Democratic Republic of Algeria

Ministry of Higher Education and Scientific Research University of Abderrahmane Mira
Faculty of Technology

جامعة بجاية
Tasdawit n Bgayet
Université de Béjaïa

ATE Department

Department of Automation, Telecommunications and Electronics

Final year project

With a view to obtaining a Master's degree in Electronics Specialization: Instrumentation

Theme:

Practical implementation of MPPT-controller for photovoltaic energy conversion system

Prepared by:
AOUCHENNI Mohand Said

Directed by :

Mr. YAHIAOUI

Mme. BERKANI

Reviewed by :

Mme.MEZZAH

Mme.IDJDARENE

Academic year: **2023/2024**

Dedication

First of all I would like to express my heartfelt gratitude to All the persons who support me from the beginning of my life my parents , brother and sisters and dedicate them this work(Yemma, vava ,noureddine , fifi, ninouh ,lili)

Also to one special person to me^^, All my friends and my mates in this work.

Acknowledgment

I would like to express my sincere gratitude to my supervisors, Mr. Yahiaoui and Mme. Berkani, for their guidance and support throughout this project. Additionally, I want to thank the members of the jury, Mme .Mezzah and Mme. Idjdarene, for their valuable scientific contributions during the evaluation of my work.

Table of contents :

List of figures

List of abbrevations :

AC: ALTERNATIVE CURRENT
AHDL: Altera HDL
AMD: Advanced Micro Devices
CUPL: Computer Aided Logic Design Programming
CLBs: Configurable logic blocks
DC: direct current
Fpga: Field Programmable Gate Arrays
HCPLDs :Complex Programmable Logic Devices
HDL: Hardware Description Language
HLS :High-Level Synthesis
IDE: integrated development environment
IEEE : Institute of Electrical and Electronics Engineers
IOBs: Input/output blocks
INC: Incremental Conductance
IP: Intellectual Property
Iph: current source
LUTs: Look-Up Tables
MPP: Maximum Power Point
MPPT: Maximum Power Point Tracking
PLDs: Programmable logic devices
PWm: pulse width modulation
P&O:Perturb & Observe
Rs: series resistance
Rsh: internal shunt resistance
RTL: Register Transfer Level code
SPLDs: Simple Programmable Logic Devices
SDRAM: Synchronous Dynamic Random Access Memory
VCO : Voltage Coefficient Fraction
VHDL: VHSIC Hardware Description Language
VHDL-AMS: VHDL-Analog Mixed Signal
VHSIC: Very High Speed Integrated Circuit
Vitis HLS: Vitis High-Level Synthesis
Xadc: Xilinx analog digital converter

General Introduction

The solar photovoltaic (PV) industry is experiencing unprecedented growth as the global shift towards renewable energy accelerates. With impressive statistics to back this trend, the solar PV industry has solidified its position as a critical player in the transition to clean and sustainable energy sources.

According to the latest data from Statista [1], global (PV) capacity is expected to reach 1.3 terawatts (TW) by 2023. This significant increase reflects the growing global adoption of solar energy, driven by technological advancements, cost reductions, and favorable renewable energy policies.

With this record capacity, the solar PV industry plays a leading role in the transition towards a more sustainable energy future. By providing a clean, renewable, and affordable source of energy, solar PV helps reduce carbon emissions and mitigate the effects of climate change. This growth underscores the increasing importance of solar energy in the global energy mix and its ability to shape a greener and more sustainable energy future. Algeria is indeed one of the largest solar energy reservoirs in the world, with sunlight durations ranging from 2,000 to 3,900 hours per year and daily irradiation of 3,000 to 6,000 Wh/m^2.

Given these particularly favorable statistics for the development of photovoltaic energy, the Algerian government decided to launch a long-term renewable energy program of 22,000 MW for electricity production, which until then was generated using vast quantities of gas.

The program in question planned to establish, by 2030, an electricity production capacity of 12,000 MW exclusively for the domestic market. This included a range of renewable energies, among which 7,200 MW from solar thermal, 2,800 MW from photovoltaic, and 2,000 MW from wind power.

An additional 10,000 MW was even planned for export to Tunisia, Morocco, and neighboring African countries. The openly stated objective was to save as much natural gas as possible to compensate for the declining quantities of gas available for export [2]. The solar energy has been considered as one of principal renewable energy sources for electric power generation.

However, the maximization of extracted power from PV system is a matter of concern as its conversion efficiency is low. Therefore, a maximum power point tracking (MPPT) controller is necessary in a PV system for maximum power extraction [3].

There are several MPPT methods, ranging from the simplest, which involves manual adjustment, to more complex methods that employ sophisticated algorithms. We will focus on the Perturbation and Observation (P&O) method, well-known among specialists in this field, Furthermore, Field-Programmable Gate Arrays (FPGAs), which are standard programmable circuits adaptable to various needs, have become indispensable in applications requiring rapid development time and guaranteed modularity. The objective of this end-of-studies project is to implement the MPPT method on an FPGA circuit.

This work involves several stages, beginning with VHDL programming, synthesis, testing, and temporal simulation, culminating in loading the program onto the FPGA board. This work is divided into three chapters as follows:

Chapter 1 provides an overview of several key topics related to photovoltaic systems and DCDC converters. It begins with an introduction and covers the photovoltaic effect, including the operating principles and characteristics of photovoltaic cells. The chapter also discusses the equivalent circuit model of solar cells and the characteristics of solar modules. It then delves into Maximum Power Point Tracking (MPPT) techniques, presenting various control algorithms used for optimizing power extraction from solar panels. Additionally, the chapter explores different types of DC-DC converters, focusing on step-up (Boost) and Buck-Boost converters, along with their topologies and their relationship to MPPT operation zones. Finally, the chapter concludes with a conclusion.

Chapter 2 provides an overview of FPGA technology and VHDL programming. It begins with an introduction and discusses the basics of FPGAs, including their circuitry and internal architecture. The chapter then introduces VHDL (VHSIC Hardware Description Language), explaining its structure within a file and distinguishing between concurrent and sequential operations. It concludes with a conclusion.

Chapter 3 covers several aspects related to the implementation of a PV panel system, focusing on both hardware and software components. It begins with an introduction and discusses the characterization of PV panels. The hardware section includes details on current and voltage sensors, as well as the Boost DC/DC converter. The software part summarizes the discussed topics, including current and voltage sensor implementations, the Perturbation and Observation (P&O) algorithm, Xadc functionality, and Pulse Width Modulation (PWM). The chapter concludes with a summary of findings and conclusions drawn from the implementation process.

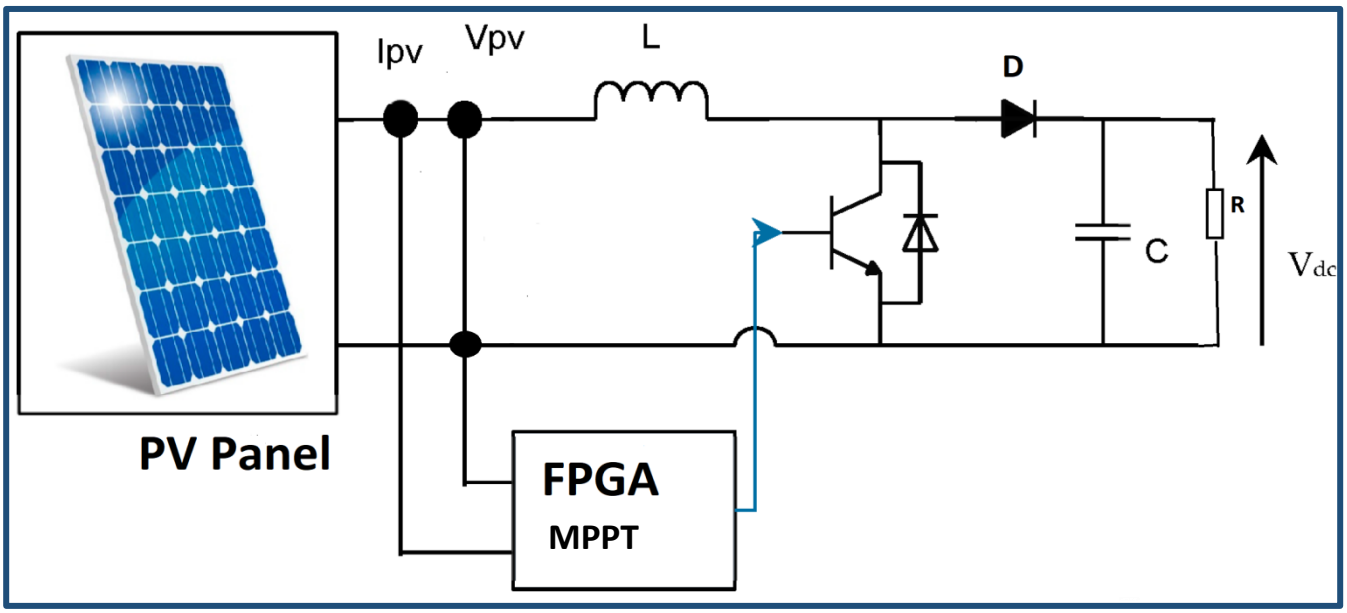


Figure 1: diagram of MPPT system[3]

CHAPTER 1

Introduction to photovoltaic

1.Introduction of chapter1:

Photovoltaics, or PV, is the process of converting sunlight directly into electricity using solar cells. These cells generate an electric current when exposed to sunlight. Over the years, PV technology has evolved significantly, becoming more efficient and cost-effective. It plays a crucial role in clean energy production and contributes to mitigating climate change. Challenges remain, such as intermittency and energy storage, but ongoing research and development continue to shape the future of PV.

In this chapter we going to review the most important thing about photovoltaic energy

2.Photovoltaic effect:

The photovoltaic effect, discovered by Edmond Becquerel in 1839, generates voltage or electric current within the cell when exposed to sunlight. This effect enables **solar panels** to efficiently convert sunlight into electrical energy. [4]

2.1.Operating principles of photovoltaic panels:

Photovoltaic solar panels capture sunlight and convert it into direct current (DC) electricity. These panels consist of **photovoltaic cells,** typically manufactured from semiconductor-grade silicon

2.2.Characteristics of photovoltaic cells (current, voltage, power):

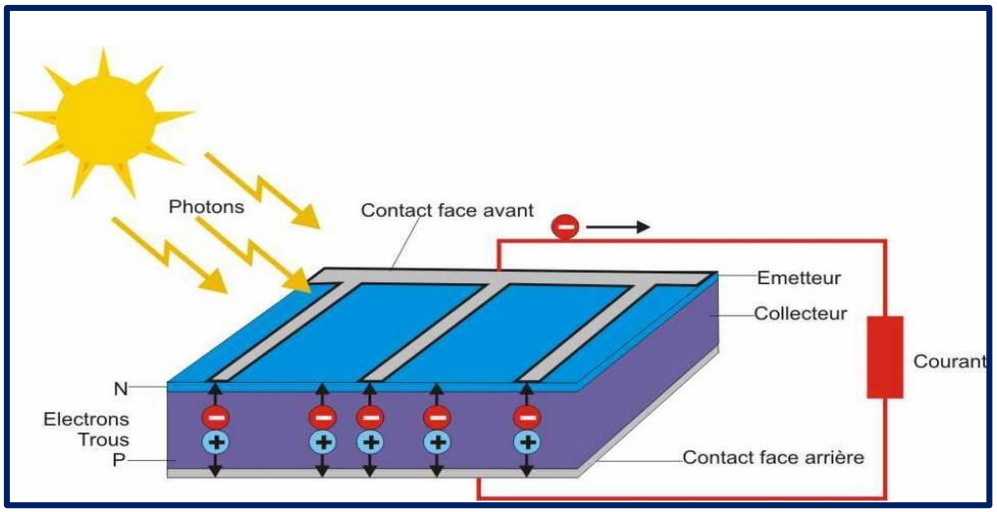


Figure 2: Description of solar cell[5]

As shown in figure 2 the solar cell is made up of several layers. It comprises an N zone (negative) and a P zone (positive), which form an electric field. When photons from the sun strike the front of the cell,

they generateelectrons and holes. These electrons and holes move through the cell, creating an electric current. The current is collected via contacts at the front and back of the cell, and can be used as a source of electrical energy.[5]

2.3.Equivalent circuit model of solar cell:

a solar cell is represented by an electrical equivalent one-diode model as shown in the following figure 3

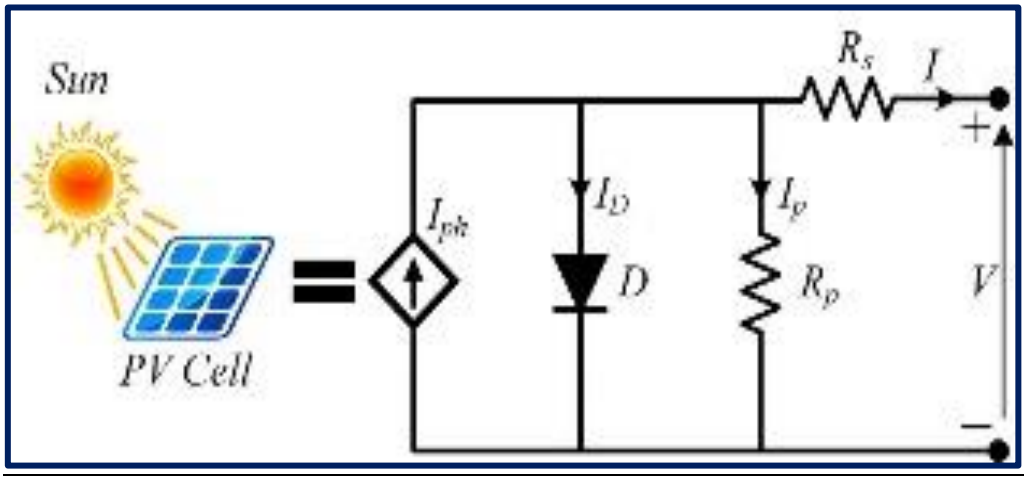


Figure 3 : equivalent circuit diagram of Photovoltaic cell [5]

As shown in Figure 3, the model consists of a current source I_{ph}, a single diode, and a series resistance R_s that indicates the resistance level within the cells. The diode comprises an internal shunt resistance R_p as well. The net current I which marks the difference between the photocurrent diode current, is calculated as :

$$I = I_{ph} - I_0 \left[exp\left(\frac{V + R_s\, I}{\alpha\, V_{th}}\right) - 1 \right] - \left(\frac{V + R_s\, I}{R_p}\right) \quad (1)$$

where, I_0 is dark saturation current (A),

α is the diode's ideality factor, it is between 1 and 2,

V_{th} is the thermodynamic potential of the cell, expressed as follows,

$$V_{th} = \frac{KT}{q} \quad (2)$$

where, K is Boltzmann's gas constant, $1.380658e^{-23}$J/K,

T est the cell's absolute temperature (Kelvin),

q is the charge of an electron, $1.60217733e^{-19}$C.

Load resistance is usually significantly smaller than shunt resistance. However, load resistance is usually much greater than series resistance, which means that a relatively small amount of power actually dissipates within the cell. If these two resistances are ignored, (1) is reduced to:

$$I = I_{ph} - I_D = I_{ph} - I_0 \left[exp\left(\frac{V + R_s\, I}{\alpha\, V_{th}}\right) - 1 \right] \qquad (3)$$

2.4.characteristics of solar cell :

In this figure 4 the **Short Circuit Current (Isc)** (current when voltage is zero), **Open Circuit Voltage (Voc)** (voltage when current is zero), and the **Maximum Power Point (MPP)** (optimal operating point for maximum power output). The shaded area under the curve up to Mpp represents power output.

