**People's Democratic Republic of Algeria**
**Ministry of Higher Education and Scientific Research**

**University M'Hamed BOUGARA – Boumerdes**


**Institute of Electrical and Electronic Engineering**

**Department of Electronics**

Final Year Project Report Presented in Partial Fulfilment of
The Requirements for the Degree of

# MASTER

In **Electrical and Electronic Engineering**

Option: **Computer Engineering**

Title:

# FPGA-Based Car-Like Robot Path Follower with Obstacle Avoidance

Presented by:

- **MOULAY Aicha**
- **LAOUFI Fatima**

Supervisor:

**Dr. A. BENZEKRI**

Registration Number:……/2016

## Dedication

*I dedicate this modest work to my dear father and mother who have provided me with their endless love, support and encouragement , to my lovely sisters and brothers .I would like to thank them for their support and unconditional love during all my studies.*

*I further extend my gratitude to my brothers in law, nephews, nieces and to all members of Moulay's family.*

*I dedicate this project to all my colleagues and friends without exception especially those who share with me the ups and downs through these five years.*

*Last and not least I dedicate this humble work to everyone who has taught, encouraged, and advised me during all my studies.*

*Aicha*

## Dedication

*My great parents, who raised me, loved me, and never stop giving of themselves in countless way.*

*My beloved sisters, who stands by me and give me the power to reach my goal.*

*To all my family, the symbol of love and giving, my dearest friends who support and encourage me.*

*All the people of my life who touch my heart, I dedicate this work.*

*Fatima*

## Acknowledgement

All praise and thanksgiving to Allah the most powerful and most merciful who give us the ability and patience to accomplish this humble work.

We would like to express our gratitude to our supervisor Dr. A. BENZEKRI for his guidance and support during our work. We are grateful to our teachers, academic staff and workers at the Institute of Electrical and Electronic Engineering who prepare for us the environment to work and offer to us their valuable help.

Special thanks to all people who supported and guided us in any respect during the completion of the project especially our colleagues.

# Abstract

This project is about the design and implementation of an SOPC based automated path following with obstacle avoidance using infrared sensors and ultrasonic sensor.

The Line follower robot is a mobile machine that can detect and follow the line drawn on the floor. The path is predefined as a black line on a white surface with a high contrasted color. This kind of robot should sense the path using infrared sensors which installed under the robot. After that the data is transmitted to the controller to be processed by specific transition buses. The processor is going to send proper commands to the driver unit circuit and thus the path will be followed by the line follower robot.

This system is provided with obstacle avoidance mechanism using ultrasonic sensor. After avoiding the obstacle either from the right side or the left side the car then has to find its path again to reach its destination with the continuity of following and scanning the path. The car will stop if an obstacle is detected on the path and on both sides of it or if the path is ended.

The digital controller is designed and implemented using both off-the-shelf integrated circuits and the FPGA. The FPGA subsystem is developed using Altera Quartus II development suite tools and the Altera monitor program software. It is realized on a Cyclone-II EP2C35F672C6 low-cost FPGA platform and tested on a physical structure constructed in the laboratory for its feasibility and functionality.

# Table of content

## Chapter V Conclusion

# List of Nomenclatures

AC: Alternative Circuit.

ADC: Analog to Digital Converter.

ALU: Arithmetic Logic Unit.

ASIC: Application Specific Integrated Circuits.

BDF: Block Diagram File.

CAD: Computer-Aided Drafting.

CPU: Central Processing Unit

CMOS: Complementary Metal Oxide Semiconductor.

DE2 board: Development and Education board.

DC: Direct Current.

DAC: Digital to Analog Converter.

FPGA: Field Programmable Gate Array.

GUI: Graphical User Interface.

HDL: Hardware Description Language.

IC: Integrated Circuit.

IDE: Integrated Development Environment.

IR: InfraRed.

I/O: Input Output.

JTAG: Joint Test Action Group.

LED: Light Emitting Diode.

LCD: Liquid Crystal Display.

LPM: Library of Parallel Modules.

MAC: Media Access Control.

NTSC: National Television System Committee.

PAL: Phase Alternative Line.

PCI: Peripheral Component Interconnect.

PWM: Pulse Width Modulation.

RAM: Random Access Memory.

RISC: Reduced Instruction Set Computer.

RUR: Rossum's Universal Robots.

SDRAM: Static Dynamic Random Access Memory.

SRAM: Static Random Access Memory.

SMA: Sub Miniature version A.

SoPC: System On Programmable Chip.

TTL: Transistor-Transistor Logic.

UART: Universal Asynchronous Receiver Transmitter.

USB: Universal Serial Blaster.

VGA: Video Graphics Array

VHDL: Very high speed integrated circuit Hardware Description Language.

# List of Figures

# List of Tables

# Chapter I: Introduction

## 1. Introduction

This chapter presents an overview, as well as the motivation and goals of our project. Moreover, it describes the application and the organization of the report.

## 2. Overview

The term « robot » was first introduced in 1920 by the Czech writer Karel Capek in his theatre play RUR (Rossum's Universel Robots). This term came from the Czech word robot, « forced labour», designated originally to an android machine capable of replacing the human in all his tasks. Between the years 1940's and 1950's, the progress of electronics allowed the possibility to miniaturize electronic circuits (invention of transistor and integrated circuit), paved the way for the manufacture of robots. In the earliest times of robotics, the robot was considered as an imitation of the human being functionality and his physical. Nowadays, the manufacturers do not try to duplicate the human appearance on a robot, favoring its functionality before all else [1].

At present, the robots are widely used in industry, particularly in automobile construction and by most of computer manufacturers. Capable of performing repetitive tasks with high speeds, they are notably used in assembly and production line. We employed them as well in a difficult environment for human to support (extreme conditions of temperature or pressure, high radioactivity, etc…). The nuclear industry has also contributed largely in development of robotics (especially in the design of remote control robotic arm) [1].

Line follower is a machine that can follow a path. The path can be visible like a black line on a white surface (or vice-versa) or it can be invisible like a magnetic field.

Line followers can be used to deliver mail within an office building and deliver medications in a hospital. The technology has been suggested for running buses and other mass transit systems, and may end up as part of autonomous cars navigating the freeway. The line follower can be used in guidance system for industrial robots moving on shop floor. An example might be in a warehouse where the robots follow 'track' to and from the shelves they stock and retrieve from. A line follower robot can be used in military as spy kids or in many other applications [2].

Obstacle avoidance is the back bone of autonomous control as it makes robot able to reach to destination without collision, robot movement would be very restrictive and fragile without it. There are several ways and algorithms to accomplish the task of obstacle avoidance within the home environment. The best technique will depend on your specific environment and what equipment you have available.

## 3. Motivation

Robotics is a part of today's communication .This later is a part of advancement of technology. Hence robotics is fast growing and interesting field [3]. So we decided to work on robotics field and design something which will make the life easier to human in today's aspect.

Path following and obstacle avoidance are two very important behaviors that must be considered in the process of Autonomous Mobile Robots development. There have been great amount of research concerning following path and obstacle avoidance projects separately. Moreover, the obstacle avoidance control includes also the obstacle detection. Our objective is to gather the two by designing and implementing a controller for a following path with obstacle avoidance car.

## 4. Project Objectives

The purpose of this project is to design and implement an FPGA-Based autonomous Following path with Obstacle avoidance car. The objectives to be achieved are:

1. Search for the black path using infrared sensors.
2. Adjust the car to follow this path using these sensors.
3. Stop the car if an obstacle is detected on the path by the ultrasonic sensor and move the car to the right to avoid it if no other obstacle is on this side.
4. Move the car to the left if an obstacle is detected on the right and no constraint is on the left side.
5. Search again for the predefined path after avoiding the obstacle.
6. Stop the car if there is no black path or if obstacles are detected on the path and on both sides.

There are several ways to achieve this goal. Today's engineers are blessed with a large number of alternatives as a result to the continuous evolution of micro and nano-technologies. Three main methodologies are possible:

- Hardware (application specific integrated circuits, ASICs).
- Software-programmed processors (general-purpose microprocessors, microcontrollers)
- Field Programmable Gate Array.

Although the first method has a very high performance and efficiency, it is very costly and has less flexibility [4].

A microprocessor based system does not have the necessary circuitry on one chip, external devices and signals are required such as memory, Input /Output buffers control signals for timing to accomplish its work. Microcontrollers are provided with extra devices and signals to overtake the constraints of microprocessors but it still time-limited and have their own permanent circuitry and instruction set that the programmer must follow in order to write codes. However, FPGA is an integrated circuit that contains millions of logic gates, and can be electrically configured to perform a certain task. **Figure 1.1** shows an FPGA chip.


**Figure 1.1** FPGA chip

## 5. Structure of the Autonomous following path with obstacle avoidance car

Due to the flexibility developing of the FPGA, we choose to implement our digital controller by integrating a heterogeneous computing platform made up of microprocessor based on the SOPC approach. The system is designed around the NIOS II/e soft-core processor instantiated in the Cyclone II EP2C35F672C6 FPGA system and some custom logic, all on a single chip. Our system consists of two main parts, hardware and software. The

software consists of the algorithms and the code executed by the microprocessor to control the hardware part. **Figure 1.2** describes the hardware used to control the car. The system is detailed as following:

- **The On-chip Hardware Unit:** contains The FPGA Controller which controls the off-chip hardware circuitry via the board's expansion header .The PC running Altera monitor program and Quartus II Software development suite tools is connected to the FPGA by means of the USB blaster mechanism.
- **The off-Chip Hardware Unit:** The off-chip hardware circuitry is built-on the proto-board consists of :
  - **Sensing Unit:** It contains the ultrasonic sensor and data acquisition. This latter is responsible for the data acquisition of the analog signals from the infrared sensors and their conversion into digital format by an ADC.
  - **Interface Block:** It performs the translation between the FPGA and the hardware components to make them compatible with each other.
  - **Motor Drivers Unit:** This unit is composed of H-bridge circuit that is controlled by the FPGA unit. It sends the appropriate signals to two DC motors.
  - **PC Unit:** It runs the Quartus II 9.1 sp2 development tool and Altera Monitor Program.

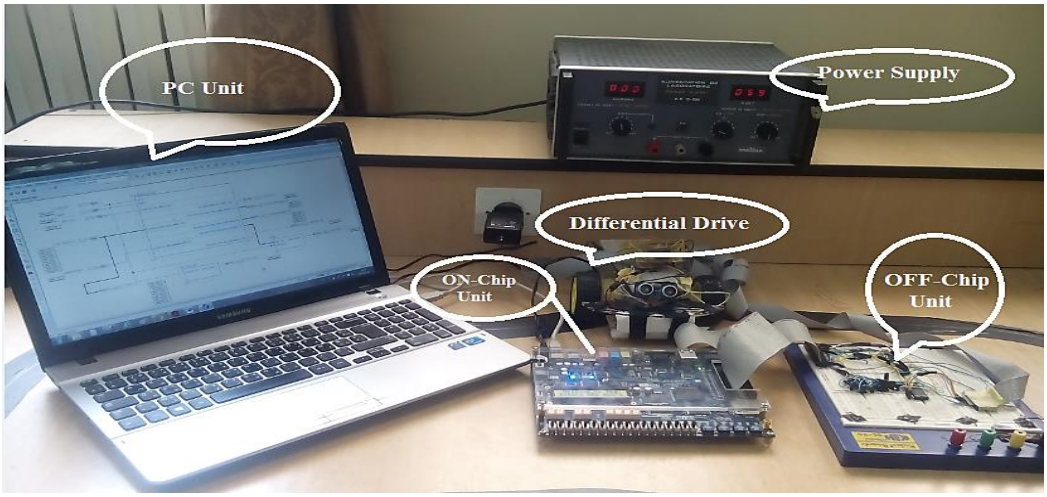


**Figure 1.2** Hardware of the FPGA-Based autonomous path follower and obstacle avoidance car.

## 6. Organization of the report

The remaining of the report is organized as follows: Chapter 2 introduces the theoretical background and describes the hardware materials used in our project. Chapter 3 constitutes the main part of the report; it describes the design and implementation of both On-chip and Off-chip subsystems. Chapter 4 presents in details the algorithms and flowcharts used to configure the FPGA. Chapter 5 summarizes the work presented in this report, mention some remarks about the difficulties faced and discuss the possible directions for further enhancement of the system. Finally, the report terminates with an extensive list of references.

