



بسم الله الرحمن الرحيم



Sudan University of Science and Technology

College of Graduate Studies

Implementation and Simulation of Rotary Automated Car Parking System

تنفيذ ومحاكاة نظام موقف سيارات آلي دوّار

A Thesis Submitted in Partial Fulfillment of the Requirements for the
Degree of M.Sc. in Electrical Engineering (Control and Microprocessor)

Prepared by:

Saad Eldin Suliman Yousif Ali

Supervised by:

Dr. Awadallah Taifour Ali

January 2016

الآية

قال تعالى:

(وَإِذْ قَالَ إِبْرَاهِيمُ رَبِّ اجْعَلْ هَذَا بَلَدًا آمِنًا وَارْزُقْ أَهْلَهُ

مِنَ الثَّمَرَاتِ)

صدق الله العظيم

سورة البقرة الآية (126)

Dedication

I would like to dedicate this work to my mother, father and the rest of my family also to all who helped me.

Thank you all...

Acknowledgement

Great thankful to Allah who lightens our ways, our life and our heart...

I would like to convey my appreciation to the Sudan University of Science and Technology.

Apart from the efforts of me, the success of any research depends largely on the encouragement and guidelines of many others. I take this opportunity to express my gratitude to the people who have been instrumental in the successful completion of this research.

I would like to show my greatest appreciation to **Dr. Awadallah Taifour Ali**. I cannot say thank you enough for his tremendous support and help.

Abstract

Lack of space availability has always been a problem in urban areas and major cities and to add to it there are cars parked randomly on the streets that further limit the space. In order to handle the issue of parking in busy places various types of vehicle parking systems are used worldwide namely such as multi-level automated car parking, Volkswagen car parking and many more, but the rotary automated car parking system is best and suitable because it availability space and it does not need big space for construction. The present research is aimed to design and implementation a control system and model for the rotary automated car parking system to accommodate six cars. This was carried out using different devices such as ATmega16, IR sensor and stepper motor. Also PROTEUS software is used for system simulation.

After testing and study the rotary automated car parking is found to be very suitable so it contribute to solve the problem of traffic congestion, as well as that of the control circuit is easy to design and programming.

مستخلص

ان عدم توفر المساحة مشكلة دائمة في المناطق الحضرية والمدن الكبرى وإضافة إلى ذلك هناك سيارات متوقفة عشوائياً في الشوارع التي تزيد إلى حد من المساحة. من أجل التعامل مع هذه المشكلة من وقوف السيارات في الأماكن المزدحمة تستخدم أنواع مختلفة من أنظمة مواقف السيارات في جميع أنحاء العالم مثل مواقف السيارات الآلي متعدد المستويات ومواقف فولكس واجن وغيرها الكثير، ولكن نظام موقف السيارات الآلي الدّوار هو الأفضل والأنسب لأنه يوفر المساحة و لا يحتاج إلى مساحة كبيرة للبناء. يهدف هذا البحث إلى تصميم وتنفيذ نظام تحكم لموقف السيارات الآلي الدّوار يستوعب ست سيارات، وتم تنفيذ ذلك باستخدام أجهزة مثل ATmega16 ، حساس الأشعة تحت الحمراء ومحرك الخطوة. أيضاً تم استخدام برنامج المحاكاة PROTEUS من أجل محاكاة النظام.

بعد الإختبار والدراسة وجد أن نظام موقف السيارات الآلي الدّوار مناسب جداً حيث أنه يساهم في حل مشكلة الإزدحام المروري، بالإضافة الي أن دائرة التحكم يسهل تصميمها و برمجتها.

Table of Contents

	Page No.
الاية	i
Dedication	ii
Acknowledgment	iii
Abstract	iv
مستخلص	v
Table of Contents	vi
List of Figures	vii
Chapter One: Introduction	
1.1 General	1
1.2 Problem Statement	1
1.3 Objectives	2
1.4 Methodology	2
1.5 Layout	2
Chapter Two: Theoretical Background and Literature Review	
2.1 History	3
2.2 Types of Parking	3
2.2.1 Traditional car parking	4
2.2.2 Automatic car parking system	4
2.3 Microcontroller	8
2.4 Seven Segment Display	9
2.5 Infrared Sensor	9
Chapter Three: Hardware and Software Considerations	
3.1 System Description	10
3.2 Microcontroller Atmega16	10

3.2.1 Atmega16 block diagram	11
3.2.2 Atmega16 pin descriptions	14
3.3 Stepper Motor	15
3.4 ULN 2003A Driver	16
3.5 Break Beam Sensors	17
3.6 Seven Segment Display	17
3.7 Push Button Switch	18
3.8 Power Supply	18
3.8.1 Step down transformer	18
3.8.2 Rectifier unit	18
3.8.3 Input filter	19
3.8.4 Regulator unit	19
3.8.5 Output filter	19
3.9 System Simulation	19
3.10 Working Principle of Circuit	20
3.11 System Flowchart	23
3.12 System Code	24
Chapter Four: System Implementation and Testing	
4.1 System Implementation	25
4.1.1 Step one	25
4.1.2 Step two	25
4.1.3 Step three	26
4.1.4 Step four	26
4.1.5 Step five	27
4.1.6 Step six	28
4.2 System Testing	28
4.2.1 Step one	28
4.2.2 Step two	29
4.2.3 Step three	30
Chapter Five: Conclusion and Recommendations	

5.1 Conclusion	31
5.2 Recommendations	31
References	32
Appendix A	

List of Figures

Figure	Title	Page No.
2.1	Traditional car parking	4
2.2	Robot car parking	5
2.3	Integrated car parking solution	6
2.4	Multi-level automated car parking	7
2.5	Rotary smart parking system	8
3.1	Block diagram for the circuit control	10
3.2	Atmega16	11
3.3	Atmega16 block diagram	13
3.4	Stepper motor 14	16
3.5	ULN2003A driver	17
3.6	Break beam sensor	17
3.7	Electrical circuit break beam sensor	18
3.8	The main circuit design	20
3.9	The main circuit design after running	21
3.10	The main circuit design after park car number one	22
3.11	System flowchart	23
4.1	Installation atmega16	25
4.2	Installation ULN2003A and stepper motor	26
4.3	Installation 7-segment display and LEDs	26
4.4	Connecting IR sensors and push button switches	27
4.5	Connecting the power supply	27
4.6	The final model	28
4.7	Entry of first car	29

4.8	Status of full parking	30
4.9	Retrieving a car number 4	30

CHAPTER ONE

Introduction

1.1 General

The rotary automated car parking system belongs to the class of rotary smart car parking systems. The traditional parking systems such as multilevel or multi-storey car parking systems (non-automated), robot car parking systems, automated multilevel car parking systems etc. have been implemented on a huge scale. But these systems have a major disadvantage of large space consumption which is successfully eliminated with the use of a rotary car parking system. Moreover, the latter provides the added benefits of flexible operation without the need of an attendant and added security and least chances of vehicle damage. Since the model makes use of composite parts, it is easy to assemble and dismantle and is thus more convenient than the traditional car parking systems. The rotary model is specifically designed to accommodate multiple cars in the horizontal space of two. The structure can accommodate six cars in the space of two and can even be customized to hold a greater number depending upon the requirements of the user and can be efficiently put to use in much space crunched areas. Parking spaces cannot cope with the growth of the number of vehicles. In many urban housing societies, the parking space ratio is 1:1.

As compared to the existing parking arrangements, the most obvious advantage is maximum space utilization; it is safer and more convenient. The rotary automated car parking system is totally automated with the user being given a unique IDentification (ID) corresponding to the trolley being allocated to him/her. This kind of equipment is useful to solve the issue of limited parking space available in busy cities [1].

1.2 Problem Statement

In metropolitan cities, vehicle parking has become a major concern in all busy areas and a good traffic system needs a good parking system. Wherefore design a stable and good control system is the solution to this problem using high technology to make it easier to deal with the

system, in order to put this system in a real model for performs work as required, and at the same time do not ignore the importance of time and the necessary safety.

1.3 Objectives

The Objectives of this research is to design and implementation of a control system for the rotary automated car parking system to accommodate six cars.

1.4 Methodology

- Study of all preview related works.
- Selecting a suitable control.
- Coding of the control algorithm is written using C language.
- Drawing control circuit using PROTEUS simulation program.
- Design a small model to testing the control system.

1.5 Layout

This thesis consists of five chapters. Chapter one deals with introduction, problem statement and objectives of the study. Chapter two discusses history and types of the car parking system in addition to explaining the elements used in the research in general. Chapter three presents the devices used in the research indeed, also explanation the system simulation, system flowchart and the system code. Chapter four illustrates implementation and testing the control circuit and model. Chapter five contains conclusion and recommendations.

CHAPTER TWO

Theoretical Background and Literature Review

2.1 History

With the increase in the number of vehicles in high-density urban housing the problem with a limited amount of parking space appeared. The solution to this problem is the automatic parking system. The first such system was created in the United States the automatic parking system attracted attention in the years 1940-1950, and the systems used were Bowser, Pigeon Hole and Roto Park. In the years 1957 to 1974 Bowser and Pigeon Hole systems were used [2], but due to frequent mechanical problems and prolonged waiting time for the vehicle, the interest in such a solution significantly decreased. This interest, however, came back since 1990, and already in 2012 there were 25 projects planned or in progress, which yielded about 6,000 parking spaces. While until 1990 the interest in the systems was mediocre in Europe, Asia and Central America, since 1970 in the automatic parking system more advanced technology was used. In Japan, since the early 1990's about 40,000 parking spaces based on the automatic parking system paternoster type were created. It is estimated that in Japan in 2012 there were about 1.6 million parking spaces.

A car parking system is a mechanical device that multiplies parking capacity inside a parking lot. Parking systems are generally powered by electric motors or hydraulic pumps that move vehicles into a storage position.

There are two types of car parking systems: traditional and automated. In the long term, automated car parking systems are likely to be more cost effective when compared to traditional parking garages. Automatic multi-storey automated car park systems are less expensive per parking slot, since they tend to require less building volume and less ground area than a conventional facility with the same capacity. Both automated car parking systems and automated parking garage systems reduce pollution cars are not running or circling around while drivers look for parking spaces [3].

2.2 Types of parking

Parking system progression through several stages of development, is generally divided into:

2.2.1 Traditional car parking

Traditional car parking shown in Figure 2.1, is one of the first forms, where it is allocated specific parking place and then draw lines by white paint or yellow paint.



Figure 2.1: Traditional car parking

2.2.2 Automatic car parking system

Automated car parking systems use a similar type of technology to that used for mechanical parcel handling and document retrieval. The driver leaves the car inside an entrance area and technology parks the vehicle at a designated area. Hydraulic or mechanical car lifters raise the vehicle to another level for proper storing. The vehicle can be transported vertically (up or down) and horizontally (left and right) to a vacant parking space until the car is needed again. When the vehicle is needed, the process is reversed and the car lifts transport the vehicle back to the same area where the driver left it. In some cases, a turntable may be used to position the car so that the driver can conveniently drive away without the need to back up. The automatic parking system has many kinds and some of these will be explained below:

• Robot car parking

Dubai is car park Capable of handling 765 vehicles, it is the first of several large-scale robotic car parks being built to address parking problems in the United Arab Emirate (UAE) as shown in Figure 2.2. All the customer sees is a parking garage with space for one car, though the 'floor' is platform which rides on the top of a robotic trolley. When the customer leaves the vehicle and collects a ticket, the wall of the garage drops away and the car is whisked to an elevator, which in turn takes the car to another trolley. From there, the machine parks the car in the dark depths of the structure. In total, the process takes around three minutes.

With this technology, you do not need to drive through the garage to find a parking space. You simply drive your car to an entry station and leave your car to be picked up by the computerized lifts that will safely place it inside the building on a shelving system. When you leave and when you return to a central point and your car is swiftly retrieved for you.



Figure 2.2: Robot car parking

• Integrated car parking solution

Integrated car parking solution Shown in Figure 2.3, has the Following characteristic:

- Customized application suitable for various types of landscapes and buildings.
- Structures available below or / and above the ground.
- Reducing management cost through precise control and low power consumption.

- User is easy control by soft touch on the operation panel screen.
- Simultaneous vertical and horizontal movement for short waiting periods.
- When a vehicle stops in front of the entrance, automatically door opens and trolley transfers the vehicle to parking cell.
- Very safe for the driver with no need to stay in the car.
- Good for underground parking lot.