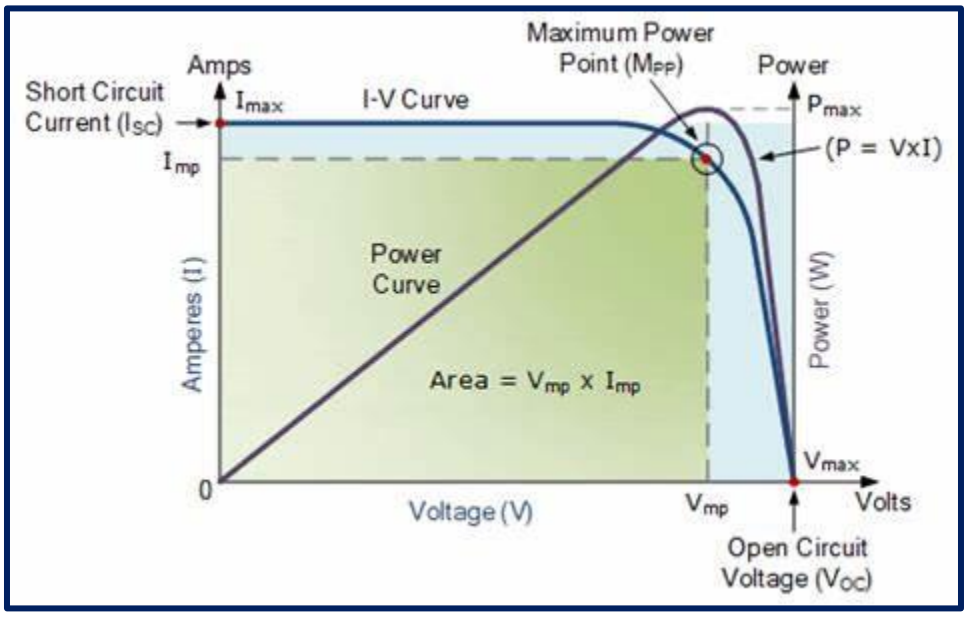


Figure 4:Photovoltaic cell performance characteristics. [6]

2.5.PV module Model

A single photovoltaic (PV) cell is not practically useful on its own, as it typically produces less than a volt of electrical potential. To achieve a usable voltage, multiple PV cells are connected together. These interconnected cells form modules, commonly known as solar panels. Furthermore, these panels can be connected in arrays, as depicted in the illustration below(figure 5)

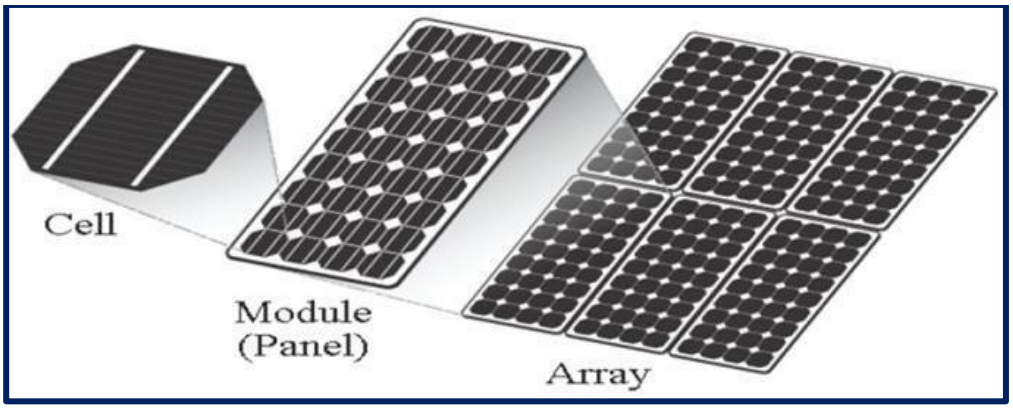


Figure 5:Evolution of solar energy systems: from cells to modules to arrays[7]

A photovoltaic (PV) module comprises solar cells connected either in series or in parallel, depending on the intended application. When cells are connected in series Voltage increases while current remains constant. In parallel connections Current increases while voltage remains constant.

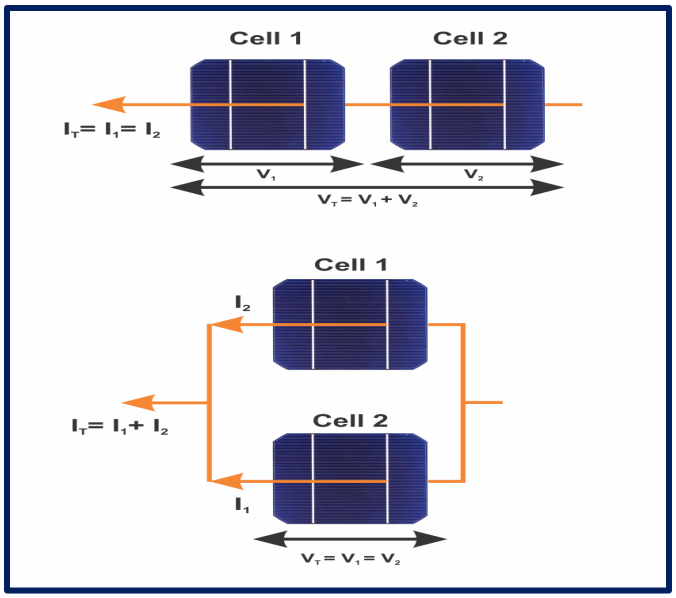


Figure 6:Schematics of series- and parallel-connected solar cells[7]

3.Maximum Power Point Tracking (MPPT) Techniques

The Maximum Power Point Tracking (MPPT) technique ensures that a photovoltaic (PV) generator operates at its maximum power, regardless of changing meteorological conditions. By automatically adjusting the duty cycle of a DC-DC converter, the system continually maximizes power output from the PV panel. Choppers (DC-DC converters) serve as power interfaces controlled by the MPPT regulator,

allowing fine-tuning of the output voltage to match load requirements.[8]

3.2.The various MPPT control algorithms:

3.2.1.Perturband Observe (P&O) Algorithm:

The P&O algorithm is one of the most commonly used MPPT algorithms today due to its ease of implementation and satisfactory cost-to-performance ratio. It involves measuring the voltage and current at the solar panel output to determine the power and output voltage. Subsequently, it calculates disturbances (deltaP and deltaV) and adjusts the duty cycle at the input of the boost converter to increase or decrease the voltage, thereby maintaining the maximum power point. You can refer to the following figures 7 and 8 to understand its exact operation.

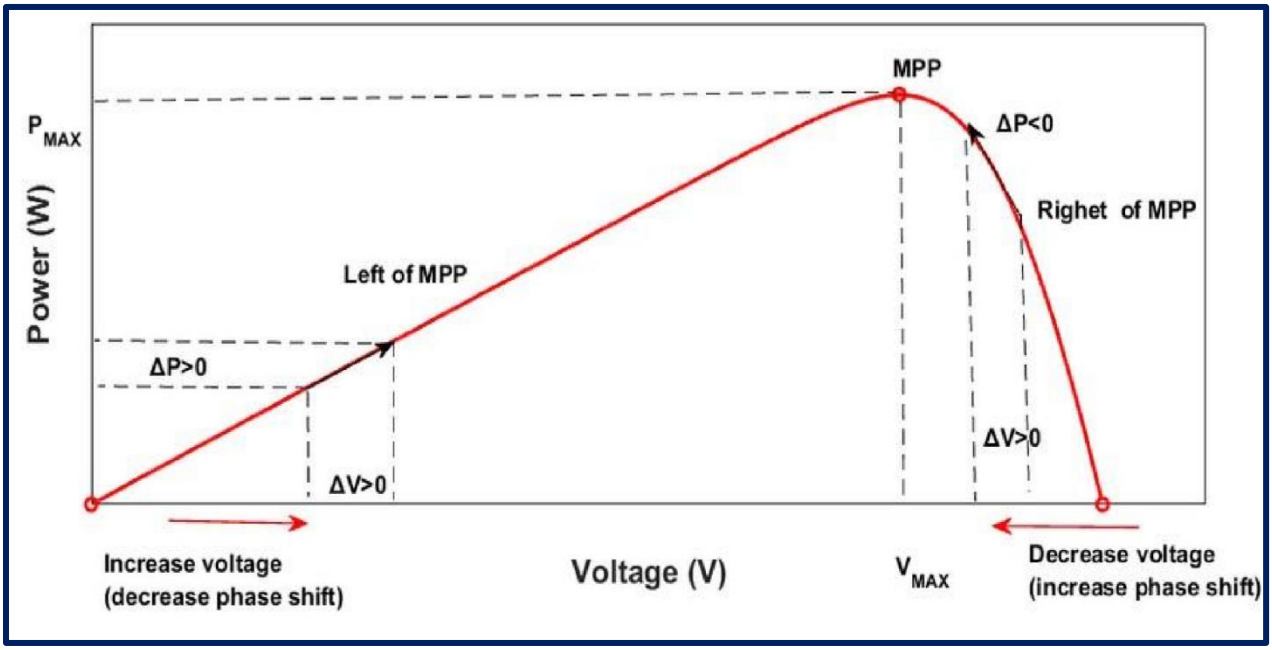


Figure 7:P&O power curve and MPP[8].

Curve Shape:

The curve in the graph represents the **power-voltage characteristic** of a **photovoltaic cell** (a solar cell).It starts from the origin (zero voltage and zero power).The curve rises to reach the **Maximum Power Point (MPP)**,Beyond MPP, the curve falls again towards a maximum voltage point (V_MAX) where power returns to zero.

- MPP (Maximum Power Point):

The MPP is the optimal operating point for the solar cell.
At MPP, the power output is at its highest (P_MAX), and the voltage is at its optimal value (V_MPP).
Operating near MPP ensures maximum efficiency.

- Left of MPP:

Increasing voltage ($\Delta V > 0$) leads to an increase in power ($\Delta P > 0$).
This region corresponds to an increase in phase shift.

- Right of MPP:

Decreasing voltage ($\Delta V < 0$) results in a decrease in power ($\Delta P < 0$).
This region is associated with a decrease in phase shift.

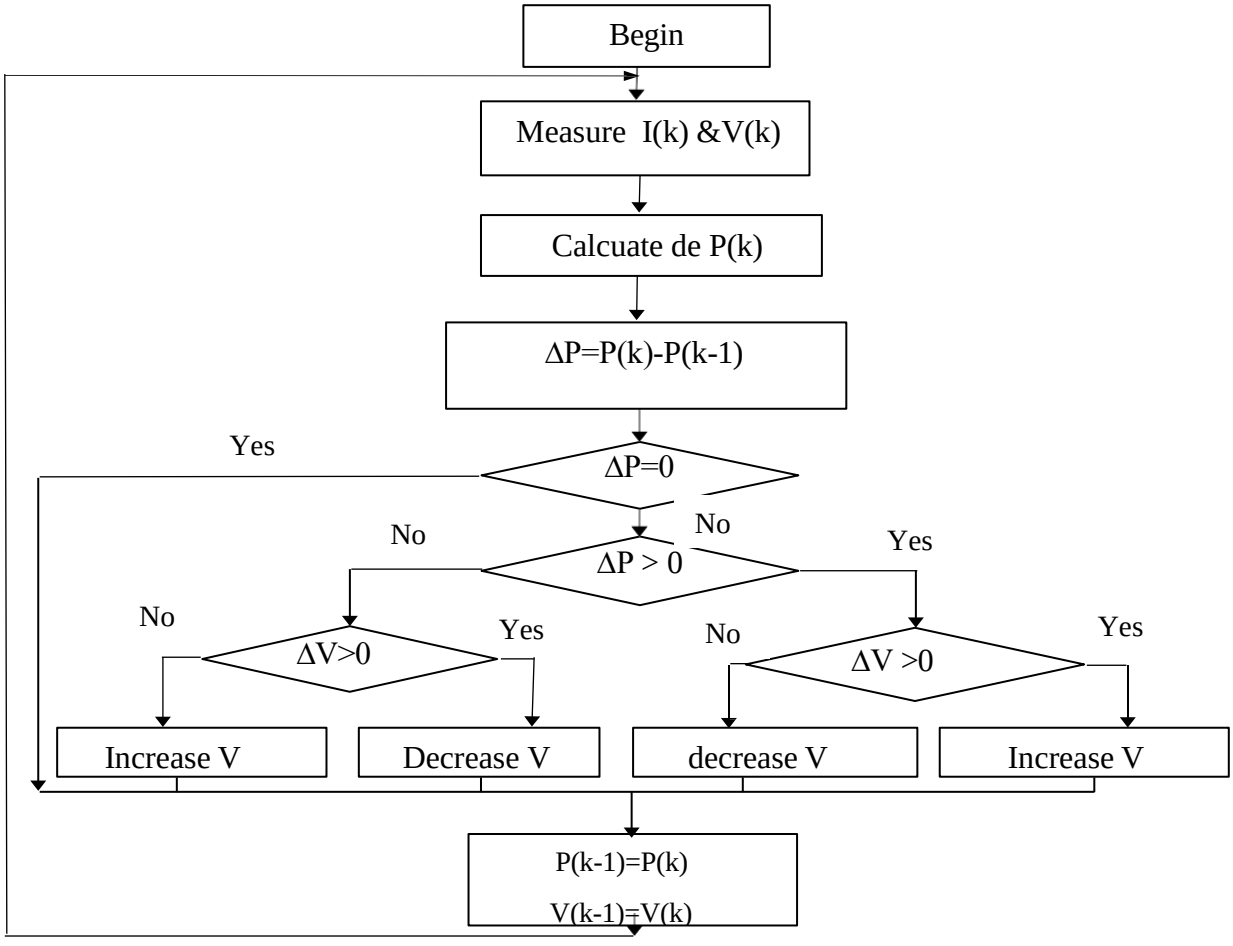


Figure 8:P&O MPPT algorithm[9]

3.2.2.Incremental Conductance (INC) Algorythm:

The Incremental Conductance (INC) MPPT technique is one of the most widely used MPPT strategies, which offers the advantage of fast tracking of the MPP. Compared to the P&O MPPT strategy , INC combines and utilizes the unique characteristics of both the P-V output curve and the I-V curve of the photovoltaic generatorand , enabling it to track the MPP more rapidly and accurately. You can refer to the following figures to understand its exact operation figure 9.[10]

This technique relies on the slope of the P-V characteristic curve (Figure 7), where the MPP is tracked when dP/dV = 0 as follows:

$$\begin{aligned} \frac{\Delta P}{\Delta V} &= \frac{\Delta(V \cdot I)}{\Delta V} = I\frac{\Delta V}{\Delta V} \\ \frac{\Delta P}{\Delta V} &= I + V\frac{\Delta I}{\Delta I}\Delta V \\ \frac{\Delta P}{\Delta V} &\simeq I + \frac{\Delta I}{\Delta V} \end{aligned} \qquad (4)$$

The previous Equation indicates that the MPP is reached when $\Delta I/\Delta V = -I/V$.

If $\Delta I/\Delta V > -I/V$, the operating point is to the left of the MPP on the P-V curve.

If $\Delta I/\Delta V < -I/V$, the operating point is to the right.

The MPP can thus be tracked by comparing the instantaneous conductance (I/V) to the incremental conductance (ΔI/ΔV) as indicated in the flowchart in Figure 9.

The size of the increment determines the tracking speed of the MPP. Faster tracking can be achieved with larger increments, but the system may not precisely operate at the MPP and may oscillate around it [10]

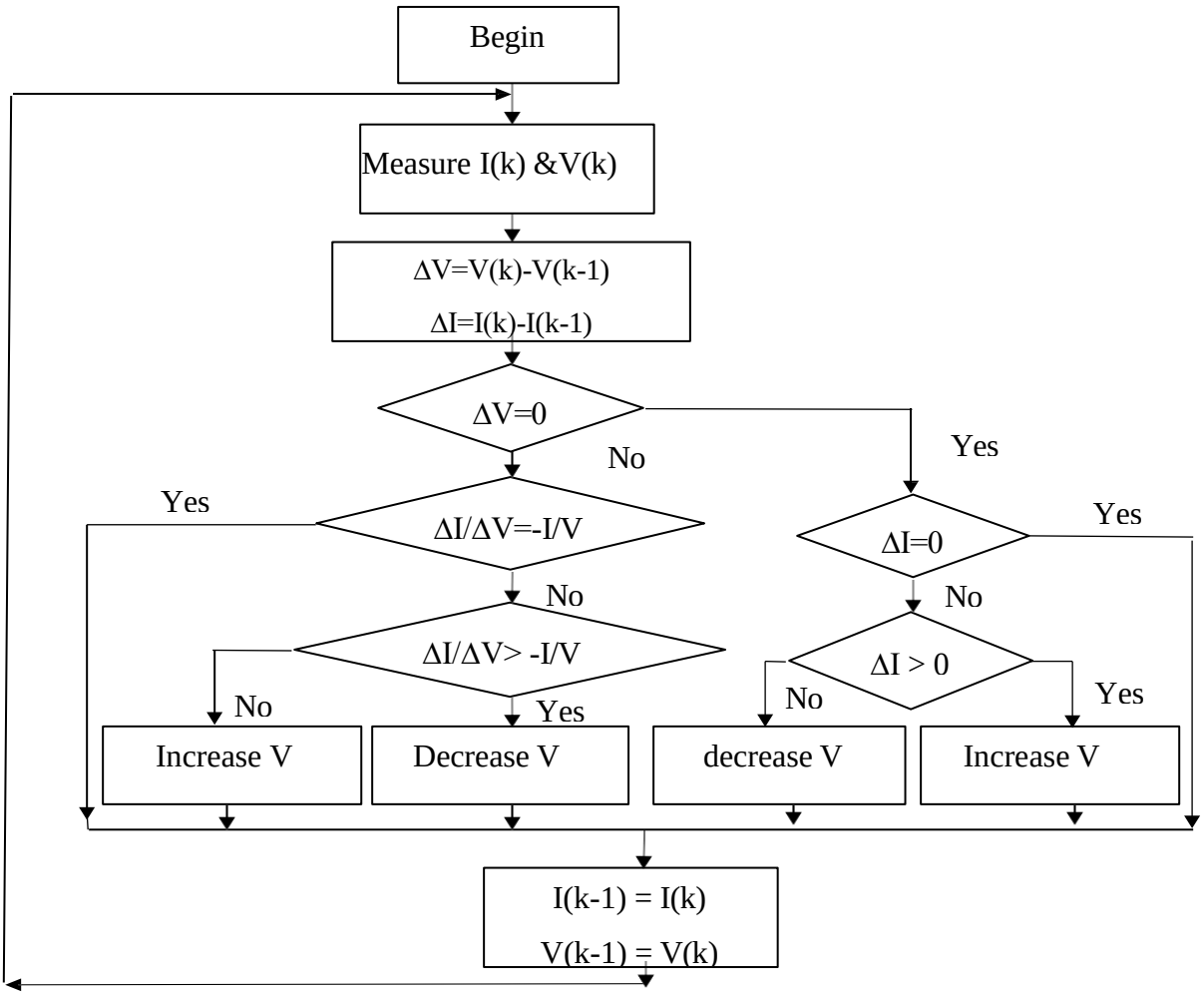


Figure 9:Incremental conductance (INC) algorythm[10]

3.2.3.Fuzzy Logic:

The FL (Fuzzy Logic) controller is an innovative approach based on artificial intelligence. Compared to the classical IC (Incremental Conductance) algorythm, FL offers better robustness, stability, and ease of implementation. Like other MPPT (Maximum Power Point Tracking) controllers, its main task is to reach the maximum power point (MPP). However, the performance of the FL controller essentially depends on human expertise.

