  tPHL => 3 ns)     -- Propagation delay from high to low

port map(
  din1 =>in_and1,    -- first input port
  din2 =>Data_out_u_1,  -- second input port
  dout =>in_nor1);  -- output port

nor1: entity dig_nor2(QuickStart)

generic map(
  tPLH => 3 ns,      -- Propagation delay from low to high
  tPHL => 3 ns)     -- Propagation delay from high to low

port map(
  din1 =>in_nor1,    -- first input port
  din2 =>por,        -- second input port
  dout =>if1_en1);  -- output port

if1_en<=if1_en1;

nand3: entity dig_nand2(QuickStart)

generic map(
  tPLH => 3 ns,      -- Propagation delay from low to high
  tPHL => 3 ns)     -- Propagation delay from high to low

port map(
  din1 =>Data_enable,  -- first input port
  din2 =>ck1,        -- second input port
  dout =>in_nand2);  -- output port

```

```

or2: entity dig_or2(QuickStart)

    generic map(
        tPLH => 3 ns,           -- Propagation delay from low to high
        tPHL => 3 ns)          -- Propagation delay from high to low

    port map(
        din1 =>Data_enable,    -- first input port
        din2 =>Data_out_u_1,   -- second input port
        dout =>in_nand22);     -- output port

nand4: entity dig_nand2(QuickStart)

    generic map(
        tPLH => 3 ns,           -- Propagation delay from low to high
        tPHL => 3 ns)          -- Propagation delay from high to low

    port map(
        din1 =>if1,           -- first input port
        din2 =>Data_enable_u,  -- second input port
        dout =>in_nand23);    -- output port

nand5: entity dig_nand3(QuickStart)

    generic map(
        tPLH => 3 ns,           -- Propagation delay from low to high
        tPHL => 3 ns)          -- Propagation delay from high to low

    port map(
        din1=>in_nand2,       -- first input port
        din2=>in_nand22,      -- second input port
        din3 =>in_nand23,     -- third input port
        dout =>ck2);         -- output port

and2: entity dig_and2(QuickStart)

    generic map(
        tPLH => 3 ns,           -- Propagation delay from low to high
        tPHL => 3 ns)          -- Propagation delay from high to low

    port map(

```



```

    din1 =>Data_in,           -- first input port
    din2 =>Data_enable_u,    -- second input port
    dout =>in_nor2);        -- output port

and3: entity dig_and2(QuickStart)

generic map(
    tPLH => 3 ns,           -- Propagation delay from low to high
    tPHL => 3 ns)          -- Propagation delay from high to low

port map(
    din1 =>Data_in_u,       -- first input port
    din2 =>Data_enable_u,   -- second input port
    dout =>in_nor3);       -- output port

nor2: entity dig_nor2(QuickStart)

generic map(
    tPLH => 3 ns,           -- Propagation delay from low to high
    tPHL => 3 ns)          -- Propagation delay from high to low

port map(
    din1 =>in_nor2,        -- first input port
    din2 =>in_nor22,       -- second input port
    dout =>d_u1);          -- output port

nor3: entity dig_nor2(QuickStart)

generic map(
    tPLH => 3 ns,           -- Propagation delay from low to high
    tPHL => 3 ns)          -- Propagation delay from high to low

port map(
    din1 =>in_nor3,        -- first input port
    din2 =>in_nor33,       -- second input port
    dout =>d1);            -- output port

ffd1: entity ff_cu(ideal)

generic map(

```

```

tPLH => 5 ns,      -- Propagation delay from clock to out (L->H)
tPHL => 5 ns,      -- Propagation delay from clock to out (H->L)
edge  => 1.0,      -- Triggering edge of the flip-flop neg for negative edge,
--pos for positive edge; 1 para positivo, 0 para negativo
Q_init => '0')     -- Initial value of output Qq

port map(
  D      =>d1,      -- input port
  clk    =>ck2,     -- input clock
  preset =>por_u,   -- input preset
  clear  =>por,     -- input clear
  Qq     =>q1,      -- output port
  nQ     =>q_u1);   -- inverted output port

ffd2: entity ff_cu(ideal)

generic map(
  tPLH => 5 ns,      -- Propagation delay from clock to out (L->H)
  tPHL => 5 ns,      -- Propagation delay from clock to out (H->L)
  edge  => 1.0,      -- Triggering edge of the flip-flop neg for negative edge,
--pos for positive edge; 1 para positivo, 0 para negativo
Q_init => '0')     -- Initial value of output Qq

port map(
  D      =>q1,      -- input port
  clk    =>ck2,     -- input clock
  preset =>por_u,   -- input preset
  clear  =>por,     -- input clear
  Qq     =>q2,      -- output port
  nQ     =>q_u2);   -- inverted output port

ffd3: entity ff_cu(ideal)

generic map(
  tPLH => 5 ns,      -- Propagation delay from clock to out (L->H)
  tPHL => 5 ns,      -- Propagation delay from clock to out (H->L)
  edge  => 1.0,      -- Triggering edge of the flip-flop neg for negative edge,
--pos for positive edge; 1 para positivo, 0 para negativo
Q_init => '0')     -- Initial value of output Qq

port map(

```

```

D      =>q2,          -- input port
clk    =>ck2,         -- input clock
preset =>por_u,       -- input preset
clear  =>por,         -- input clear
Qq     =>q3,          -- output port
nQ     =>Data_out_u_1); -- inverted output port

nor4: entity dig_nor2(QuickStart)
generic map(
  tPLH => 3 ns,      -- Propagation delay from low to high
  tPHL => 3 ns)      -- Propagation delay from high to low

port map(
  din1 =>Data_enable_u, -- first input port
  din2 =>q3,             -- second input port
  dout =>in_nor33);     -- output port

nor5: entity dig_nor2(QuickStart)

generic map(
  tPLH => 3 ns,      -- Propagation delay from low to high
  tPHL => 3 ns)      -- Propagation delay from high to low

port map(
  din1 =>Data_enable_u, -- first input port
  din2 =>Data_out_u_1,  -- second input port
  dout =>in_nor22);    -- output port

xnor1: entity dig_xnor(QuickStart)

generic map(
  tplh => 3 ns,      -- Propagation delay from low to high
  tphl => 3 ns)      -- Propagation delay from high to low

port map(
  din1=>a1, -- first input port
  din2=>q2, -- second input port
  dout=>in_nand6); -- output port

```

```

xnor2: entity dig_xnor(QuickStart)

generic map(
    tplh => 3 ns,           -- Propagation delay from low to high
    tphl => 3 ns)          -- Propagation delay from high to low

port map(
    din1=>a2, -- first input port
    din2=>q1, -- second input port
    dout=>in_nand62);      -- output port
--q31<=q3 after 50ns;
nand6: entity dig_nand3(QuickStart)

generic map(
    tPLH => 3 ns,           -- Propagation delay from low to high
    tPHL => 3 ns)          -- Propagation delay from high to low

port map(
    din1=>in_nand6,      -- first input port
    din2=>in_nand62,     -- second input port
    din3 =>q3,           -- third input port
    dout =>out_nand6);   -- output port

nor6: entity dig_nor2(QuickStart)

generic map(
    tPLH => 3 ns,           -- Propagation delay from low to high
    tPHL => 3 ns)          -- Propagation delay from high to low

port map(
    din1 =>out_nand6,     -- first input port
    din2 =>Data_enable,   -- second input port
    dout =>ck4);          -- output port

ffd4: entity ff_cu(ideal)

generic map(
    tPLH  => 5 ns,        -- Propagation delay from clock to out (L->H)
    tPHL  => 5 ns,        -- Propagation delay from clock to out (H->L)
    edge  => 1.0,        -- Triggering edge of the flip-flop neg for negative edge,
    --pos for positive edge; 1 para positivo, 0 para negativo

```

```

Q_init => '0')      -- Initial value of output Qq

port map(
  D      =>Data_enable_u,      -- input port
  clk    =>ck4,                -- input clock
  preset =>por_u,              -- input preset
  clear  =>por,                -- input clear
  Qq     =>Data_enable,        -- output port
  nQ     =>Data_enable_u);     -- inverted output port

Gn_i<= vh when (Gn1='1') else vl;
Gp_i<= vh when (Gp1='1') else vl;
if_en_i<= vh when (if1_en1='1') else vl;
Gn_analogic_v == Gn_i'Ramp(tr,tf);
Gp_analogic_v == Gp_i'Ramp(tr,tf);
if_en_analogic_v == if_en_i'Ramp(tr,tf);
end architecture;

```

A.19 Current Reference Code

```

library ieee;
use ieee.electrical_systems.all;
entity current_ref_nmos is
port(terminal vmas, vmenos, vref, vrefp, vrefn, irefp, irefn:electrical);
end entity current_ref_nmos;
architecture ideal of current_ref_nmos is

begin

trans1: entity work.pmos(arch_pmos)
generic map(cgso=>1.5e-010,
            cgdo=>1.5e-010,
            mu0=>209.0,
            cbd=>5.0e-014,
            cbs=>5.0e-014,
            Iss=>1.0e-015,
            cox=>3.45e-6,
            l=>1.0e-6,
            vt0=>-0.2,
            cgbo=>4.0e-010,

```