Figure 2.3: Integrated car parking solution

• Multi-level automated car parking

A multi-level car parking shown in Figure 2.4, is essentially a building with number of floors or layers for the cars to be parked. The different levels are accessed through interior or exterior ramps. An automated car parking has mechanized lifts which transport the car to the different levels. Therefore, these car parks need less building volume and less ground space and thus save on the cost of the building. It also does away the need for employing too many personal to monitor the place. In an automated multi-level car parking, the cars are left at the entrance and are further transported inside the building by robot trolley. Similarly, they are retrieved by the trolley and placed at the exit for the owner to drive away.



Figure 2.4: Multi-level automated car parking

• Rotary smart parking system

The rotary smart parking system shown in Figure 2.5, consists of one automated mechanical system, which rotates at an angle of being perpendicular to the ground. By this way, cars parking and retrieving is reliable. It runs in a rotating way that driven by the reducer, transmission device drives the parking pallets rotating for cars parking and retrieving. The smart rotary parking system has the advantages of [4]:

- Adopting the Rubik is cube technique, saving lots of land coverage.
- Small land coverage, flexible equipment and high space utilization ratio.
- Low cost of fire control, external package and civil engineering base.
- Brief structure, package available, good-looking appearance, permeability good and coordination with the surrounding landscape.
- It ensures quick and automated parking and easy retrieval of vehicles.
- The surface space required is equivalent to the parking space of two cars only.
- Most suitable for parking in offices, malls and similar places.

- Low maintenance levels are required by the system.
- Does not require any parking attendant.
- It can be easily constructed in a small area, just requiring a simple concrete base and three phase power supply.

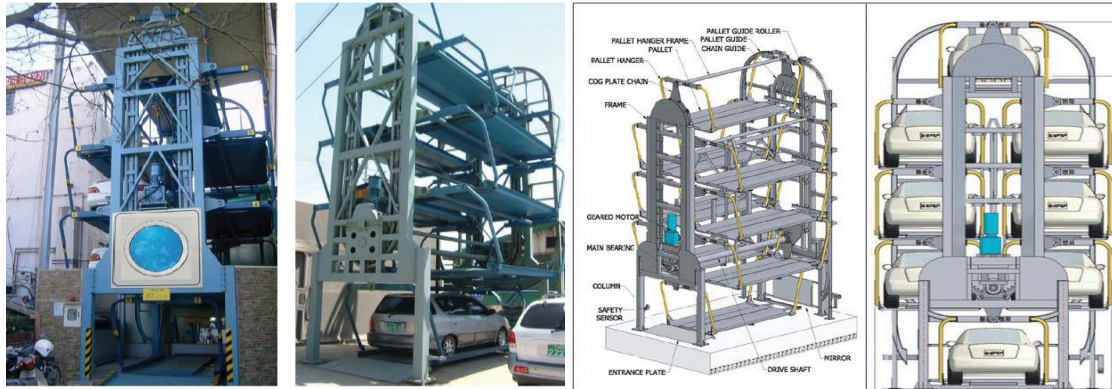


Figure 2.5: Rotary smart parking system

2.3 Microcontroller

A microcontroller is a small computer on a single integrated circuit containing a processor core, memory, and programmable input/output peripherals. Program memory in the form of ferroelectric Random Access Memory (RAM), NOR flash or One Time Programmable Read Only Memory (OTP ROM) is also often included on chip, as well as a typically small amount of RAM. Microcontrollers are designed for embedded applications, in contrast to the microprocessors used in personal computers or other general purpose applications.

Microcontrollers are used in automatically controlled products and devices, such as automobile engine control systems, implantable medical devices, remote controls, office machines, appliances, power tools, toys and other embedded systems. By reducing the size and cost compared to a design that uses a separate microprocessor, memory, and input/output devices, microcontrollers make it economical to digitally control even more devices and processes. Mixed signal microcontrollers are common, integrating analog components needed to control non-digital electronic systems.

Some microcontrollers may use four-bit words and operate at clock rate frequencies as low as 4 kHz, for low power consumption (single-digit mill watts or microwatts). They will generally

have the ability to retain functionality while waiting for an event such as a button press or other interrupt; power consumption while sleeping (Central Processing Unit (CPU) clock and most peripherals off) may be just Nano watts, making many of them well suited for long lasting battery applications. Other microcontrollers may serve performance-critical roles, where they may need to act more like a Digital Signal Processor (DSP), with higher clock speeds and power consumption [5].

2.4 Seven Segment Display

A 7-segment display, or seven-segment indicator, is a form of electronic display device for displaying decimal numerals that is an alternative to the more complex dot matrix displays. seven segment displays are widely used in digital clocks, electronic meters, basic calculators, and other electronic devices that display numerical information.

2.5 Infrared Sensor

InfRared sensor (IR) detectors have been called the eyes of the digital battlefield. Military applications in Western countries have spearheaded and dominated the requirements in this field akin to many other emerging fields. In addition to many military applications for IR systems such as target acquisition, search and track, missile seeker guidance, there is a great potential for IR systems in the commercial market. IR systems enhance automobile and aircraft safety, medical diagnosis, and manufacturing quality and control. Industry is looking to expand into the commercial market because the military market is decreasing and concurrently becoming more specialized. Today, only about 20% of the market is commercial. After a decade, the commercial market is estimated to grow by over 70% in volume and 40% in value [6].

CHAPTER THREE

Hardware and Software Considerations

3.1 System Description

The rotary automated car parking system block diagram shown in Figure 3.1, mainly consists of a microcontroller ATmega16, stepper motor, ULN2003A driver, infrared sensor, 7-segment display, push button switch, power supply.

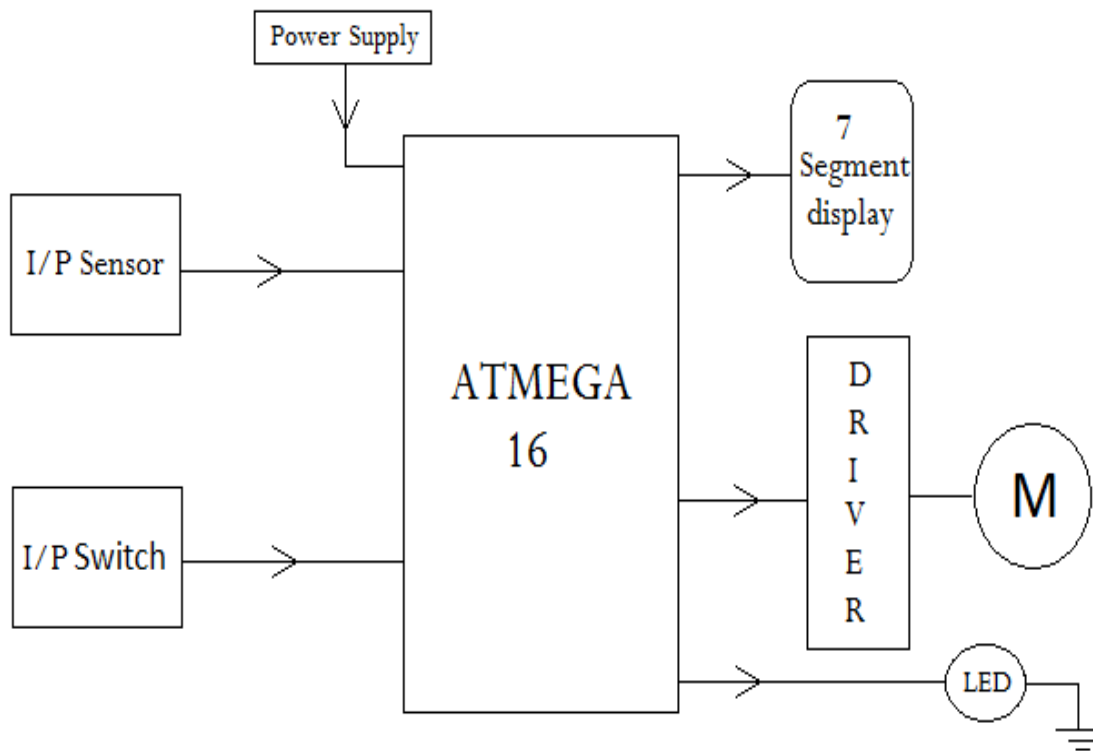


Figure 3.1: Block diagram for the circuit control

3.2 Microcontroller ATmega16

The ATmega16 microcontroller is a 40-pin wide Dual In Line (DIP) package chip. This chip was selected because it is robust, and the DIP package interfaces with prototyping supplies like solderless bread boards and solder-type perf-boards. This same microcontroller is available in a surface mount package, about the size of a dime. Surface mount devices are more useful for circuit boards built for mass production. Figure 3.2 shows the ‘pin-out’ configuration of the ATmega16. This diagram is very useful, because it tells you where power and ground should

be connected, which pins tie to which functional hardware, etc [7].

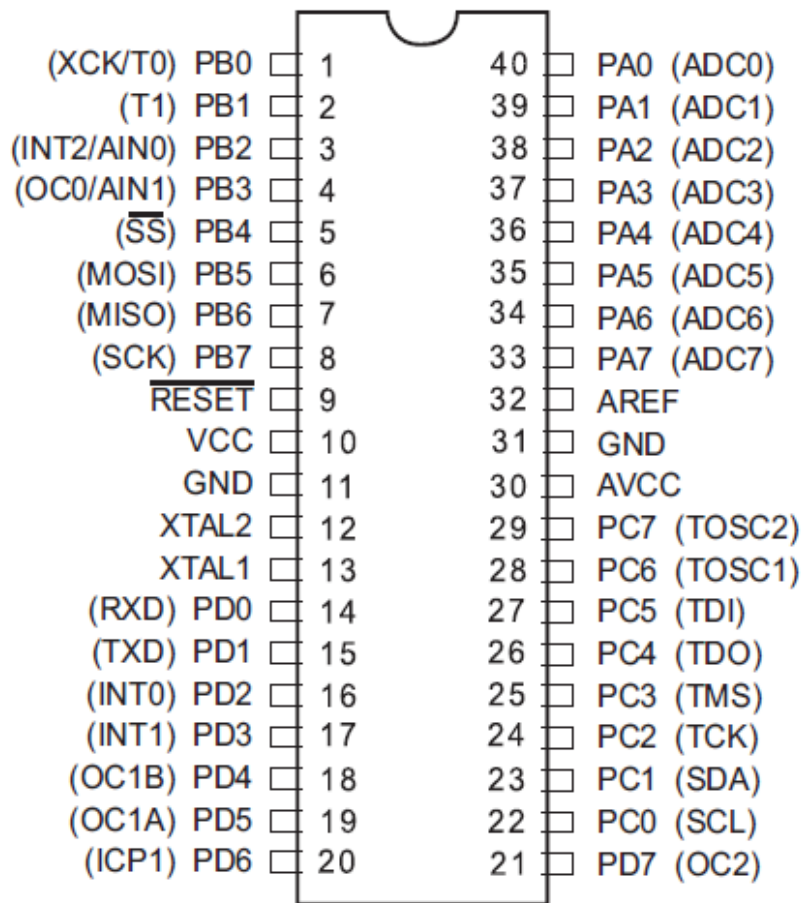


Figure 3.2: ATmega16

3.2.1 ATmega16 Block Diagram

The AVR core combines a rich instruction set with 32 general purpose working registers. All the 32 registers are directly connected to the Arithmetic Logic Unit (ALU), allowing two independent registers to be accessed in one single instruction executed in one clock cycle. The resulting architecture is more code efficient while achieving throughputs up to ten times faster than conventional Complex Instruction Set Computer (CISC) microcontrollers.