The FL approach stems from decomposing a range of variation of a real variable into linguistic variables, each assigned a membership function. The rules, developed from human operator expertise, are expressed linguistically and determine the dynamic performance of the FL controller.

The proposed FL controller consists of four basic elements: the fuzzification unit, the basic rules, the inference engine, and the defuzzification. The fuzzification unit converts real variables into fuzzy variables. The inputs E(k) and CE(k) of the FL controller are given by the equations. E(k) represents the derivative of photovoltaic power (PPV) at instant k, and becomes zero when the operating point reaches the MPP. CE(k) quantifies the error of E(k). Voltage and current are measured for PPV power calculation. The FL controller's output, denoted as dD, corresponds to the variable duty cycle step.[11]

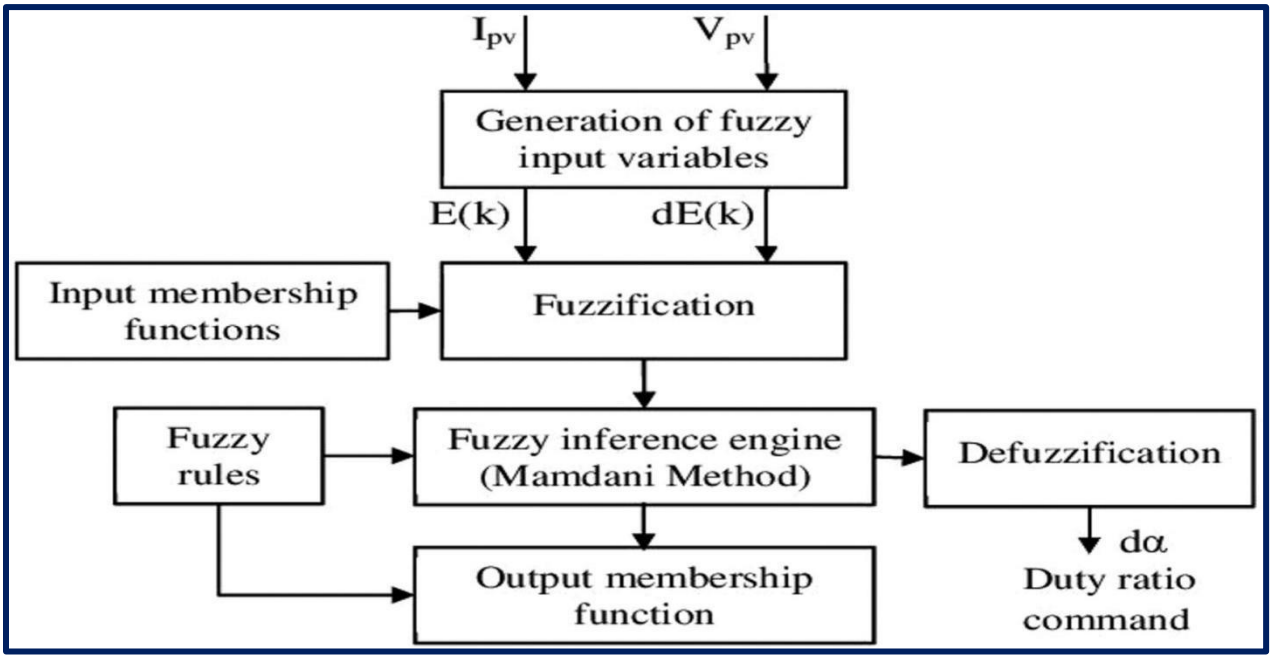


Figure 10:Block diagram of the Mamdani fuzzy controller.[11]

$$E(k) = \frac{\mathrm{I(k)} - \mathrm{I(k-1)}}{\mathrm{V(k)} - \mathrm{V(k-1)}} \quad (5)$$

$$CE(k) = \mathrm{E(k)} - \mathrm{E(k-1)} \quad (6)$$

4.DC-DC converter:

A DC-DC converter, also called a DC-DC power converter or voltage regulator, is an electrical system (device) which converts direct current (DC) sources from one voltage level to another. In other words, a DC-DC converter takes as input a DC input voltage and outputs a different DC voltage. The output DC voltage can be **higher** or **lower** than the DC input voltage. As the name implies, a DC-DC converter only works with direct current (DC) sources and not with alternative current (AC) sources.

CHAPTER 1: Introduction to photovoltaics

4.1.Types of DC-DC converters:

4.1.1. Buck converter:

The step-down DC–DC converter, commonly known as a buck converter, is shown in figure11. It consists of DC input voltage source VS , controlled switch S, diode D, filter inductor L, filter capacitor C, and load resistance R

The buck converter takes an input voltage (supply voltage) and converts it to a lower output voltage (load voltage).

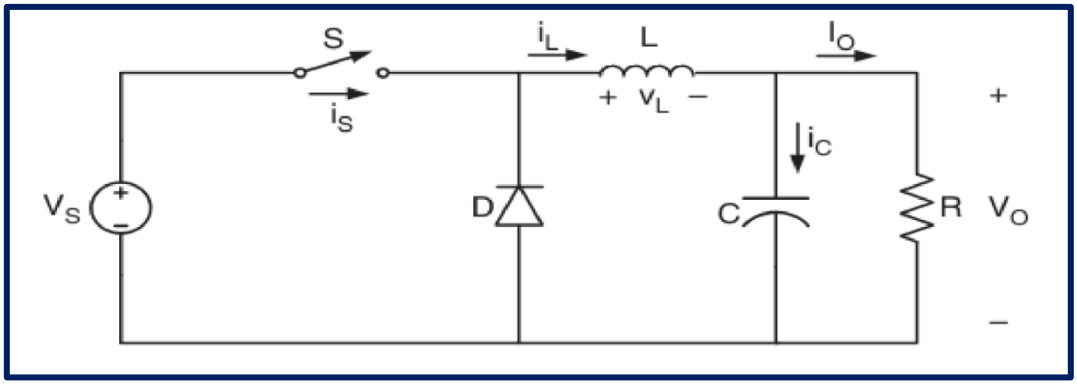


Figure 11:Circuit diagram of buck converter[12]

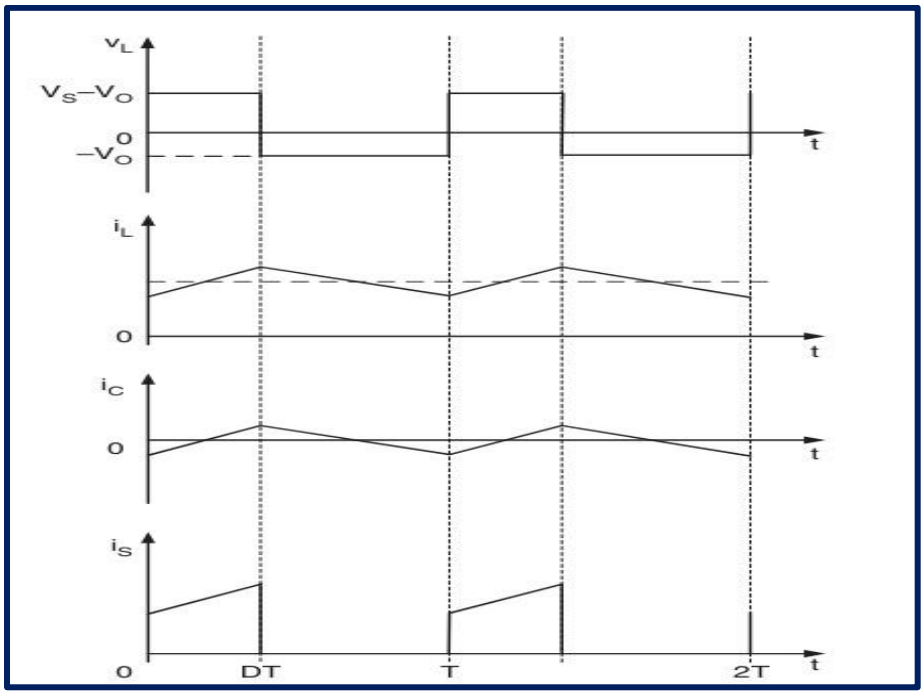


Figure 12:Waveforms of Buck converter. [12]

3. 2. Step-up (Boost) converter :

Figure 14 shows a boost converter which converts a DC voltage into a higher DC voltage. It comprises an inductor (L), a switch (S) (transistor), a diode (D) which protects the transistor from reverse current flow and a capacitor (C) which smoothes the output voltage using a resistor as a load.

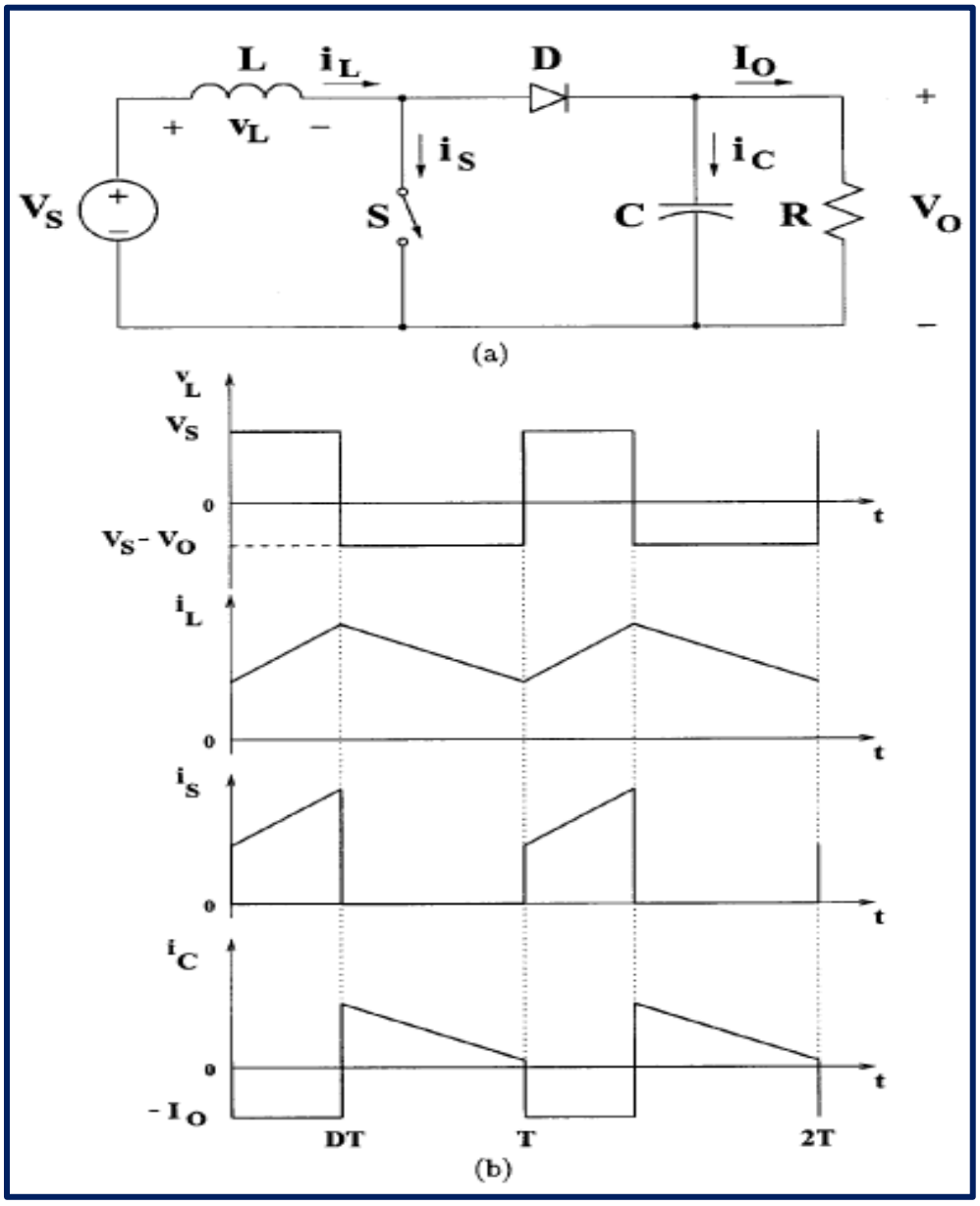


Figure 13 : Boost converter and its input and output voltages and currents as a function of the duty cycle D and the period T[12]

- When switch S is in the on position, the current in the inductor increases linearly and diode D is extinguished at this time (0<t<DT).

- using the law of mesh:

$$\frac{di}{dt} = \frac{VS}{L} \Rightarrow i - Imin = \frac{VS}{L}t$$

and $i(DT) = Imax$ (7)

$$Imax - Imin = \frac{VS}{L}DT$$

- When switch S is off, the energy stored in the inductor is released through the diode (on) to the load (DT<t<T).
using the law of mesh:

$$\frac{di}{dt} = \frac{VS - Vo}{L} \Rightarrow i - Imax = \frac{VS - Vo}{L}(T - DT)$$

and and $I(T) = Imin$ (8)

$$Imax - Imin = \frac{Vo - Vs}{L}(1 = D)T$$

Transfer function by identifying (7) and (8) we obtain

$$\frac{Vo - Vs}{L}(1 - D)T = \frac{VS}{L}DT => \frac{Vo}{Vs} = \frac{1}{1 - D} \quad (9)$$

$$V_s \times I_s = V_O \times I_O$$

replace (8) in (9)

$$V_S \times I_S = \frac{V_S}{1 - D} \quad (10)$$

$$I_O = (1 - D) \times I_S$$

To calculate L:

$$T=\frac{1}{f} = t1+t2=DT + (T - DT)$$

from (8) and (9) we have

$$T=\frac{\Delta I*L}{Vs}+\frac{\Delta I*L}{Vo-Vs}=\frac{\Delta I*L*Vo}{Vs(Vo-Vs)}$$

$$L=\frac{Vs(Vo-Vs)}{f*\Delta I*V0} \quad (11)$$

To calculate C: in the 0<t<D part Io=Ic

$$\Delta Vc = \frac{1}{C}\int_0^{\mathrm{DT}} Ic\, dt = \frac{1}{C}\int_0^{\mathrm{DT}} Io\, dt = \frac{IoDT}{C} \quad (12)$$

of (8)

$$\mathrm{DT} = \frac{Vo - Vs}{V0 * f}. \quad (13)$$

replace (12) in (13)

$$\Delta Vc = \frac{Io(Vo-Vs)}{Vo*f*C} \Rightarrow \boxed{C = \frac{Io(Vo-Vs)}{Vo*f*\Delta V}} \quad (14)$$

3. 3. Buck-Boost converter :

A non-isolated (transformerless) topology of the buck–boost converter is shown in Figure 14 The converter consists of DC input voltage source V_S , controlled switch S, inductor L, diode D, filter capacitor C, and load resistance R. With the switch "on", the inductor current increases while the diode is maintained "off". When the switch is turned "off", the diode provides a path for the inductor current. Note the polarity of the diode which results in its current being drawn from the output

The Buck-Boost converter can "step up" or "step down" the DC voltage.