# Chapter II: Theoretical Background

# 1. Introduction

This chapter includes a detailed theoretical analysis on the different components and approach used in the project.

# 2. The Field Programmable Gate Array

The Field Programmable Gate Array, FPGA for short, is a digital integrated circuit designed to be electrically configured by the customer after manufacturing to become almost any customized digital system [5].

The "field programmable" portion of the FPGAs name refers to the fact that its programming takes place "in the field" (as opposed to devices whose internal functionality is hard wired by the manufacturer). This may mean that FPGAs are configured in the laboratory, or it may refer to modifying the function of a device resident in an electronic system that has already been deployed in the outside world. If a device is capable of being programmable while remaining resident in a higher-level [6].

## 2.1. FPGA architecture

Each FPGA vendor has its own FPGA architecture, but in general terms they are all a variation of that shown in **Figure 2.1**. The architecture consists of configurable logic blocks, configurable I/O blocks, and programmable interconnect. Also, there will be clock circuitry for driving the clock signals to each logic block, and additional logic resources such as ALUs, memory and decoders may be available. The two basic types of programmable elements for an FPGA are Static RAM and anti-fuses [6].

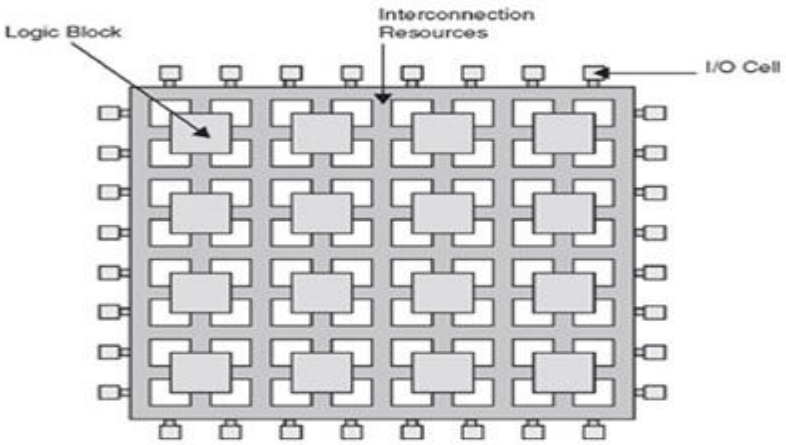


**Figure 2.1** General structure of FPGA [7]

## 2.2. Application of FPGAs

Configurable hardware is an approach for realizing optimal performance by tailoring its architecture to the characteristics of a given problem. FPGAs implementation of some designs is very attractive because of the high flexibility that can achieved through the reprogram ability nature of these circuits [6].

Their advantages lies in that they are sometimes significantly faster for some applications due to their parallel nature and optimality in terms of the number of gates used for a certain process. Specific applications of FPGAs include digital signal processing, software-defined radio, ASIC prototyping, medical imaging, computer vision, speech recognition, cryptography, bioinformatics, computer hardware emulation, radio astronomy, metal detection, and a growing range of other areas [8].

## 2.3. The DE2 Development Board

Development and Education board is an ideal platform designed for university laboratories to perform wide range of tasks on digital logic, computer organizations and FPGAs. These tasks might be simple or advanced design. The power of the board is such that it is also highly suitable for a variety of design projects as well as for the development of sophisticated digital systems [9]. **Figure 2.2** represents the DE2 board.

The DE2 box includes:

- The 8*6 inch with a Cyclone II EP2C35 (672-pin package) FPGA.

- 9V AC/DC adaptor.
- USB cable.
- Plexiglas cover for the DE2 board.
- Installation guide.

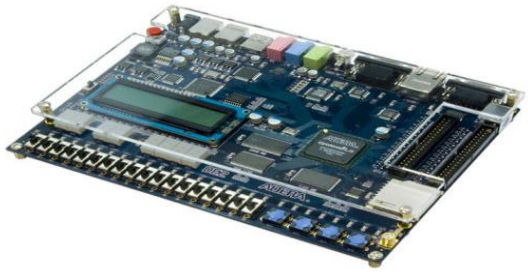

**Figure 2.2** DE2 Development and Education Board

The table below shows the features of DE2 board.

**Table 2.1** DE2 board information

| Feature | Description |
|---|---|
| **FPGA** | Cyclone II EP2C35F672C6 with EPCS16 16-Mbit serial configuration device |
| **I/O Interfaces** | • Built-in USB-Blaster for FPGA configuration<br>• Line In/Out, Microphone In (24-bit Audio CODEC)<br>• Video Out (VGA 10-bit DAC)<br>• Video In (NTSC/PAL/Multi-format)<br>• RS232<br>• Infrared port<br>• PS/2 mouse or keyboard port<br>• 10/100 Ethernet<br>• USB 2.0 (type A and type B)<br>• Expansion headers (two 40-pin headers) |
| **Memory** | • 8 MB SDRAM, 512 KB SRAM, 4 MB Flash<br>• SD memory card slot |
| **Displays** | • Eight 7-segment displays<br>• 16 x 2 LCD display |
| **Switches and LEDs** | • 18 toggle switches<br>• 18 red LEDs<br>• 9 green LEDs<br>• Four debounced pushbutton switches |
| **Clocks** | • 50 MHz clock<br>• 27 MHz clock<br>• External SMA clock input |

## 2.4. Soft-Core Processors

A soft-core processor is a microprocessor fully described in software, usually in an HDL, which can be synthesized in programmable hardware. A soft-core processor targeting FPGAs is flexible because its parameters can be changed at any time by reprogramming the device. Traditionally, systems have been built using general-purpose processors implemented as Application Specific Integrated Circuits (ASIC), placed on printed circuit boards that may have included FPGAs if flexible user logic was required. Using soft-core processors, such systems can be integrated on a single FPGA chip, assuming that the soft-core processor provides adequate performance. Recently, two commercial soft-core processors have become available: Nios II from Altera Corporation, and MicroBlaze from Xilinx Inc. In an SOPC, the processor, memories and components are created by using the available resources in a programmable logic device and it can be customized according to a particular set of specifications [10].

### a. Nios II Soft Processor

A Nios II processor system is equivalent to a microcontroller or "computer on a chip" that includes a CPU and a combination of peripherals and memory on a single chip.

The term "Nios II processor system" refers to a Nios II processor core, a set of on-chip peripherals, on chip memory and interfaces to off-chip memory, all implemented on a single Altera® chip. Like a microcontroller family, all Nios II processor systems use a consistent instruction set and programming model. **Figure 2.3** shows the architecture of embedded system Nios II/Avalon [11].

In practice, most FPGA designs implement some extra logic in addition to the processor system. Altera FPGAs provide flexibility to add features and enhance performance of the Nios II processor system. Conversely, you can eliminate unnecessary processor features and peripherals to fit the design in a smaller and lower-cost device.

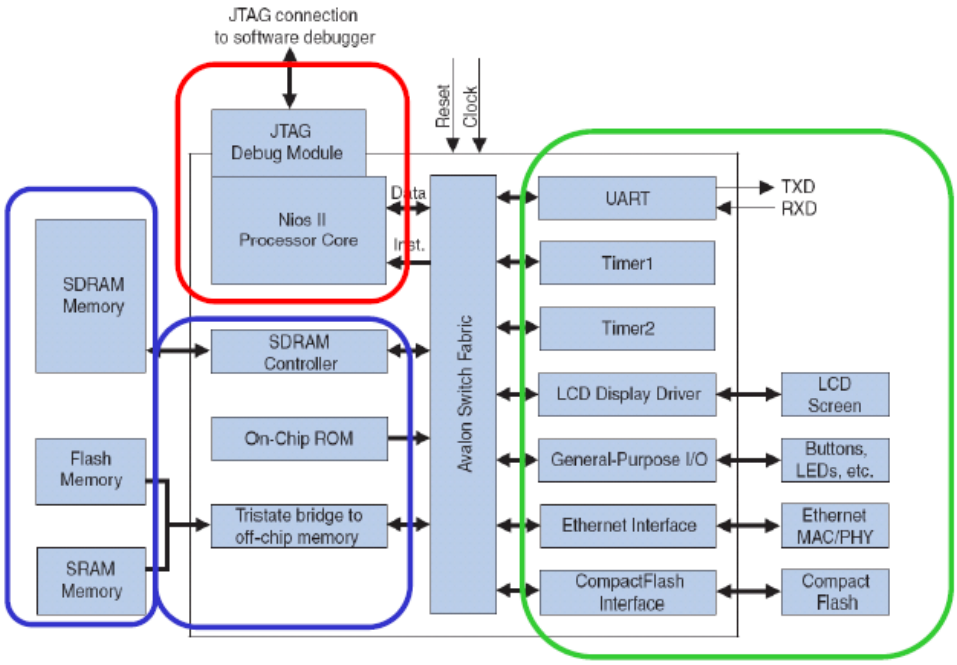


**Figure 2.3** Embedded system Nios II/Avalon Architecture [11]

## b. Overview of Nios II Processor Features

The Nios II processor is a general-purpose RISC processor core; it is a load/store machine with 32 general purpose registers. The Nios II processor has a number of features that can be configured by the user to meet the demands of a desired system. The processor can be implemented in three different configurations as it is represented in **Figure 2.4.**

• **Nios2/f** – "fast" version, optimal for performance-critical applications and applications with large amounts of code and/or data (e.g. a system running a full-featured operating system). This variant has separate instruction and data caches. It provides the highest performance but is much larger in size.

• **Nios2/s** – "standard" version, optimal for cost-sensitive, medium-performance applications, including those with large amounts of code and/or data. This variant has an instruction cache, but no data cache. It provides a smaller-sized core, without sacrificing too much in the way of performance.

• **Nios2/e** – "economy" version, optimal for cost-sensitive applications, such as those found in the automotive and consumer industries. This variant has no instruction or data cache. It is around half the size of the Nios2/s, but this comes at the expense of execution performance [12].

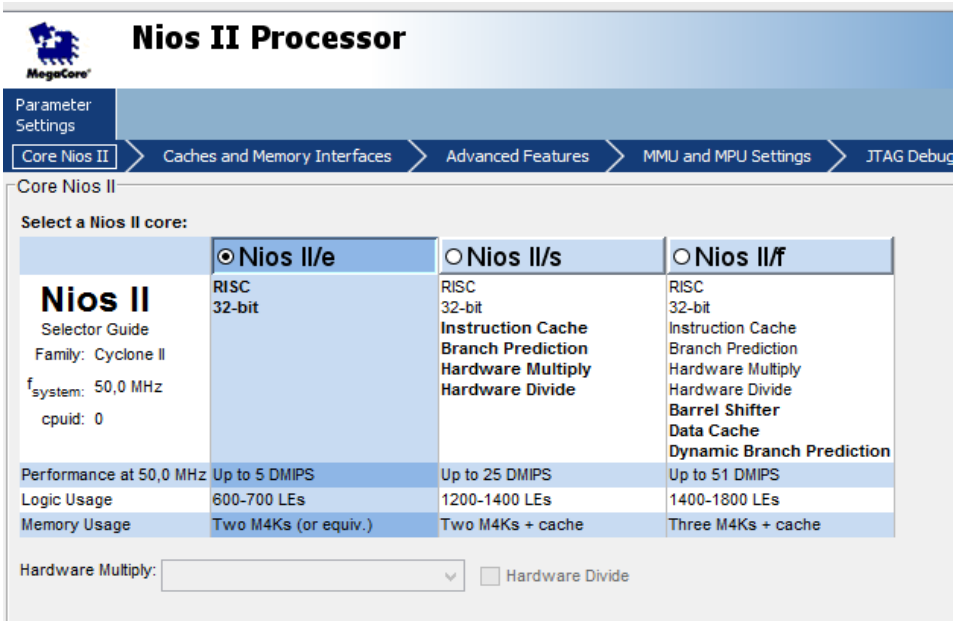


**Figure 2.4** Nios II core configuration wizard

All instructions are 32-bits wide. In the "standard"(Nios2/s) and "fast" (Nios2/f) variants of the core, most instructions execute in a single clock cycle. In the "economy" variant (Nios2/e), instructions execute in six clock cycles [12].