The ATmega16 provides the following features: 16 Kbytes of In-System Programmable Flash Program memory with Read-While-Write capabilities, 512 bytes Electrically Erasable Programmable Read Only Memory (EEPROM), 1 Kbyte SRAM, 32 general purpose input/output lines, 32 general purpose working registers, a Joint Test Action Group (JTAG) interface for Boundary scan, On-chip Debugging support and programming, three flexible

Timer/Counters with compare modes, Internal and External Interrupts, a serial programmable Universal Synchronous/Asynchronous Receiver/Transmitter (USART), a byte oriented Two-wire Serial Interface, an 8-channel, 10-bit Analog-to-Digital Converter (A/D) with optional differential input stage with programmable gain (Thin Quad Flat Package (TQFP) only), a programmable Watchdog Timer with Internal Oscillator, an Serial Peripheral Interface (SPI) serial port, and six software selectable power saving modes. The Idle mode stops the CPU while allowing the USART, Two-wire interface, A/D Converter, SRAM, Timer/Counters, SPI port, and interrupt system to continue functioning. The Power-down mode saves the register contents but freezes the Oscillator, disabling all other chip functions until the next External Interrupt or Hardware Reset. In Power-save mode, the Asynchronous Timer continues to run, allowing the user to maintain a timer base while the rest of the device is sleeping. The ADC Noise Reduction mode stops the CPU and all Input/Output (I/O) modules except Asynchronous Timer and ADC, to minimize switching noise during ADC conversions. In Standby mode, the crystal/resonator Oscillator is running while the rest of the device is sleeping. This allows very fast start-up combined with low-power consumption. In Extended Standby mode, both the main Oscillator and the Asynchronous Timer continue to run.

The device is manufactured using Atmel's high density nonvolatile memory technology. The Onchip In-System Programming (ISP) Flash allows the program memory to be reprogrammed in-system through an SPI serial interface, by a conventional nonvolatile memory programmer, or by an On-chip Boot program running on the AVR core. The boot program can use any interface to download the application program in the Application Flash memory. Software in the Boot Flash section will continue to run while the Application Flash section is updated, providing true Read-While-Write operation. By combining an 8-bit RISC CPU with In-System Self-Programmable Flash on a monolithic chip, the ATmega16 is a powerful microcontroller that provides a highly-flexible and cost-effective solution to many embedded control applications.

The ATmega16 AVR is supported with a full suite of program and system development tools including: C compilers, macro assemblers, program debugger/simulators, in-circuit emulators, and evaluation kits [7]. The ATmega16 block diagram shown in Figure 3.3.

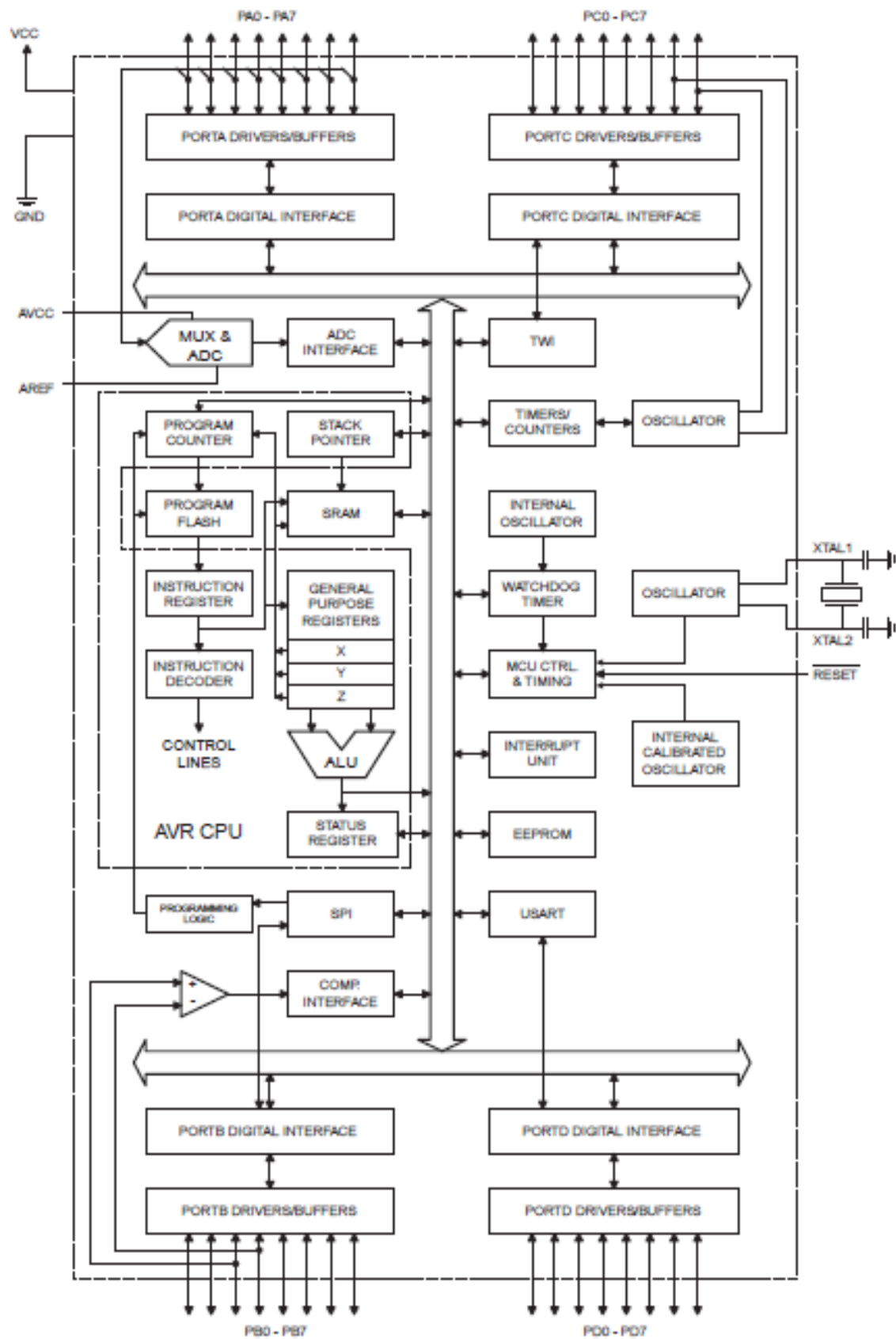


Figure 3.3: ATmega16 block diagram

3.2.2 ATmega16 Pin Descriptions

VCC

Digital supply voltage.

GND

Ground.

Port A (PA7..PA0)

Port A serves as the analog inputs to the A/D Converter. Port A also serves as an 8-bit bi-directional input/output port, if the A/D Converter is not used. Port pins can provide internal pull-up resistors (selected for each bit). The Port A output buffers have symmetrical drive characteristics with both high sink and source capability. When pins PA0 to PA7 are used as inputs and are externally pulled low, they will source current if the internal pull-up resistors are activated. The Port A pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port B (PB7..PB0)

Port B is an 8-bit bi-directional input/output port with internal pull-up resistors (selected for each bit). The Port B output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port B pins that are externally pulled low will source current if the pull-up resistors are activated. The Port B pins are tri-stated when a reset condition becomes active, even if the clock is not running. Port B also serves the functions of various special features of the ATmega16.

Port C (PC7..PC0)

Port C is an 8-bit bi-directional input/output port with internal pull-up resistors (selected for each bit). The Port C output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port C pins that are externally pulled low will source current if the pull-up resistors are activated. The Port C pins are tri-stated when a reset condition becomes active, even if the clock is not running. If the JTAG interface is enabled, the pull-up resistors on pins PC5(TDI), PC3(TMS) and PC2(TCK) will be activated even if a reset occurs. Port C also serves the functions of the JTAG interface and other special features of the ATmega16.

Port D (PD7..PD0)

Port D is an 8-bit bi-directional input/output port with internal pull-up resistors (selected for each bit). The Port D output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port D pins that are externally pulled low will source current if the pull-up resistors are activated. The Port D pins are tri-stated when a reset condition becomes active, even if the clock is not running. Port D also serves the functions of various special features of the ATmega16.

RESET

Reset Input. A low level on this pin for longer than the minimum pulse length will generate a reset, even if the clock is not running. Shorter pulses are not guaranteed to generate a reset.

XTAL1

Input to the inverting Oscillator amplifier and input to the internal clock operating circuit.

XTAL2

Output from the inverting Oscillator amplifier.

AVCC

AVCC is the supply voltage pin for Port A and the A/D Converter. It should be externally connected to VCC, even if the ADC is not used. If the ADC is used, it should be connected to VCC through a low-pass filter.

AREF

AREF is the analog reference pin for the A/DC [7].

3.3 Stepper Motor

A stepper motor shown in Figure 3.4, is an electric motor that rotates in discrete step increments. The movement of each step is precise and repeatable; therefore, the motor is position can be controlled precisely without any feedback mechanism, as long as the motor is carefully sized to the application. This type of control eliminates the need for expensive sensing and feedback devices such as optical encoders. The position is known simply by keeping track of the input step pulses. It is one of the most versatile forms of positioning systems. They are typically digitally controlled as part of an open loop system, and are simpler and more rugged than closed loop servo systems.

The essential property of the stepping motor is its ability to translate switched excitation changes into precisely defined increments of rotor position ('steps'). Stepping motors are categorised as doubly salient machines, which means that they have teeth of magnetically permeable material on both the stationary part (the stator) and the rotating part (the rotors). Most stepping motors can be identified as variations on the two basic types: variable-reluctance or hybrid. For the hybrid motor the main source of magnetic flux is a permanent magnet, and the currents flowing in one or more windings direct the flux along alternative paths.

There are two configurations, for the variable-reluctance stepping motor, but in both cases the magnetic field is produced solely by the winding currents [8].

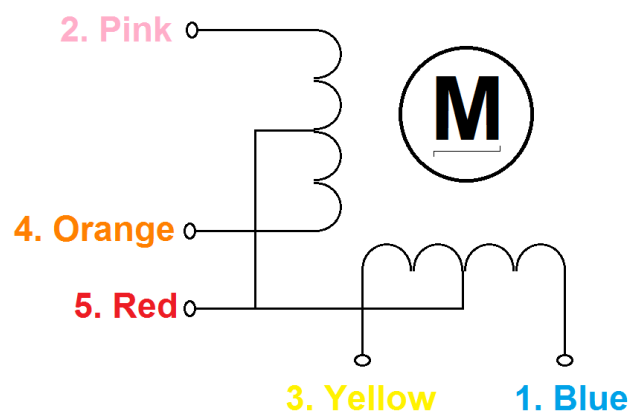


Figure 3.4: Stepper motor

3.4 ULN 2003A Driver

ULN2003 driver shown in Figure 3.5, is a high voltage and high current Darlington array Integrated Circuit (IC). It contains seven open collector Darlington pairs with common emitters. A Darlington pair is an arrangement of two bipolar transistors.

ULN2003 belongs to the family of ULN200X series of ICs. Different versions of this family interface to different logic families. ULN2003 is for 5V Transistor-Transistor Logic (TTL), Complementary Metal Oxide Semiconductor (CMOS) logic devices. These ICs are used when driving a wide range of loads and are used as relay drivers, display drivers, line drivers etc. ULN2003 is also commonly used while driving stepper motors. refer stepper motor interfacing using ULN2003.

Each channel or Darlington pair in ULN2003 is rated at 500mA and can withstand peak current of 600mA. The inputs and outputs are provided opposite to each other in the pin layout.

Each driver also contains a suppression diode to dissipate voltage spikes while driving inductive loads [9].

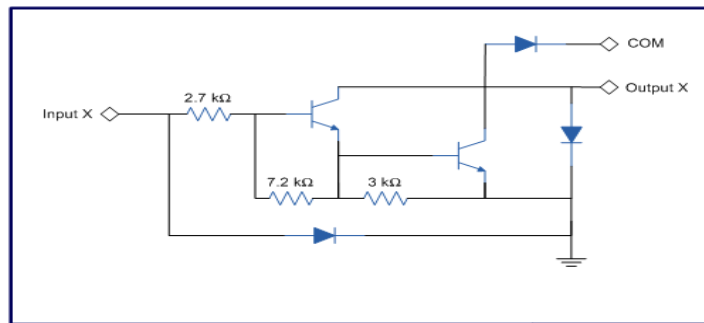


Figure 3.5: ULN2003A driver

3.5 Break Beam Sensors

These types of active IR sensor have emitter and receiver placed in such a way that the IR emitted by the emitter falls directly in to the receiver. During the operation as shown in Figure 3.6, IR beam is emitted continuously towards the receiver. The flow of IR can be interrupted by placing an object between the emitter and receiver. If the IR is transmitted but altered then receiver generates output based on the change in radiation. Similarly if the radiation is completely blocked the receiver can detect it and provide the desired output. For example: let us consider a break beam sensor and a conveyer belt. When an opaque object interrupts the IR flow the receiver does not receive any signal thus the conveyer belt stops [6].