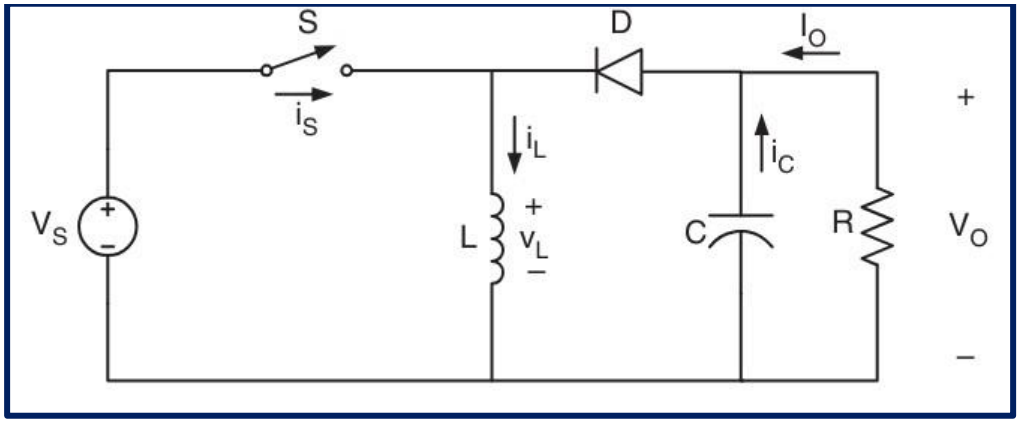


Figure 14:Circuit diagram of buck-Boost converter[12]

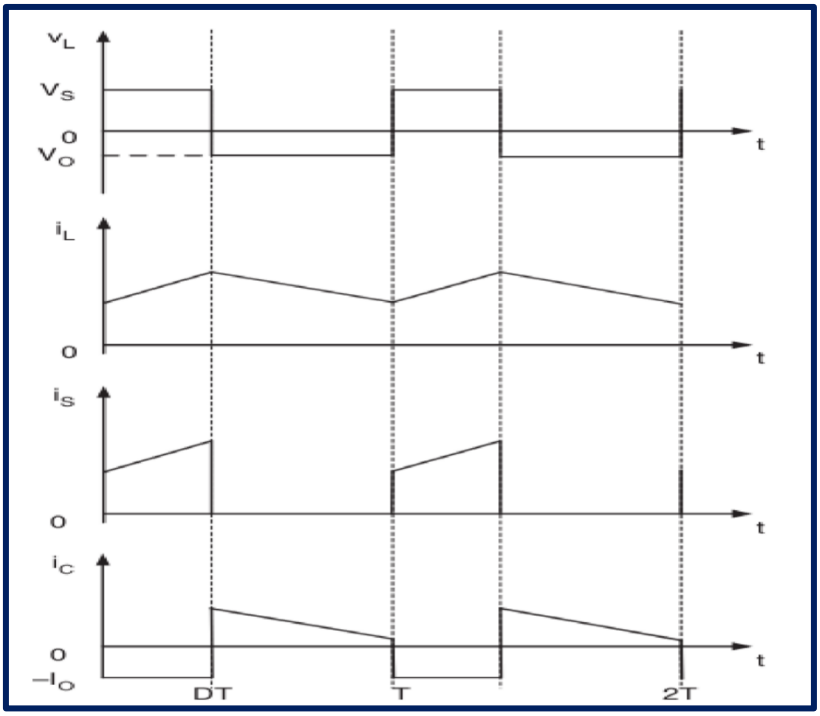


Figure 15:Waveforms of Buck-Boost converter [12]

5-Conclusion of chapter1:

In this chapter, we introduced the concept of photovoltaics. We covered the photovoltaic effect, which explains how solar cells convert sunlight into electricity. Additionally, we discussed the characteristics of solar cells, including current, voltage, and power. The equivalent circuit model of a solar cell was also explored.finnaly we talked a bout several type od DC-DC converter.

CHAPTER 2

FPGA &VHDL

CHAPTER 2: FPGA &VHDL

1.Introduction of chapter2:

Field-Programmable Gate Arrays (FPGAs) are versatile integrated circuits that allow users to configure digital logic circuits. They find applications in various fields, including telecommunications, aerospace, and industrial automation. VHDL (VHSIC Hardware Description Language) is a powerful language used for designing and simulating digital systems. It provides a structured way to describe hardware behavior and facilitates efficient FPGA programming. Understanding both FPGAs and VHDL is essential for engineers and developers working on digital design projects.[13]

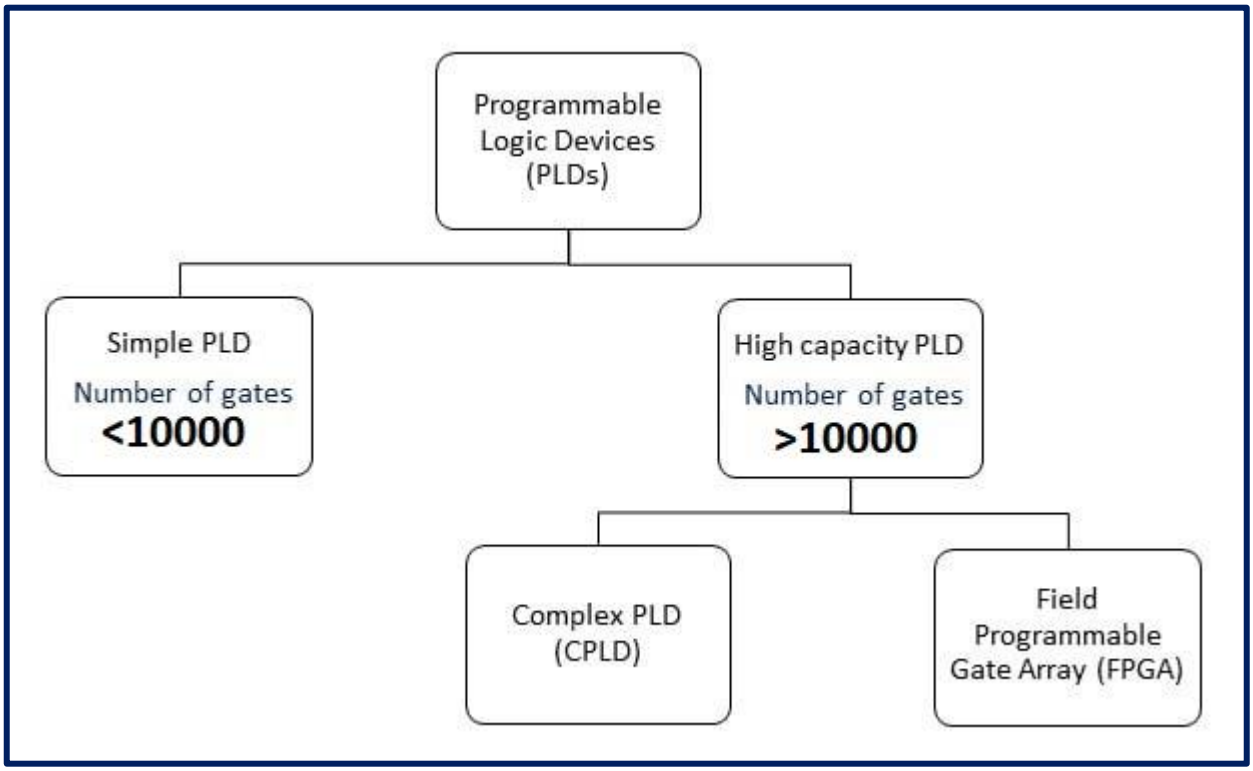


Figure 16: Classification of PLDs [13]

2.FPGAs:

2.1.Circuit FPGAs:

FPGAs are reprogrammable integrated circuits based on arrays of configurable logic blocks whose connections can be modified according to the designer's needs. This means that FPGAs are programmable, which gives them great flexibility of use, since their programming can be modified to improve them or correct bugs [13].

CHAPTER 2: FPGA &VHDL

Figure 20 shows the NEXYS 4 DDR[14]

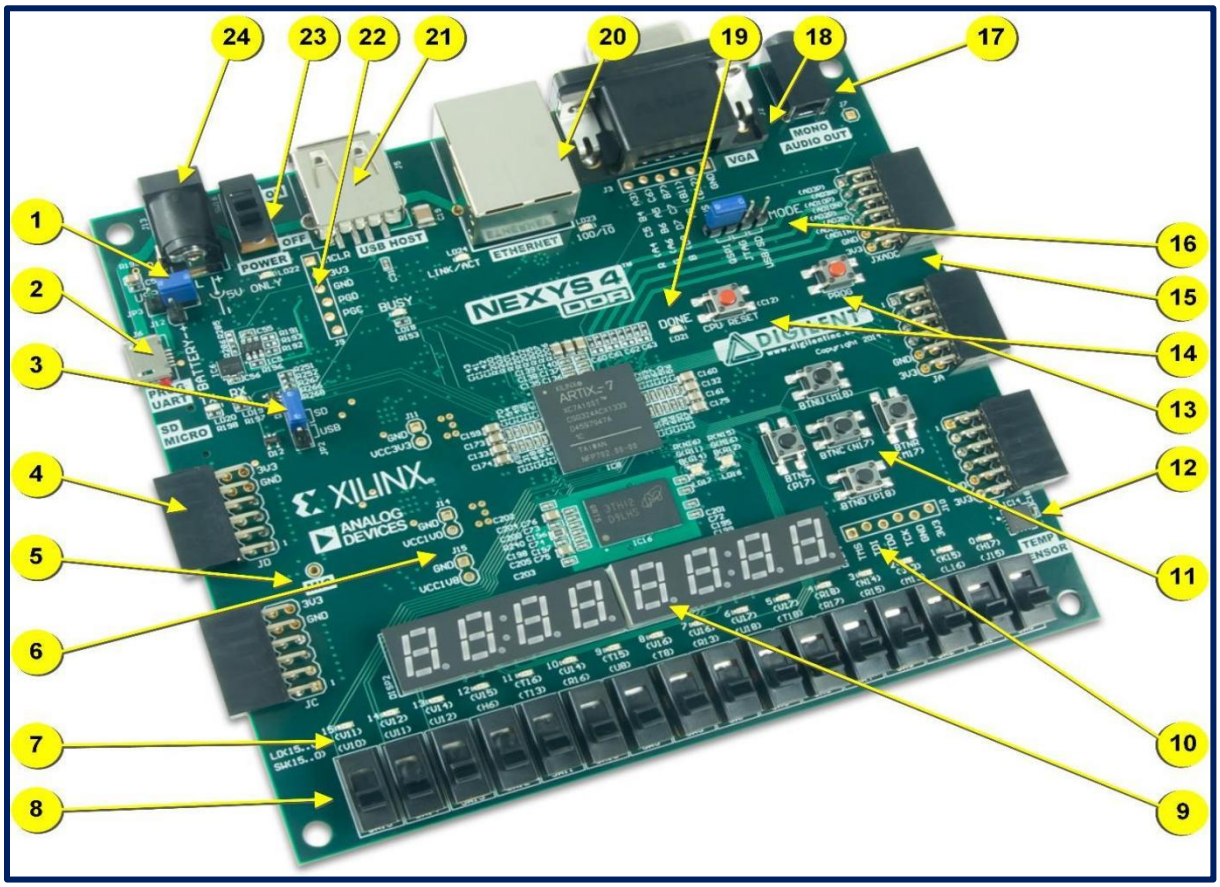


Figure 17 NEXYS 4 DDR[14]

Callout	ComponentDescription	Callout	ComponentDescription
1	Power select jumper and battery header	13	FPGA configuration reset button
2	Shared UART/JTAG USB port	14	CPU reset button (forsoftcores)
3	Externa lconfiguration jumper(SD/USB)	15	Analog signal Pmodport(XADC)
4	Pmodport(s)	16	Programming mode jumper
5	Microphone	17	Audio connector
6	Power supply test point(s)	18	VGA connector
7	LEDs(16)	19	FPGAprogrammingdoneLED
8	Slideswitches	20	Ethernet connector
9	Eight digit7-seg display	21	USB host connector
10	JTAG port for(optional)external cable	22	PIC24 programming port(factoryuse)
11	Five push buttons	23	Power switch
12	Temperature sensor	24	Power jack

The Nexys4 DDR features an on-chip XADC that allows you to perform analog-to-digital conversions.

It provides 16 single-ended analog input channels (AD0 to AD15) with a voltage range of **0V to 1V.** The XADC can also operate in differential mode, measuring the voltage difference between two pins.

FPGA: The board uses an Artix-7 XC7A100T-1CSG324C FPGA with 15,850 logic slices, 240 DSP slices, and clock speeds of 100 MHz.
Peripherals: It includes user switches, LEDs, VGA output, accelerometer, temperature sensor, and more.
XADC Port: The Nexys4 DDR has a Pmod port for XADC signals, allowing you to connect external analog sensors or devices.[14]

2.2.Internal architecture of FPGA:

The architecture of an FPGA varies from one manufacturer to another and also according to the technology used. In generic terms, as shown in Figure 21, an FPGA is made up of configurable logic blocks (CLBs), surrounded by input/output blocks (IOBs), interconnected via vertical and horizontal communication channels [15]

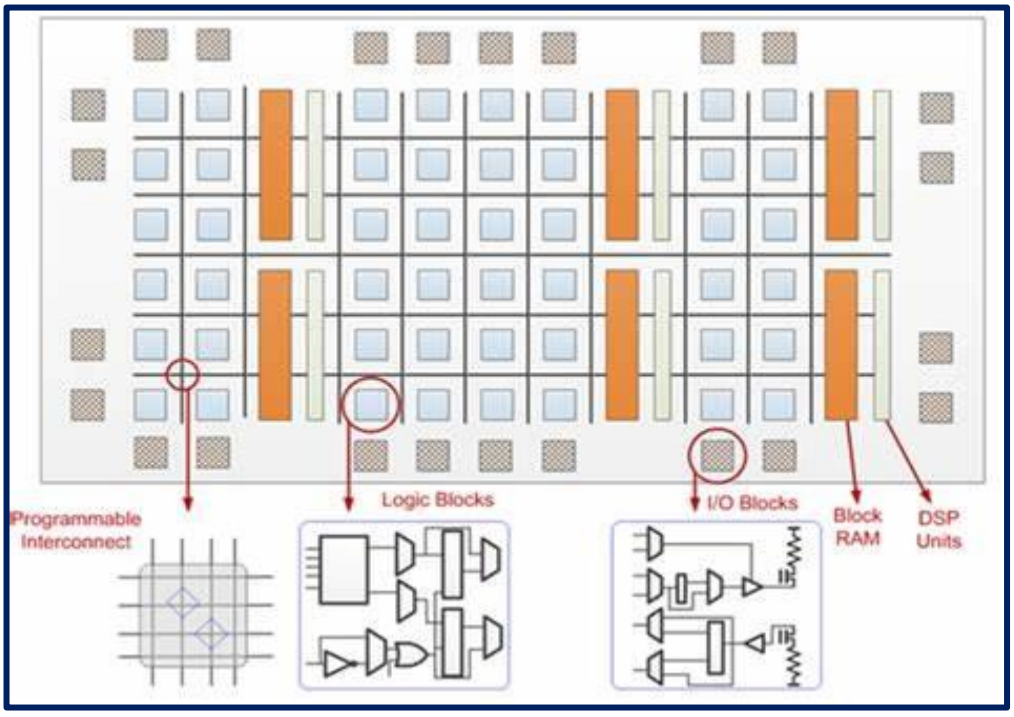


Figure 18:FPGA board architecture[15]

Logic blocs: These are the basic blocks used to build a digital circuit. All FPGAs have configurable logic blocks. This is the heart of the FPGA, which enables the various logic functions to be implemented.

Programmable interconnect: This is a multi-purpose, multi-level interconnection structure between FPGA components.

In order to implement the logic elements presented in the previous sections, they must not only be correctly configured, but also connected to each other. The internal interconnect structure of an FPGA consists of a set of wires connected by programmable elements

I/O Blocks : IOBs provide the interface between the FPGA component pins and the internal logic. They are present throughout the periphery of the FPGA circuit. Each IOB block controls one pin of the component and can be defined as input, output, bidirectional signals, etc. These are divided into banks configured to interconnect the logic of different electrical standards independently

Block Ram: is a type of memory resource within the FPGA that provides high-speed storage for data that can be quickly accessed by the DSP units or other logic within the FPGA.

DSP Units:are specialized circuits designed to perform high-speed arithmetic operations, which are crucial for tasks like digital signal processing, including audio and video processing.

3.VHDL:

3.1.Introduction to VHDL (VHSIC Hardware Description Language)

VHDL as Verilog is a language for creating digital circuits without drawing them. It lets you write instructions for how the circuit should behave, using simple terms instead of complex diagrams or equations as assembly language, and it closer to the human language instead of the machine language. These instructions are saved in ".vhd" files, and a compiler turns them into the actual circuit.[16]

3.2.Structure of a VHDL file:

A digital system is described by its inputs and outputs and the relationship between them.
VHDL, on the one hand, describes the external aspect of the circuit: inputs and outputs; and on the other hand, links the inputs to the outputs. The description of the input/output ports is in the entity part, and the description of the circuit's behaviour is in the architecture part. Each architecture must be associated

with an entity.