## 2.5. System on Programmable Chip

The term System on a Programmable Chip (SoPC) refers to the combination of processor core with custom hardware, which is implemented using FPGA logic elements and memory blocks. SoPC development requires both hardware and software design elements. The software is typically written (C or assembly language for AMP, C or C++ for IDE) and compiled using a stand-alone tool (Altera's Integrated Development Environment for Nios II or Xilinx's Embedded Development Kit for Xilinx MicroBlaze). The hardware part is usually implemented using specific tools provided in FPGA manufacturer's CAD suite, which can be used for traditional FPGA design as well (Altera's Quartus II and Xilinx's ISE) [13].

The SoPC Builder is a tool used in conjunction with the Quartus II CAD software. It allows the user to easily create a system based on the Nios II processor, by simply selecting the desired functional units and specifying their parameters [14]. These functional units range from simple blocks of fixed logic, to complex, parameterized, and dynamically generated subsystems. SoPC Builder library components include:

- Processors.
- Microcontroller peripherals (including arbitrarily defined general-purpose I/O).
- Interface Protocols (ASI, Ethernet, PCI, Serial).
- Bridges and Adapters (Memory Mapped, Streaming).
- Memories and Memory Controllers (on-chip or off-chip, SRAM, SDRAM).
- University Program (Audio and Video, bridges communication).
- Video and Image Processing.
- Software components (Header files, Generic C drivers, Operating system (OS) kernels, Middleware libraries).

Additionally, FPGA companies provide extensive support tools to ease the customization and use of their cores, including high-level compilers and debuggers targeted at the custom cores.

In the case of Altera and Xilinx, the Processor Core Configuration Tool block shown in **Figure 2.5** is realized in a user-friendly GUI interface that allows the designer to customize the processor for a particular project. A screen shot of Altera's processor configuration wizard is seen in **Figure 2.6**. The configurable parameters can include the data path width, memory, address space, and peripherals (including arbitrarily defined general-purpose I/O, UARTs, etc.) as it is mentioned above. Once the processor parameters are specified in the GUI interface, SoPC Builder connects multiple modules together by generating system interconnect fabric that contains logic to manage the connectivity of all modules in the system. It also generates a system-level hardware description language (HDL) file in Verilog HDL or VHDL, depending on which language you specified when you first set up the system in SoPC Builder that defines all components of the system [15].

Specific pin assignments and additional user logic can be included at this point like any other FPGA design. Next, the full hardware design (processor core and any additional user logic) is compiled (synthesis, place and route, etc.), and the FPGA can be programmed with the resulting file using the standard tools. The hardware design is complete, and the FPGA logic has been determined [15].

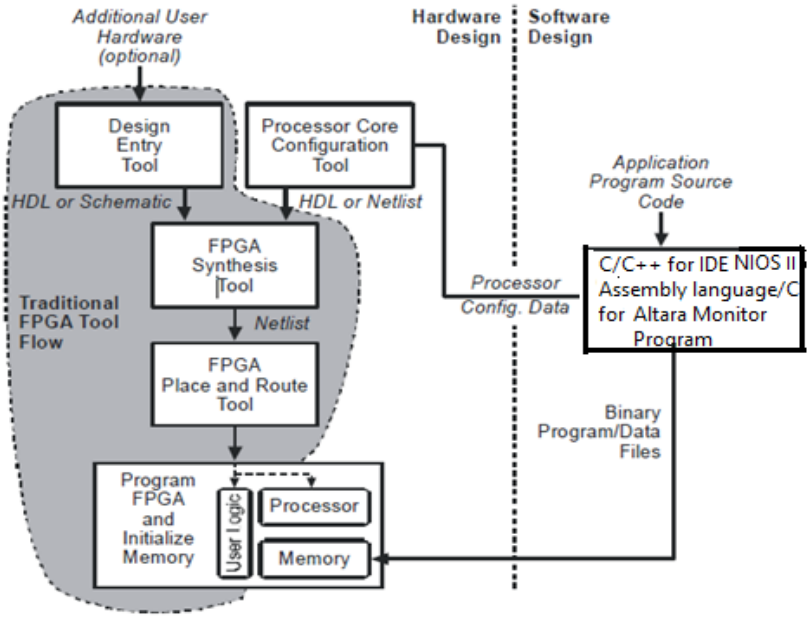


**Figure 2.5** The CAD tool flow for SoPC design [15]

So, In addition to SoPC Builder enables the user to define and generate a complete system-on-a-programmable-chip (SoPC) in much less time than using traditional, manual integration methods, it integrated within the Altera Quartus II software to give FPGA designers immediate access to a revolutionary new development tool [16].

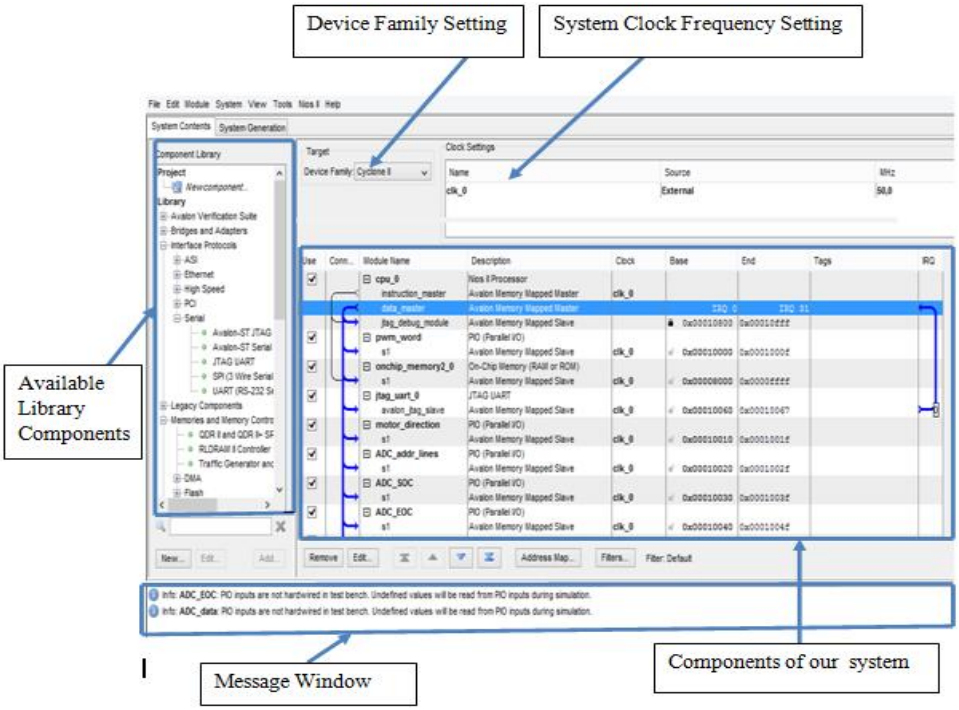


**Figure 2.6** Processor core configuration tool GUI for the Nios II soft processor core

## 3. Motor driver

Generally, even the simplest robot requires a motor to rotate a wheel or performs particular action. However, motors require more current than the microcontroller (FPGA) pin can typically provide, for this reason we need some kind of switches which can accept a small current in the input, and generate a larger current at the output. This operation is done by using what is known as motor driver [17].

### 3.1. Principle of Operation

Motor driver is a little current amplifier that takes a low current control signal and then turns it into higher current signal that can drive a motor [18]. The process of driving the motor in both directions can be achieved by using four switches that are arranged in a shape that resembles the alphabet "H", this circuit is known as the "H-bridge circuit". The H-bridge circuit not only can drive the motor, but also controls its direction.

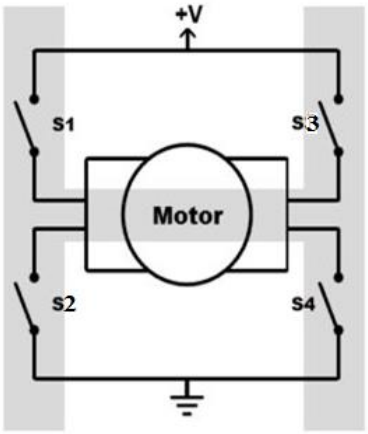


**Figure 2.7** H-bridge configuration [19]

### 3.2. The H-bridge

An H bridge is an electronic circuit that enables a voltage to be applied across a load in either direction [20]. These circuits are often used in robotics and other applications to allow DC motors to run forwards and backwards.

An H bridge is built with four switches as shown in **Figure 2.7** S1, S2, S3 and S4. To power the motor, the pair of switches that are diagonally opposed are turned ON.

Switching ON S1 and S4 will supply the motor with a positive voltage and hence turning the motor in the forward direction, **Figure 2.8** demonstrates the current flow through

the H-bridge circuit. In the same manner, switching ON S2 and S3 will give the motor a negative voltage causing the motor to run in the reverse direction. **Figure 2.9** shows the current flow in this case.

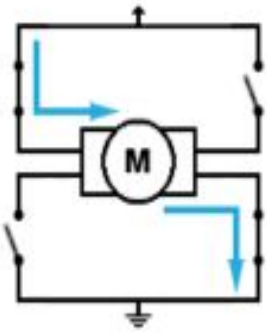


**Figure 2.8** Forward current flow [19]

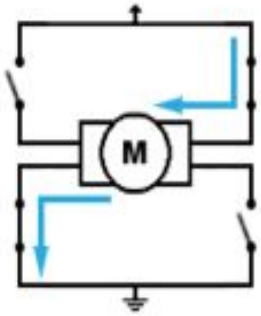


**Figure 2.9** Reverse current flow [19]

However, if we switch ON S1 and S2, there will be a short circuit as the positive power supply will be connected with ground, and that causes circuit damage, overheating, fire or explosion. The same thing will happen with closing both S3 and S4 at the same time as instantiated in **Figure 2.10**.

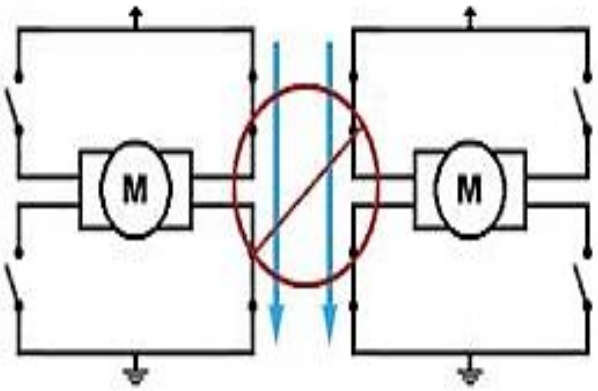


**Figure 2.10**Short circuit [19]

The following table summarizes the H-bridge operation modes, with S1-S4 corresponding to **Figure 2.7**.

**Table 2.2** Operation mode of H-bridge Switching

| S1 | S2 | S3 | S4 | Result |
|---|---|---|---|---|
| 1 | 0 | 0 | 1 | Forward direction |
| 0 | 1 | 0 | 1 | Backward direction |
| 1 | 0 | 1 | 0 | Brake motor |
| 0 | 1 | 0 | 1 | Brake motor |

### 3.3. L298N H-bridge

The choice of the H-bridge has a very important effect on the success of driving any kind of motor. The L298N as shown in **Figure 2.11** is a dual H-bridge motor driver integrated circuit (IC) which contains two inbuilt H-bridge driver circuits, so that we can drive two DC motors simultaneously.

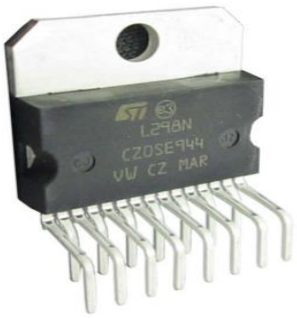


**Figure 2.11** L298 H-bridge

### 3.4. DC Motor

DC motor is a device that converts electrical energy into mechanical energy. DC motors are widely used, cheap, small and powerful for their size. They are most easy to control. One DC motor requires only two signals for its operation. They are non-polarized, means you can reverse the voltage without any damage to motor [21].