```

        T=>298.0,
        w=>1.0e-6,
        twophi=>0.65,
        gamma=>0.69,
        lambda=>1.1)

port map(g=>vrefp, d=>vrefn, s=>vmas, b=>electrical_ref);

trans2:entity work.nmos(arch_nmos)
generic map(cgso=>0.5e-010,
            cgdo=>0.5e-010,
            cgbo=>2.0e-009,
            mu0=>442.2,
            cbd=>5.0e-014,
            cbs=>5.0e-014,
            Iss=>1.0e-015,
            cox=>0.08e-6,
            l=>10.0e-6,
            vt0=>0.2,
            T=>298.0,
            w=>1.0e-6,
            twophi=>0.65,
            gamma=>0.2,
            lambda=>0.6)
port map(g=>vrefn, d=>vrefn, s=>vmenos, b=>electrical_ref);

trans3: entity work.pmos(arch_pmos)
generic map(cgso=>1.5e-010,
            cgdo=>1.5e-010,
            mu0=>209.0,
            cbd=>5.0e-014,
            cbs=>5.0e-014,
            Iss=>1.0e-015,
            cox=>3.45e-6,
            l=>1.0e-6,
            vt0=>-0.2,
            cgbo=>4.0e-010,
            T=>298.0,
            w=>1.0e-6,
            twophi=>0.65,
            gamma=>0.69,

```

```

        lambda=>1.1)

port map(g=>vrefp, d=>vrefp, s=>vmass, b=>electrical_ref);

trans4:entity work.nmos(arch_nmos)
generic map(cgso=>0.5e-010,
            cgdo=>0.5e-010,
            cgbo=>2.0e-009,
            mu0=>442.2,
            cbd=>5.0e-014,
            cbs=>5.0e-014,
            Iss=>1.0e-015,
            cox=>0.08e-6,
            l=>10.0e-6,
            vt0=>0.2,
            T=>298.0,
            w=>1.0e-6,
            twophi=>0.65,
            gamma=>0.2,
            lambda=>0.6)
port map(g=>vrefn, d=>vrefp, s=>vref, b=>electrical_ref);

res1: entity work.resistor(ideal)
generic map(res=>10000.0e3)
port map(p1=>vref,p2=>vmeno);

trans5: entity work.pmos(arch_pmos)
generic map(cgso=>1.5e-010,
            cgdo=>1.5e-010,
            mu0=>209.0,
            cbd=>5.0e-014,
            cbs=>5.0e-014,
            Iss=>1.0e-015,
            cox=>3.45e-6,
            l=>1.0e-6,
            vt0=>-0.2,
            cgbo=>4.0e-010,
            T=>298.0,
            w=>1.0e-6,
            twophi=>0.65,
            gamma=>0.69,

```

```

        lambda=>1.1)

port map(g=>vrefp, d=>irefp, s=>vmas, b=>electrical_ref);

trans6:entity work.nmos(arch_nmos)
generic map(cgso=>0.5e-010,
            cgdo=>0.5e-010,
            cgbo=>2.0e-009,
            mu0=>442.2,
            cbd=>5.0e-014,
            cbs=>5.0e-014,
            Iss=>1.0e-015,
            cox=>0.08e-6,
            l=>1.0e-6,
            vt0=>0.7,
            T=>298.0,
            w=>1.0e-6,
            twophi=>0.65,
            gamma=>0.2,
            lambda=>0.6)
port map(g=>vrefn, d=>irefn, s=>vmenos, b=>electrical_ref);

end architecture;

```

A.20 Tag Testbench in The Wrong Address Case

```

library ieee;
use ieee.electrical_systems.all;
use IEEE.math_real.all;
    use ieee.std_logic_1164.all;
use IEEE.FUNDAMENTAL_CONSTANTS.all;
library DISCIPLINES;
entity test_ds_cu is

end entity test_ds_cu;

architecture ideal of test_ds_cu is

terminal senal,mas,menos,env,ref,vref_c,vpos,vneg,ipos,ineg,gn,ck1,a11,b11,ck12,Gn_ana,

```



```

Gp_ana,if_en_ana:electrical;
signal ck11: std_logic;
signal por1,por_u1:std_logic;
signal data_in1:std_logic;
signal data_in_u1:std_logic;
signal in_invert: std_logic;
signal add0:std_logic:='0';
signal add1:std_logic:='1';
signal Gn_ctrl:std_logic;
signal Gp_ctrl:std_logic;
signal if_en_ctrl:std_logic;
signal clk_ctrl:std_logic:='1';
signal Data:std_logic;
signal ovf:std_logic:='0';
signal clk_if_out:std_logic:='1';
signal clk_if_out1:std_logic:='1';
signal uno:std_logic:='1';
signal out_count:std_logic_vector(2 downto 0);
signal in_if:std_logic_vector(1 downto 0);
signal sig1:std_logic;
signal sigx:std_logic:='0';
signal sigy:std_logic;
signal in_clk_ctrl:std_logic:='1';
begin

source: entity work.func_rf(ideal)
generic map(f=>2.4e9,
            v=>1.0,
            Vhi=> 0.3,          -- Voltage level corresponding to high level
            Vlo=>0.0,          -- Voltage level corresponding to low level
            level=>1.0,        -- 1: increasing pulses are Vhi
                                -- 0: increasing pulses are Vlo
            toff=>200ns,       -- Duration time of OFF time
            ton =>200ns,       -- Duration time of ON time (increasing pulses)
            trise=> 5ns,       -- Output rise time
            tfall=> 5ns,       -- Output fall time
            n => 20)           -- Number of pulses per cycle);
port map(salida=>senal);

rect: entity Rect_Diodes(ideal)
port map(vmas=>mas,vmenos=>menos,rf=>senal);

```

```

det: entity detector(ideal)
port map(rf_in=>senal,Venv=>env,Vref=>ref);

porent: entity por(ideal)
port map(vmas=>mas, vmenos=>menos, a1=>a11, b1=>b11,por0=>por1,por_u=>por_u1);

curr_ref:entity current_ref_nmos(ideal)
port map(vmas=>mas, vmenos=>menos, vref=>vref_c,
vrefp=>electrical_ref, vrefn=>electrical_ref, irefp=>ipos, irefn=>ineg);

comp:entity comparator_nmos(ideal)
port map(Venv=>env, G_n=>Gn_ana, Vref=>ref, Vmas=>mas, Vmenos=>menos,
Iref_n=>ineg, clk=>ck12,clk_out=>in_invert,ck1=>ck11);

decoder:entity dig_dff3(QuickStart)
generic map(
    tPLH=> 5 ns,      -- Propagation delay from clock to out (L->H)
    tPHL=>5 ns,      -- Propagation delay from clock to out (H->L)
    edge=>1.0,      -- Triggering edge of the flip-flop neg for negative edge,
    -- pos for positive edge
    Q_init=>'0')    -- Initial value of output Qq
port map(
    D      =>data_in1,      -- input port
    clk    =>ck11,          -- input clock
    preset =>por_u1,        -- input preset
    clear  =>por1,          -- input clear
    Qq     =>data_in_u1,    -- output port
    nQ     =>data_in1);    -- inverted output port

cu:entity ctrl_log(ideal)
port map(Gn_a=>Gn_ana,
    Gp_a=>Gp_ana,
    if_en_a=>if_en_ana,
    Data_in=>data_in1,
    Data_in_u=>data_in_u1,
    if1=>clk_if_out, --senial de reloj
    if1_en=>if_en_ctrl, --activacion senial reloj
    Gn=>Gn_ctrl, --salida
    Gp=>Gp_ctrl, --salida
    ck1=>ck11, --salida

```

```

        por=>por1,
        por_u=>por_u1,
        Data_out_u=>Data, --salida de datos
        a1=>add0,          --direccion (in) que va a q2
        a2=>add1);        --direccion (in) que va a q1.

count:      entity counter(QuickStart)

generic map(
    n =>3)          -- No. of counter o/p bits

port map(
    clk=>if_en_ctrl, -- Counted clock input.
    reset=>uno,      -- Counter reset.
    overflow=>ovf,   -- counter overflow
    dout=>out_count); -- counter o/p
sigx<='1' after 2.405us;--(out_count(0)='1' and out_count(1)='1');
sigy<=not sigx;

clk:entity clk_gen(QuickStart)
generic map(
    fclk=> 1.0e6,      -- Clock frequency.
    duty_cycle =>0.1,  -- Duty cycle.
    tD_on  => 0 fs,    -- Clock on delay.
    tD_off =>0 fs)    -- Clock off delay.
port map(
    clk=>clk_if_out,  -- Clock output.
    enable=>sigx);   -- Clock enable.

end architecture;
```

A.21 Tag Testbench in The Right Address Case

```

library ieee;
use ieee.electrical_systems.all;
use IEEE.math_real.all;
    use ieee.std_logic_1164.all;
use IEEE.FUNDAMENTAL_CONSTANTS.all;
library DISCIPLINES;
```

```

entity test_ds_cu_work is

end entity test_ds_cu_work;
architecture ideal of test_ds_cu_work is

terminal senal,mas,menos,env,ref,vref_c,vpos,vneg,ipos,ineg,gn,ck1,
a11,b11,ck12,Gn_ana,Gp_ana,if_en_ana:electrical;
signal ck11: std_logic;
signal por1,por_u1:std_logic;
signal data_in1:std_logic;
signal data_in_u1:std_logic;
signal in_invert: std_logic;
signal add0:std_logic:='1';
signal add1:std_logic:='0';
signal Gn_ctrl:std_logic;
signal Gp_ctrl:std_logic;
signal if_en_ctrl:std_logic;
signal clk_ctrl:std_logic:='1';
signal Data:std_logic;
signal ovf:std_logic:='0';
signal clk_if_out:std_logic:='1';
signal clk_if_out1:std_logic:='1';
signal uno:std_logic:='1';
signal out_count:std_logic_vector(2 downto 0);
signal in_if:std_logic_vector(1 downto 0);
signal sig1:std_logic;
signal sigx:std_logic:='0';
signal sigy:std_logic;
signal in_clk_ctrl:std_logic:='1';
begin

source: entity work.func_rf(ideal)
generic map(f=>2.4e9,
            v=>1.0,
            Vhi=> 0.3,          -- Voltage level corresponding to high level
            Vlo=>0.0,          -- Voltage level corresponding to low level
            level=>1.0,        -- 1: increasing pulses are Vhi
                                -- 0: increasing pulses are Vlo
            toff=>200ns,       -- Duration time of OFF time
            ton =>200ns,       -- Duration time of ON time (increasing pulses)

```