In addition, although it is not strictly necessary, libraries and packages can also be defined, which will indicate what types of ports and operators can be used. The definition of libraries and packages must always appear before the definition of the entity.[16]

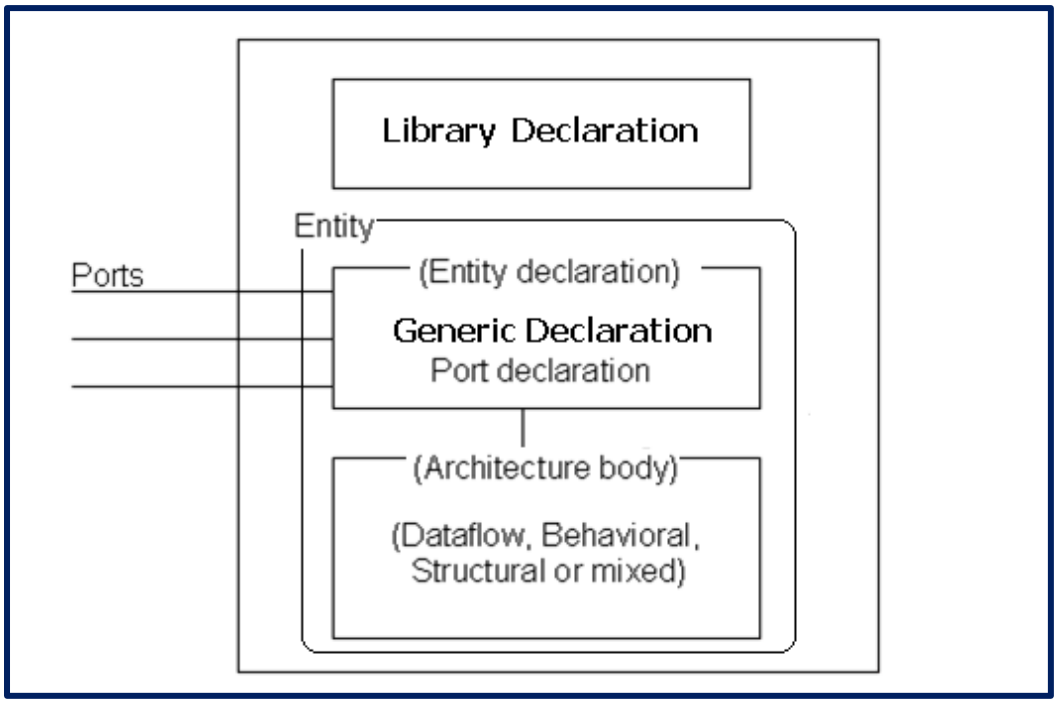


Figure 19: Structure of a VHDL code[16]

3.2.1.Bookshop declaration (Library) :

A library is a library where precompiled design entities are stored. They are declared in the header of a VHDL description using the 'library' instruction. Descriptions are compiled in the 'work' library by default unless another library is currently activated in the compilation tool.

3.2.2.Fundamental VHDL packages (Package)

After declaring the libraries, the 'use' instruction is used to access the objects that can be used in the rest of the VHDL description.

The table above shows the packages developed and standardized by the IEEE (Institute of Electrical and Electronics Engineers).

Table II. 1: Packages associated with the IEEE library

Packages	Their functions
std_logic_1164	To define std_logic types .
std_logic_arith	To use the arithmetic operators '+', '-', '*', '/' for std_logic types.
numeric_std	Use of decimal values for the std_logic type.
std_logic_signed	std_logic types with signed values.
std_logic_unsigned	The std_logic type with unsigned value.
math_complex	provides a numerical calculation using complex numbers.
math_real	provides a numerical calculation using real numbers.
std_logic_textio	Use of ASCII values for the std_logic type.
numeric_bit	Use of bit values for the std_logic type.

3.2.3.Declaration of the entity (Entity):

An entity declaration is a blueprint that outlines how to connect a circuit component to its environment. It's a common template for various designs, specifying the connections and their types without detailing the internal workings, similar to a symbol on a circuit diagram.

```
entity ENTITY_NAME is
  port (
    PORT_NAME1 : DIRECTION DATA_TYPE;
    PORT_NAME2 : DIRECTION DATA_TYPE;
    -- Add more ports as needed
  );
end ENTITY_NAME;
```

Figure 20: Entity declaration syntax

- **ENTITY_NAME** is the name you give to your entity.
- **PORT_NAME** is the name of each interface port.
- **DIRECTION** can be in, out, inout, or buffer, indicating the direction of data flow.

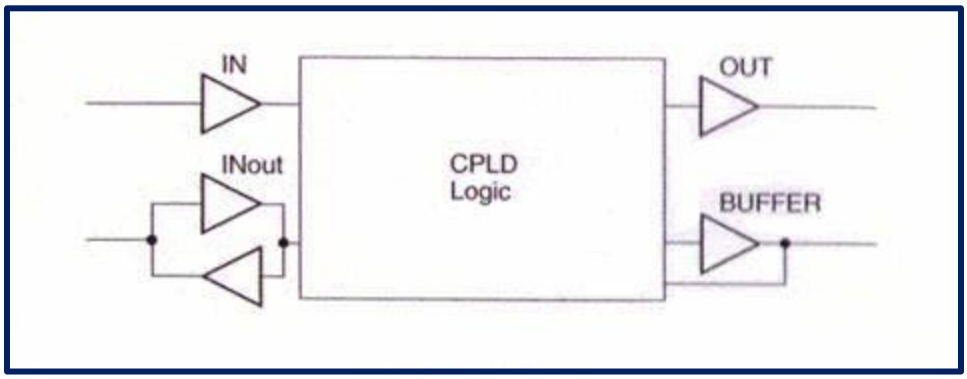


Figure 21: Port modes of a VHDL entity[16]

- **IN** The entity reads an input signal supplied externally
- **OUT** The entity provides an output signal, but cannot read back this signal.
- **INOUT** The signal is bidirectional
- **BUFFER** The entity's architecture produces a usable output signal, which can also be read back by the entity as an internal signal

- **DATA_TYPE** specifies the type of data for each port:
  1. **bit**: Represents a single binary value, either '0' or '1'.
  2. **std_logic**: Similar to bit, but includes additional states like 'U' (undefined) and 'Z' (high impedance).
  3. **bit_vector**: An array of bit values, used for representing multiple binary values.
  4. **std_logic_vector:** An array of std_logic values, also used for multiple binary values.
  5. **integer**: Represents whole numbers.
  6. **real:** Used for floating-point numbers.
  7. **boolean:** Represents true or false values.
  8. **character:** Represents a single character.
  9. **string:** Represents a sequence of characters.

3.2.3.Architecture statement (Architecture)

The body of an architecture defines the body of a design entity. It specifies the relationships between inputs and outputs. This specification can be expressed in behavioural, structural or dataflow form, and the three forms can coexist within the same architecture body. The declarative part declares objects

(internal signals, functions, procedures, constants, etc.) which will be visible and usable by the design entity. All the instructions that describe the body of the architecture are concurrent instructions that execute asynchronously with each other.

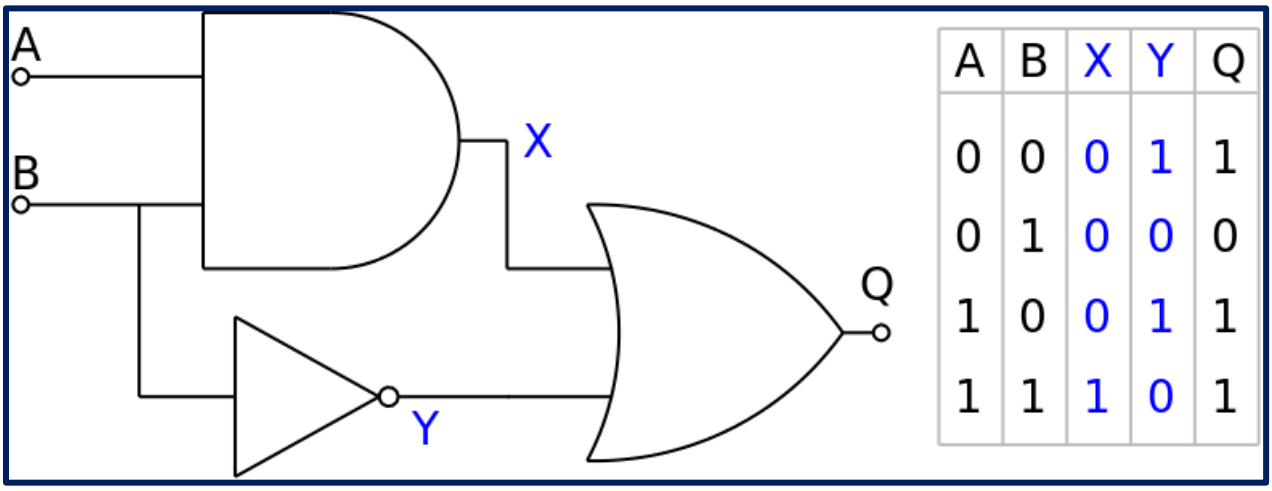
```
architecture ARCHITECTURE_NAME of ENTITY_NAME is
   -- Declarations (signals, constants, types, etc.)
begin
   -- Concurrent statements (processes, assignments, etc.)
end ARCHITECTURE_NAME;
```

Figure 22:Architecture statement

3.3.Concurrent operation and sequential

3.3.1.Operation competitor

In a logic circuit, all the gates operate simultaneously. We then say that the gates operate concurrently, i.e. all the operations take place in parallel , The order in which these processes appear in the description is not important .Taking the logic function in the figures below as an example:



A	B	X	Y	Q
0	0	0	1	1
0	1	0	0	0
1	0	0	1	1
1	1	1	0	1

Figure 23:Simple logic function

The logic gates of this function run in parallel here is the architecture that describes this

```
architecture Behavioral of LogicCircuit is
begin
    Q <= X NOR Y;
    X <= (A AND B);
    Y <= NOT B;

end Behavioral;
```

Figure 24:Description of a logic function

3.3.2.The 5 concurrent instructions of VHDL:

- simple assignment: **... <= ....**
- conditional assignment: **when ... else**
- selective assignment: **with ... select**
- component instantiation: **When a component is used, it must be declared and the entity-architecture pair to be used must be specified. This is called configuration.**
- Conditional generation:

The generic syntax for each instruction is as follows:

```
begin

-- Simple assignment
my_signal <= '1'; -- Assigns the value '1' to my_signal

-- Conditional assignment
output1 <= '0' when input1 = '1' else '1'; -- Assigns '0' to output1 if input1 is '1', else assigns '1'

-- Selective assignment
with input2 select
output2 <= '0' when '0',
           '1' when others; -- Assigns '0' to output2 if input2 is '0', else assigns '1'

-- Component instantiation
my_component_inst: my_component PORT MAP(
    comp_input => input1,
    comp_output => my_signal
); -- Instantiates my_component with mappings

-- Conditional generation
gen: if input1 = '1' generate
    -- VHDL statements that are conditionally generated
end generate gen;

end Behavioral;
```

Figure 25: 5 concurrent instructions of VHDL

3.4.Operation sequential:

Since concurrent operation is not always the most comfortable way of describing systems, particularly very complex systems, VHDL also allows algorithmic operation, which combines instructions executed in series with others in concurrent operation.

- **The process**

Is a particular VHDL structure that is mainly reserved for containing instructions that do not necessarily have to have their value defined for all inputs. This forces the process structure to store its signal values and can (not always) lead to sequential sub-circuits. Furthermore, in simulation only instructions internal to this structure are executed when one of the signals in its sensitivity list changes value.

The generic syntax of a process is as follows:

```
process (sensitivity_list)
begin
    -- process instructions
end process;
```

Figure 26: VHDL generic syntax of a process

- **Instruction case...is...when**

The generic syntax of the **case...is...when instruction** is as follows:

```
case expression is
    when value1 =>
        -- instructions for case 1
    when value2 =>
        -- instructions for case 2
    when others =>
        -- instructions for all other cases
end case;
```

Figure 27: VHDL syntax of the case...is...when

- **Loop instructions**

The generic syntax for loop instructions is as follows:

```
-- Simple loop with exit condition
loop
    -- loop instructions
    exit when condition;
end loop;

-- For loop
for i in range loop
    -- loop instructions using i
end loop;

-- While loop
while condition loop
    -- loop instructions
end loop;
```

Figure 28: VHDL syntax for loop instructions

4. Conclusion of chapter2 :

At the beginning of this chapter we discussed PLDs and FPGA circuits, illustrating the architecture of the latter, then we presented the VHDL language, which is one of the languages used to program FPGA circuits, we explained the structure of a code written in this language and then we determined the two concurrent and sequential operating modes and the instructions used.

CHAPTER 3

Practical applications

CHAPTER 3: practical applications

1.Introduction of chapter 3:

In our practical application, we decided to implement our Maximum Power Point Tracking (MPPT) system using FPGA boards called Nexys 4 DDR and the Vivado software. We coded the MPPT algorithm in VHDL, incorporating a current sensor (AC712) and a simple voltage divider for voltage sensing. Additionally, we utilized a 80W solar panel(STP085-12/Bb). Considering the benefits of the Perturb and Observe (P&O) algorithm, especially its ease of implementation, we chose this method for our project.

2. PV panel:

The Unisun 80W 12V Monocrystalline solar panel from Uniteck is a reliable choice for solar power generation.

The Unisun 80W 12V Monocrystalline Solar Panel boasts a multilayer structure that ensures remarkable efficiency even in demanding sunlight conditions (such as high heat or low sunlight). Additionally, it features an anti Hot-spot system to safeguard against cell shading effects. Its robust construction includes an anodized aluminum frame and tempered glass, making it resistant to shocks, corrosion, and outdoor elements. Furthermore, this versatile panel can power low-consumption LED bulbs, bedside radios, and phone chargers.[17]

Parameters:

- **Model Number**: STP085-12/Bb
- **Rated Maximum Power (Pmax)**: 80W
- **Current at Pmax (Imp)**: 4.65A
- **Voltage at Pmax (Vmp)**: 17.2V
- **Short-Circuit Current (Isc)**: 5.1A
- **Open-Circuit Voltage (Voc)**: 21.6V
- **Nominal Operating Cell Temp. (NOCT)**: 50±2°C
- **Weight**: 8kg
- **Dimensions**: 1195×541×30(mm)
- **Maximum System Voltage**: 715V
- **Maximum Series Fuse Rating**: 8A .

CHAPTER 3: practical applications

2.1. Real-world characterization

Materials:

1. Solar Panel (STP085-12/Bb) ;
2. Variable Resistor (0-33Ω) ;
3. Voltmeter ;
4. Ammpermeter ;
5. Pyranometer: Measures solar irradiance ;
6. Environmental Multimeter: temperatur ;
7. Cables.

-We took our measurements in middle of the days with average 37°C for temperature and 300W/m^2 for solar irradiance and all those conditions (shading)figure 30 to 37.

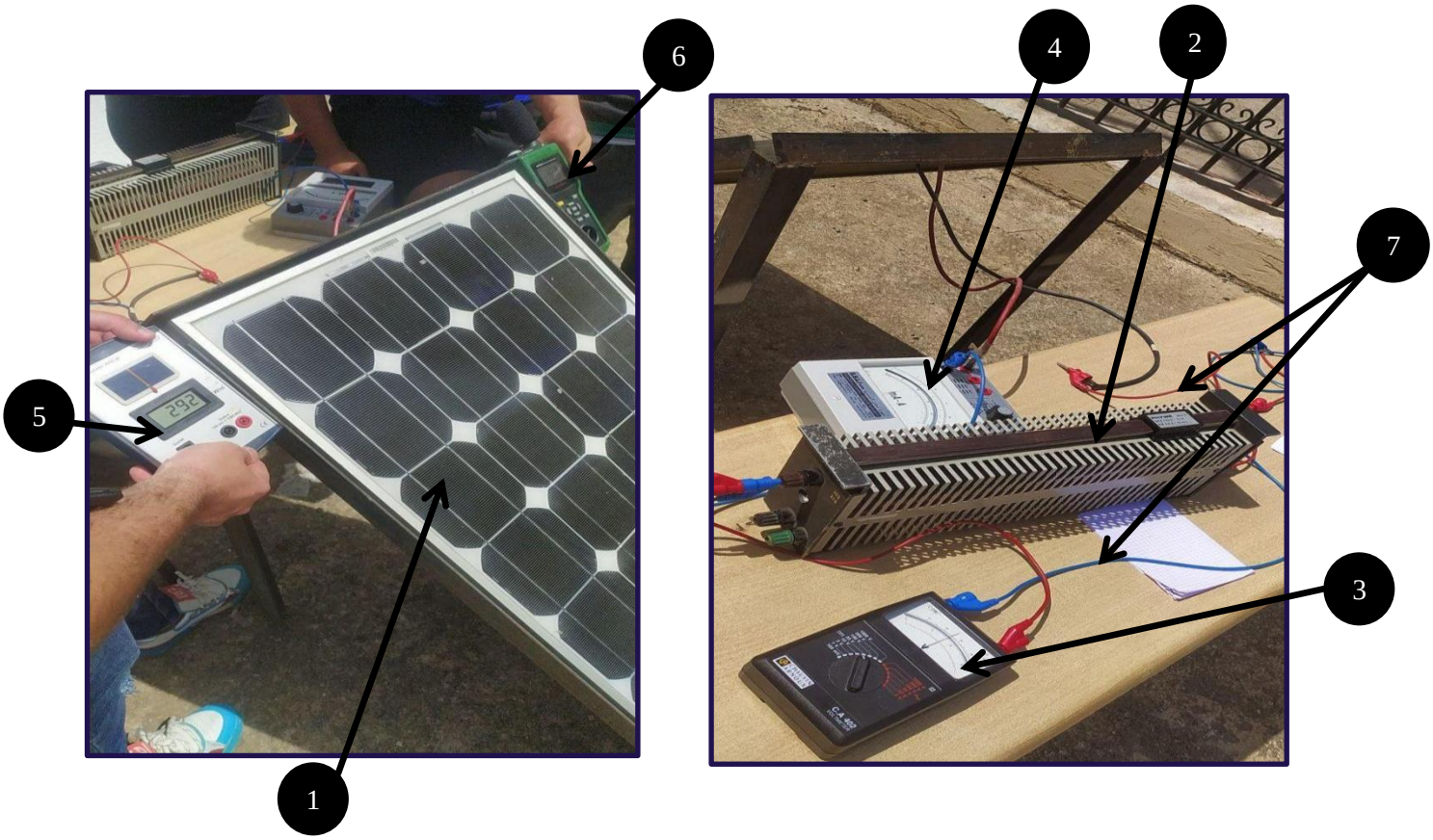


Figure 29:Materials used for the experiences

Figure 30:Clean panel, shading 34%

Figure 31:Clean panel, shading 0%

Figure 32:Clean panel, shading 25%

Figure 33:Clean panel, shading 68%

CHAPTER 3: practical applications

Figure 35 :Tree leaf

Figure 34: Water 100%

Figure 36: Sand

Figure 37:More sand

From the measurements we took we got those graphics :