DC gear motor is used in this project; geared DC motors can be defined as an extension of DC motor. The gear assembly helps in increasing the torque and reducing the speed [22]. **Figure 2.12** shows the geared motor.

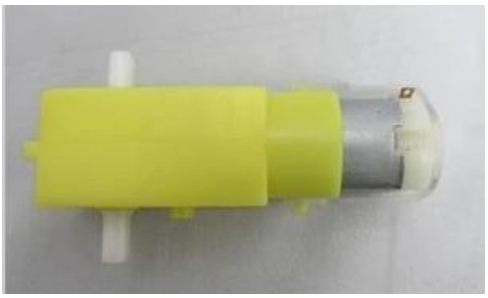

**Figure 2.12** The geared DC motor

## 4. Sensors

As electronics seamlessly weave their way into our lives, proximity sensing becomes a simple and inexpensive companion that enables a wide range of the applications to make our daily life easier [23]. Proximity sensors are sensors that based upon infrared signal detection without any physical contact.

### 4.1. Types of sensors

There are many types of proximity sensors, each one suited to specific applications and environments. The most common types are inductive proximity sensors, capacitive proximity sensors, photoelectric proximity sensors, and ultrasonic proximity sensors. From only two will be discussed in this part: ultrasonic and infrared sensors.

### 4.2. Infrared Sensors

Infrared technology addresses a wide variety of wireless applications. The main areas are sensing and remote controls. Infrared sensor is an electronic device that emits and/or detects infrared radiation in order to sense some aspect of its surroundings. It can detect motion, as well as measure how “bright” the object is, and that what we need for line tracking task [24]. Figure 2.13 represents CNY70 infrared sensor.


**Figure 2.13** CNY70 sensor

### 4.3. Principle of Operation

A reflective sensor includes an infrared emitter and phototransistor adjacent to each other as shown in **Figure 2.14**. It works by illuminating a surface with infrared light, the sensor then picks up the reflected infrared radiation and based on its intensity determines the reflectivity of the surface in question. White surfaces will reflect more light than dark surfaces, resulting in their appearing brighter to the sensor. This allows the sensor to detect a dark line on a white background, or a white line on a dark background [25]. **Figure 2.15** illustrates the operation of IR sensor to measure brightness.

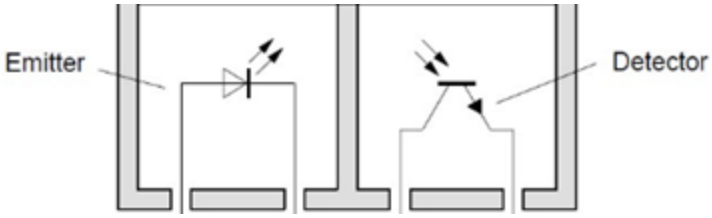


**Figure 2.14** The basic design of infrared sensor

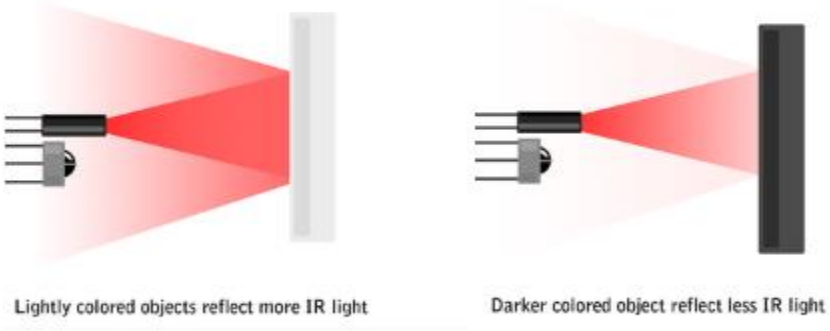


**Figure 2.15** Depiction of The Operation of an IR sensor to measure brightness [26]

### 4.4. Ultrasonic Sensor

A human ear can hear sound frequency around 20HZ to 20 KHZ, the ultrasonic is the sound beyond the human ability of 20KHZ [27].

An ultrasonic sensor is a device capable of converting electrical energy into higher frequency sound waves and also converting sound waves back into electrical energy.

The HC-SR04 ultrasonic sensor uses sonar to determine distance to an object like bats and dolphins do. It provides excellent non-contact range detection from 2cm to 400 cm

with high accuracy reach to 3mm. Its operation is not affected by sunlight or black materials. The module includes transmitter, receiver and control circuits [28].


**Figure 2.16** HC-SR04 ultrasonic sensor

## 4.5. Principle of Operation

To start measurement, Trig of SR04 must receive a pulse of high (5V) for at least 10µs, this will initiate the sensor to transmit out 8 cycle of ultrasonic burst at 40 kHz and wait for the reflected ultrasonic burst. When the sensor detected ultrasonic from receiver it will set the Echo pin to high (5V) and delay for a period (width), which is proportional to distance. To obtain the distance (D) between the obstacle and the transmitter Equation (2.1) is used [29].

. $D=340*\frac{T}{2}$ Equation (2.1)

Where: T is the width of Echo pulse.

The timing diagram is as represented in **Figure 2.17**.

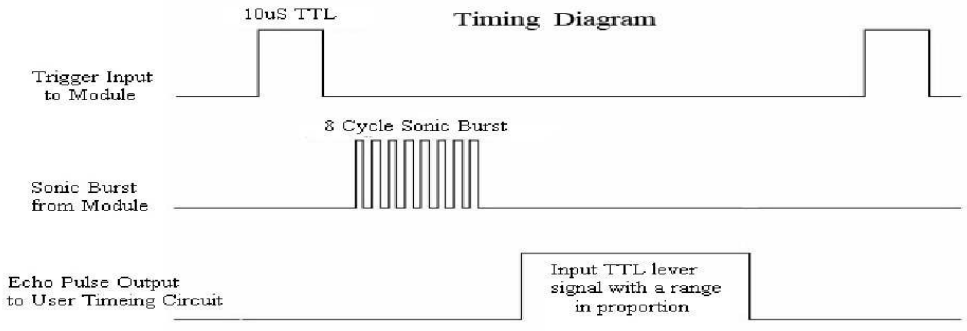


**Figure 2.17** Timing diagram [28]

**Wire connecting direct as following**

- 5V Supply
- Trigger Pulse Input
- Echo Pulse Output
- 0V Ground

## 5. Analog to Digital Converter

Data acquisition component is a monolithic CMOS device with an 8-bit analog-to-digital converter and eight input analog channels. The input which is to be converted to digital form can be selected by using three address lines. The voltage reference can be set using the (Vref+) and (Vref-) pins. The step size is decided based on set reference value. Step size is the change in analog input to cause a unit change in the output of ADC. The default step size is 19.53mV corresponding to 5V reference voltage [30].

### 5.1. Principle of Operation

To get the digital output from ADC after hooking an analog signal up it is required to provide the ADC with seven control signals that must be sent from FPGA. These are addresses lines A, B and C, Address Latch Enable (ALE), clock, start of conversion and output enable .There is also one control signal which is sent to the FPGA, it is the end of conversion signal which goes high when the conversion is complete to indicate that the data is ready to be picked up [31]. **Figure 2.18** represents ADC chip.

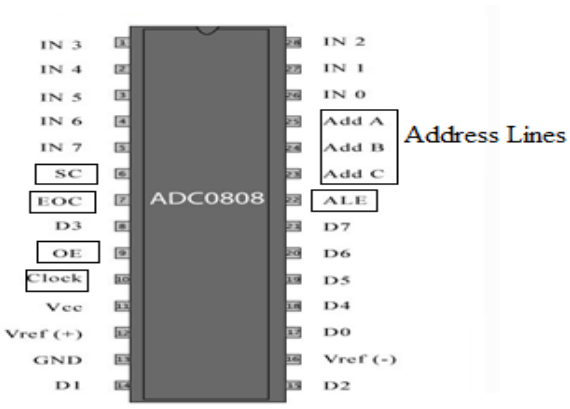


**Figure 2.18** Analog to Digital Converter chip

## 6. Differential drive

Many mobile robots use a drive mechanism known as differential drive. It consists of two drive wheels mounted on a common axis, and each wheel can independently being driven either forward or backward. The drive wheels are usually placed on each side of the robot and toward the rear. The Differential drive that we have used is shown in **Figure 2.19**

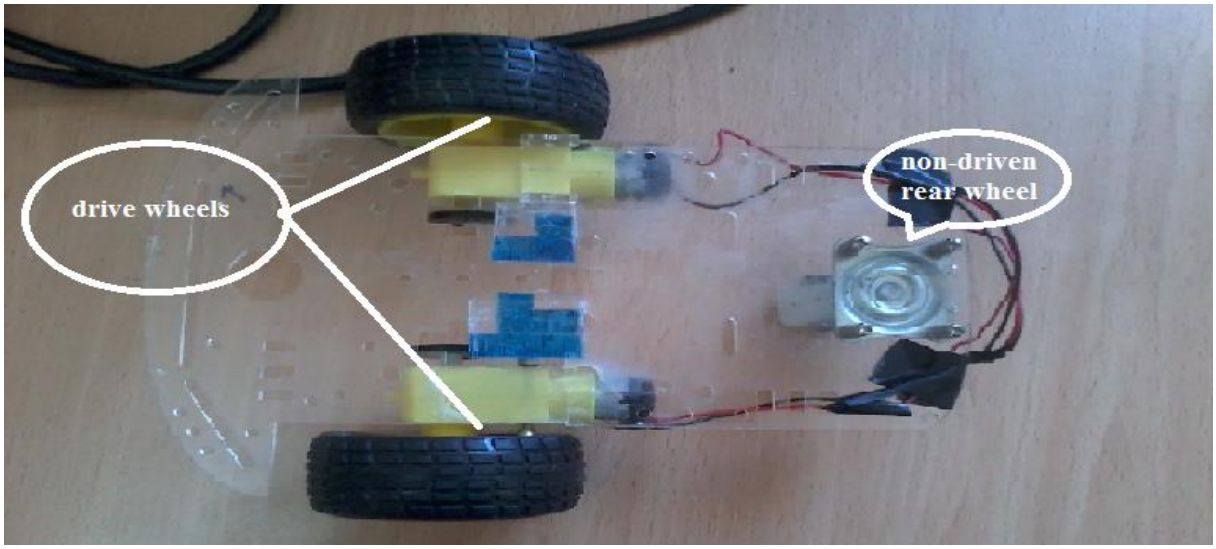


**Figure 2.19** The Differential drive used

# Chapter III: Hardware System Design

## 1. Introduction

This chapter describes in details the design and development of system hardware. This part is crucial. It must be correctly design to ensure that the operation will work appropriately as desired.

The principle of this part is to define how the DC motors are controlled using the H-bridge and the sensors circuit and how these combined circuits are connected to the FPGA.

To fulfill the previous operation, the system is made up of two main subsystems as shown in **Figure 3.1.**

- **OFF-Chip Hardware Subsystem**

This subsystem consists of the proto-board built circuits. It includes the data acquisition unit, the sensors, the interface block (buffers) and the motor driving unit.

- **ON-Chip Hardware Subsystem**

This subsystem represents the brain of the system. It mainly consists of the SoPC system which includes Nios II/e soft processor, memory, JTAG and I/O ports. In addition to some non-SoPC logic units developed in VHDL or LPMs, each has a specific function.