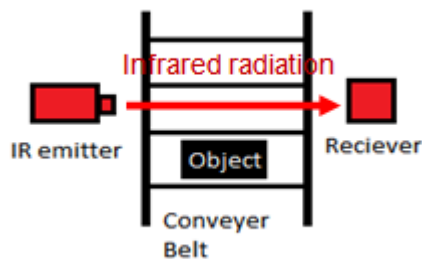


Figure.3.6: Break beam sensor

For connect break beam sensors in the electrical circuit we need driveway as shown in Figure 3.7.

3.6 Seven Segment Display

In the common cathode display, all the Anode connections of the LED segments are joined

together to logic “1” or ground. The individual segments are illuminated by application of a “LOW”, or logic “0” signal via a current limiting resistor to forward bias the individual anode terminals (a-g).

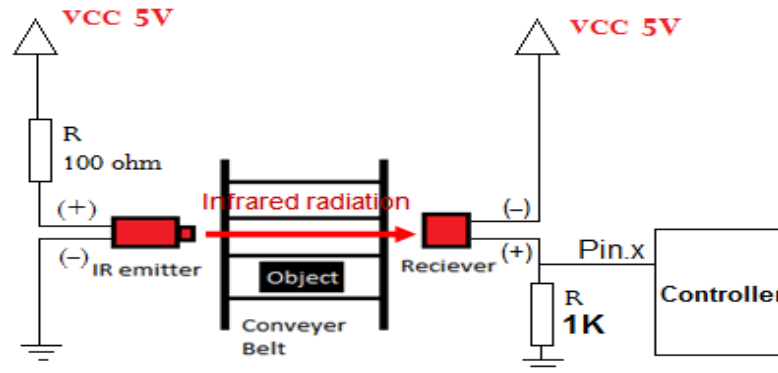


Figure.3.7: Electrical circuit break beam sensor

3.7 Push Button Switch

The system designed to accommodate six cars. The key pad used for the system contains six keys to be restored by the Special number for the car, as well as key to perform this operation (key to confirm).

3.8 Power Supply

ATmega16 can operate with a power supply +2.7 to +5.5 usually [10]. Charger unit consists of the following units [11]:

3.8.1 Step down transformer

It is used to step down the main supply voltage by using step down transformer. It consists of primary and secondary coils. The output from the secondary coil is also Alternating Current (AC) voltage output. Therefore, we have to convert the easy Direct Current (DC) voltage by using rectifier unit.

3.8.2 Rectifier unit

We have to convert AC voltage to DC using rectifier. Bridge rectifier is used. This output voltage often rectifier is in rippled form, so we have to remove ripples from DC voltage.

3.8.3 Input filter

Capacitor acts as filter. The principle of the capacitor is charging and discharging. It charges in the positive half cycle of the AC voltage and it will discharge in the negative half cycle. So this allows only AC voltage and does not allow the DC voltage. This filter is fixed before the regulator.

3.8.4 Regulator unit

Regulator regulates the output voltage constant depends upon the regulator. It is classified as follows:

Positive regulator L7805CV

Input pin.

Ground pin.

Output pin.

It regulates the positive voltage.

Negative regulator 7905

Ground pin.

input pin.

output pin.

It regulate the negative voltage.

3.8.5 Output filter

Capacitor acts as filter. The principle of the capacitor is charging and discharging. It charges in positive half cycle of the AC voltage an it will discharge in negative half cycle. So it allow only allows AC voltage and does not allow the DC voltage. This filterer fixed after the regulator.

3.9 System Simulation

Computer simulations have become a useful part of mathematical modelling of many natural systems to observe their behaviour. It allows the engineer to test the design before it is built in the real situation.

As mentioned earlier, the simulations for this research were performed in PROTEUS program

shown in Figure 3.8. These software applications are widely used in control engineering for both simulation and design.

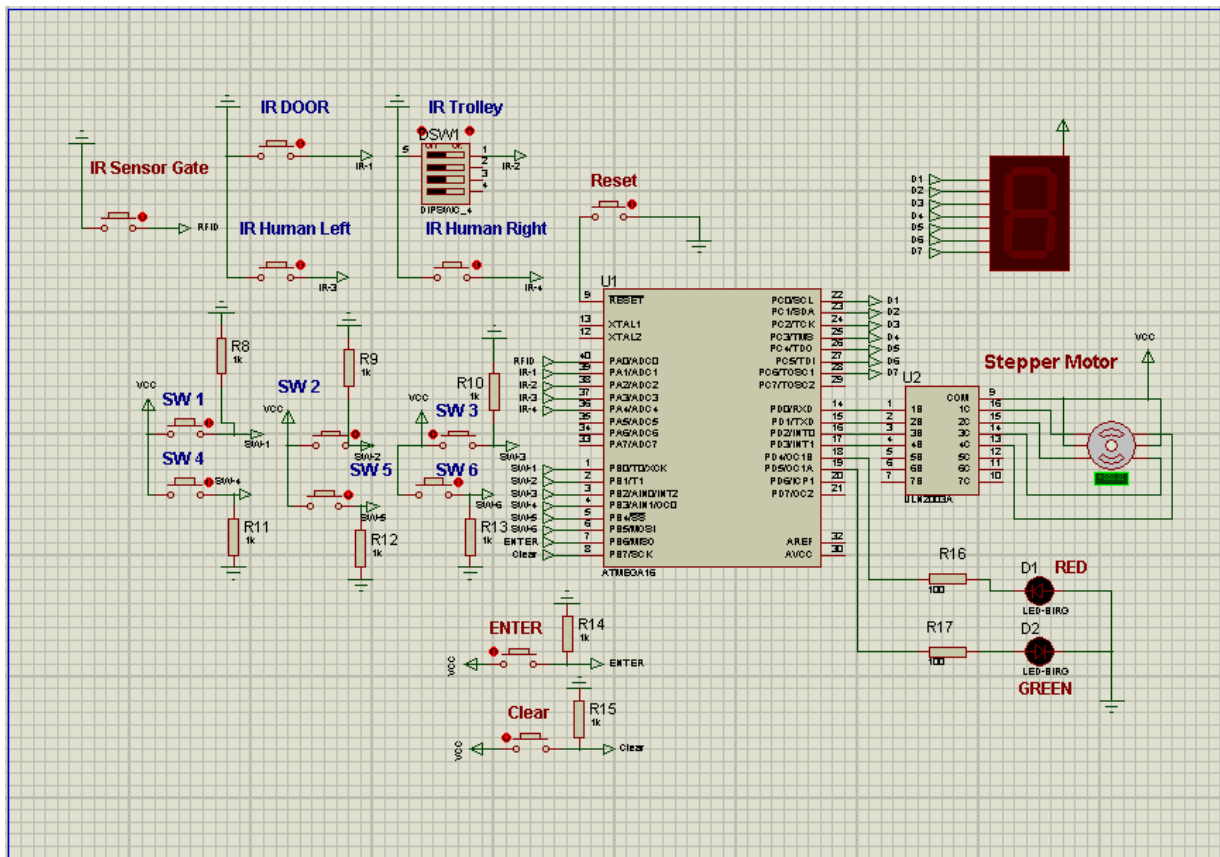


Figure 3.8: The main circuit design

3.10 Working Principle of Circuit

The control circuit contains five IR sensors that responsible to park the car and in order to complete this process there are some conditions series that should be fulfilled, and these conditions are represented as:

The activation of IR sensor Gate one indicates that there is an arrived car, also show whether or not parking is full by turning on lamps green for empty or red for full.

The activation of IR Door indicates to continuation of car entry.

Activation of IR Trolley indicates to occupation of trolley.

As for the IR Human Left or Right, they indicate that driver got off the car.

Once IR Door has been activated for the second time the stepper motor will start running to park the car and to provides a new empty trolley and display number of trolley in the 7-

segment display.

In addition, the control circuit contains eight keys in order to retrieve the car, where six of which refers to a number trolley, and the keys seven and eight refer to confirmation and clear respectively.

While the circuit is running, the 7-segment display initial value (number one) to indicate the states of trolley in the parking as shown in Figure 3.9.

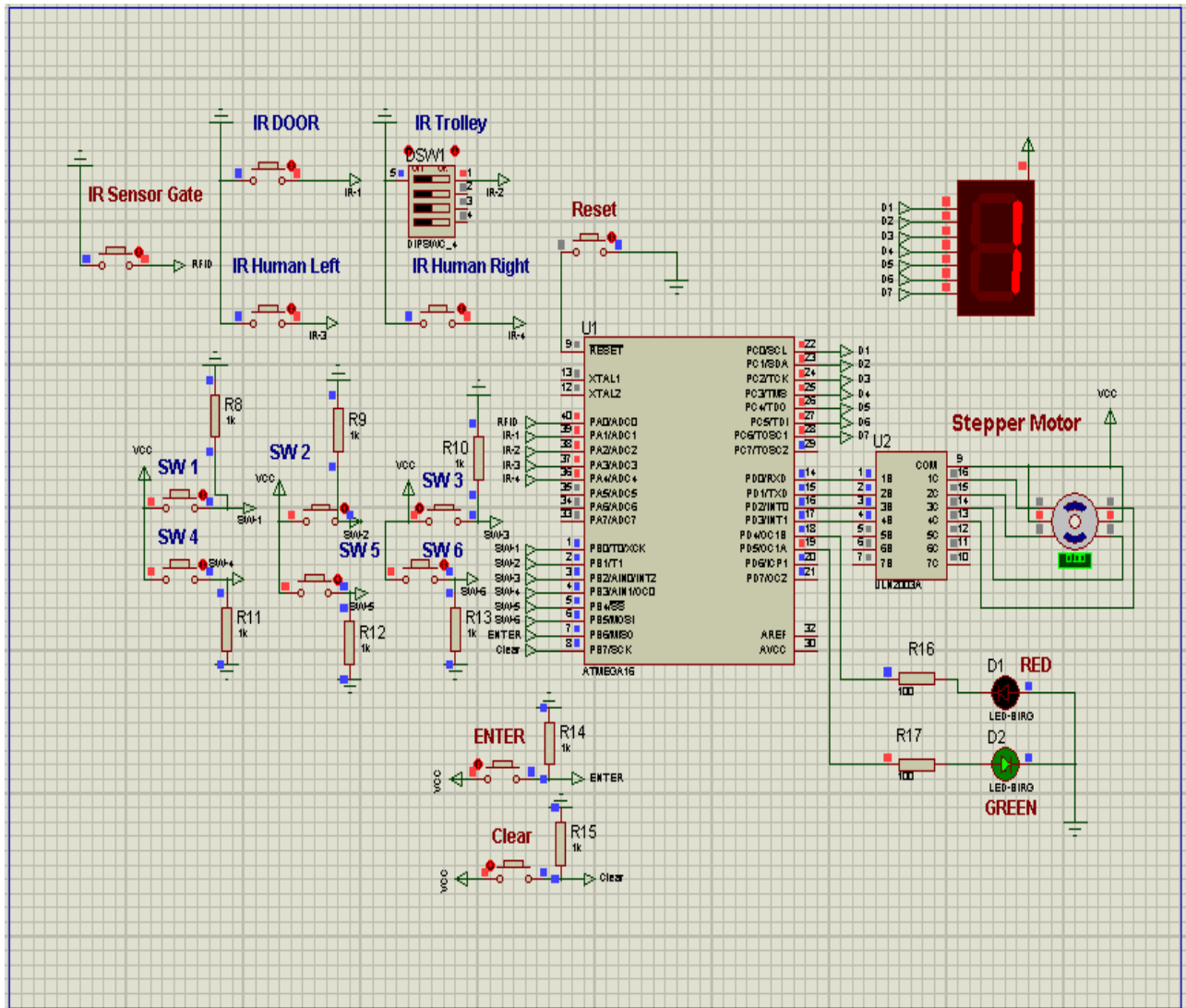


Figure 3.9: The main circuit design after running

After apply all conditions respectively the stepper motor will start running to park the car and to provides a new empty trolley and display number of trolley in the seven-segment display as shown in Figure 3.10, and we can repeat this process until the parking becomes full. In addition to this, we can retrieve the car by press any push button switch (sw1 to sw6) then press ENTER

switch for confirmation.