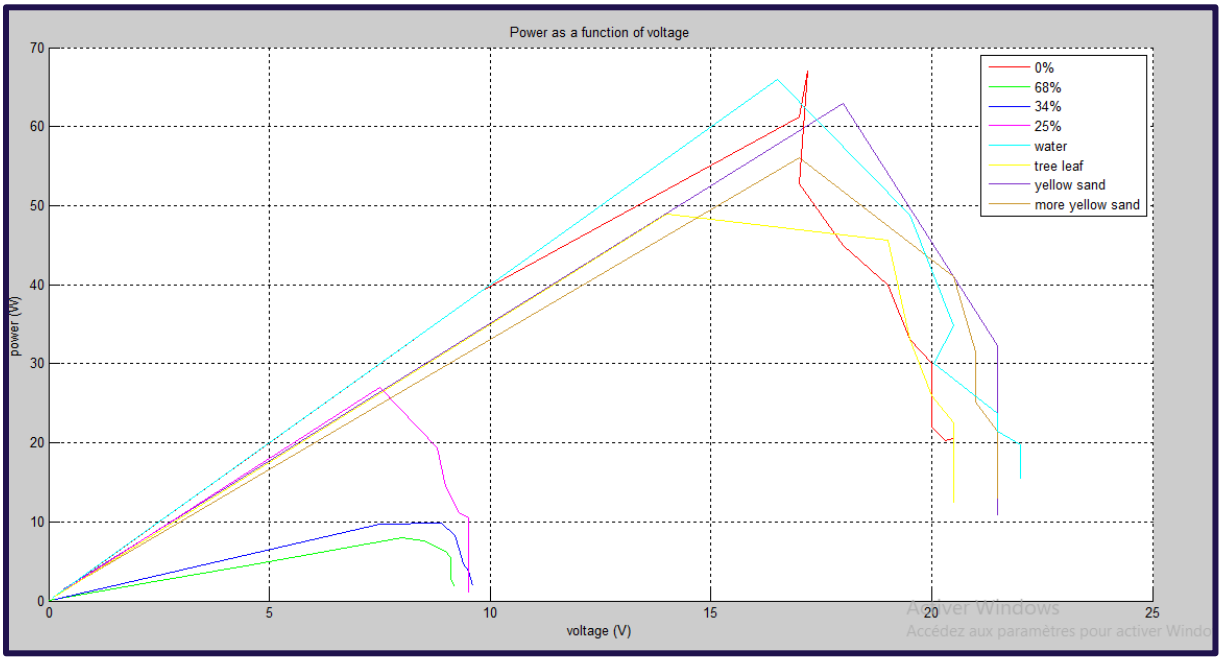


Figure 36: P-V curve of photovoltaic panel under varying conditions

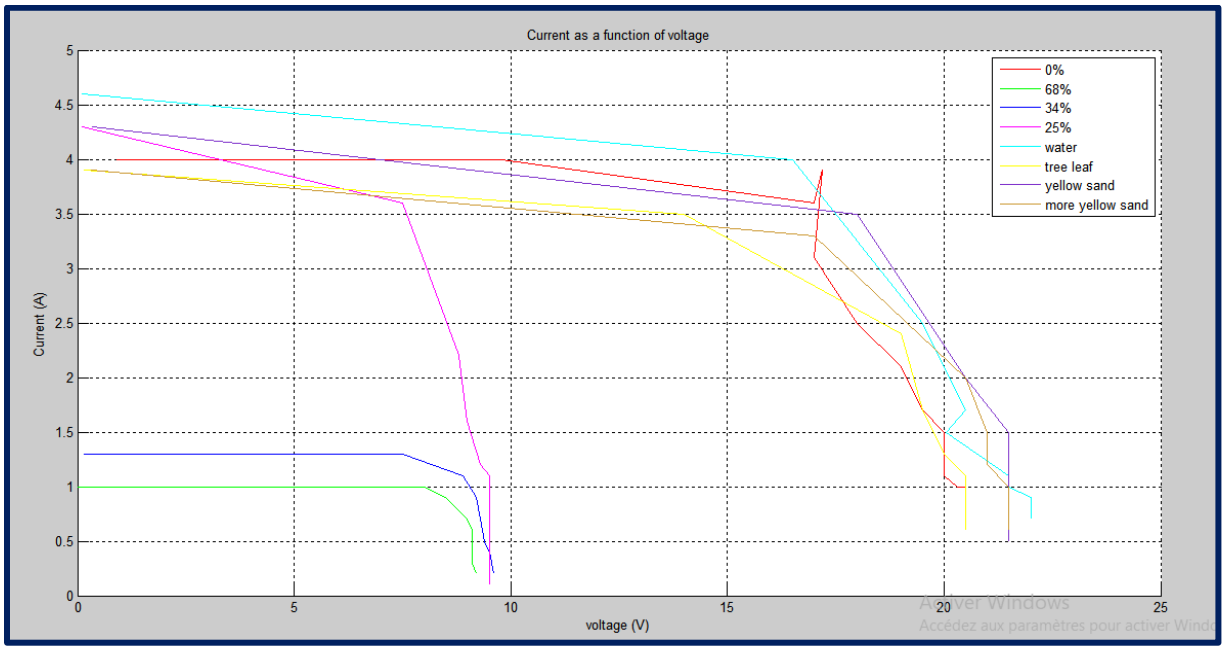


Figure 37: I-V curve of photovoltaic panel under varying conditions

Discussion & results:

Graph Analysis:

CHAPTER 3: practical applications

The behavior of the solar panel, as observed in the P-V (power-voltage) and I-V (current-voltage) curves, closely aligns with theoretical expectations.

This alignment suggests that the panel is performing as expected based on established principles.

Measurement Logic:

However, the water condition seems to be more efficient than the 0% condition (presumably no water). This implies that water has a positive impact on the panel's performance.

Based on the data, we can conclude that water helps keep the surface of the solar panel cooler (lower temperature).

As a result, it minimizes heat loss, contributing to better overall efficiency.

3. Hardware

2.1.Voltage sensor:

As a voltage sensor we used just a voltage divider but in this case as we have our PV panel ranging between 0V and 17V we can juste choose our resistances to convert the output voltage of the pv panel to a proportional signal of 0V to 1V

equation :

$$Vout=\frac{1k\Omega}{1k\Omega+16k\Omega}\,Vin$$

$$Vout=\frac{1}{17}\,Vin$$

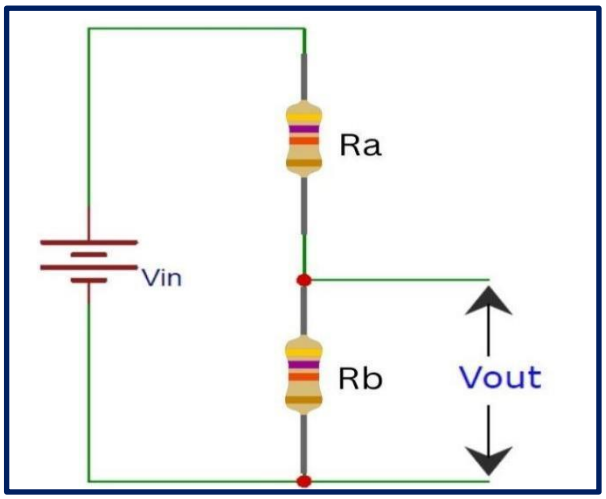


Figure 40:Voltage divider

CHAPTER 3: practical applications

2.2.Current sensor:

The ACS712 shown in figure 41 is a versatile current sensor that can measure both AC (alternating current) and DC (direct current).

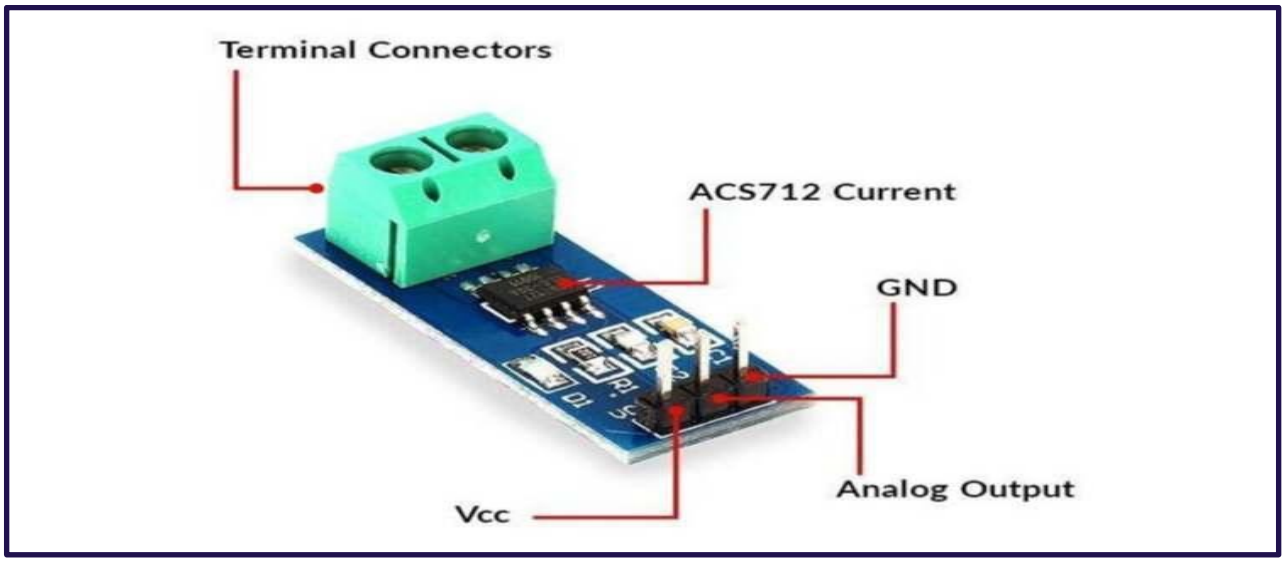


Figure 41: ACS712 PINS[18]

Working Principle:

The ACS712 employs the Hall effect to detect current(figure42). When current flows through its onboard Hall sensor circuit, it generates a magnetic field. The Hall effect sensor senses this magnetic field and produces a voltage proportional to the current passing through the sensor. This voltage output directly corresponds to the magnitude of the current. In essence, the ACS712 provides an accurate and reliable means of measuring current in variousapplications.(figure42)[18]

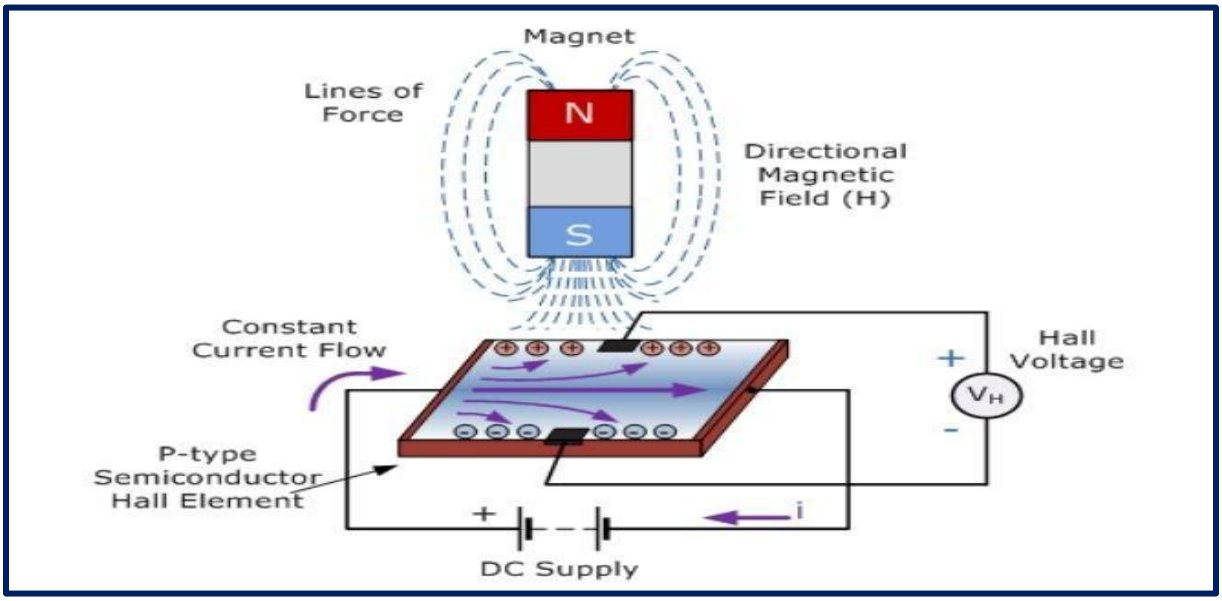


Figure 42:ACS712 Hull effect[18]

Note:at 0A the sunsor give us 2.5V at the outputs and 5V at 5A
(0A-----2.5V 5A-------5V)

As we have the ACS712 sensor designed for Arduino and the analog input of the Arduino's ADC, it means that the analog output of our sensor provides a signal ranging between 0V and 5V. To adapt this signal to the 0-1V range required by the XADC of the Nexys 4 DDR board, we employed a voltage divider. Additionally, even the VCC pin must receive external 5V power.

equation :

Vout=$\frac{100k\Omega}{400k\Omega+100k\Omega}$ Vin

Vout=$\frac{1}{5}$ Vin

2.3.Boost DC/DC converter:

2.3.1.Calculations:

EX: in our case

Let's take the duty cycle **D=0.5** and the **F**requency = **10Khz.**

Vo =$\frac{17.2}{0.5} = \mathbf{34.4V}$

=>$\Delta V = 17.2V$

Io = (1-D)*Is

Io=(1*0.5)4.65=2.325A =>ΔI=2.325A

Calculate L :

from (9):

L=$\frac{Vs(Vo-Vs)}{f*\Delta I*V0}$ =$\frac{17.2(34.4-17.2)}{10000*2.325*34.4}$

L=**0.369mH**

Calculate C

from (12):