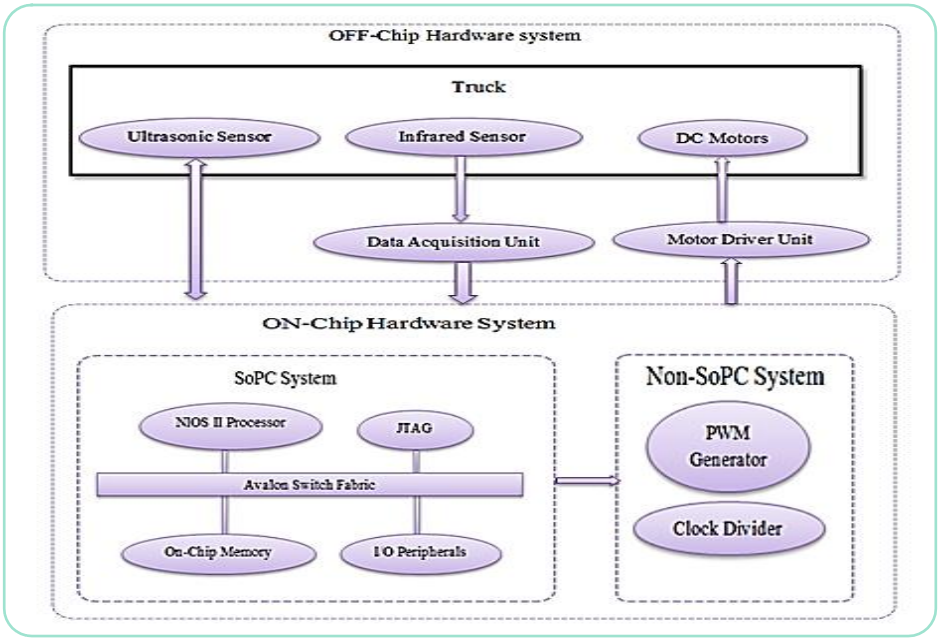


**Figure 3.1** Block diagram of the overall system.

## 2. OFF-Chip Hardware Design

This part deal with the hardware circuit on the proto-board. The different parts of the system can be divided into three units which are: sensing, data acquisition and motor driving units.

### 2.1. Motor Driving Unit

#### a. L298N H-bridge

In order to make the robot move in all directions with different speeds, a motor driving is needed for the robot to operate properly.

The two DC motors are controlled by the PWM generator implemented in the FPGA and L298N H-bridge circuit. The L298N is an IC with two parallel H-bridges. To control the movement of each DC motor, three pins of L298N are used. Two of them are used for controlling the direction by setting one input high and the other low, and the third one is used by setting it high to enable the DC motor.

To control the speed of the two DC motors, a technique called Pulse Width Modulation is applied to the enable pin.

Pulse Width Modulation is a periodic square wave signal which has two levels, high or low, with constant frequency and variable duty cycle.

The speed of the motors is controlled by adjusting the pulse width or in other word duty cycle of PWM signal. With 100% duty cycle, the motor will rotate with a voltage similar to the voltage of the power supply, and 50% duty cycle is similar to half voltage of the power supply. **Figure 3.2** illustrates the internal circuit of the L298N.

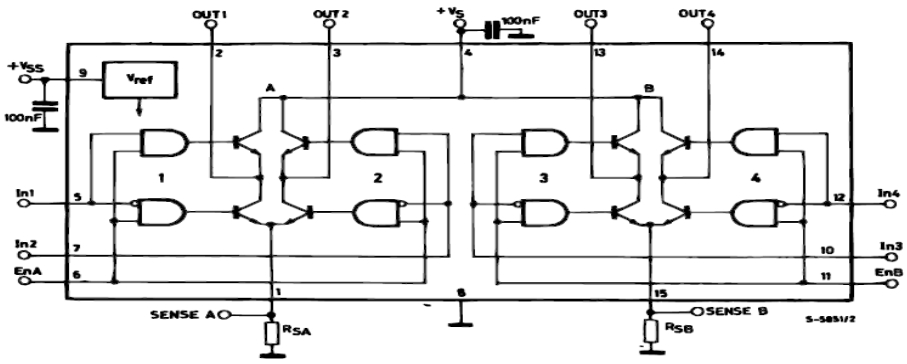


**Figure 3.2** The internal circuit of L298N [32]

### b. Protective Circuit

To protect the whole circuit from any damages a protective circuit is provided using Schottky diodes connected in parallel and in reverse to prevent power flow from the power supply to the motors, and some capacitors in order to absorb noise coming from the motor. **Figure 3.3** shows the motor driver and the protective circuits.

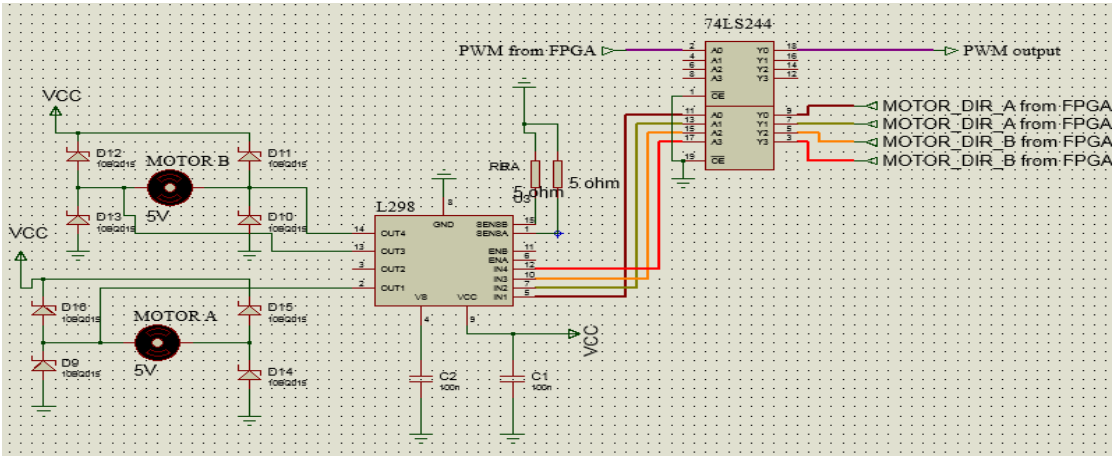


**Figure 3.3** Motor driver circuit

## 2.2. Buffer Interface Block

All the components of the OFF-Chip circuitry implemented on the proto-board operate under a voltage of approximately 5V. However, the DE2 board supplies a voltage of 3.3V. In order to make the signals exchanged between the FPGA and proto-board's components in a compatible way, SN74LS244 buffers are integrated as an interface between them and the FPGA.

## 2.3. Sensing and Data Acquisition Unit

### 2.3.1. Obstacle Avoidance

In this part the ultrasonic sensor is used to detect and avoid any obstacle stands in the path of the robot. The sensor has two openings on its front. One opening transmits ultrasonic bursts and the other receives them. This module is interfaced with the FPGA through two pins, the Trig and the Echo pins through the expansion-header and SN74LS244 buffer. **Figure 3.4** shows the ultrasonic sensor circuit.


**Figure 3.4** The ultrasonic sensor circuit

The trigger pin receive from the FPGA a pulse of at least 10µS.If there is an obstacle in front of the robot, the transmitter module will reflect back the signal as an input through the Echo pin of the sensor to the FPGA; this signal will be high during the whole time from transmitting the ultrasonic burst until receiving it. After getting the time signal, the distance between the robot and the obstacle can be calculated.

## 2.3.2. Path follower detection

Our robot is designed to follow a black path on a white surface. In order to do that, optical sensors are used.

We have used the CNY70. It is a reflective optical sensor that includes an infrared emitter and phototransistor in a leaded package which blocks visible light [34]. **Figure 3.5** illustrates the circuit of the CNY70 IR sensor.

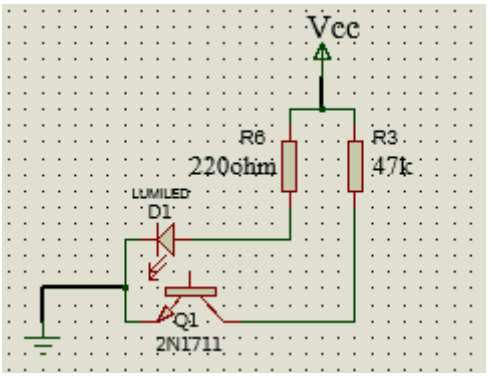


**Figure 3.5** The circuit of CNY70 IR sensor

Its principal of operation is based on the intensity of the IR light reflected back to the phototransistor.

This variation of the intensity will lead to a variation in the resistance between collector-emitter of the phototransistor which will further change the voltage at the output. **Figure 3.6** shows the schematic diagram of the sensors interfaced with the ADC. In the case of white surface, the intensity of the IR light reflected back and received by the phototransistor will increase, results in a decreasing in the output voltage of the sensors. While in the case of black surface, it absorbs IR light means less intensity reflected back and therefore increasing in the output voltage of the sensors.

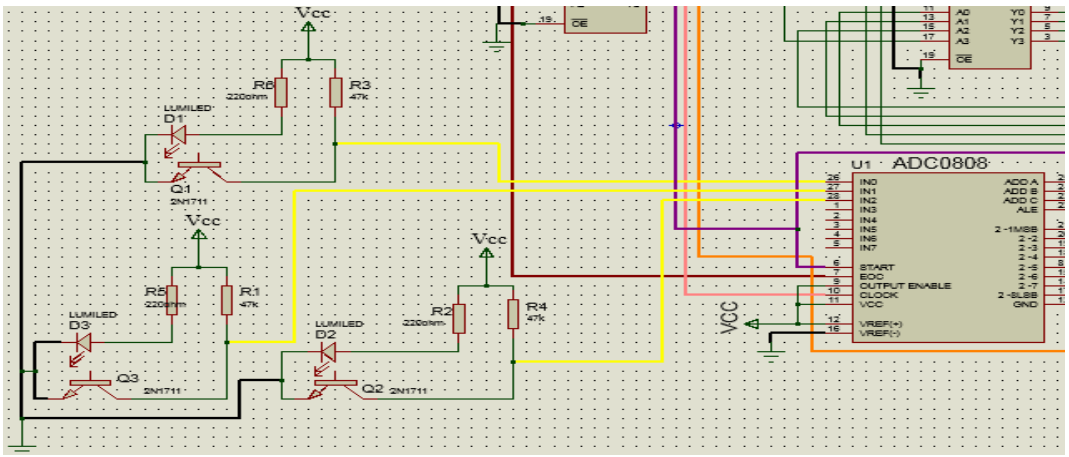


**Figure 3.6** Schematic diagram of interfacing IR sensors with ADC

We have used three IR sensors arranged in a ‘V’ shape for the line detection; based on the data which read from the three sensors the robot will move forward, right and left direction. **Figure 3.7** represents the arrangement of the IR sensors. The width of the line here should be greater or equal the distance between the center sensor and the right or the left one (to make sure that one sensor is out of the path another one is on it).

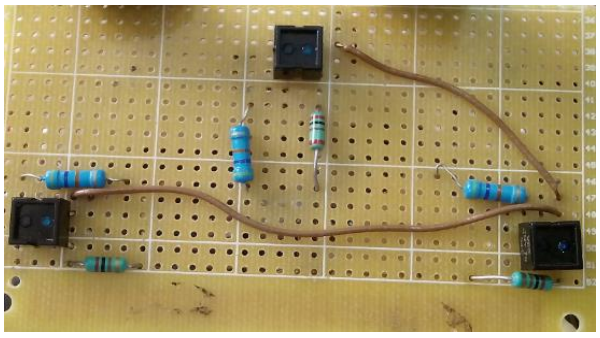

**Figure 3.7** Arrangements of IR sensors

### 2.3.3. Data Acquisition Unit

Since we need digital value from the sensors instead of analog, an analog-to-digital converter is used. The ADC0808 is 8-channel, 8-bit analog-to-digital converter. It works on the principle of successive approximation. It requires a clock signal of 640 KHz to perform a conversion in 100µS. The ADC has a total of eight analog input channels which can be selected using the address lines (A, B and C). Since we have used three sensors, three channels are needed; in this case IN0, IN1 and IN2 are used.

To visualize the digital output of the converter, a collection of eight LEDs have been used, where each LED is connected to respective data lines D0 through D7. The interfacing between the ADC and the DE2 is done through the expansion header. **Figure 3.8** illustrates the interface between the ADC and the FPGA.