```

        trise=> 5ns,          -- Output rise time
        tfall=> 5ns,         -- Output fall time
        n => 20)             -- Number of pulses per cycle);

port map(salida=>senal);

rect: entity Rect_Diodes(ideal)
port map(vmas=>mas,vmenos=>menos,rf=>senal);

det: entity detector(ideal)
port map(rf_in=>senal,Venv=>env,Vref=>ref);

porent: entity por(ideal)

port map(vmas=>mas, vmenos=>menos, a1=>a11, b1=>b11,por0=>por1,por_u=>por_u1);

cap1: entity work.capacitor(leakage)
generic map(
c=>10000.0e-12,
r_leak=>10.0e6)
port map(node1=>mas, node2=>electrical_ref);

cap2: entity work.capacitor(leakage)
generic map(
c=>10000.0e-12,
r_leak=>10.0e6)
port map(node1=>menos, node2=>electrical_ref);

curr_ref:entity current_ref_nmos(ideal)
port map(vmas=>mas, vmenos=>menos, vref=>vref_c, vrefp=>electrical_ref,
vrefn=>electrical_ref, irefp=>ipos, irefn=>ineg);

comp:entity comparator_nmos(ideal)
port map(Venv=>env, G_n=>Gn_ana, Vref=>ref, Vmas=>mas, Vmenos=>menos, Iref_n=>ineg,
clk=>ck12,clk_out=>in_invert,ck1=>ck11);

decoder:entity dig_dff3(QuickStart)
generic map(
tPLH=> 5 ns,          -- Propagation delay from clock to out (L->H)
tPHL=>5 ns,          -- Propagation delay from clock to out (H->L)

```

```

    edge=>1.0,          -- Triggering edge of the flip-flop neg for negative edge,
    --pos for positive edge
    Q_init=>'0')       -- Initial value of output Qq
port map(
    D          =>data_in1,          -- input port
    clk        =>ck11,             -- input clock
    preset     =>por_u1,          -- input preset
    clear      =>por1,            -- input clear
    Qq         =>data_in_u1,       -- output port
    nQ         =>data_in1);       -- inverted output port

cu:entity ctrl_log(ideal)
port map(Gn_a=>Gn_ana,
        Gp_a=>Gp_ana,
        if_en_a=>if_en_ana,
        Data_in=>data_in1,
        Data_in_u=>data_in_u1,
        if1=>clk_if_out, --senial de reloj
        if1_en=>if_en_ctrl, --activacion senial reloj
        Gn=>Gn_ctrl, --salida
        Gp=>Gp_ctrl, --salida
        ck1=>ck11, --salida
        por=>por1,
        por_u=>por_u1,
        Data_out_u=>Data, --salida de datos
        a1=>add0, --direccion (in)
        a2=>add1); --direccion (in)

count:      entity counter(QuickStart)
generic map(
    n =>3)          -- No. of counter o/p bits
port map(
    clk=>if_en_ctrl, -- Counted clock input.
    reset=>uno, -- Counter reset.
    overflow=>ovf, -- counter overflow
    dout=>out_count); -- counter o/p

sigx<='1' after 2.405us;--(out_count(0)='1' and out_count(1)='1');
sigy<=not sigx;

```

```
clk:entity clk_gen(QuickStart)
  generic map(
    fclk=> 1.0e6,      -- Clock frequency.
    duty_cycle =>0.1,  -- Duty cycle.
    tD_on  => 1 fs,    -- Clock on delay.
    tD_off =>1 fs)     -- Clock off delay.
  port map(
    clk=>clk_if_out,  -- Clock output.
    enable=>sigx); -- Clock enable.
    sig1<='1' when out_count="010";