C=$\frac{Io(Vo-Vs)}{Vo*f*\Delta V}$=$\frac{4.65(17.2)}{34.4*10000*17.2}$

C =**13.5μf**

CHAPTER 3: practical applications

3.3.2.Proteus simulation

With 50% duty cycle

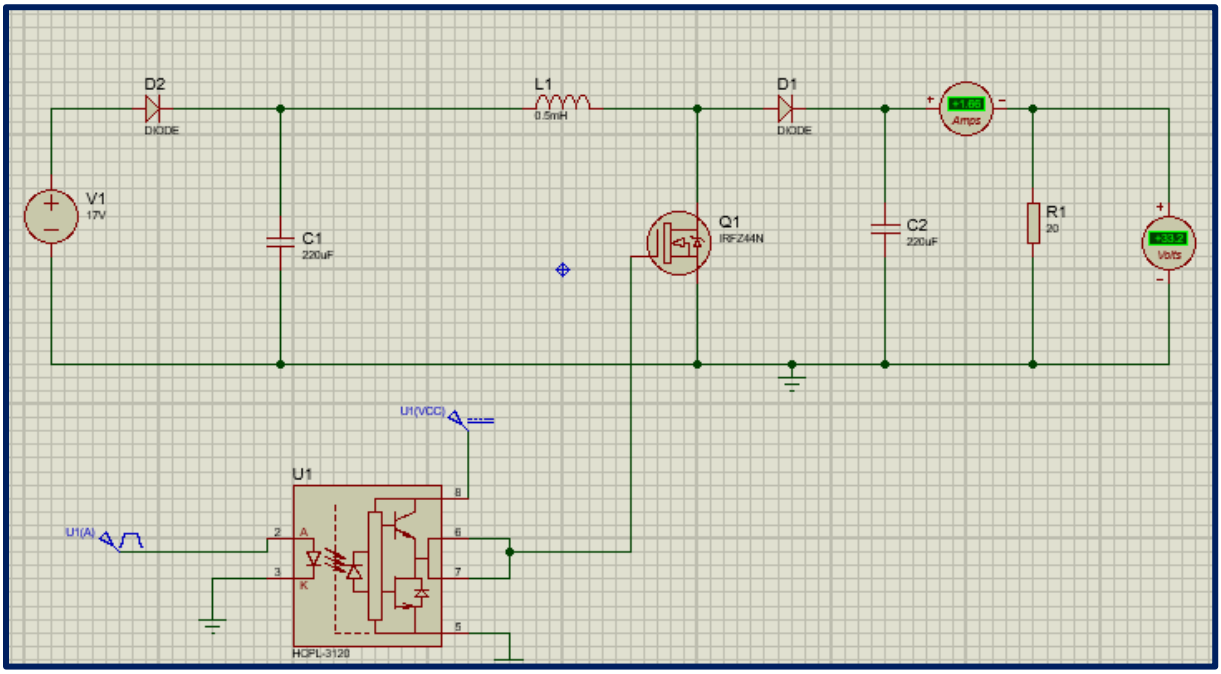


Figure 43: Proteus simulation of boost converter

CHAPTER 3: practical applications

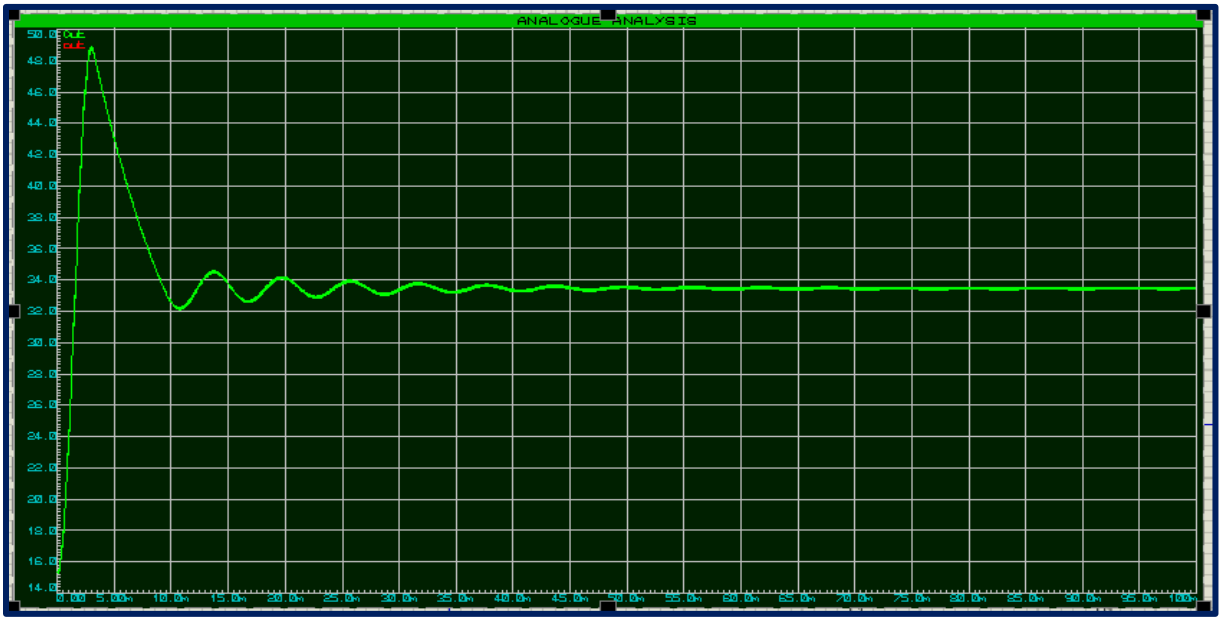


Figure 44: Result of the simulation c=200uF l:0.4mH

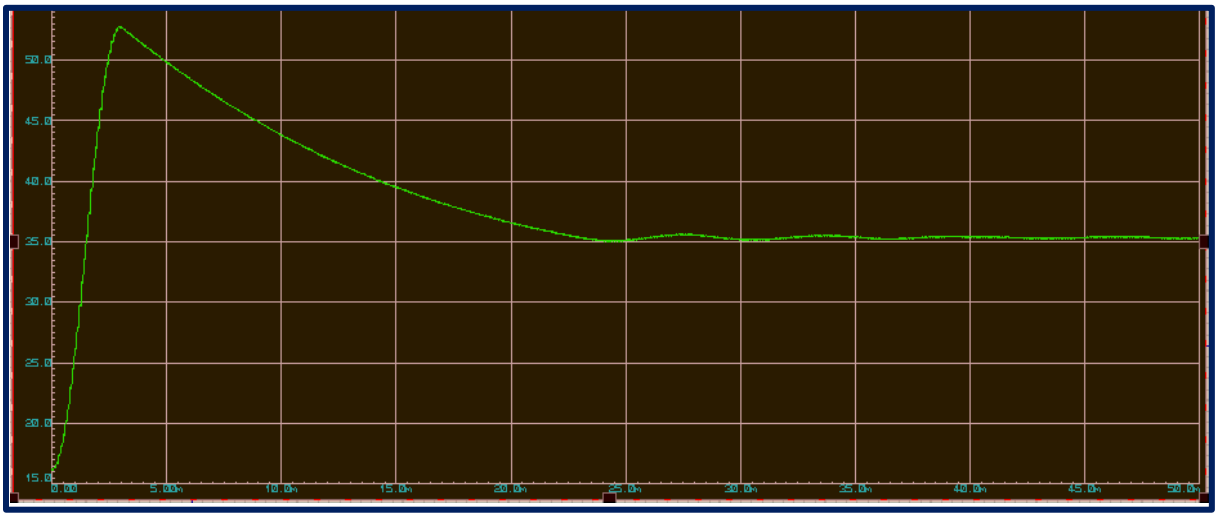


Figure 45: Result of the simulation c:500uF l=0.4mH

2.3.3.Realization and result of the boost

-with 50% duty cycle also as shown in figures 46,47,48

Components:

1-Capacitors

2-Inductor

3- diode(1N5408)

4- resistor

5-mosfet (IRFZ44N)

6-optocopler (HCPL-3120)

CHAPTER 3: practical applications

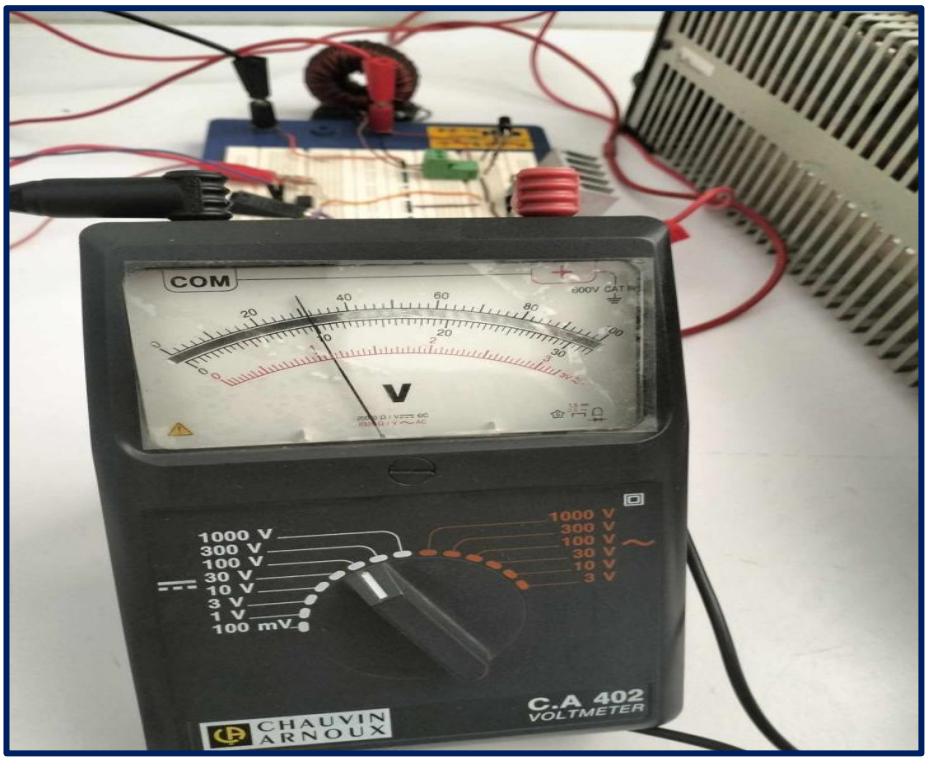


Figure 48: Vout at the load of the boost

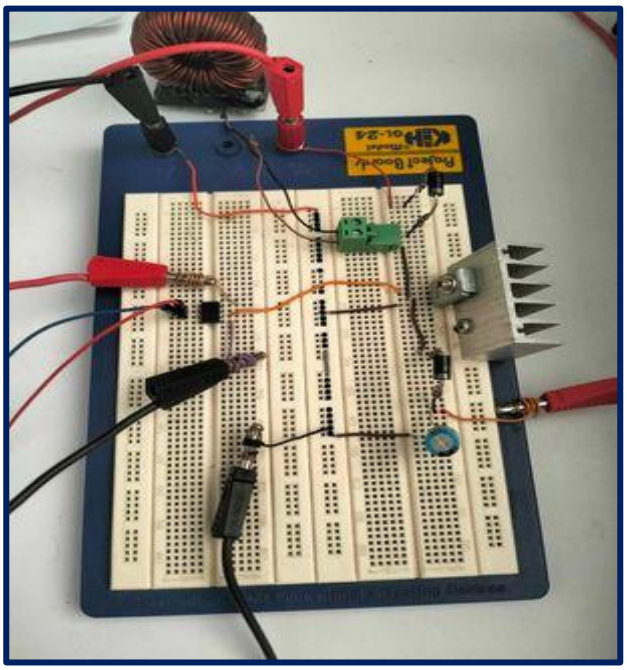


Figure 46: Realization of the boost

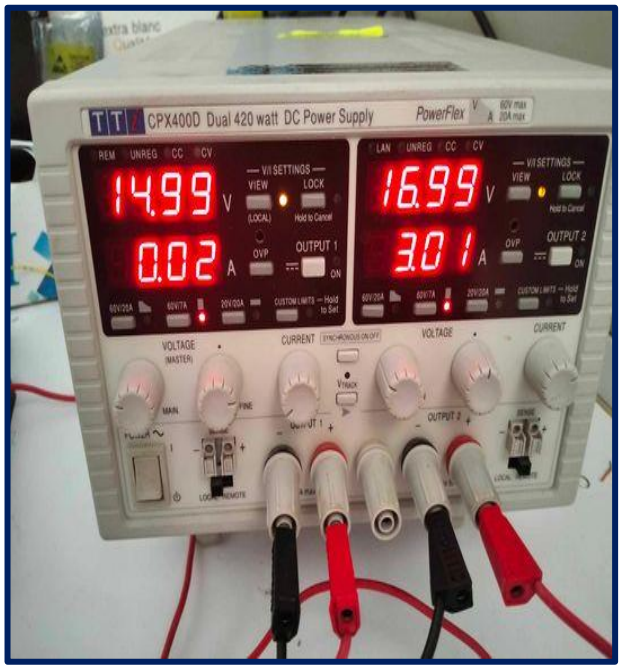


Figure 47: VCC of the optocopler and vin of the boost

Discuss & results:

-as we see here according to figures 43 to 48:

CHAPTER 3: practical applications

Note: Vin=17V Iin=3A

- Theoric Vout =34V Iout=1.5A
- the proteus simulation Iout=1.6 A Vout=32.6V , Vout is lower due to the diode losses
- and real realization Vout=31V losses ar due , Vout is lower due to the diodes and wires losses (pratical losses)

-Satisfactory results.

-Also We notice that as we approach the theoretically calculated values of capacitance and inductance, the boost becomes faster, with slightly less overshoot.

3.Software:

3.1.Summary:

1.Current and Voltage Sensors:

The current_out and voltage_out are the outputs of the current and voltage sensors, These sensor outputs are connected to the MPPT controller.

2.P_and_O:

The P_and_O uses the outputs of the current and voltage sensors data (inputs) and (P&O) algorithm to track the maximum power point of the PV system,The mppt_voltage is the output of the P_and_O, and it goes to the PWM input .

3.Clock divider :

clk (100Mhz) signal is the clock signal delivered by our boards (nexys 4 DDR) and our code clk_div divided our clk to 2550000hz**(sclk)** to separate clocks at first and to have our pwm signal at 10000hz cause PWM generator devide the **sclk** with it counter .

This sclk signal is used in the PWM generator component.

4.PWM Generator:

The mppt_voltage output is connected to our PWM generator, the PWM generator generates PWM signals based on the MPPT voltage provided by the previous code, then we use it to control our mosfet gate .

CHAPTER 3: practical applications

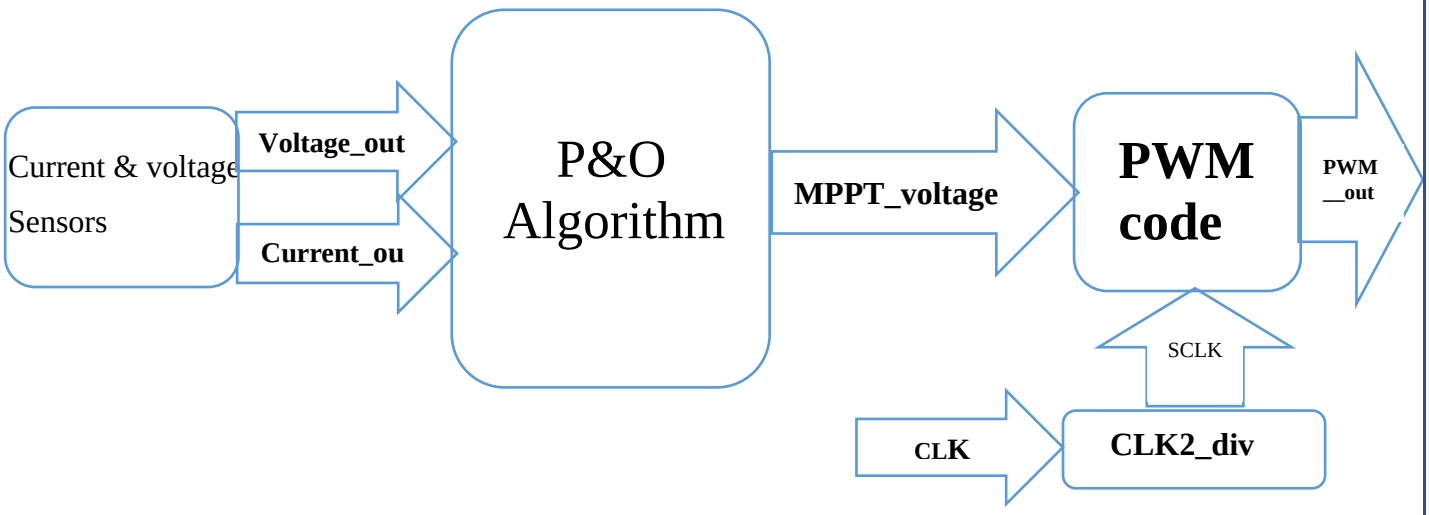


Figure 49: Diagram representing ours codes

3.2.Current sensor :

Code1:

```
ibrut <= conv_integer(unsigned(data1)) * 1000;
voltageadc <= ((ibrut * 5000) / 4094);
current_out <= ((voltageadc - 2400000) * 5000) / 2500000;
```

1-Here we have data1 as an outputs of our xadc which it mean that data1 is a std_logic_vector of 12 bits we convert it to integer (range between 0 to 4094).

2-To have the voltage that the sensor should provide (voltageadc) we use a simple rule of three.

ibrut → 4094

Voltageadc → 5V

3- As we Know from the datasheet the acs712 provide 2.5V (VCC/2) on his outputs when he measure 0A ,that's why we subtract 2.5V to Voltage that the sensor provide(voltageadc) and then we use our second rule of three to have the real current in Amperes .

Deltavoltageadc→ 2.5V

Current_out → 5A

in the codes sometimes we multiply by 1000 and sometimes we divide just to not deal with the floats numbers and have our results in mV and mA.

3.3.Voltage sensor :

Code2:

```
vbrut <= conv_integer(unsigned(data2)) * 1000;
voltage_out <= (vbrut * 17000) / 4094000;
```

Here as we said in the hardware part we just using a voltage divider to have our real voltage we just use a rule of three.

Vbrut → 4095

voltage_out → 17V

In the codes sometimes we multiply by 1000 and sometimes we divide just to not deal with the floats numbers cause of VHDL and vivado and have our results in mV and mA.