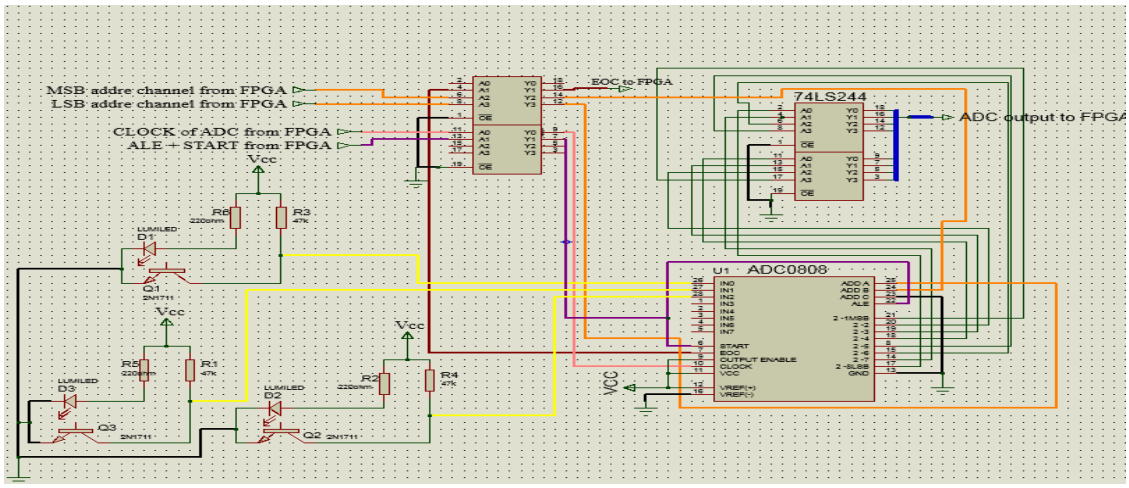


**Figure 3.8** Interfacing between the ADC and the FPGA

## 3. ON-Chip Hardware Design

The main purpose of the ON-Chip part is to maintain control over the hardware part at all the time and determine where the robot has to move. This system is implemented onto the FPGA and is partitioned into SoPC and non-SoPC modules.

### 3.1. The SoPC System

The SoPC subsystem is built around the Nios-II 32-bit embedded soft-core processor, it includes the following items:

- **NIOS II Processor:** the base of the system is the NIOS II Soft-Core Processor under a clock of 50MHz.
- **On-Chip memory:** a memory of 32Kbytes is used to store the program.
- **JTAG UART:** a component that is used to allow the host computer to control the Nios II system.
- **I/O Peripherals:** to control the hardware part, several input/output peripherals are needed. The peripherals are described as following:
    - **ADC_SOC:** 1-bit output signal refer to Start-of-Conversion of the ADC, which triggers the conversion of analog data.
    - **ADC_addr_lines:** is a 2-bit output signal that connected to the address lines of the ADC; it is used to select between the infrared sensors.
    - **ADC_EOC:** is a 1-bit input signal that refer to the End-of-Conversion of the ADC, it is go high when the conversion is finished.
    - **Motor_direction:** is a 4-bit output signal connected to the H-bridge chip.
    - **ADC_data:** is an 8-bit input signal referring to the data coming out from the ADC.
    - **Ultrasonic_Trig:** is a 1-bit output signal that is responsible of triggering the ultrasonic sensor.
    - **Ultrasonic_Echo:** is a 1-bit input signal that receives the digital signal back from the ultrasonic sensor.
    - **Pwm_word:** is a 32-bit output signal that is used to adjust the speed of the two dc motors depending on the requirements of the system.

Once all the peripherals are specified and assigned base addresses, the system is generated. **Figure 3.9** shows the SoPC builder with the whole SoPC system whereas **Figure 3.10** represents the Nios II system block.

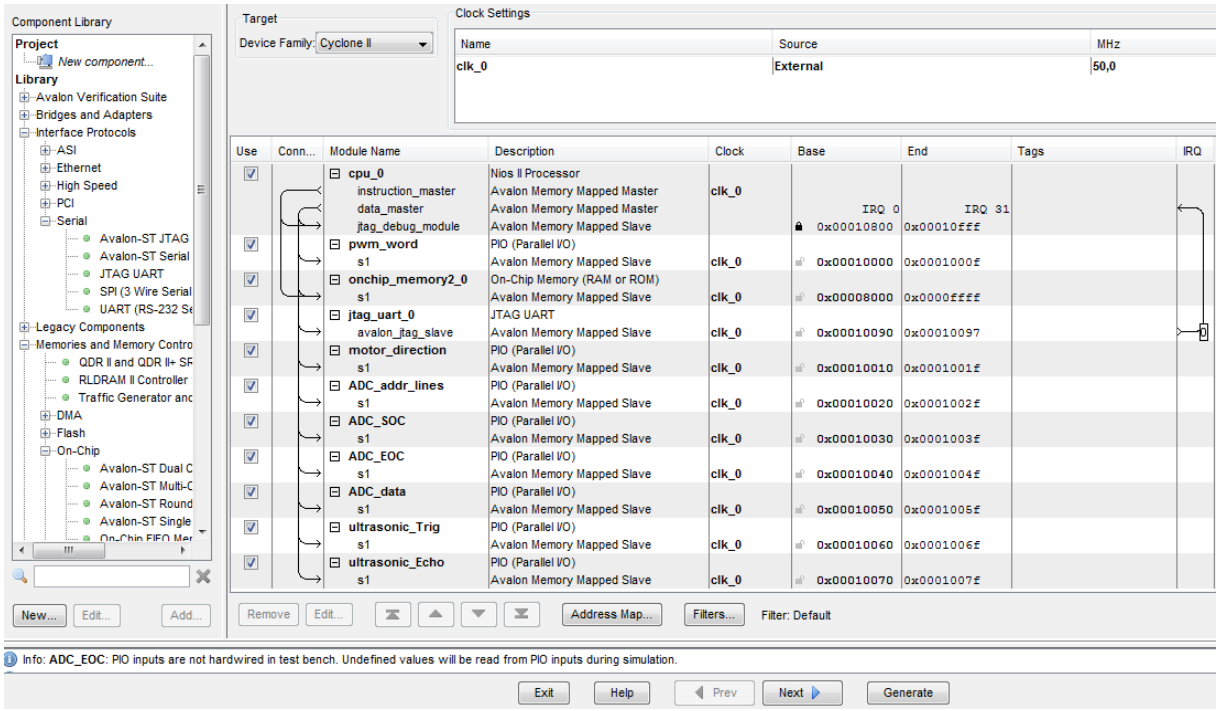


**Figure 3.9** SoPC builder system

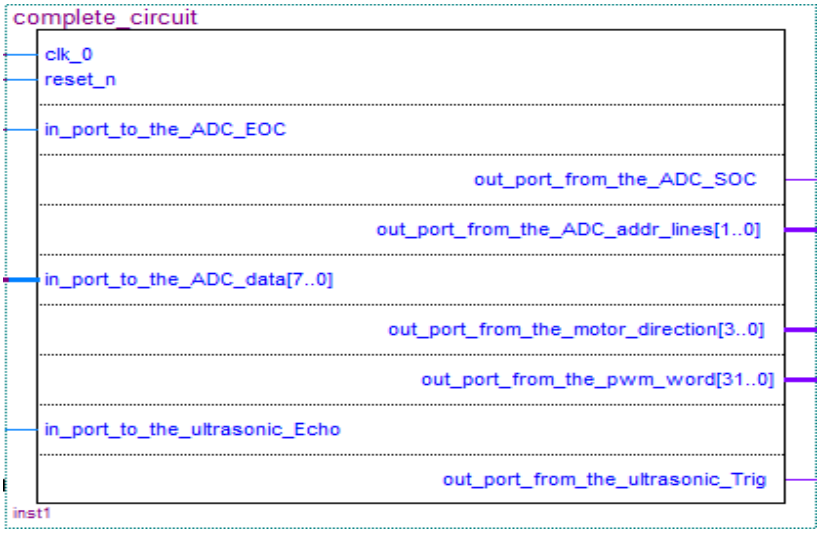


**Figure 3.10** Nios II system block

## 3.2. Non-SoPC System

In addition to using the SoPC system, we have non-SoPC units. These units are the PWM generator for DC motors and the clock divider which are controlled by the SoPC system.

### 3.2.1. PWM Generator

In order to control the speed of the DC motors a PWM is used by applied the PWM signal to the enable pin of the H-bridge. The PWM is implemented using a 5-bit binary counter connected to the select lines of a 32-to-1 multiplexer. The duty cycle of PWM will changed by changing the number of successive ones applied on the data inputs of the multiplexer, therefore, the more successive ones applied, the higher the duty cycle**. Figure 3.11** represents the PWM generator block which has been implemented on Quartus II with frequency of 5 KHz.

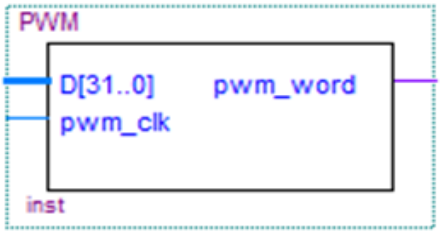


**Figure 3.11** PWM generator block

### 3.2.2. The Clock Divider

As mentioned in section 2.3.3, the ADC0808 requires a clock signal of 640 KHz. To achieve this, we have implemented a counter using an LPM counter of modulus 79 provided in the Quartus II library to divide the system's clock frequency of 50 MHz and generate the required clock signal. Clock divider block is instantiated in **Figure 3.12** using the schematic capture of the Quartz II environment.

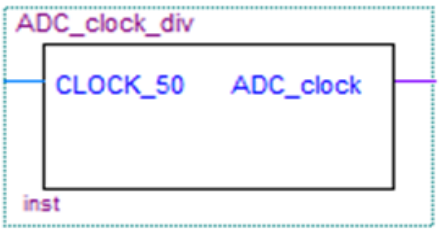


**Figure 3.12** Clock divider block

After specifying all the settings and assigning the base-addresses, the system is generated.

A symbol file of the NIOS II/e system is created, the PWM and Clock divider units are connected to it. The pins are assigned to each signal. **Figure 3.13** shows the final block diagram/schematic file (bdf) of the line follower with obstacle avoidance system in Quartus II.

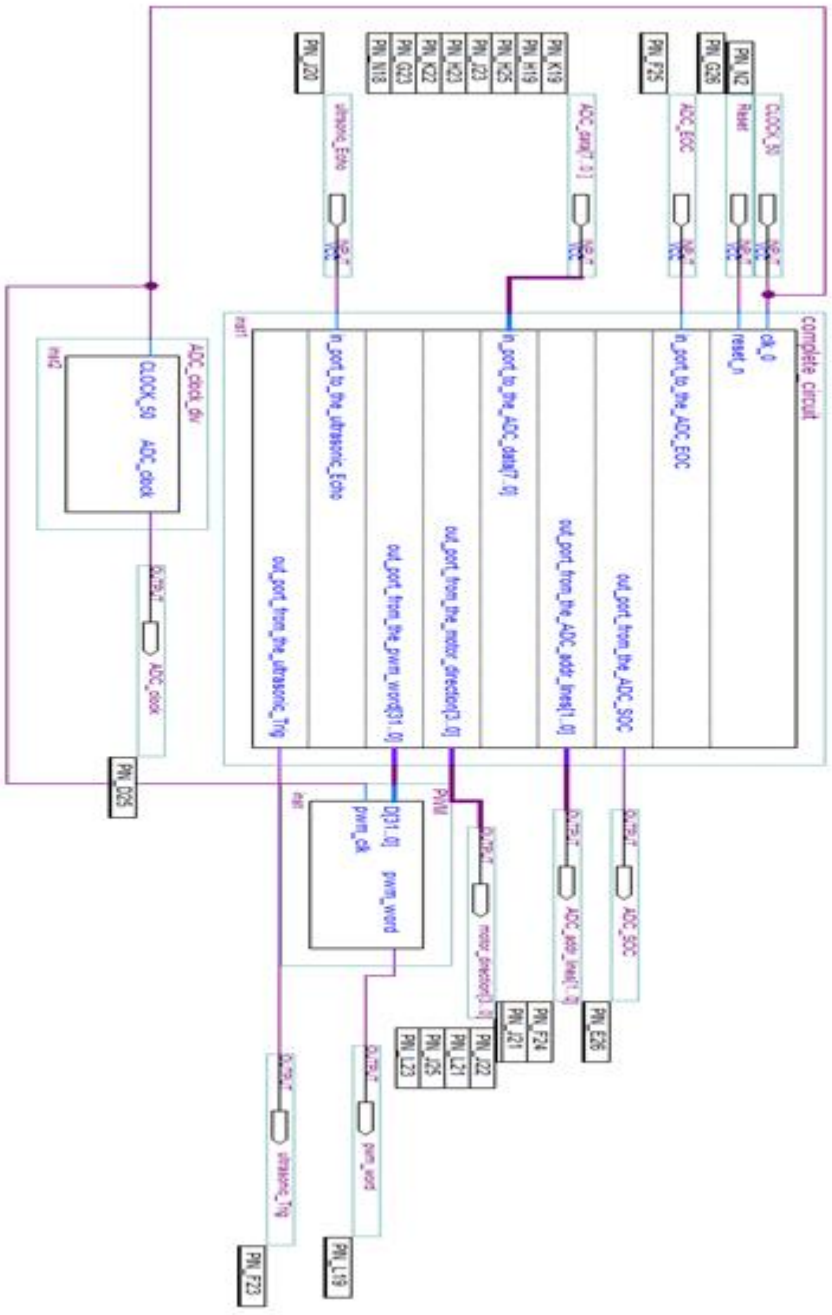


**Figure 3.13** The block diagram file (.bdf) of the final system

The compilation of the whole system was successful. The summary of the compilation shown in **Figure 3.13,** we can see that the entire system uses 5% of the total logic elements, 769 registers, 7% of the total pins and 56% of the total memory bits.