end architecture;
```

Appendix B

Signals of The Blocks

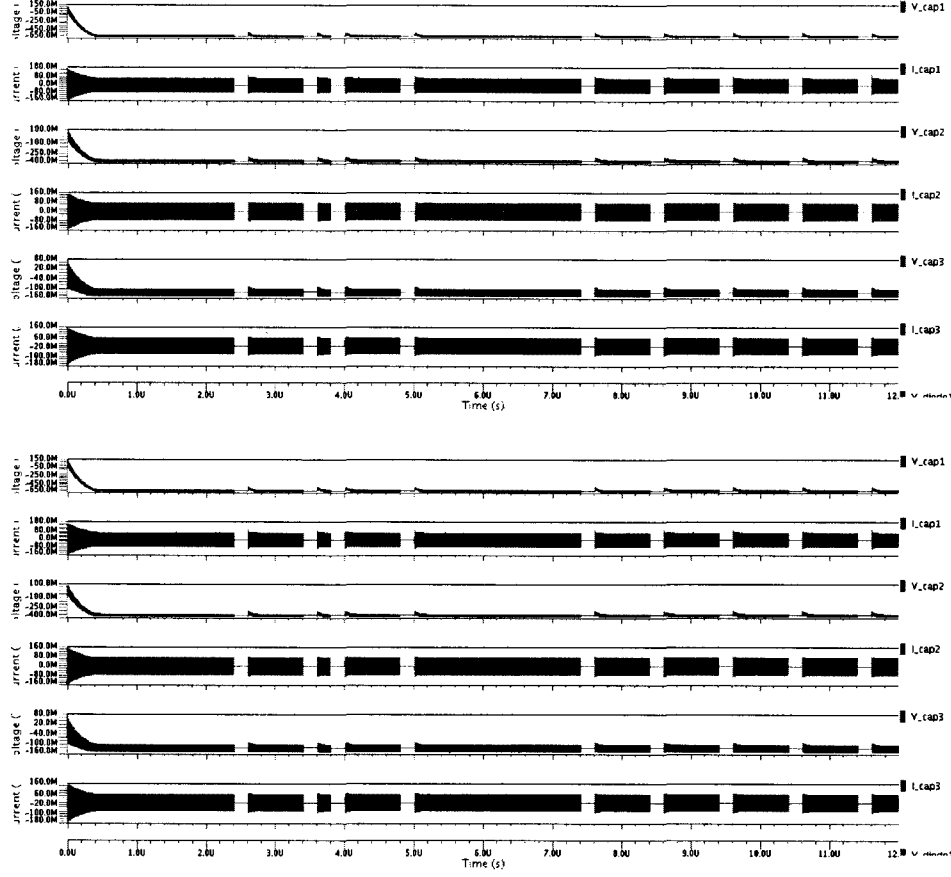


Figure B.1: Rectifier's Signals 1

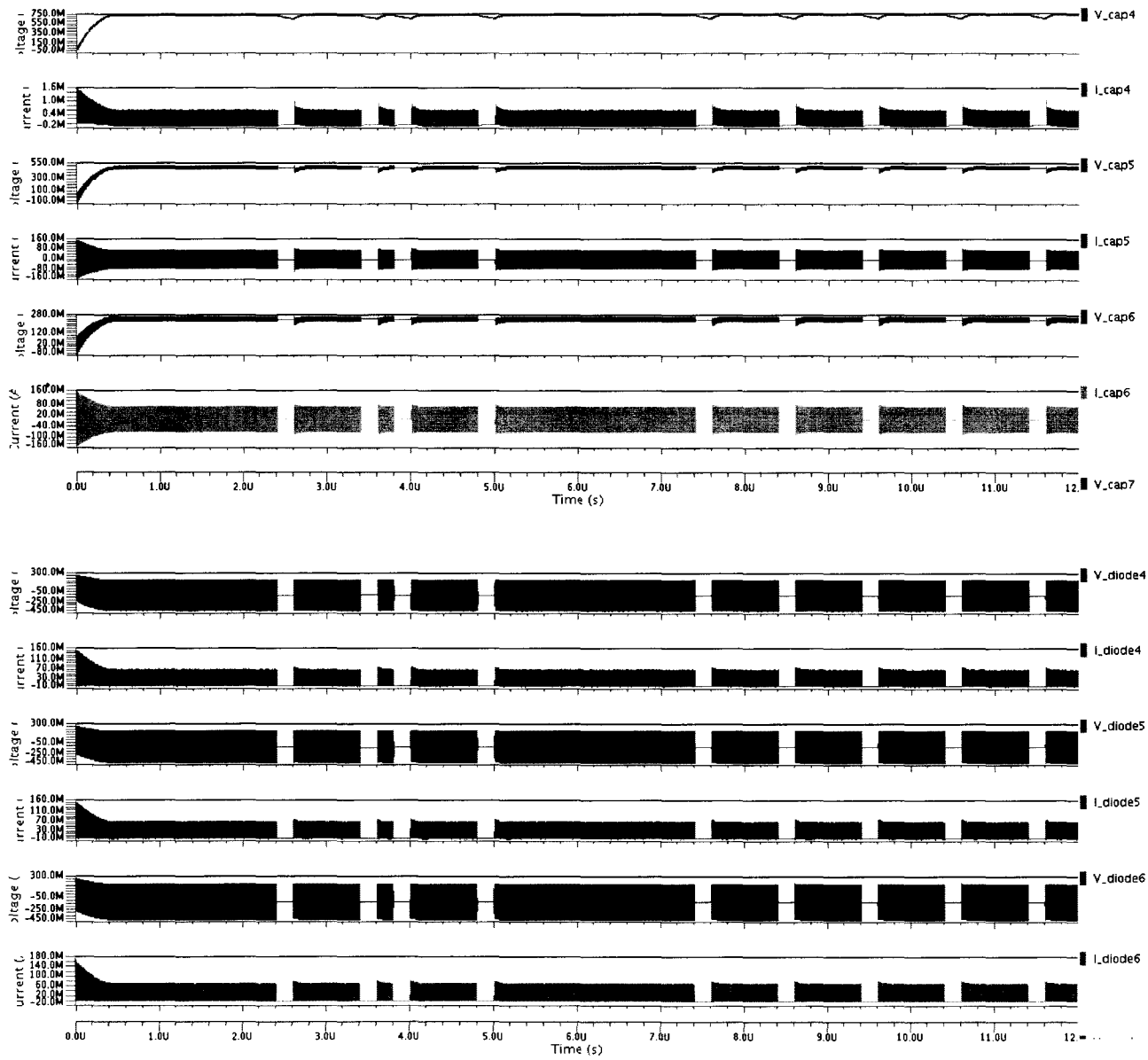


Figure B.2: Rectifier's Signals 2

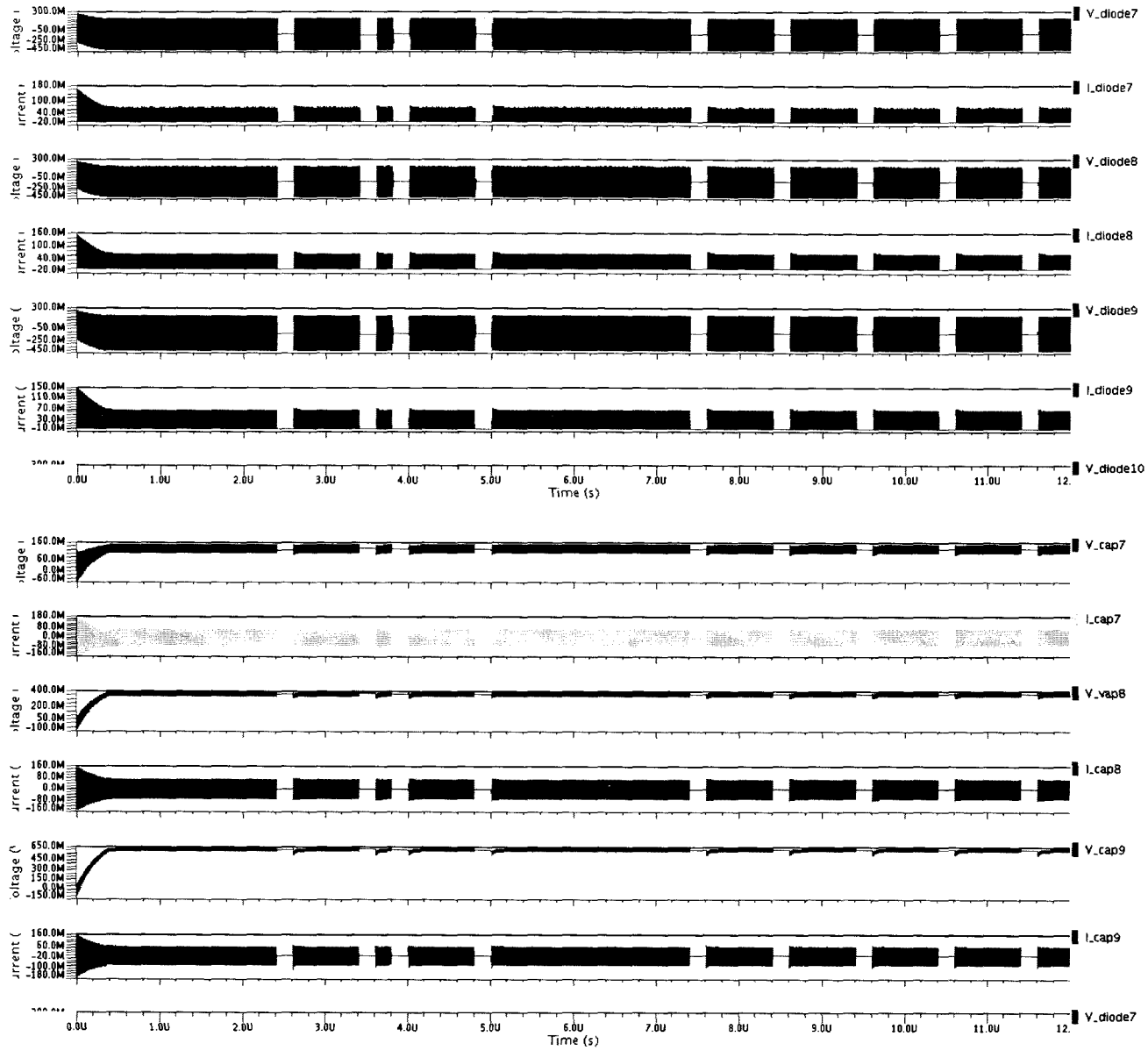


Figure B.3: Rectifier's Signals 3

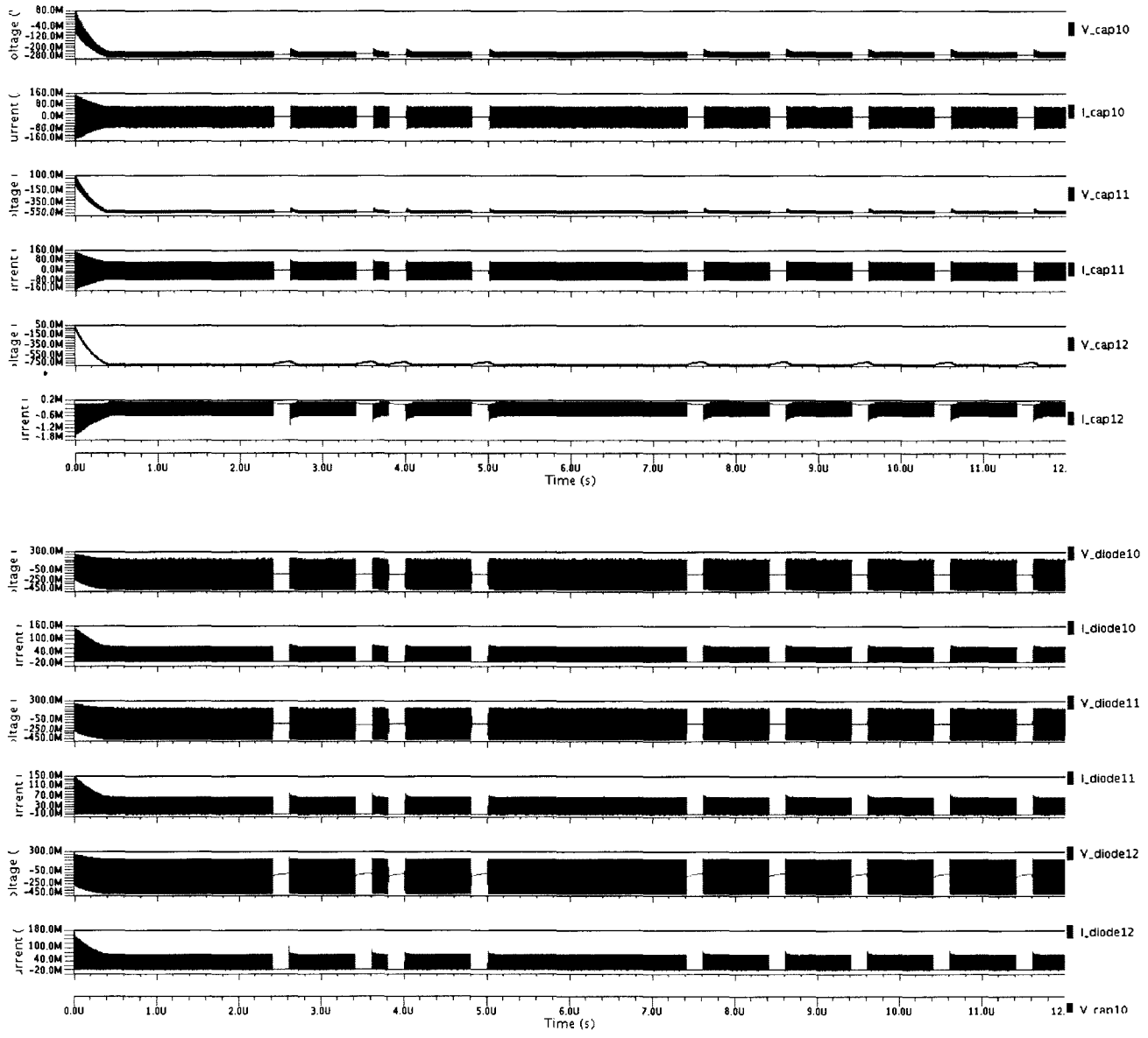


Figure B.4: Rectifier's Signals 4