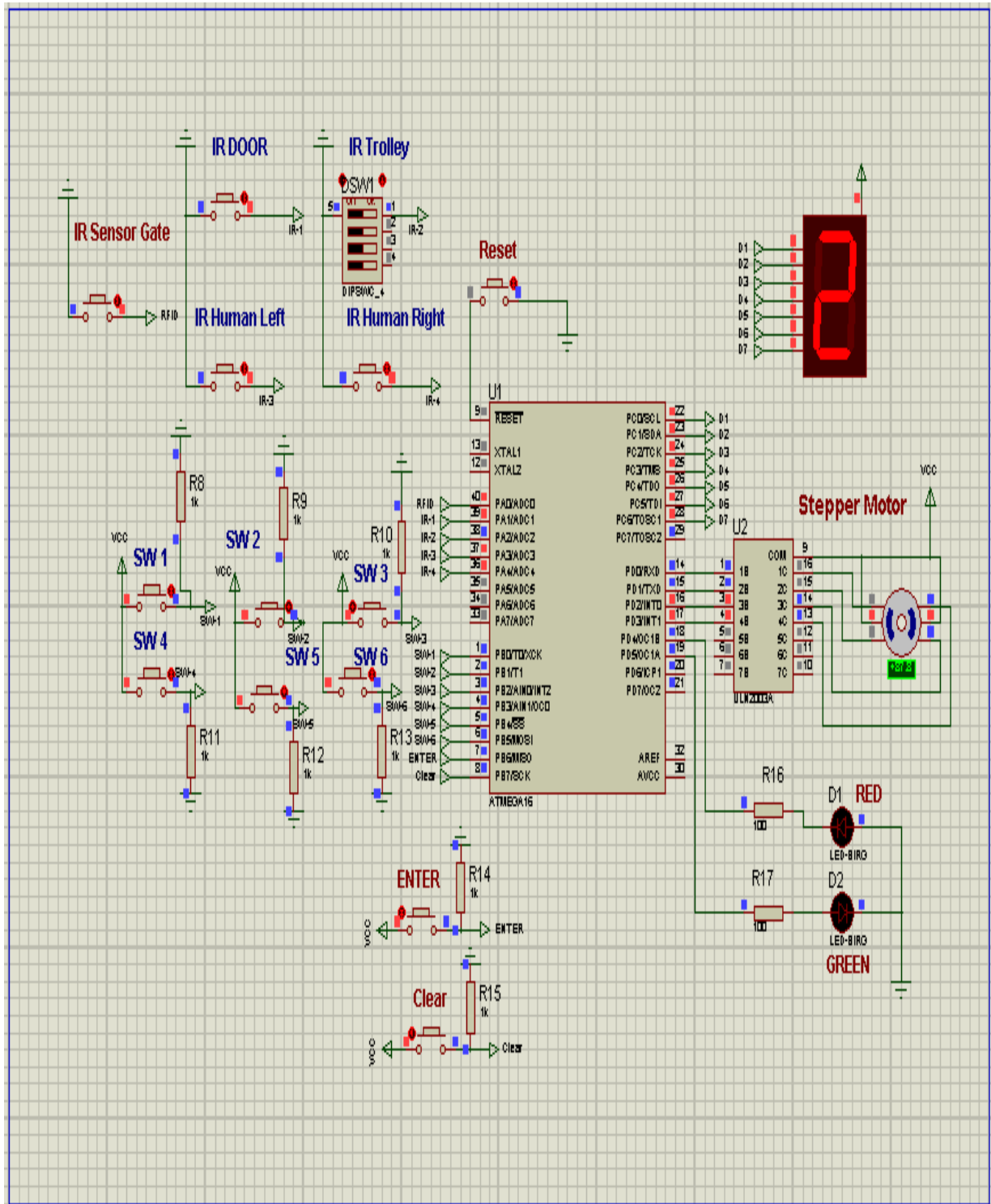


Figure 3.10: The main circuit design after park car number one

3.11 System Flowchart

A flowchart is a type of diagram that represents an algorithm, workflow or process, showing

the steps as boxes of various kinds, and their order by connecting them with arrows. This diagrammatic representation illustrates a solution model to a given problem. Flowcharts are used in analysing, designing, documenting or managing a process or program in various fields, in this thesis the flowchart illustrates the principle of system code as shown in Figure 3.11.

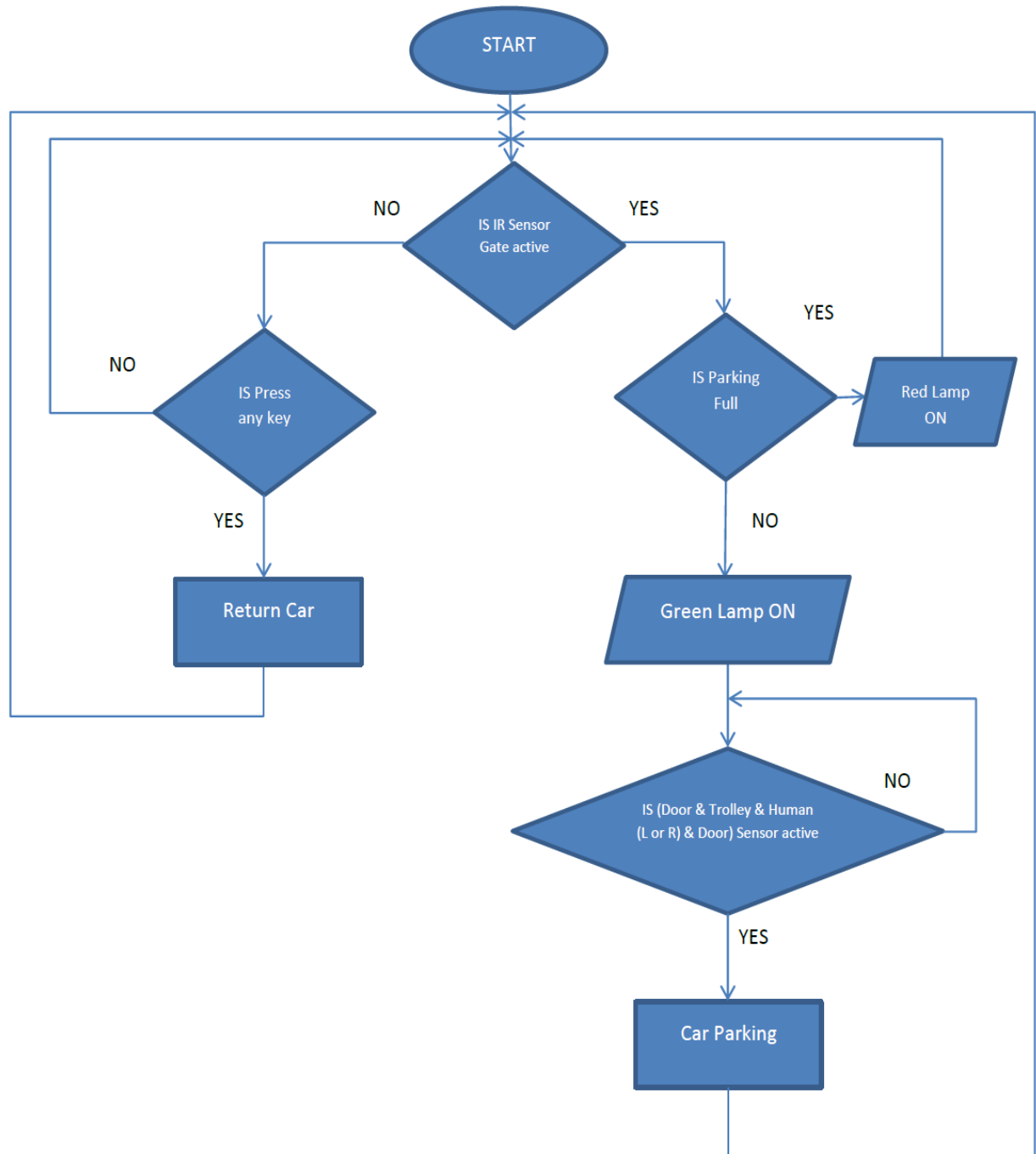


Figure 3.11: System flowchart

3.12 System Code

The Code has been written in C language using program AVR Studio to program ATmega16, after program is completed it has been converted to Hex file to install it in ATmega16. Where was the use of primary and non-complex instructions in writing program such as IF-statement instruction and While-loop instruction to implement the control conditions. The illustrative code in Appendix A.

CHAPTER FOUR

System Implementation and Testing

4.1 System Implementation

The control circuit consists of different devices represented in ATmega16, stepper motor (5V), ULN2003A driver, IR sensor transmitter and receiver, 7-segment display, push button switch, resistors and DC supply 5V and 2A.

For the implementation of the control circuit correctly follow the following steps:

4.1.1 Step one

Programming of ATmega16 and installation in the port as shown in Figure 4.1.



Figure 4.1: Installation atmega16

4.1.2 Step two

Installation of ULN2003A driver in the port then connecting it with ATmega16 (port D pin0 with 1B, pin1 with 2B, pin2 with 3B and pin3 with 4B respectively) also connecting ULN2003A (com, 1c, 2c, 3 and 4c respectively) with stepper motor as shown in Figure 4.2.

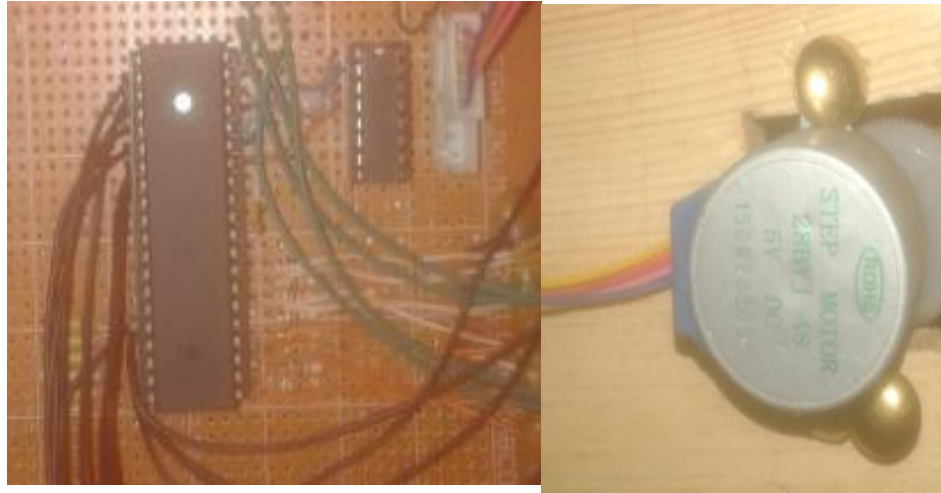


Figure 4.2: Installation ULN2003A and stepper motor

4.1.3 Step three

Installation of 7-segment displays in the port then connecting it with ATmega16 (port C pin0 ~ pin6), also connecting the LEDs (Green and Red) with ATmega16 (port D pin4 and pin5) as shown in Figure 4.3.

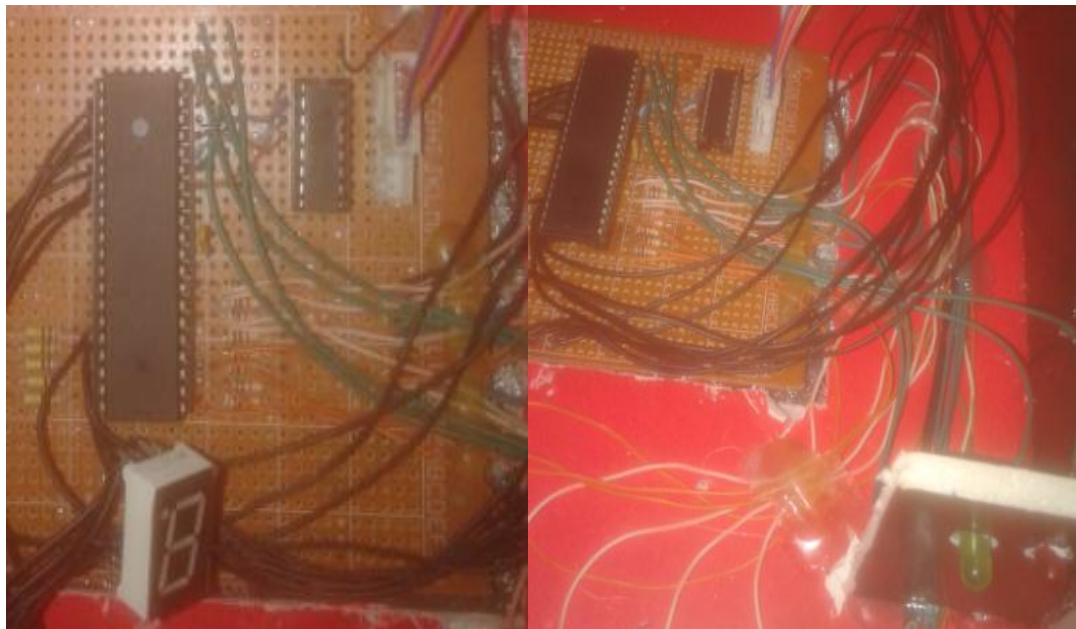


Figure 4.3: Installation 7-segment display and LEDs

4.1.4 Step four

Connecting five IR transmitters with 5V and resistances (100 ohm) then connecting five IR receivers with ATmega16 (port A pin0 ~ pin4) and resistors (1K), also connecting eight push button switches with ATmega16 (port B pin0 ~ pin7) as shown in Figure 4.4.