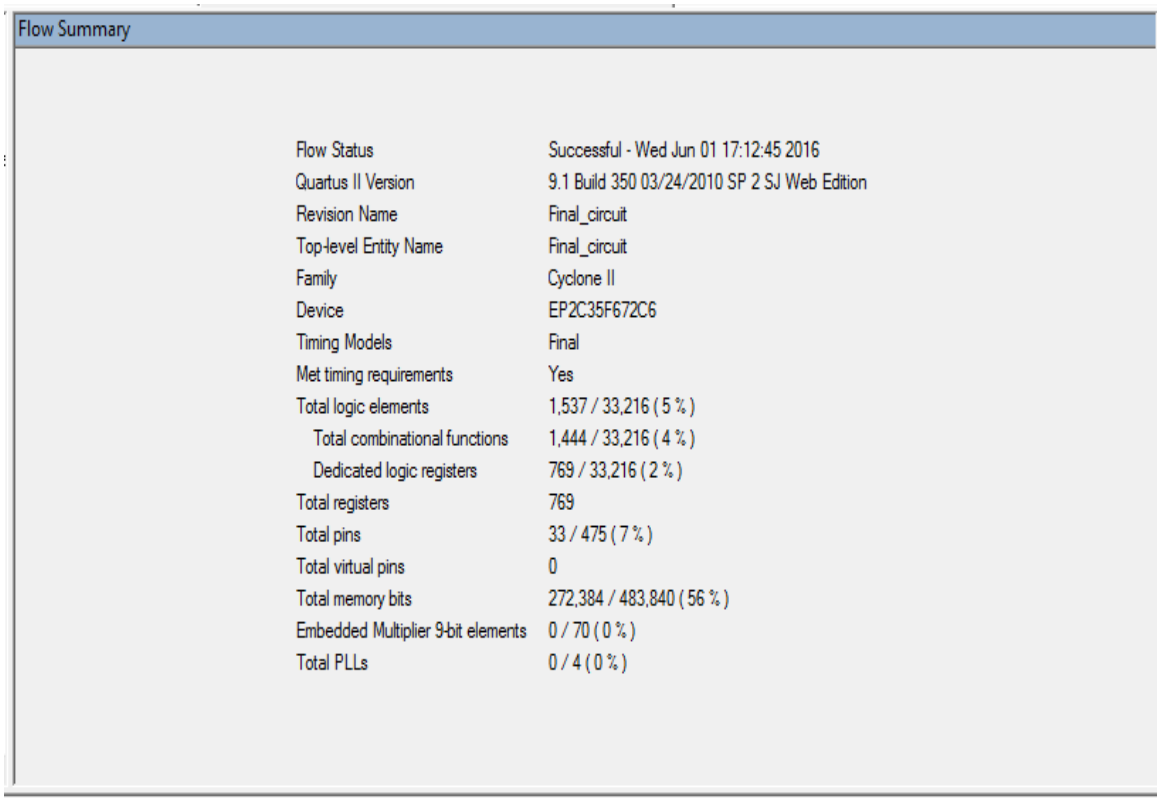


**Figure 3.14** Compilation summary

# Chapter IV: Software System Design

## 1. Introduction

This chapter describes the software part of the digital controller. Different firmware codes used to program the Nios II subsystem are represented by flowcharts.

## 2. The Firmware

An embedded system consists of two parts: hardware and software (firmware). Each part is useless with the absence of the other part. After building the hardware design, the software takes its role to manipulate the data and take decision accordingly. This rule is ensured through a code written in assembly language or a high-level language and executed by a microprocessor.

Our project depends on the data read from the three infrared sensors and the ultrasonic one. The program receives data from these sensors and generates the output corresponding to the current situation. The generated output is used to control the rotation of motors such as forward, turn left and turn right.

## 3. Structure of the Program

The complete code is divided into several subroutines; each one is responsible for achieving a desired task. The main code allows the connection between these subroutines to reach the objective of the system.

## 4. Main Program

In this section we are going to explain how the main program is implemented and how the different parts are connected. The car first looks for the path by moving one dc motor direction and the other one left, at the same time reading data from the three infrared sensors to track the black line. Once this is done the car has to follow the path by adjusting the two dc motors in such a way it still on the path. If an obstacle is detected on the path, the car will move to the right direction to scan it; it will avoid the obstacle on the path from the right if nothing is on this side. The car will move to the left direction to scan it too when an obstacle is noticed on the right; it will continue its way from this direction if no constraint is detected there. When the car avoids the obstacle it will look for the path again using the previous mentioned procedure. The car will stop when the path is ended or when obstacles are detected

on the path and on both sides of it. The flowchart in **Figure 4.1** is illustrated in more details the algorithm of our system

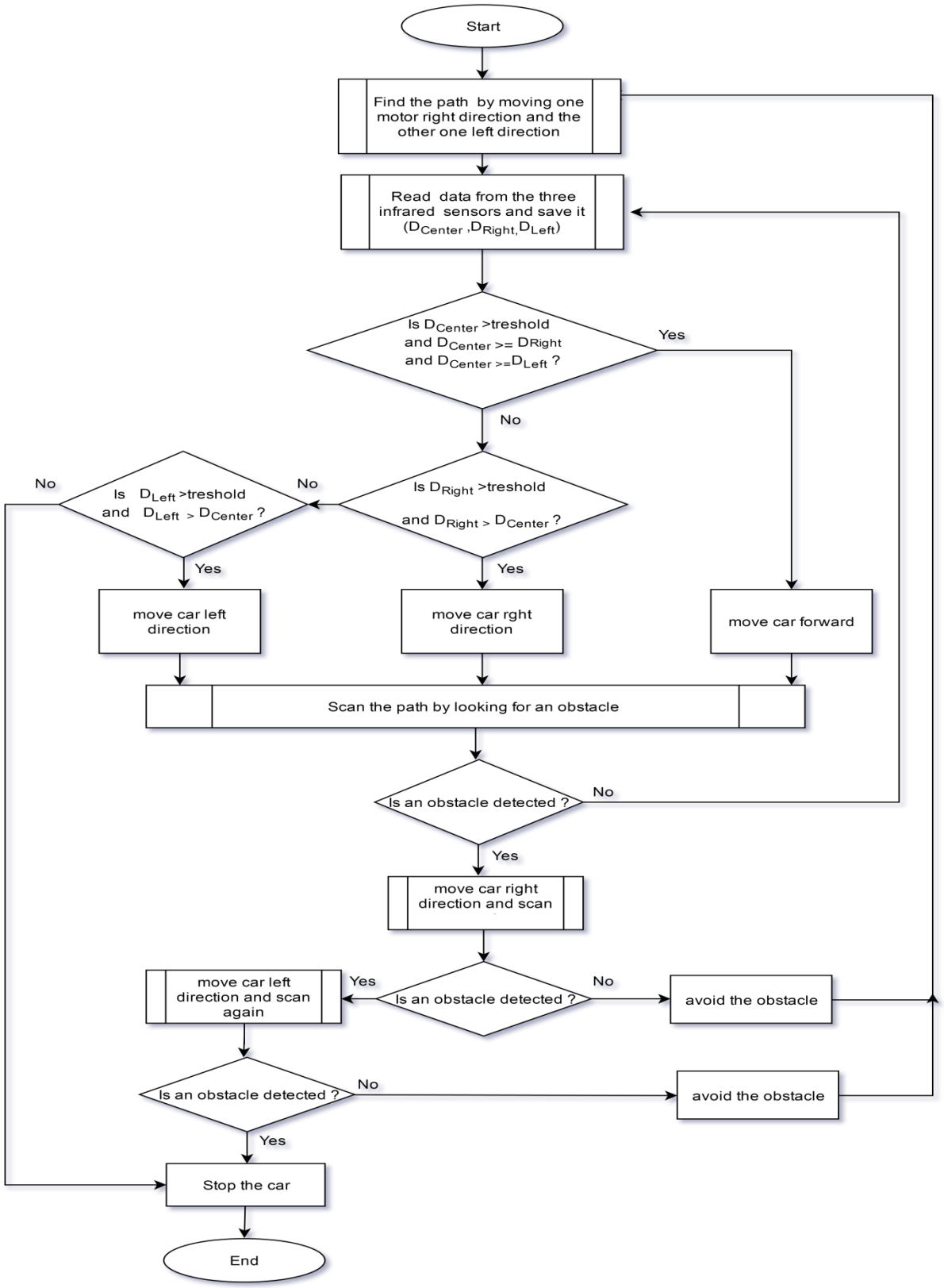


**Figure 4.1** The flowchart of the main program.

## 5. Data Acquisition Routine

The routine described by the flowchart in **Figure 4.2** performs the acquisition of resulted data from three infrared sensors. The routine selects sequentially the channel to convert then sends the Start/ALE pulse to latch the address of the selected channel and starts the conversion. Next, it keeps polling the end of conversion output signal until the conversion is over. Once acquired, the data is temporarily stored into specific registers before it is processed by the Nios II system.

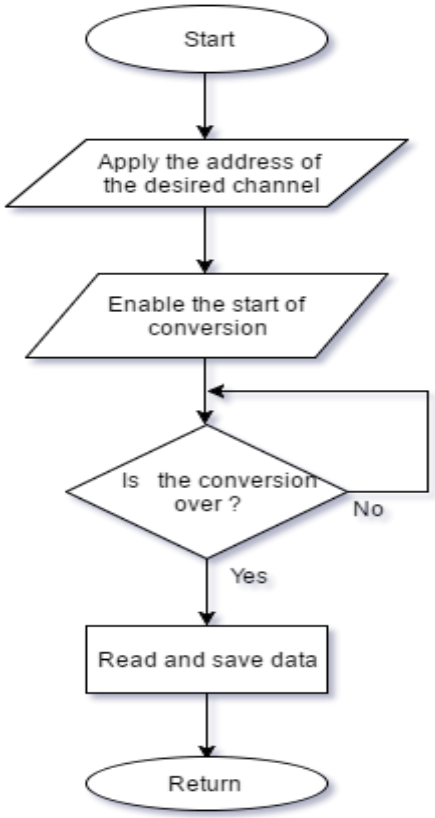


**Figure 4.2** Data Acquisition Routine

## 6. Path Following Procedure

In this section, we are going to describe in more details how we implemented the path follower. **Figure 4.3** shows the flowchart to achieve this goal.