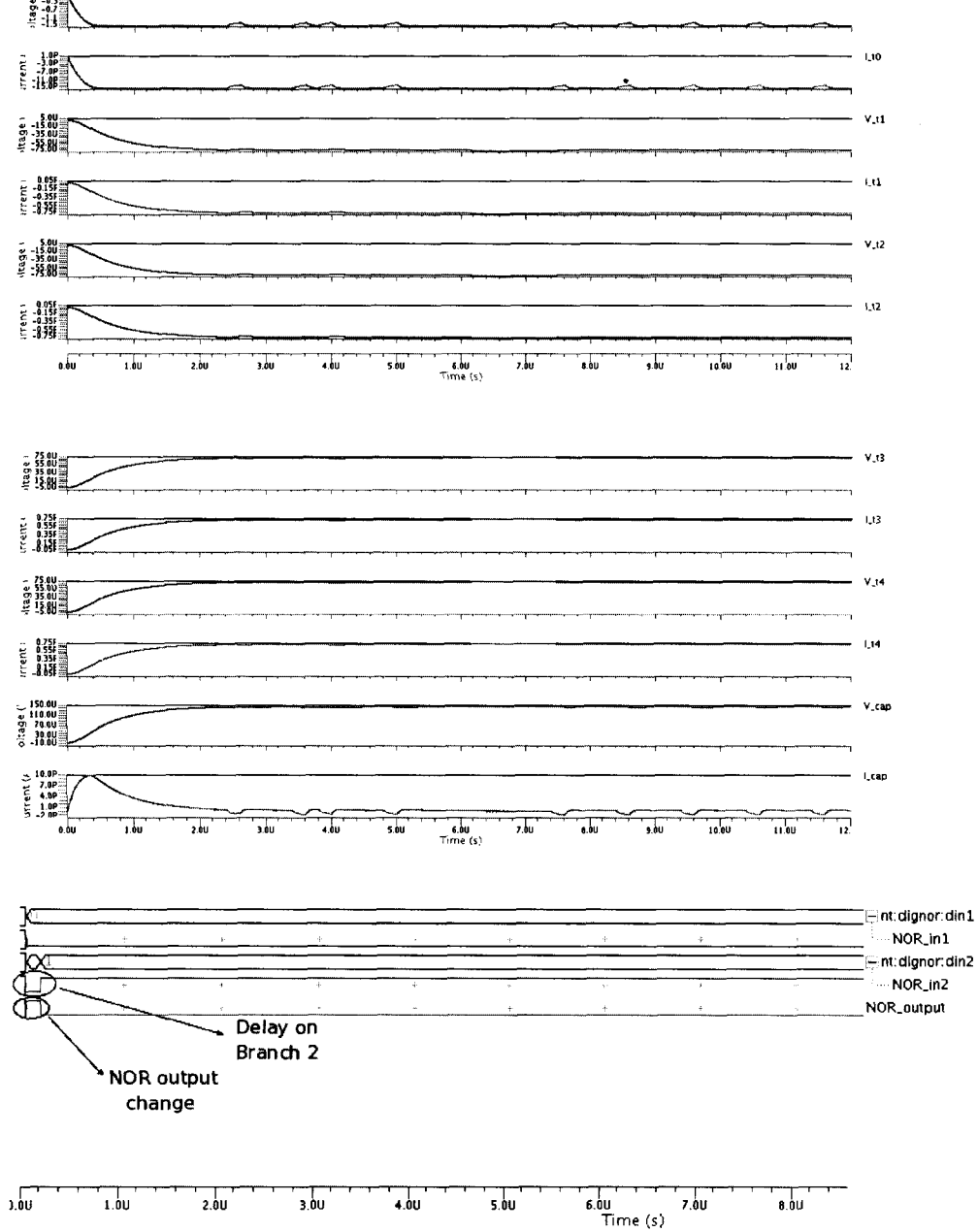


Figure B.5: Power-on-Reset's Signals

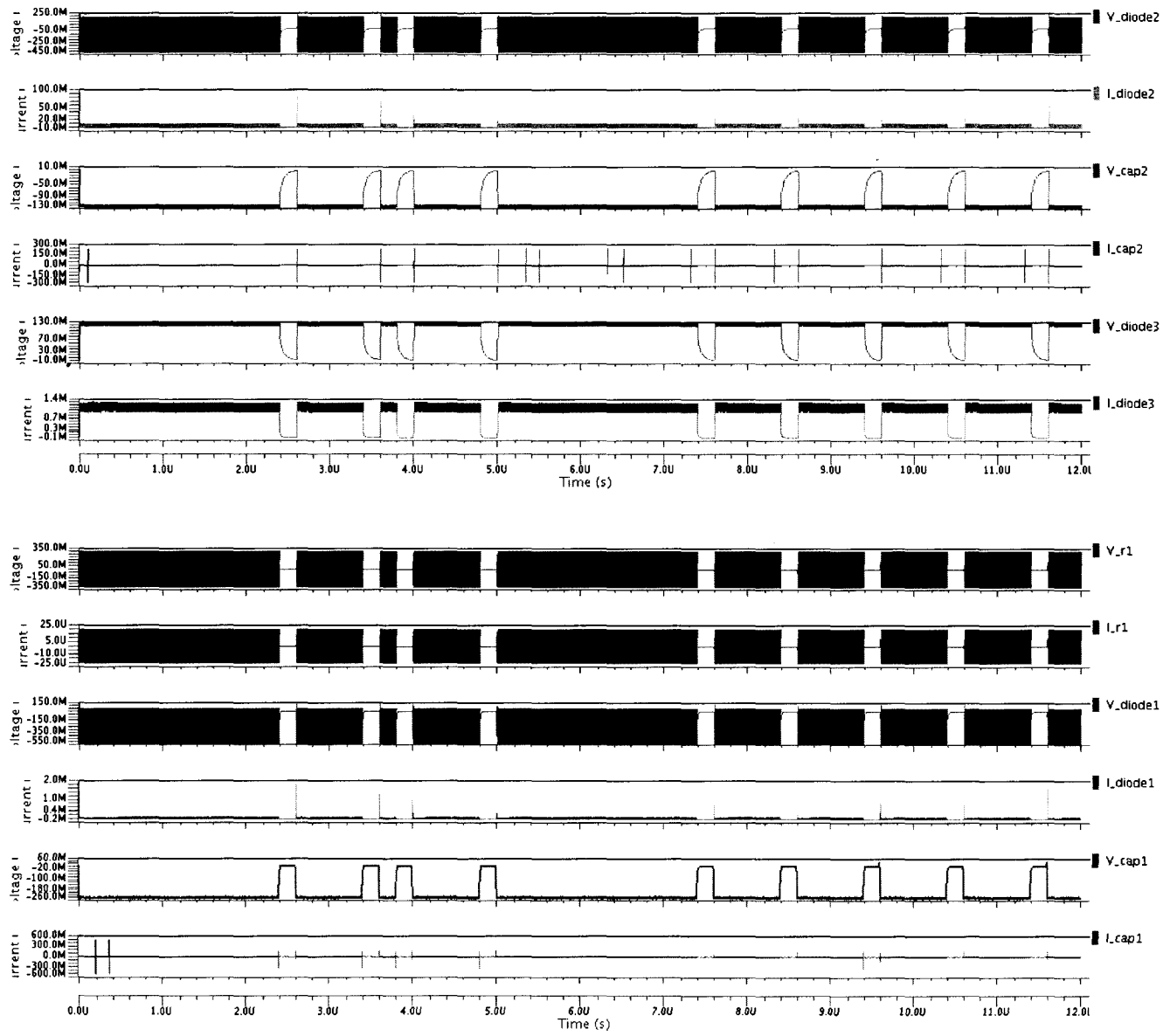


Figure B.6: Detector's Signals

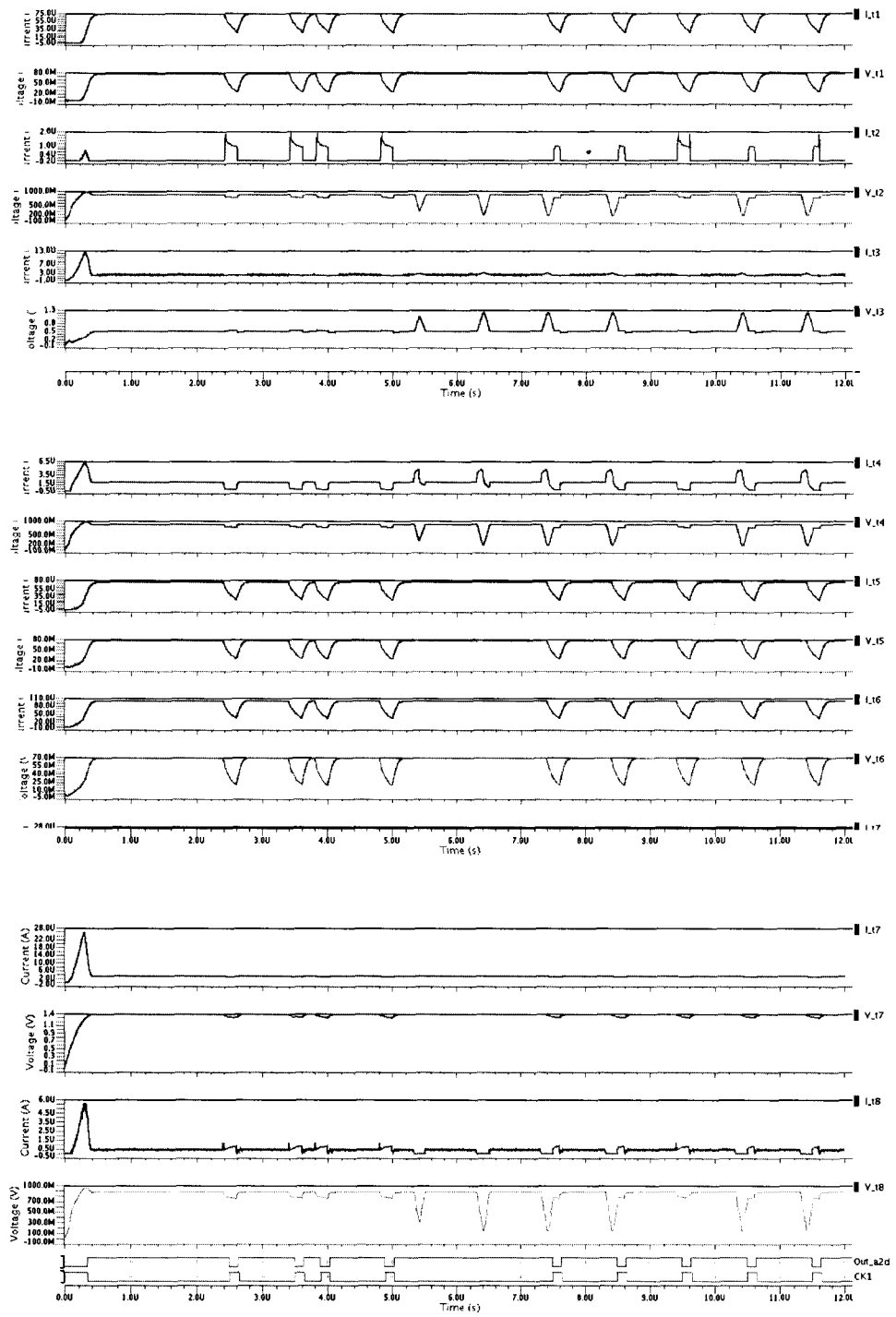


Figure B.7: Comparator's Signals

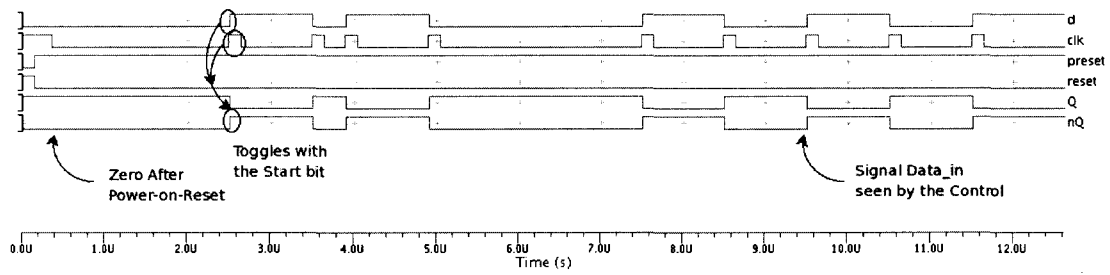


Figure B.8: Decoder's Signals

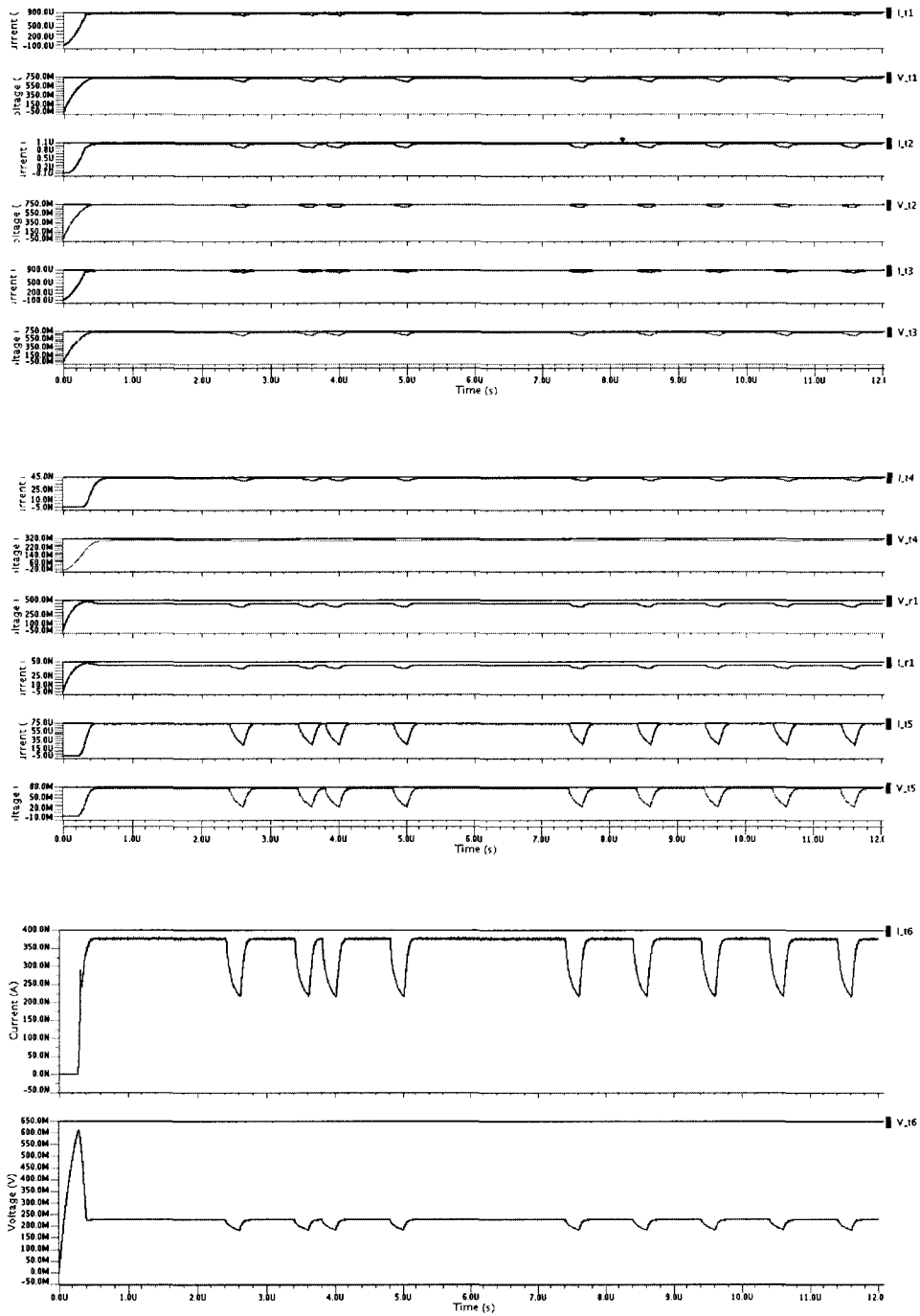