Figure 4.4: Connecting IR sensors and push button switches

4.1.5 Step five

Connecting the power DC supply 5V and 2A for the electrical circuit as shown in Figure 4.5.



Figure 4.5: Connecting the power supply

4.1.6 Step six

Finally, the installation of an electrical circuit in the final model as shown in Figure 4.6.



Figure 4.6: The final model

4.2 System Testing

After the electrical circuit is completed, it has been installed in small model to simulate the real parking and to test the electrical circuit and assure its efficiency and apply the possible conditions and it has been illustrated in the following steps:

4.2.1 Step one (entry of first car)

Every car that enters the parking must fulfill the conditions that mentioned previously so the system could deals with it and provide new free trolley and take in consideration the nearest free trolley to save time. The Figure 4.7 demonstrates the five levels of car entry.

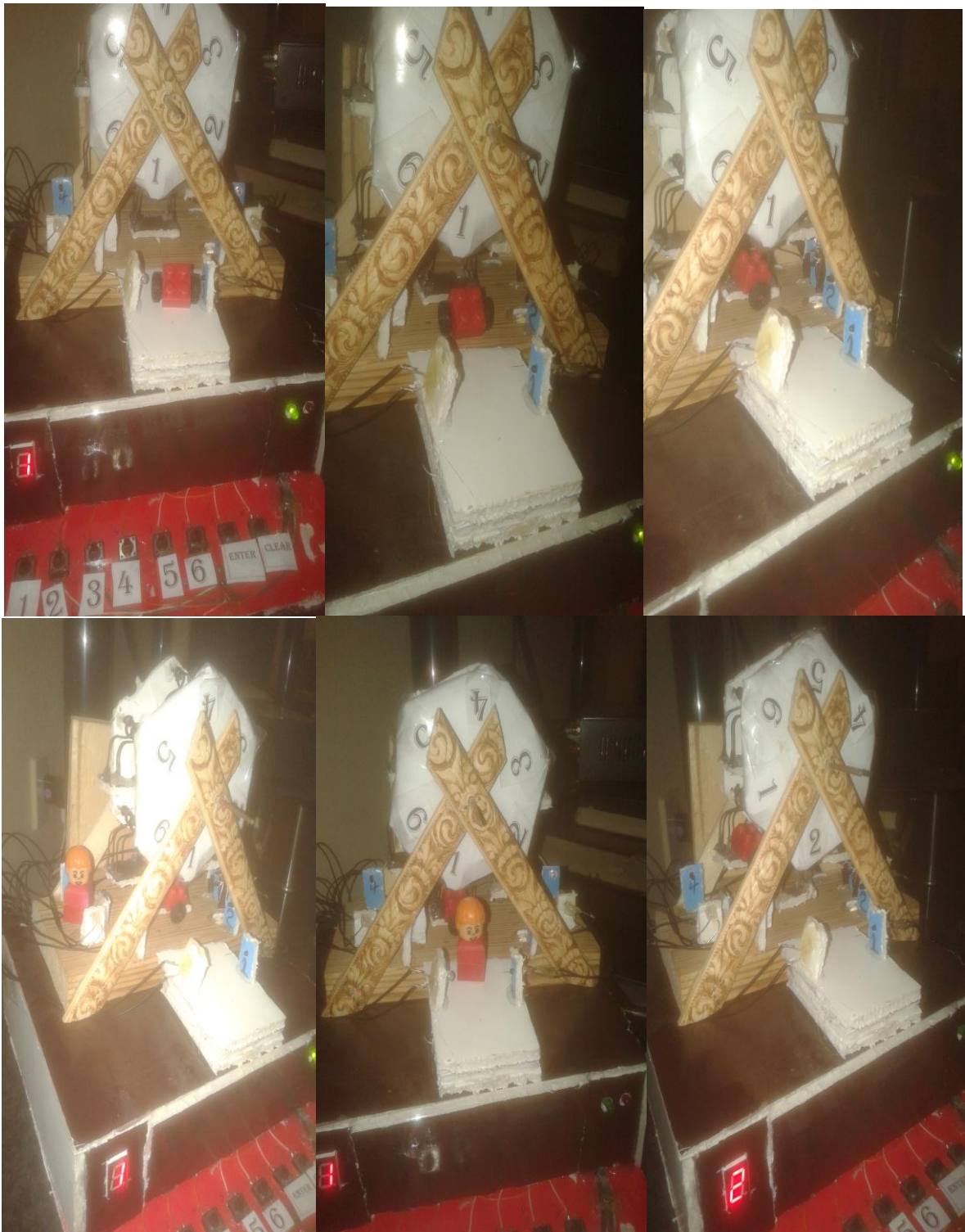


Figure 4.7: Entry of first car

4.2.2 Step two

Status of Full parking and the entry of new car where the system turns on a red LED (or green LED for the empty) and to demonstrate the status of car parking as shown in Figure 4.8.



Figure 4.8: Status of full parking

4.2.3 Step three

Retrieving a car number 4 from parking (for illustration) by pressing the key that assigned to it where system retrieves it from nearest direction to save time. This is shown in Figure 4.9.



Figure 4.9: Retrieving a car number 4

CHAPTER FIVE

Conclusion and Recommendations

5.1 Conclusion

The model has been designed and the software as well as the control circuit has been implemented successfully. It demonstrates the working of the planned rotary automated car parking system. ATmega16 is the most suitable controller whence the cost, ease of use, reliable in the work and available in the markets.

5.2 Recommendations

The proposed rotary automated car parking system will be provided with safety devices such as firefighting system.

The platforms can also be equipped with safety sensors guiding the movement of vehicles in the platforms.

Add of a rotary platform to route the car to street after going out of parking.

Add of Radio Frequency IDentification (RFID) in order to allow specific cars to login into parking by using database of the server.

References

- [1] M Childs, "Parking Spaces; A Design, Implementation and Use Manual for Architects Planners and Engineers" McGraw-Hill, 2009.
- [2] McDonald S. Shannon, "Cars Parking and Sustainability", The Transportation Research Forum, 2012.
- [3] Beebe Richard, "Automated Parking", Status in the United States, 2001.
- [4] H Hwang, S Lee, "Expected Service Time Model for A Rotary Parking System", Computer Industrial Engineering, 1998.
- [5] Steven F. Barrett & Daniel J. Pack, "Atmel AVR Microcontroller Primer: Programming and Interfacing", 2008.
- [6] C.desilva, "Control Sensors and Actuators", prentice hall, 1989.
- [7] Atmel Corporation, "Datasheetatmega16", San Jose USA, 2010.
- [8] Paul Acarnley,"Stepping Motors a guide to theory and practice", 4th Edition, IEE, 2002.
- [9] STMicroelectronics, "DatasheetULN2003A", Italy, 2002.
- [10] Dogn Ibrahim, "Microcontroller Based Temperature Monitoring and control", Elsevier Science & Technology Books, 2002.
- [11] C.B. Gray, "Electrical Machines and Drive System", 1989.

Appendix A

Code:

```
#include <avr/io.h>

#include <util/delay.h>

unsigned char T1=0x00;
unsigned char T6=0x00;
unsigned char RFID=0x00;
unsigned char Car1=0x00;
unsigned char SW_1=0x00;
unsigned char SW_2=0x00;
unsigned char SW_3=0x00;
unsigned char SW_4=0x00;
unsigned char SW_5=0x00;
unsigned char SW_6=0x00;
unsigned char SW_11=0x00;
unsigned char T2=0x00;
unsigned char T3=0x00;
unsigned char T4=0x00;
unsigned char T5=0x00;
unsigned char SW_22=0x00;
unsigned char SW_33=0x00;
unsigned char SW_44=0x00;
unsigned char SW_55=0x00;
unsigned char SW_66=0x00;
```

```

unsigned char Error=0x00;

unsigned char j = 0;

unsigned char Human = 0x00;

unsigned char IR_Trolley = 0x00;

unsigned char IR_DOOR = 0x00;


void main (void)

{

    DDRA = 0x00; /* set PORTB for Input*/

    /*PORTA = 0x1F; /*in Simulation*/

    DDRB = 0x00; /* set PORTD for Input*/

    DDRC = 0xFF; /* set PORTB for output*/

    DDRD = 0xFF; /* set PORTD for output*/

    PORTC = 0b01111001; /* num 1 */

    while (1) {

if(PINA == 0x1E | PINA == 0x1A) /* Test RFID sensor */

    { if (T1==0x01 & T2==0x01 & T3==0x01 & T4==0x01& T5==0x01 & T6==0x01)

        {      PORTD = 0x10; } else {

            PORTD = 0x20;

            RFID = 0x01; }}

if (RFID == 0x01 & PINA == 0x1D)

    {      IR_DOOR = 0x01;}

    if (PINA == 0x1B )

        {      IR_Trolley = 0x01;    }

if (IR_DOOR == 0x01 & IR_Trolley == 0x01)

    {

if(PINA == 0x13 | PINA == 0x0B)/*Test Human sensor Left or Right */

```

```

    {      Human = 0x01;}}

    if(Human == 0x01)

    {      Human = 0x00;

    RFID = 0x00;

    IR_DOOR = 0x00; while(PINA != 0x19); /*Test Door sensor */

    IR_Trolley == 0x00;

/***** Test Trolley1 *****/

if (Car1 == 0x00)/* Test num of trolley in Door Parking */

    {

if ( T2 == 0x00) /* To know trolley num 2 empty */

    {      One_cycle();

    Forward();

PORTC = 0b00100100; /* 7seg num 2 */

    Car1 = 0x01;

    T1 = 0x01; } else {

if ( T6 == 0x00) /*To know trolley num 6 empty */

{ One_cycle(); /* For go to trolley 2 */

    Backward();

PORTC = 0b00000010; /* num 6 */

    Car1 = 0x05;

    T1 = 0x01; } else {

if ( T3 == 0x00)

    {      Two_cycle();

    Forward();

PORTC = 0b00100100; /* 7seg num 2 */

    _delay_ms(1000);

    _delay_ms(1000);

    _delay_ms(1000);

```

```

PORTC = 0b00110000; /* num 3 */

    Car1 = 0x02;

    T1 = 0x01;    } else {

if ( T5 == 0x00)

{    Two_cycle();

    Backward();

PORTC = 0b00000010; /* num 6 */

    _delay_ms(1000);

    _delay_ms(1000);

    _delay_ms(1000);

PORTC = 0b00010010; /* num 5 */

    Car1 = 0x04;

    T1 = 0x01; } else {

if ( T4 == 0x00)

{    Three_cycle();

    Forward();

PORTC = 0b00100100; /*num 2 */

    _delay_ms(1000);

    _delay_ms(1000);

    _delay_ms(1000);

PORTC = 0b00110000; /* num 3 */

    _delay_ms(1000);

    _delay_ms(1000);

    _delay_ms(1000);

PORTC = 0b00011001; /* num 4 */

    Car1 = 0x03;

    T1 = 0x01;    } else {

if ( T2 == 0x01)

```

```

    {
        One_cycle();
        Forward();
PORTC = 0b00100100; /* 7seg num 2 */

        Car1 = 0x01;
        T1 = 0x01; } } } } } else {
/***** Test Trolley 2 *****/

        if(Car1 == 0x01) /* in Door Parking trolley 2 */
        {
            if ( T3 == 0x00)
            {
                One_cycle();
                Forward();
PORTC = 0b00110000; /* num 3 */

                Car1 = 0x02;
                T2 = 0x01; } else {
                    if ( T1 == 0x00)
                    {
                        One_cycle();
                        Backward();
PORTC = 0b01111001; /* num 1 */

                        Car1 = 0x00;
                        T2 = 0x01; } else {
                            if ( T4 == 0x00)
                            {
                                Two_cycle();
                                Forward();
PORTC = 0b00110000; /* num 3 */

                                _delay_ms(1000);
                                _delay_ms(1000);
                                _delay_ms(1000);
PORTC = 0b00011001; /* num 4 */