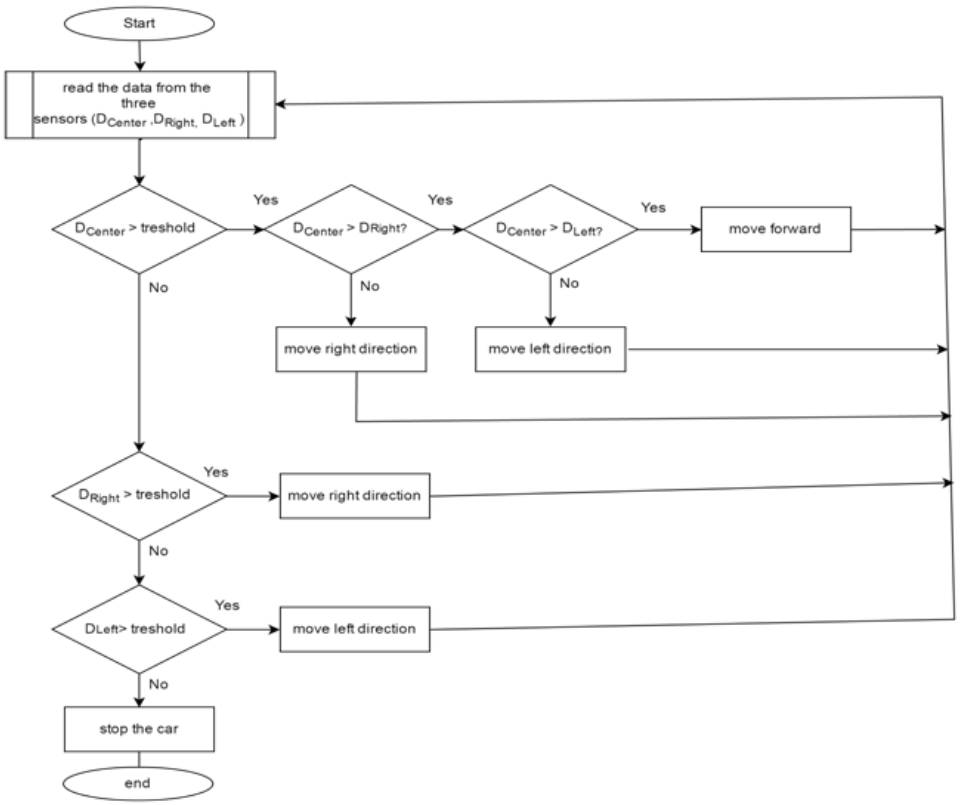


**Figure 4.3** path following flowchart

## 7. Obstacle Scanning Procedure

The aim of this procedure is to compute the duration for the Echo port to be high since this indicates the presence of an obstacle.

The procedure starts by triggering the sensor with high for at least 10 µs and waits for Echo port to be high. While this port is high (1) a counter is incremented. Once the Echo port turns low (0) the final count is translated into distance according to specific calculations as shown in **Figure 4.4.**

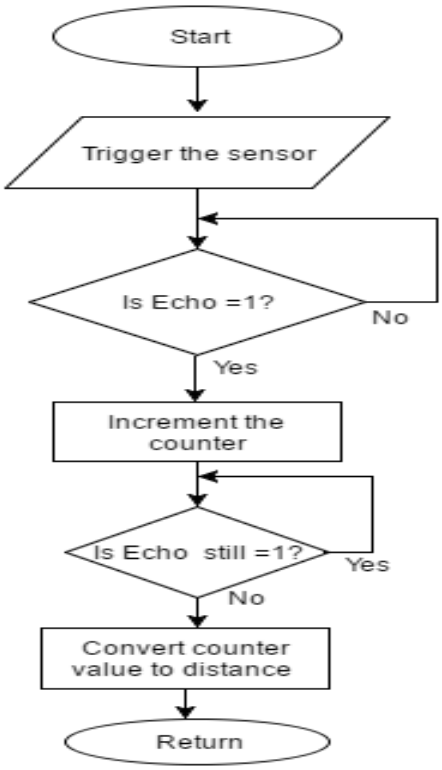


**Figure 4.4** Obstacle scanning routine

## 8. Programming Language

Nios II offers the possibility to work with assembly or high level language C. The native language of any microprocessor is the assembly one which is the lower language. Therefore no need for the compiler to convert it like other higher languages. So it is very useful to communicate with the hardware level. **Figure 4.5** depicts a portion of the Nios II assembly language.

```
                                .equ ultrasonic_Echo, 0x00010070
                                .equ pwm_left_motor, 0x00010080
                                .global _start
                                 _start:
                                  movia r2, ADC_SOC
                                _start:
0x00008000    00800074              orhi     r2, zero, 0x1
0x00008004    10800c14              ori      r2, r2, 0x30
                                  movia r7, ADC_addr_lines
0x00008008    01c00074              orhi     r7, zero, 0x1
0x0000800c    39c00814              ori      r7, r7, 0x20
                                  movia r9, pwm_word
0x00008010    02400074              orhi     r9, zero, 0x1
0x00008014    4a400074              orhi     r9, r9, 0x1
                                  movia r12, pwm_left_motor
0x00008018    03000074              orhi     r12, zero, 0x1
0x0000801c    63002014              ori      r12, r12, 0x80
                                  movia r10, motor_direction
0x00008020    02800074              orhi     r10, zero, 0x1
0x00008024    52800414              ori      r10, r10, 0x10
                                  movia r15, ADC_EOC
0x00008028    03c00074              orhi     r15, zero, 0x1
```

**Figure 4.5** Portion of Nios II assembly language

# Chapter V: Conclusion

# Conclusion

An FPGA-based path following with obstacle avoidance controller was designed and implemented. The autonomous car successfully tracks the black line above the white surface. The robot senses the line and follows the track in all directions.

The system is designed using SOPC approach. It is designed around the Nios II/e soft-core processor instantiated in control of the motors based on the data provided by the infrared sensors.

The SOPC approach has several benefits. It is flexible in both hardware and software. The designer can use the exact components that fit his application. Moreover the reconfiguration and modification of the system can be done at any time by adding a new component or deleting an exist one.

The economic version of the Nios II/e is bounded. Some instructions needed for software are not available which causes an increase in response. However, working with System on Programmable Chip has allowed us to deal with new concepts and learn a lot of things.

During the realization of this project troubleshooting has taken a long period delaying our work. Some of these troubleshooting are:

- The circuits on the proto-board: they are so sensitive due the several connected wires. If the system does not work it is necessary to check the whole circuit to discover the problem.
- The motor driven unit: L298 has caused drooping of voltage several times due to its shape which is difficult to connect tightly.
- Over sensitivity of the infrared sensors: they respond as well today light.
- The dissimilar nature of two motors: this is one of the most critical problems that we have faced. The motors were driven in different speeds in identical operating conditions. The major cause of this is the high friction of one motor unit.

The problems mentioned above can be solved by using printed circuit board PCB instead of proto-board circuit which is more efficient and reliable. L293 is a driven motor chip which can replace L298. In addition, it is easy to connect it. It is provided with the protective circuit so there is no need to add this latter to the whole driven circuit. The PWM word which is sent by the controller to enable the DC motor can be a correction method of dissimilarity so each motor has its own PWM even that it is not very efficient in our case due the limitation of the multiplexer we have used (5 bits), but by increasing the bits of multiplexer the speed will be more controlled.

As future enhancement and further work:

- Providing the car with more infrared sensors (5 sensors) so the car may follow the path more accurately.
- Providing the two projects with an android application using Smart phone to select a mode that can be obstacle avoidance or a path follower.
- Using color sensors so the robot can sense different colors. It can be used in the robotic game competition and other fields.
- Adding a path recording mechanism or rather making the robot track the path from a start point till the end of the path so it memorize the followed path then derivate the reverse path without much effort and more rapidly .
- Optimizing the best path; the car tracks the different paths from a certain origin to a given destination with memorizing the shortest path to reach the destination so the robot will optimize its path from different paths in a short time as maze solver robot does.

# References


[1] KAUSHIK VELUSAMI, DEEPTHI VENKITA RAMANAN, SHRIRAM KUSUDEVAN, Introduction to AI-AN Intelligent Approach. Lulu Publication,United States of America.Edition 2013.

[2] M. S. Islam & M. A. Rahman, Design and Fabrication of Line Follower Robot, Department of Electrical and Electronic Engineering, Rajshahi University of Engineering and Technology, Rajshahi-6204, BANGLADESH, 2013 .

[3] Shireen Matar, Umera Rana , Line Following and Obstacle Avoidance Robot Using PIC MICROCONTROLLER, Semester project.

[4] HADDJI Isma, HAFNAOUI Iman , SOPC Based Automated Self Parking Car 'INGENIEUR D'ETAT' In Electrical Engineering Option :Control , June 2011.

[5] Benzekri Azzouz "FPGA-Based Intelligent Dual-Axis Solar Tracking Control System" Dlibrary univ-Boumerdes.dz.8080, 2014

[6] MEBARKI Mohamed Razine, TOUATI Meziane, MOUHOU Hamimi, FPGA Based PID Controller, Final Year Project in Electrical and Electronic Engineering. Option: Computer. Institute of Electrical and Electronic Engineering, UMBB,June 2014.

[7] FPGA architecture (www.globalspec.com).

[8] AZZOUGUI Yasmina, MAKHLOUF Yasmine,FPGA-Based Autonomous Obstacle Avoidance Robot, Final Year Project in Electrical and Electronic Engineering, Option:Computer.Institute of Electrical and Electronic Engineering, UMBB, June 2015.

[9] DE2 Development and Education Board (http://wl.altera.com).

[10] Franjo Plavec , Soft-Core Processor Design ,Master of Applied Science, Graduate Department of Electrical and Computer Engineering, University of Toronto 2004

[11] René Beuchat, Real Time Embedded Systems "System On Programmable Chip " NIOS II Architecture and Interrupt services Some Avalon Peripherals PDF, 2013.

[12] NIOS II 32-bit RISC Processor.pdf , CR0164(V2.0) March 11, 2008 (www.altium.com).

[13] http://ati.ttu.ee/~alsu/05_SoPC.pdf.

[14] Introduction to the Altera SOPC Builder Using VHDL Design.pdf. (https://people.ece.cornell.edu).

[15] James O. Hamblen ,Tyson S. Hall, Georgia Institute of Technology, Atlanta, GA 30332, USA, Southern Adventist University, Collegedale, TN 37315, USA, 3, Sept. 2006.

[16] Introduction to SOPC builder.pdf. (https://www.uni-ulm.de).

[17] ALLOUANE Souhila, NEFFAD Amira, Remote Control Robot using Android Mobile Device, Final Year Project in Electrical and Electronic Engineering Option:Computer. Institute of Electrical and Electronic Engineering,UMBB, June 2015 .

[18] motor-driver.(www.futureelectronics.com).

[19] DIY AutoLab, h-bridge.(www.diyautolab.blogspot.com).

[20] http://www.wikipedia.com.

[21] Manan Shaileshkumar Shah, Line Following and Obstacle Avoiding Car, Master of Science, Electrical and Electronic Engineering, CALIFORNIA State University, SACRAMENTO, 2010.

[22] Pintu and Alka Dubey, Design and Implementation of IR based Line. pdf

[23] Light Sensor Applications.pdf (www.intersil.com).

[24] what is an Infrared Sensor? ( www.wisegeek.com).

[25] Sensor Accessories.pdf.

[26] what is an IR sensor? education.rec.ri.cmu.edu.

[27] HC-SR4 User Guide.pdf (www.elefreaks.com).

[28] Ultrasonic Ranging Module HC-SR04. (www.elefreaks.com)

[29] HC-SR04 User's Manual , Cytron Technologies Sdn Bhd,Taman Universiti Johor, Malaysia. May 2013.

[30] ADC0808/ADC0809 8-Bit μP Compatible A/D Converters with 8-Channel Multiplexer.PDF (www.ti.com).

[31] Shauna Rae, Getting an ADC0809 Analogue to Digital Converter to Work for You, November 1999.

[32] Datasheet of L298N.