Figure B.9: Current Reference's Signals

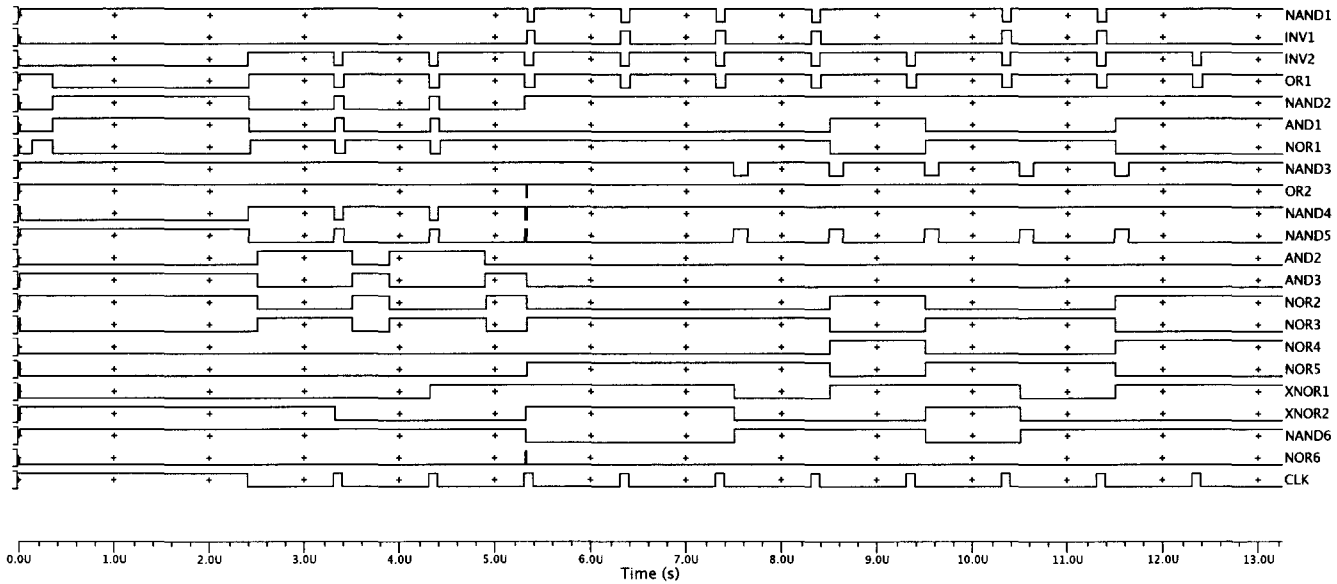
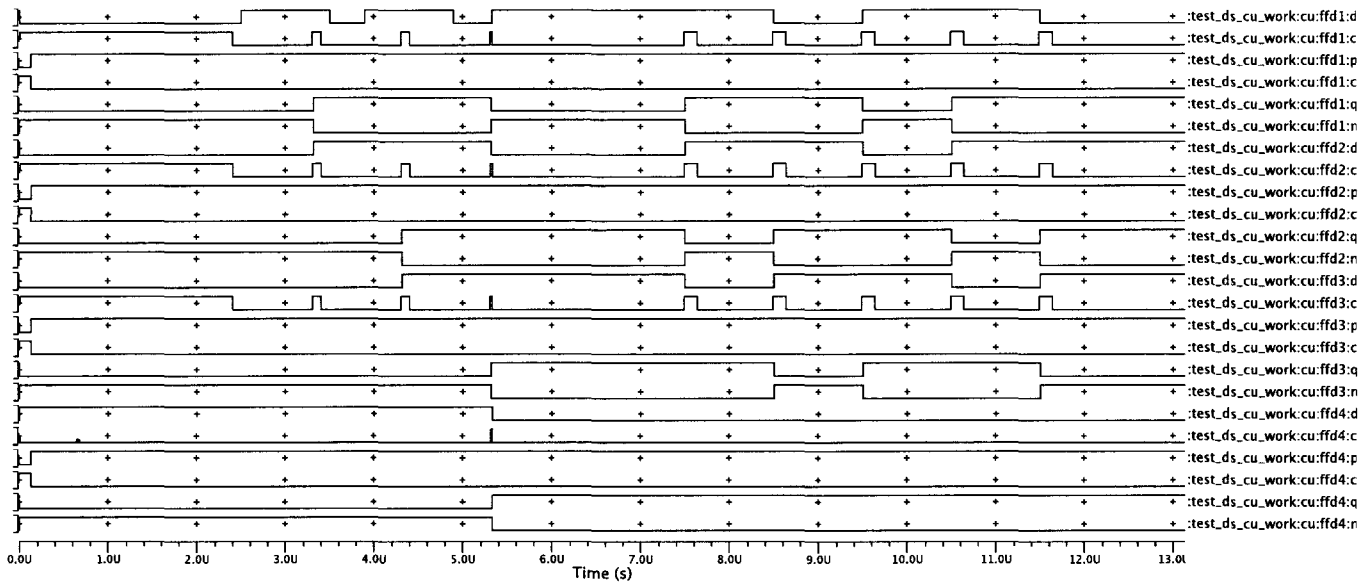


Figure B.10: SR and Control's Signals Right Address Case

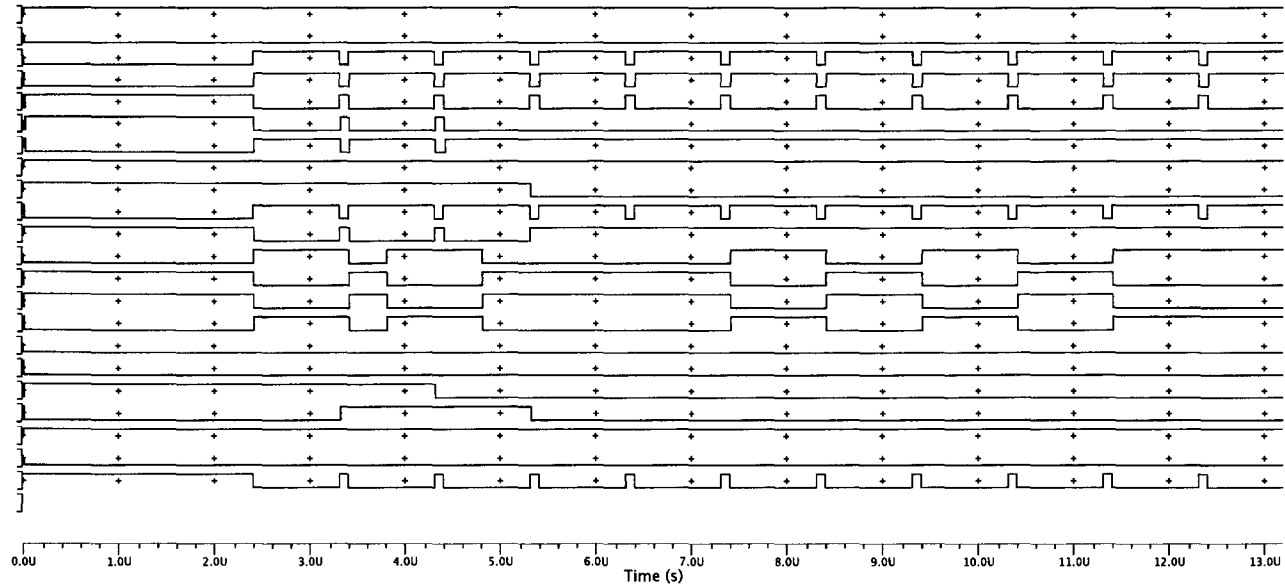
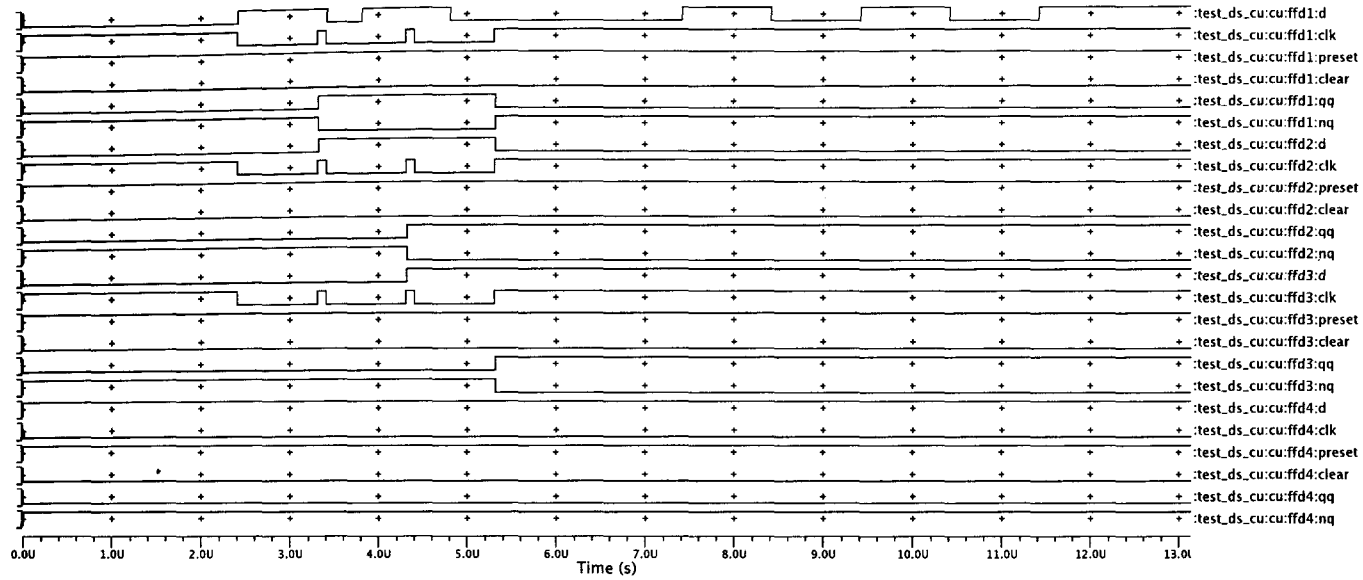


Figure B.11: SR and Control's Signals Wrong Address Case

Appendix C

EPFL EKV 2.6 Transistor Code in Matlab