```



```

        Car1 = 0x03;

        T2 = 0x01;    } else {
if ( T6 == 0x00)
{
    Two_cycle();

    Backward();

PORTC = 0b01111001; /* num 1 */

    _delay_ms(1000);

    _delay_ms(1000);

    _delay_ms(1000);

PORTC = 0b00000010; /* num 6 */

    Car1 = 0x05;

    T2 = 0x01;    } else {
if ( T5 == 0x00)
{
    Three_cycle();

    Forward();

PORTC = 0b00110000; /* num 3 */

    _delay_ms(1000);

    _delay_ms(1000);

    _delay_ms(1000);

PORTC = 0b00011001; /* num 4 */

    _delay_ms(1000);

    _delay_ms(1000);

    _delay_ms(1000);

PORTC = 0b00010010; /* num 5 */

    Car1 = 0x04;

    T2 = 0x01;    } else {
if ( T3 == 0x01)
{
    One_cycle();

```

```

        Forward();

PORTC = 0b00110000; /* num 3 */

        Car1 = 0x02;

        T2 = 0x01;    }   } } } } } else {

/***** Test Trolley 3 *****/

        if (Car1 == 0x02)

        {
            if ( T4 == 0x00)

            {
                One_cycle();

                Forward();

PORTC = 0b00011001; /* num 4 */

                Car1 = 0x03;

                T3 = 0x01;    } else {

if ( T2 == 0x00)

            {
                One_cycle();

                Backward();

PORTC = 0b00100100; /* num 2 */

                Car1 = 0x01;

                T3 = 0x01;    } else {

if ( T5 == 0x00)

            {
                Two_cycle();

                Forward();

PORTC = 0b00011001; /* num 4 */

                _delay_ms(1000);

                _delay_ms(1000);

                _delay_ms(1000);

PORTC = 0b00010010; /* num 5 */

                Car1 = 0x04;

                T3 = 0x01;    } else {

```

```

if ( T1 == 0x00)
{
    Two_cycle();

    Backward();

    PORTC = 0b00100100; /* num 2 */

    _delay_ms(1000);

    _delay_ms(1000);

    _delay_ms(1000);

    PORTC = 0b01111001; /* num 1 */

    Car1 = 0x00;

    T3 = 0x01;    } else {
if ( T6 == 0x00)
{
    Three_cycle();

    Forward();

    PORTC = 0b00011001; /* num 4 */

    _delay_ms(1000);

    _delay_ms(1000);

    _delay_ms(1000);

    _delay_ms(1000);

    PORTC = 0b00010010; /* num 5 */

    _delay_ms(1000);

    _delay_ms(1000);

    _delay_ms(1000);

    _delay_ms(1000);

    PORTC = 0b00000010; /* num 6 */

    Car1 = 0x05;

    T3 = 0x01;    } else {
if ( T4 == 0x01)
{
    One_cycle();

```

```

        Forward();

PORTC = 0b00011001; /* num 4 */

        Car1 = 0x03;

        T3 = 0x01;    }   } } } } else {

/***** Test Trolley 4 *****/

        if (Car1 == 0x03)

        {
            if ( T5 == 0x00)

            {
                One_cycle();

                Forward();
PORTC = 0b00010010; /* num 5 */

                Car1 = 0x04;

                T4 = 0x01;    } else {

if ( T3 == 0x00)

{
    One_cycle();

    Backward();

PORTC = 0b00110000; /* num 3 */

    Car1 = 0x02;

    T4 = 0x01;    } else {

if ( T6 == 0x00)

{

    Two_cycle();

    Forward();

PORTC = 0b00010010; /* num 5 */

        _delay_ms(1000);

        _delay_ms(1000);

        _delay_ms(1000);

PORTC = 0b00000010; /* num 6 */

        Car1 = 0x05;

```

```

        T4 = 0x01;    } else {
if ( T2 == 0x00)
{
    Two_cycle();

    Backward();

PORTC = 0b00110000; /* num 3 */

    _delay_ms(1000);

    _delay_ms(1000);

    _delay_ms(1000);

PORTC = 0b00100100; /* num 2 */

    Car1 = 0x01;

    T4 = 0x01;    } else {
if ( T1 == 0x00)
{
    Three_cycle();

    Forward();

PORTC = 0b00010010; /* num 5 */

    _delay_ms(1000);

    _delay_ms(1000);

    _delay_ms(1000);

PORTC = 0b00000010; /* num 6 */

    _delay_ms(1000);

    _delay_ms(1000);

    _delay_ms(1000);

PORTC = 0b01111001; /* num 1 */

    Car1 = 0x00;

    T4 = 0x01;    } else {
if ( T5 == 0x01)
{
    One_cycle();

    Forward();

```

```

PORTC = 0b00010010; /* num 5 */

    Car1 = 0x04;

    T4 = 0x01;    }   } } } } } else {

/***** Test Trolley 5 *****/

        if (Car1 == 0x04)

        {
            if ( T6 == 0x00)

            {
                One_cycle();

                Forward();
PORTC = 0b00000010; /* num 6 */

                Car1 = 0x05;

                T5 = 0x01;    } else {

if ( T4 == 0x00)

            {
                One_cycle();

                Backward();
PORTC = 0b00011001; /* num 4 */

                Car1 = 0x03;

                T5 = 0x01;    } else {

if ( T1 == 0x00)

            {
                Two_cycle();

                Forward();
PORTC = 0b00000010; /* num 6 */

                _delay_ms(1000);

                _delay_ms(1000);

                _delay_ms(1000);
PORTC = 0b01111001; /* num 1 */

                Car1 = 0x00;

                T5 = 0x01;    } else {

if ( T3 == 0x00)

```

```

        {      Two_cycle();

                Backward();

PORTC = 0b00011001; /* num 4 */

                _delay_ms(1000);

                _delay_ms(1000);

                _delay_ms(1000);

PORTC = 0b00110000; /* num 3 */

                Car1 = 0x02;

                T5 = 0x01;    } else {

if ( T2 == 0x00)

        {      Three_cycle();

                Forward();

PORTC = 0b00000010; /* num 6 */

                _delay_ms(1000);

                _delay_ms(1000);

                _delay_ms(1000);

PORTC = 0b01111001; /* num 1 */

                _delay_ms(1000);

                _delay_ms(1000);

                _delay_ms(1000);

PORTC = 0b00100100; /* num 2 */

                Car1 = 0x01;

                T5 = 0x01;    } else {

if ( T6 == 0x01)

        {      One_cycle();

                Forward();

PORTC = 0b00000010; /* num 6 */

                Car1 = 0x05;

```

```

        T5 = 0x01;    }   } } } } } else {

/***** Test Trolley 6 *****/

        if (Car1 == 0x05)

        {
            if ( T1 == 0x00)

            {
                One_cycle();

                Forward();

PORTC = 0b01111001; /* num 1 */

                Car1 = 0x00;

                T6 = 0x01;    } else {

if ( T5 == 0x00)

            {
                One_cycle();

                Backward();

PORTC = 0b00010010; /* num 5 */

                Car1 = 0x04;

                T6 = 0x01;    } else {

if ( T2 == 0x00)

            {
                Two_cycle();

                Forward();

PORTC = 0b01111001; /* num 1 */

                _delay_ms(1000);

                _delay_ms(1000);

                _delay_ms(1000);

PORTC = 0b00100100; /* num 2 */

                Car1 = 0x01;

                T6 = 0x01;    } else {

if ( T4 == 0x00)

            {
                Two_cycle();

                Backward();

```


14

```

if(PINB == 0x80) /* Test SW clear */

{PORTC = 0b01111111; }/* Clear 7 segment */

/***** Test SW 1 *****/

    if(PINB == 0x01) /* Test SW 1 */

    { SW_1 = 0x01;

PORTC = 0b01111001; }/* num 1 */

if (SW_1 == 0x01 & PINB == 0x40)

    {      SW_1 = 0x00;

      SW_11 = 0x01;      }

    if (SW_11 == 0x01)

    { if(Error == 0x02)/* Error 2 */

      {      SW_22 = 0x00;

      T2 = 0x01;      } else {

if(Error == 0x03)/* Error 3*/

      {      SW_33 = 0x00;

      T3 = 0x01;      } else {

if(Error == 0x04)/*Error 4*/

      {      SW_44 = 0x00;

      T4 = 0x01;      } else {

if(Error == 0x05)/* Error 5*/

      {      SW_55 = 0x00;

      T5 = 0x01;      } else {

if(Error == 0x06)/*Error 6*/

      {      SW_66 = 0x00;

      T6 = 0x01;      } } } }

      if (Car1 == 0x01)

      {      One_cycle();

Backward();

```

```

PORTC = 0b01111001; /* num 1 */

    Car1 = 0x00; } else {

    if (Car1 == 0x02)

    {      Two_cycle();

    Backward();

PORTC = 0b00100100; /* num 2 */

_delay_ms(1000);

_delay_ms(1000);

_delay_ms(1000);

PORTC = 0b01111001; /* num 1 */

    Car1 = 0x00; } else {

    if (Car1 == 0x03)

    {      Three_cycle();

    Backward();

PORTC = 0b00110000; /* num 3 */

_delay_ms(1000);

_delay_ms(1000);

_delay_ms(1000);

PORTC = 0b00100100; /* num 2 */

_delay_ms(1000);

_delay_ms(1000);

_delay_ms(1000);

PORTC = 0b01111001; /* num 1 */

    Car1 = 0x00;      } else {

    if (Car1 == 0x04)

    {      Two_cycle();

    Forward();

PORTC = 0b00000010; /* num 6 */

```

```

_delay_ms(1000);

_delay_ms(1000);

_delay_ms(1000);

PORTC = 0b01111001; /* num 1 */

    Car1 = 0x00; } else {

if (Car1 == 0x05)

    {    One_cycle();

    Forward();

PORTC = 0b01111001; /* num 1 */

    Car1 = 0x00; } } } }

if(PINA == 0x1B)

{Error = 0x01; /*Error 1;*/

    Car1 = 0x00; } else {

    Error = 0x00; /*Error 0;*/

    SW_11 = 0x00;

    T1 = 0x00;

    Car1 = 0x00; } }

/***** Test SW 2 *****/

    if(PINB == 0x02) /* Test SW 2 */

    { SW_2 = 0x01;

PORTC = 0b00100100; } /* num 2 */

if (SW_2 == 0x01 & PINB == 0x40)

    {    SW_2 = 0x00;

        SW_22 = 0x01; }

    if (SW_22 == 0x01)

    {

if(Error == 0x01) /*Error == 0x01)*/

    {    SW_11 = 0x00;

```

```

        T1 = 0x01;    } else {
if(Error == 0x03)/*Error == 0x03)*/
    {
        SW_33 = 0x00;
        T3 = 0x01;    } else {
if(Error == 0x04)/*Error == 0x04)*/
    {
        SW_44 = 0x00;
        T4 = 0x01;    } else {
if(Error == 0x05)/* Error 5*/
    {
        SW_55 = 0x00;
        T5 = 0x01;    } else {
if(Error == 0x06)/*Error 6*/
    {
        SW_66 = 0x00;
        T6 = 0x01;    } } } }
        if (Car1 == 0x00)
        {
            One_cycle();
        }
        Forward();
PORTC = 0b00100100; /* num 2 */
        Car1 = 0x01; } else {
        if (Car1 == 0x02)
        {
            One_cycle();
        }
        Backward();
        PORTC = 0b00100100; /* num 2 */
        Car1 = 0x01; } else {
        if (Car1 == 0x03)
        {
            Two_cycle();
        }
        Backward();
PORTC = 0b00110000; /* num 3 */
        _delay_ms(1000);
        _delay_ms(1000);