3.4.P&O:

In this code we have current_out (input for solar panel current), voltage_out (input for solar panel voltage), and mppt_voltage (output for adjusted voltage). Inside the architecture, the process calculates current power, monitors power and voltage variations, and adjusts the output voltage based on power changes. The goal is to maximize power extraction from the solar panel by dynamically adjusting the output voltage.

3.5.Xadc:

Code3:

```
entity xadc_wiz_0 is
   port
   (
    daddr_in        : in  STD_LOGIC_VECTOR (6 downto 0);     -- Address bus for the dynamic reconfiguration port
    den_in          : in  STD_LOGIC;                         -- Enable Signal for the dynamic reconfiguration port
    di_in           : in  STD_LOGIC_VECTOR (15 downto 0);    -- Input data bus for the dynamic reconfiguration port
    dwe_in          : in  STD_LOGIC;                         -- Write Enable for the dynamic reconfiguration port
    do_out          : out  STD_LOGIC_VECTOR (15 downto 0);   -- Output data bus for dynamic reconfiguration port
    drdy_out        : out  STD_LOGIC;                        -- Data ready signal for the dynamic reconfiguration port
    dclk_in         : in  STD_LOGIC;                         -- Clock input for the dynamic reconfiguration port
    reset_in        : in  STD_LOGIC;                         -- Reset signal for the System Monitor control logic
    vauxp2          : in  STD_LOGIC;                         -- Auxiliary Channel 2
    vauxn2          : in  STD_LOGIC;
    vauxp10          : in  STD_LOGIC;                         -- Auxiliary Channel 10
    vauxn10          : in  STD_LOGIC;
    busy_out        : out  STD_LOGIC;                        -- ADC Busy signal
    channel_out     : out  STD_LOGIC_VECTOR (4 downto 0);    -- Channel Selection Outputs
    eoc_out         : out  STD_LOGIC;                        -- End of Conversion Signal
    eos_out         : out  STD_LOGIC;                        -- End of Sequence Signal
    alarm_out       : out STD_LOGIC;                         -- OR'ed output of all the Alarms
    vp_in           : in  STD_LOGIC;                         -- Dedicated Analog Input Pair
    vn_in           : in  STD_LOGIC
);
end xadc_wiz_0;
```

CHAPTER 3: practical applications

Here we got the code of xadc compenent generated by vivado software**(code 3)** and all those ports that we can use as we please, as in the following example that we use to read our analogs signals :
As we have **two analog signals** to convert, and the Nexys 4 DDR has only **one ADC block,** we opted to use a small state machine**(code4)**. This state machine reads **port vaux2** for current and **vaux10** for voltage simultaneously, depending on the **daddr, eoc, drdy**, and **den** signals of the XADC port

Code4:

```
process(clk)
begin
    if clk'event and clk='1' then
        case state is
        when idle =>  -- waiting for eoc
            den <= '0';
            daddr <= "0010010"; -- Vaux2
            if eoc = '1' then
                state <= read1; -- read the first data
                den <= '1';
            end if;
        when read1 =>
            den <= '0';
            if drdy = '1' then
                data1 <= data;
                daddr <= "0100000"; -- vaux10
                state <= tempo;
            end if;
            if eoc = '1' then  -- normally it never happens
                state <= idle; -- but better to put it
            end if;
        when tempo =>
            den <= '1';
            state <= read2;
        when read2 =>
            if drdy = '1' then
                data2 <= data;
                state <= idle;
            end if;
        end case;
    end if;
```

3.6. PWM :

In this code we initialize a counter to 255, convert our MPPT voltage signal that represent the duty cycle to an integer in the range 0-255, and increment the counter on each rising edge. Then, we compare if the MPPT voltage is greater than the counter. If it is, we set our PWM output to a high state (1); otherwise, it remains at a low state (0) .

CHAPTER 3: practical applications

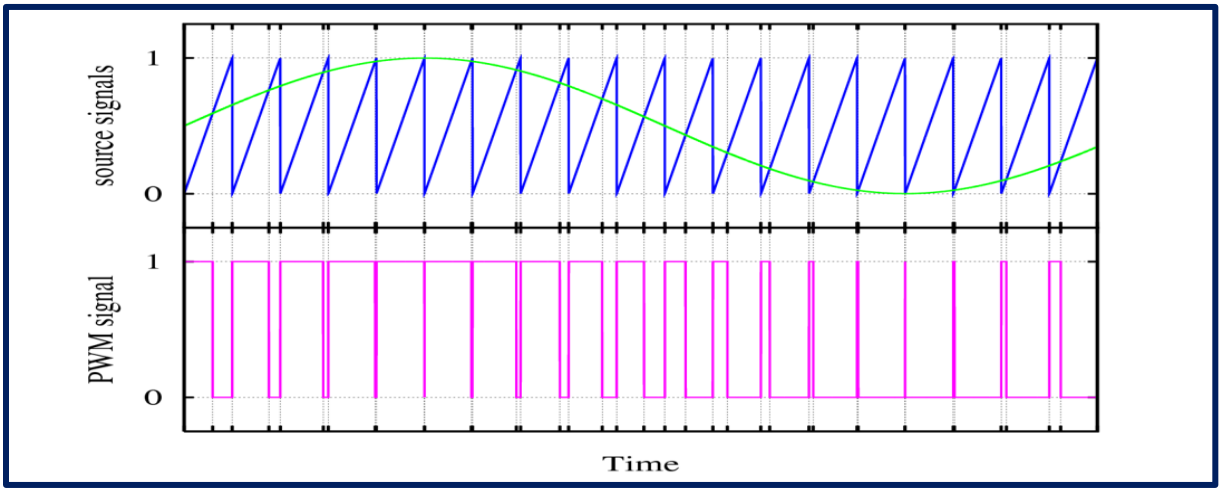
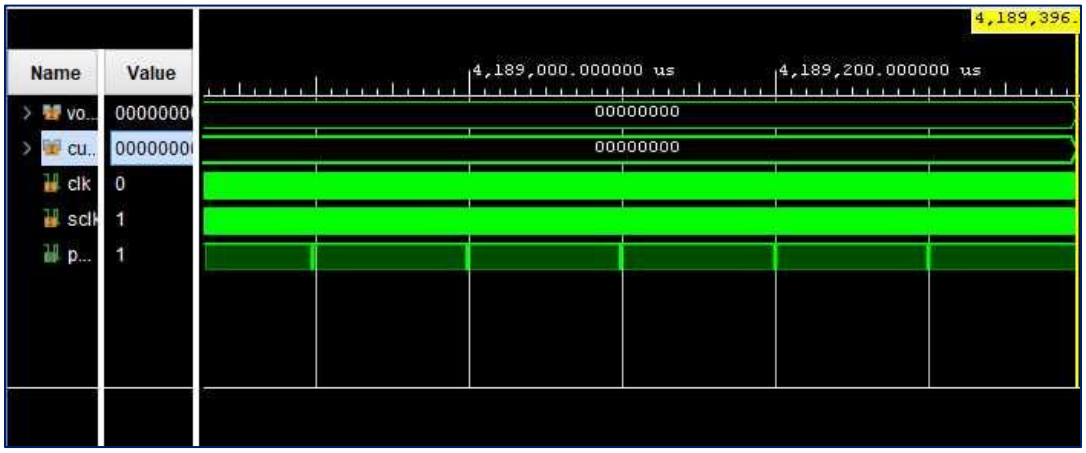
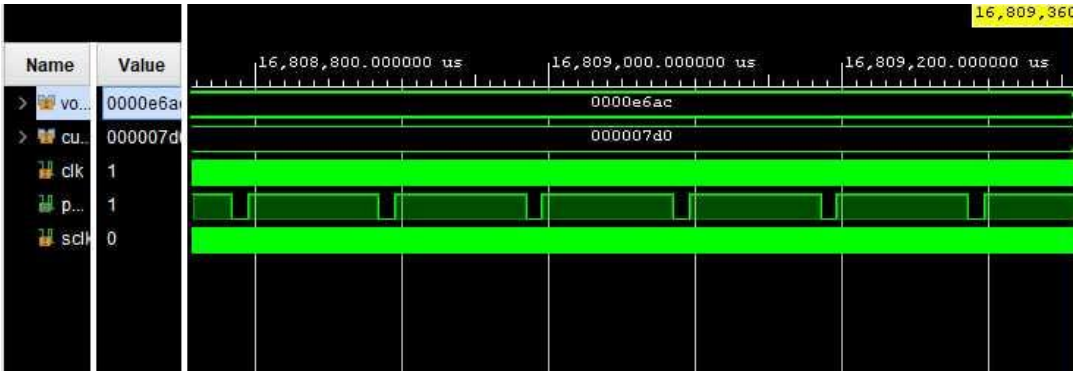


Figure 50: How Pulse Width Modulation (PWM) signals been generated

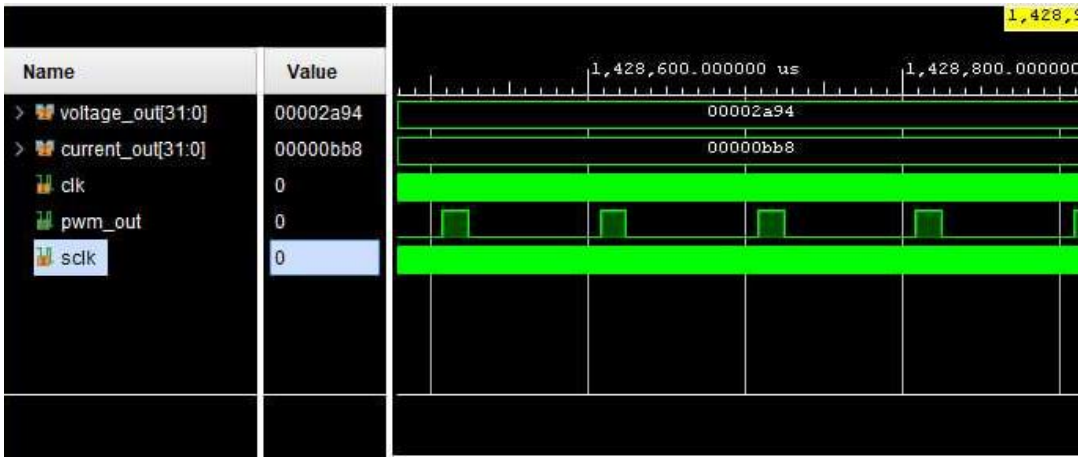
Here are some results from the practical implementation of a PWM signal in our MPPT controller(figure 51). We conducted tests using Vivado testbench .



a-PMW when MPPT voltage decrease sudently



b-PWM when MPPT voltage decrease



c-PWM When MPPT voltage increase

Figure51: Differents states of the PWM

Discussion of the results :

We observe that our system behaves as predicted by theory (Figure 7) when specific variations of delta p and delta v are applied to it by increasing or decreasing the MPPT voltage, which directly affects the low and high states of the PWM.

4.Conclusions of chapter3:

The efficiency of PV systems remains a concern, prompting the necessity for Maximum Power Point Tracking (MPPT) controllers to optimize power extraction. Various MPPT methods exist, ranging from manual adjustments to sophisticated algorithms like the Perturbation and Observation (P&O) method, which we are implementing using Field-Programmable Gate Arrays (FPGAs). FPGAs are versatile programmable circuits essential for applications demanding rapid development and modular design capabilities.

General Conclusion

General conclusion

This project focuses on implementing the MPPT method on an FPGA platform, involving stages such as VHDL programming, synthesis, testing, and simulation. Chapters 1 to 3 of this study provide comprehensive insights into photovoltaic systems, DC-DC converters, FPGA technology, VHDL programming, and the practical implementation of MPPT techniques. These chapters collectively aim to contribute to enhancing the efficiency and effectiveness of solar energy utilization in future energy systems.

In this work, we aimed to explore and fully utilize the resources offered by FPGA circuits by implementing a Perturbation and Observation (P&O) based MPPT method used in photovoltaic systems. We began by introducing theoretical concepts related to photovoltaic energy, followed by an exploration of MPPT techniques. Subsequently, we developed DC-DC converters, which are essential for understanding the remainder of this work.

In the second chapter, our focus shifted to FPGA circuits and the VHDL language.

In the last chapter, we found that shading positions affect panel efficiency to varying degrees, depending on cell connections. Additionally, photovoltaic cells perform better at lower temperatures. Practical tests showed slight voltage reductions due to diode and wiring losses, aligning closely with theoretical predictions. Moreover, adjusting capacitance and inductance values led to faster boost performance with minimal overshoot, confirming theoretical expectations and influencing PWM states

Sources

5. Sources :

[1]:Statista, "Solar PV - statistics & facts", Apr 2024. https://www.statista.com/topics/993/solar-pv/#editorsPicks ,last consulted on 04/07/2024.

[2]:Expert from Portail Algérien des énergies renouvelables, "L'enjeu capital des énergies renouvelables pour palier aux restrictions des débouchées du gaz naturel ", Algérie, 2021. https://portail.cder.dz/2021/02/15/lenjeu-capital-des-energies-renouvelables-pour-palier-aux-restrictions-des-debouchees-du-gaz-naturel/, last consulted on 04/07/2024.

[3]:S. Necaibia M. Kelaiaia, H. Labar, A. Necaibia, O. Djelailia, F. Merad, "Practical Implementation of a Proposed MPPT Control Strategy to Mitigate Inaccurate Responses for Photovoltaic Systems", International Journal on Electrical Engineering and Informatics - Volume 10, Number 4, Desember 2018.

[4]:G.Boyle. Renewable Energy: Power for a Sustainable Future, 2nd ed. Oxford, UK: Oxford University Press, 2004.
[5]:Temperature coefficient of a photovoltaic cell. (n.d.). Alternative Energy Tutorials. https://www.alternative-energy-tutorials.com/photovoltaics/temperature-coefficient.html,last consulted on 09/07/2024.
[6]Labouret, A., & Villoz, M. (n.d.). Energie solaire photovoltaïque.

[7]:Sarang, S.A., Raza, M.A., Panhwar, M. et al. Maximizing solar power generation through conventional and digital MPPT techniques: a comparative analysis. Sci Rep 14, 8944 (2024) https://doi.org/10.1038/s41598-024-59776-z

[8]:Ahteshamul Haque (2014) Maximum Power Point Tracking (MPPT) Scheme for Solar Photovoltaic System, Energy Technology & Policy, 1:1, 115-122, DOI: 10.1080/23317000.2014.979379

[9]:Fevzi Çakmak, Zafer Aydoğmuş & Mehmet Rıda Tür (25 Dec 2023): Analysis of Open CircuitVoltageMPPTMethodwithAnalyticalAnalysiswithPerturbandObserve(P&O)MPPTMethodinPV Systems,ElectricPower Componentsand Systems,DOI:10.1080/15325008.2023.2296958

[10]:M.AlDhaifallah,A.M.Nassef,H.Rezk,andK.S.Nisar,"Optimalparameterdesignoffractionalordercontrolbased inc-mpptfor pvsystem,"Solar Energy,vol.159, pp.650–664, 201

[11]:6Allataifeh, A.A., Bataineh, K., Al-Khedher, M.: 'Maximum power point tracking using fuzzy logic controller under partial conditions', Smart Grid Renew. Energy, 2015,

[12]:H. Rashid, Muhammad (2001). Power Electronics Handbook, DC-DC Converters (p. 211-216). Presse Académique 2001, San Diego, San Francisco, New York, Boston, Londres, Sydney, Tokyo

[13]: Shet, R. (2020, March 5). Programmable Logic Devices – A summary of all types of PLDs. Technobyte. https://technobyte.org/programmable-logic-devices//,last consulted on 14/07/2024.

[14]:NEXYS4 DDRTM FPGA Board Reference Manual Nexys4 DDR rev. C; Revised April 11, 2016

Sources

[15]:Kilts, S. (2007). Advanced FPGA design: Architecture, Implementation, and Optimization. John Wiley & Sons.

[16]:Sánchez-Élez,M.(n.d.).Introducción a la programación en VHDL.

[17] :Panneau Solaire Monocristallin 80W - 12V - Unisun 80.12 M - fabricant français. (n.d.). https://uniteck.fr/fr/panneaux-solaires-photovoltaiques_unisun-80.12-m_0095,last consulted on 13/07/2024.

[18] :Kumar,P.(2024,February24). ACS712 current sensor and arduino. Makerguides.com. https://www.makerguides.com/acs712-current-sensor-and-arduino-a-completeguide/,
last consulted on 14/07/2024.

Abstract:

In our work, we discussed photovoltaics (PV) in chapter 1, FPGA technology in chapter 2, and practical applications related to solar energy. In the PV section, we explored the conversion of sunlight into electricity using solar cells, The FPGA and VHDL segment delves into versatile integrated circuits (FPGAs) and the powerful VHDL language for digital system design. Finally, practical applications involve PV panels, hardware components (current and voltage sensors, DC-DC converters), and software techniques (P&O, PWM) for optimizing solar energy utilization.

Keywords:

Photovoltaic(PV),Maximum Power Point Tracking(MPPT),Perturb and Observe(P&O) , Field-Programmable Gate Array(FPGA),VHSIC Hardware Description Language(VHDL)