```
% EKV - Charge based model on SOS mosfets
clear

% load SOS transistor data to check results
%load "\\Cronus\users\euge\Advanced Research\SOS\SOS5data\Lavida data\RN2.5X2.5.MAT"
%load "\\Cronus\users\euge\Advanced Research\SOS\SOS5data\Paul data\chip 1 (left1)\Array4_RN25x25(L).mat"
%load RN2.5X2.5.MAT;
%load Array4_RN25x25(L).mat Id Vs Vg Vd %Is Ig

% general parameter set:
q = 1.602e-19;
kToq = 300*8.617e-5; % temperature
ni = 1.45e16; % [1/cm3] intrinsic concentration in Si
EPSi = 1.04e-10; % [F/m]
e0 = 8.85e-12; % [F/m]
eox = e0*3.97;
UT = kToq; % thermoelectric voltage

% SOS RN general parameter set:
W = 2.5e-6;
L = 2.5e-6;
NS = 1; % number of series devices
XJ = 1e-7; % junction depth
TOX = 105e-10; % [m] sos cadence tools 2.6
COX = eox/TOX;
VTO = -0.2;%0.7; % euge extraction ( VTO = VFB + PSIO + GAMMA*sqrt(PSIO) )
GAMMA = 0.711; % default parameter ( GAMMA = sqrt(2*q*EPSi*Nsub)/COX )
PHI = 0.8; % default Fermi potential *2 (equivalent to PSIO without correction factor)
KP = 82e-6;%8.14e-5; % euge extraction
THETA = 0.157;%0.157; % euge extraction
UCRIT = 7.0e6; % vcrit in SOS rn spice longitudinal critical field
DW = -0.05e-6; % default parameter
```

```

DL = -0.1e-6; % default parameter

% Fine fitting parameters:
LAMBDA = 2.5; % euge extraction
WETA = 0.1; % default parameter
LETA = 0.3; % default parameter

% impact ionization parameters:
IBN = 0.6;
IBA = 3e8;
IBB = 2e8;

% Model:
expion = 0; % toggles the exponential ionization model off and enables Euge's model
Nsub = (GAMMA*COX)^2/(2*q*EPSi);
FIF = UT*log(Nsub/ni);
PSIO = 2*FIF + 3*UT; % add 2-4 UT for fitting
Weff = W + DW;
Leff = L + DL;
Vc = UCRIT*NS*Leff; % velocity saturation voltage
Lc = sqrt((EPSi/COX)*XJ);