```

```

        _delay_ms(1000);
PORTC = 0b00100100; /* num 2 */

        Car1 = 0x01; } else {

        if (Car1 == 0x04)

        {      Three_cycle();

        Backward();

PORTC = 0b00011001; /* num 4 */

        _delay_ms(1000);

        _delay_ms(1000);

        _delay_ms(1000);

PORTC = 0b00110000; /* num 3 */

        _delay_ms(1000);

        _delay_ms(1000);

        _delay_ms(1000);

PORTC = 0b00100100; /* num 2 */

        Car1 = 0x01;      } else {

        if (Car1 == 0x05)

        {      Two_cycle();

        Forward();

PORTC = 0b01111001; /* num 1 */

        _delay_ms(1000);

        _delay_ms(1000);

        _delay_ms(1000);

PORTC = 0b00100100; /* num 2 */

        Car1 = 0x01; } } } }

if(PINA == 0x1B)

        {Error = 0x02; /*Error 2;*/

        Car1 = 0x01; } else {

```

```

        Error = 0x00; /*Error 0;*/

        SW_22 = 0x00;

        T2 = 0x00;

        Car1 = 0x01; }    }

/***** Test SW 3 *****/

        if(PINB == 0x04) /* Test SW 3 */

        { SW_3 = 0x01;

PORTC = 0b00110000;} /* num 3 */

        if (SW_3 == 0x01 & PINB == 0x40)

                {      SW_3 = 0x00;

                SW_33 = 0x01;      }

                if (SW_33 == 0x01)

{ if(Error == 0x01)/*Error == 0x01)*/

        {      SW_11 = 0x00;

        T1 = 0x01;    } else {

if(Error == 0x02)/* Error 2 */

        {      SW_22 = 0x00;

        T2 = 0x01;    } else {

if(Error == 0x04)/* Error 4 */

        {      SW_44 = 0x00;

        T4 = 0x01;    } else {

if(Error == 0x05)/* Error 5*/

        {      SW_55 = 0x00;

        T5 = 0x01;    } else {

if(Error == 0x06)/*Error 6*/

        {      SW_66 = 0x00;

        T6 = 0x01;    } } } }

        if (Car1 == 0x00)

```

```

        {    Two_cycle();

            Forward();

PORTC = 0b00100100; /* num 2 */

        _delay_ms(1000);

        _delay_ms(1000);

        _delay_ms(1000);

PORTC = 0b00110000; /* num 3 */

        Car1 = 0x02; } else {

            if (Car1 == 0x01)

                {    One_cycle();

                    Forward();

PORTC = 0b00110000; /* num 3 */

                    Car1 = 0x02; } else {

                        if (Car1 == 0x03)

                            {    One_cycle();

                                Backward();

PORTC = 0b00110000; /* num 3 */

                                Car1 = 0x02; } else {

                                    if (Car1 == 0x04)

                                        {    Two_cycle();

                                            Backward();

PORTC = 0b00011001; /* num 4 */

                                            _delay_ms(1000);

                                            _delay_ms(1000);

                                            _delay_ms(1000);

PORTC = 0b00110000; /* num 3 */

                                            Car1 = 0x02; } else {

```



```

        if (Car1 == 0x05)
        {
            Three_cycle();

            Backward();

PORTC = 0b00010010; /* num 5 */

            _delay_ms(1000);                _delay_ms(1000);
        _delay_ms(1000);

PORTC = 0b00011001; /* num 4    _delay_ms(1000);                _delay_ms(1000);
        _delay_ms(1000);

PORTC = 0b00110000; /* num 3 */

            Car1 = 0x02; } } } }

        if(PINA == 0x1B)

    {
        Error = 0x03; /*Error 3 */

        Car1 = 0x02; } else {

        Error = 0x00; /*Error 0*/

        SW_33 = 0x00;

        T3 = 0x00;

        Car1 = 0x02; } }

/***** Test SW 4 *****/

        if(PINB == 0x08) /* Test SW 4 */

    {
        SW_4 = 0x01;

PORTC = 0b00011001; } /* num 4 */

        if (SW_4 == 0x01 & PINB == 0x40)

            {
                SW_4 = 0x00;

                SW_44 = 0x01;            }

            if (SW_44 == 0x01)

    {
        if(Error == 0x01)/*Error 1 */

            {
                SW_11 = 0x00;

                T1 = 0x01;    } else {

```

```

        if(Error == 0x02)/*Error 2*/
        {
            SW_22 = 0x00;
            T2 = 0x01;    } else {
        if(Error == 0x03)/*Error 3*/
        {
            SW_33 = 0x00;
            T3 = 0x01;    } else {
        if(Error == 0x05)/* Error 5 */
        {
            SW_55 = 0x00;
            T5 = 0x01;    } else {
        if(Error == 0x06)/* Error 6*/
        {
            SW_66 = 0x00;
            T6 = 0x01;    } } } }
    if ( Car1 == 0x00)
    {
        Three_cycle();
        Forward();
        PORTC = 0b00100100; /* num 2 */
        _delay_ms(1000);
        _delay_ms(1000);
        _delay_ms(1000);
        PORTC = 0b00110000; /* num 3 */
        _delay_ms(1000);
        _delay_ms(1000);
        _delay_ms(1000);
        PORTC = 0b00011001; /* num 4 */
        Car1 = 0x03; } else {
        if ( Car1 == 0x01)
        {
            Two_cycle();
            Forward();

```

```

PORTC = 0b00110000; /* num 3 */
    _delay_ms(1000);
    _delay_ms(1000);
    _delay_ms(1000);
PORTC = 0b00011001; /* num 4 */
    Car1 = 0x03; } else {
    if ( Car1 == 0x02)
    {
        One_cycle();
        Forward();
PORTC = 0b00011001; /* num 4 */
    Car1 = 0x03; } else {
    if ( Car1 == 0x04)
    {
        One_cycle();
        Backward();
PORTC = 0b00011001; /* num 4 */
    Car1 = 0x03; } else {
    if ( Car1 == 0x05)
    {
        Two_cycle();
        Backward();
PORTC = 0b00010010; /* num 5 */
    _delay_ms(1000);
    _delay_ms(1000);
    _delay_ms(1000);
PORTC = 0b00011001; /* num 4 */
    Car1 = 0x03; } } } }
    if(PINA == 0x1B)
{
    Error = 0x04; /*Error 4*/
    Car1 = 0x03; } else {

```

```

        Error = 0x00; /*Error 0;*/

        SW_44 = 0x00;

        T4 = 0x00;

        Car1 = 0x03; } }

/***** Test SW 5 *****/

        if(PINB == 0x10) /* Test SW 5 */

        { SW_5 = 0x01;

PORTC = 0b00010010; }/* num 5 */

        if (SW_5 == 0x01 & PINB == 0x40)

                {      SW_5 = 0x00;

                SW_55 = 0x01;      }

                if (SW_55 == 0x01)

        { if(Error == 0x01)/*Error 1 */

                {      SW_11 = 0x00;

                T1 = 0x01;      } else {

                if(Error == 0x02)/*Error 2*/

                {      SW_22 = 0x00;

                T2 = 0x01;      } else {

                if(Error == 0x03)/*Error 3*/

                {      SW_33 = 0x00;

                T3 = 0x01;      } else {

                if(Error == 0x04)/* Error 4 */

                {      SW_44 = 0x00;

                T4 = 0x01;      } else {

                if(Error == 0x06)/* Error 6*/

                {      SW_66 = 0x00;

                T6 = 0x01;      } } } }

        if ( Car1 == 0x00)

```

```

        {    Two_cycle();

            Backward();

PORTC = 0b00000010; /* num 6 */

        _delay_ms(1000);

        _delay_ms(1000);

        _delay_ms(1000);

PORTC = 0b00010010; /* num 5 */

        Car1 = 0x04; } else{

            if ( Car1 == 0x01)

                {    Three_cycle();

                    Forward();

PORTC = 0b00110000; /* num 3 */

                _delay_ms(1000);

                _delay_ms(1000);

                _delay_ms(1000);

PORTC = 0b00011001; /* num 4 */

                _delay_ms(1000);

                _delay_ms(1000);

                _delay_ms(1000);

PORTC = 0b00010010; /* num 5 */

                Car1 = 0x04; } else {

                    if ( Car1 == 0x02)

                        {    Two_cycle();

                            Forward();

PORTC = 0b00011001; /* num 4 */

                            _delay_ms(1000);

                            _delay_ms(1000);

                            _delay_ms(1000);

```

```

PORTC = 0b00010010; /* num 5 */

    Car1 = 0x04; } else{
if ( Car1 == 0x03)
    {
        One_cycle();

        Forward();
PORTC = 0b00010010; /* num 5 */

        Car1 = 0x04; } else {
if ( Car1 == 0x05)
    {
        One_cycle();

        Backward();
PORTC = 0b00010010; /* num 5 */

        Car1 = 0x04; } } } }

if(PINA == 0x1B)
{
    Error = 0x05; /*Error 5*/

    Car1 = 0x04;

    } else {

    Error = 0x00; /*Error 0 */

    SW_55 = 0x00;

    T5 = 0x00;

    Car1 = 0x04;

    } }

/***** Test SW 6 *****/

if(PINB == 0x20) /* Test SW 6 */

{ SW_6 = 0x01;

PORTC = 0b00000010; } /* num 6 */

if (SW_6 == 0x01 & PINB == 0x40)

    {
        SW_6 = 0x00;

        SW_66 = 0x01;
    }

```

```

        if (SW_66 == 0x01){
if(Error == 0x01)/*Error 1 */
{
    SW_11 = 0x00;

    T1 = 0x01;    } else {
if(Error == 0x02)/*Error 2*/
{
    SW_22 = 0x00;

    T2 = 0x01;    } else {
if(Error == 0x03)/*Error 3*/
{
    SW_33 = 0x00;

    T3 = 0x01;    } else {
if(Error == 0x04)/* Error 4 */
{
    SW_44 = 0x00;

    T4 = 0x01;    } else {
if(Error == 0x05)/* Error 5*/
{
    SW_55 = 0x00;

    T5 = 0x01;    } } } }
    if ( Car1 == 0x00)
    {
        One_cycle();

        Backward();
PORTC = 0b000000010; /* num 6 */

        Car1 = 0x05; } else {

            if ( Car1 == 0x01)

            {
                Two_cycle();

                Backward();
PORTC = 0b01111001; /* num 1 */

                _delay_ms(1000);

                _delay_ms(1000);

                _delay_ms(1000);

PORTC = 0b000000010; /* num 6 */

```

```

        Car1 = 0x05; } else {
            if ( Car1 == 0x02)
            {
                Three_cycle();

                Forward();

PORTC = 0b00011001; /* num 4 */

                _delay_ms(1000);

                _delay_ms(1000);

                _delay_ms(1000);

PORTC = 0b00010010; /* num 5 */

                _delay_ms(1000);

                _delay_ms(1000);

                _delay_ms(1000);

PORTC = 0b00000010; /* num 6 */

                Car1 = 0x05; } else {
                    if ( Car1 == 0x03)
                    {
                        Two_cycle();

                        Forward();

PORTC = 0b00010010; /* num 5 */

                        _delay_ms(1000);

                        _delay_ms(1000);

                        _delay_ms(1000);

PORTC = 0b00000010; /* num 6 */

                            Car1 = 0x05; } else {
                                if ( Car1 == 0x4)
                                {
                                    One_cycle();

                                    Forward();

PORTC = 0b00000010; /* num 6 */

                                    Car1 = 0x05; } } } }

```



```

        if(PINA == 0x1B)
        { Error = 0x06; /*Error 6*/
          Car1 = 0x05; } else {
          Error = 0x00; /*Error 0;*/
          SW_66 = 0x00;
          T6 = 0x00;
          Car1 = 0x05;
        } } } } /* else main */

```

```

void Forward() /* Clokwise */      {
    while(j!=0)
{
    j--;
    PORTD = 0b00000011;
        _delay_ms(900);
    PORTD = 0b00001001;
        _delay_ms(900);
    PORTD = 0b00001100;
        _delay_ms(900);
    PORTD = 0b00000110;
        _delay_ms(900); }
    return;
}

```

```

void Backward() /* UN Clokwise */
{
    while(j!=0)
{
    j--;
    PORTD = 0b00000110;
        _delay_ms(900);
    PORTD = 0b00001100;

```

```

        _delay_ms(900);
PORTD = 0b00001001;
        _delay_ms(900);
PORTD = 0b00000011;
        _delay_ms(900);    }

    return;    }

void One_cycle()
{
    j = 85;
    return;    }

void Two_cycle()
{
    j = 170;
    return;    }

void Three_cycle()
{
    j = 255;
    return;    }

```