Vs = 0;
Vg = [0:0.5:3.5];
Vd = [0:0.1:3.5]*1;

for i = 1:size(Vg,2)
    for j = 1:size(Vd,2)

% calculate Vp:
if Vg(i) > 0
    Vp(i,j) = Vg(i) - VTO - GAMMA*(sqrt(Vg(i)-VTO+(sqrt(PSIO)+GAMMA/2)^2)-(sqrt(PSIO)+GAMMA/2));
else
    Vp(i,j) = -PHI;
end

npar = 1 + GAMMA./(2*sqrt(PSIO+Vp(i,j)));
vpp = Vp(i,j)/UT;
vdd = Vd(j)/UT;
vss = Vs/UT;
%Qpinv = -COX*npar*(Vp-Vch);

% forward normalized current:
viff = (Vp(i,j) - Vs)/UT;
iff = (log(1 + exp(viff/2)))^2;

% velocity saturation and channel length modulation effects:
VDSS = Vc*(sqrt(1/4 + (UT/Vc)*sqrt(iff)) - 1/2); % this is 1/2 of the saturation voltage!
VpDSS = Vc*(sqrt(1/4 + (UT/Vc)*(sqrt(iff) - (3/4)*log(iff))) - 1/2) + UT*(log(Vc/(2*UT)) - 0.6);
deV = 4*UT*sqrt(LAMBDA*(sqrt(iff) - VDSS/UT) + 1/64);
Vds = (Vd(j) - Vs)/2;

```

```

Vip = sqrt(VDSS^2 + deV^2) - sqrt((Vds-VDSS)^2 + deV^2);
deL = LAMBDA*Lc*log(1 + (Vds-Vip)/(Lc*UCRIT));
%deL = LAMBDA*Lc*log(1 + VpDSS/(Lc*UCRIT)); % test for problems
Lpr = NS*Leff - deL - (Vds+Vip)/UCRIT; % account for multiple device in series ---- is this wrong?
%Lpr = NS*Leff - deL + VDSS/UCRIT; % test for problems
Lmin = NS*Leff/10; % account for multiple device in series
Leq = 1/2*(Lpr + sqrt(Lpr^2 + Lmin^2));
%Leq = Leff; %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% to remove vel sat fx

% mobility reduction due to vertical field
Vppr = 1/2*(Vp(i,j) + sqrt(Vp(i,j)^2 + 2*UT^2));
BETA = NS*Weff/Leq*KP/(1 + THETA*Vppr);
Iss = 2*npar*BETA*UT^2; % normalization current

% Euge's kink model:
KST = 1.9;
KSLOPE = 0.06;
if Vd(j) >= (KST + VDSS)
    Vp(i,j) = Vp(i,j) + KSLOPE*(Vd(j) - KST - VDSS);
    % re-calculate forward normalized current:
    viff = (Vp(i,j) - Vs)/UT;
    iff = (log(1 + exp(viff/2)))^2;
end

% reverse normalized current:
virr = (Vp(i,j) - Vds - Vs - sqrt(VpDSS^2 + deV^2) + sqrt((Vds-VpDSS)^2 + deV^2))/UT;
irr = (log(1 + exp(virr/2)))^2;
%irr = (log(1 + exp((vpp-vdd)/(2))))^2; % simple model, no short channel effect
idd = iff - irr; % sum forward and reverse currents
Idsm(i,j) = idd*Iss; % compute model drain current

% impact ionization current:
Vib = Vd(j)-Vs-IBN*2*VDSS; % 2*VDSS is the saturation voltage
if (Vib > 0 & expion == 1)
    IDB = Idsm(i,j)*IBA/IBB*Vib*exp(-IBB*Lc/Vib);
else
    IDB = 0;
end
Idm(i,j) = Idsm(i,j)+IDB;

end
end

plot(Vd, Idsm, 'ro', Vd, Idm, 'k-')
xlabel('Vds [V]'); ylabel('Ids [A]');

% ionization:
plot(Vd, Idsm, 'ro', Vd, Idm, 'k-')
xlabel('Vds [V]'); ylabel('Ids [A]');

```

Bibliography

- [1] J. Curty, M. Declercq, C. Dehollain, and N. Joehl, *Design and Optimization of Passive UHF RFID Systems*, Springer, Ed. New York, NY, USA: Springer, 2007.
- [2] J.-P. Curty, N. Joehl, F. Krummenacher, C. Dehollain, and M. Declercq, "A model for u-power rectifier analysis and design," *Circuits and Systems I: Regular Papers, IEEE Transactions on*, vol. 52, no. 12, pp. 2771–2779, Dec. 2005.
- [3] A. Jones, R. Hoare, S. Dontharaju, S. Tung, R. Sprang, J. Fazekas, J. Cain, and M. Mickle, "A field programmable rfid tag and associated design flow," April 2006, pp. 165–174.
- [4] A. Jones, R. Hoare, S. Dontharaju, S. Tung, R. Sprang, J. Fazekas, and J. Cain, "An automated, reconfigurable, low-power rfid tag," 2006, pp. 131–136.
- [5] A. Janek, C. Steger, R. Weiss, J. Preishuber-Pfluegl, and M. Pistauer, "Functional verification of future higher class uhf rfid tag architectures based on cosimulation," April 2008, pp. 336–343.
- [6] A. Abrial, J. Bouvier, P. Senn, M. Renaudin, and P. Vivet, "A new contactless smartcard ic using an on-chip antenna and an asynchronous micro-controller," Sept. 2000, pp. 97–100.
- [7] A. Shameli, A. Safarian, A. Rofougaran, M. Rofougaran, J. Castaneda, and F. De Flaviis, "A uhf near-field rfid system with fully integrated transponder," *Microwave Theory and Techniques, IEEE Transactions on*, vol. 56, no. 5, pp. 1267–1277, May 2008.
- [8] H. Yan, H. jianyun, L. Qiang, and M. Hao, "Design of low-power baseband-processor for rfid tag," Jan. 2006, pp. 4 pp.–63.

- [9] A. Huber, "Rfid fundamentals," *Informationsforum RFID*, pp. 1–20, 2006.
- [10] J. Landt, "The history of rfid," *Potentials, IEEE*, vol. 24, no. 4, pp. 8–11, Oct.-Nov. 2005.
- [11] "The History of RFID Technology," <http://www.rfidjournal.com/article/view/1338/1/129>, January 2005.
- [12] "Ti-rfid history of innovation," <http://www.ti.com/rfid/timeline/timeline.shtml>, January 2007.
- [13] H. Bhatt and B. Glover, *RFID Essentials*, O'Reilly, Ed., 2006.
- [14] P. Lindstrom and F. Thornton, *RFID Security*. Syngress Publishing, 2005.
- [15] K. Finkenzeller, *RFID Handbook: Fundamentals and Applications in Contactless Smart Cards and Identification*. New York, NY, USA: John Wiley & Sons, Inc., 2003.
- [16] J. Flores, S. Srikant, B. Sareen, and A. Vagga, "Performance of rfid tags in near and far field," Jan. 2005, pp. 353–357.
- [17] D. Pozar, *Microwave Engineering*. 111 River Street, Hoboken, NJ: John Wiley & Sons, Inc., 2005.
- [18] T. Carlisle, "Rfid...coming age," *ITAA White Paper*, pp. 1–8, 2004.
- [19] "SpeedPass Overview," <https://www.speedpass.com/forms/frmHowItWorks.aspx?pgHeader=how>, January 2009.
- [20] M. Ollivier, "Rfid-a practical solution for problems you didn't even know you had!" Nov 1996, pp. 3/1–3/6.
- [21] T. Nakamura, H. Matsushashi, and Y. Nagatomo, "Silicon on sapphire (sos) device technology," *OKI Technical Review*, pp. 66–69, 2004.
- [22] J.-P. Curty, N. Joehl, C. Dehollain, and M. Declercq, "Remotely powered addressable uhf rfid integrated system," *Solid-State Circuits, IEEE Journal of*, vol. 40, no. 11, pp. 2193–2202, Nov. 2005.

- [23] “A web interface for diode rectification ability evaluation,” <http://legwww.epfl.ch/direct>, January 2004.
- [24] S. Badam, “An improved ekv transistor model with deep submicron effects modeled in vhdl-ams,” Master’s thesis, University of Cincinnati, 2600 Clifton Ave., Cincinnati, Ohio 45221, September 2007.
- [25] A. S. Sedra and K. C. Smith, *Microelectronic Circuits*, 4th ed. Oxford University Press.
- [26] “Ekv v2.6 parameters for 0.5um cmos,” <http://legwww.epfl.ch/ekv/paramset.html>, January 1999.
- [27] R. Khouri, V. Berouille, T.-P. Vuong, and S. Tedjini, “Wireless system validation using vhdl-ams behavioral antenna models: radio-frequency identification case study,” 2004, pp. 185–188.
- [28] D. Hodges, H. Jackson, and R. Saleh, *Analysis and Design of Digital Integrated Circuits*, 3rd ed. McGraw Hill.
- [29] Z. Fu, P. Weerakoon, and E. Culurciello, “Nano-watt silicon-on-sapphire adc using 2c-1c capacitor chain,” *Electronics Letters*, vol. 42, no. 6, pp. 341–343, March 2006.

Tecnológico de Monterrey, Campus Monterrey



30002007213010

<http://biblioteca.mty.itesm.mx>