**DESIGN OF SMART CONVEYANCE VEHICLE FOR AN AIRPORT USING PIC MICROCONTROLLER**

PROJECT REPORT 2007-2008

Submitted by

**M. PREM CHANDAN [04D078]**



**SAMSON THOMAS [04D082]**

**J. SOLOMON JOHN [04D103]**

**S. A. MANIKANDAN [04D607]**

Under the guidance of

**Miss. K. PRESCILLA, M.E.**

(Lecturer, ECE)

Submitted in partial fulfillment of the requirements for the award of the degree

**BACHELOR OF ENGINEERING**

**IN**

**ELECTRONICS & COMMUNICATION ENGINEERING**

Degree of Karunya University.

**Department of Electronics and Communication Engineering**

**School of Electrical Sciences**

**KARUNYA UNIVERSITY**

(Declared as Deemed-to-be University under Section 3 of the UGC Act, 1956)

**Coimbatore-641 114.**

**APRIL 2008**

**Department of Electronics and Communication Engineering**

**School of Electrical Sciences**

**KARUNYA UNIVERSITY**

(Declared as Deemed-to-be University under Section 3 of the UGC Act, 1956)

**Coimbatore-641 114.**

**BONAFIDE CERTIFICATE**

This to certify that the project report entitled **“DESIGN OF SMART CONVEYANCE VEHICLE FOR AN AIRPORT USING PIC MICROCONTROLLER”** is a bonafide record of the project work done by

**M. PREM CHANDAN [04D078]**

**SAMSON THOMAS [04D082]**

**J. SOLOMON JOHN [04D103]**

**S. A. MANIKANDAN [04D607]**

during the academic year 2007-2008.

Guide Head of Department

**Submitted for the University Exam held on ­­­­­­­­­­­­­­­­­­­­­\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Internal Examiner External Examiner**

**ACKNOWLEDGEMENT**

We articulate our heartfelt gratitude to our precious **LORD JESUS** **CHRIST** for being with us during our project work. We are grateful to Him for giving us wisdom to think of the logics in this project. His hand was there in every task we undertook.

We express our gratitude to our beloved Late Founder Chancellor Emeritus **Dr. D. G. S. DHINAKARAN** **C.A.I.I.B., Ph.D., D.D (USA), D.Litt,** and Chancellor **Dr. PAUL DHINAKARAN M.B.A., Ph.D.,** for their prayers that were of great encouragement during the course of project.

We fondly submit our humble gratitude to our Vice-Chancellor **Dr. PAUL P. APPASAMY M.S.(Chemical Engg.), M.B.A., Ph.D** and Registrar **Dr.(Mrs.)ANNE MARY FERNANDEZ M.A., M.Phil., Ph.D.,** for their encouragement.

We thank **Mrs. KEZI SELVA VIJILA M.E. (Ph.D),** HOD of Electronics and Communication Department, for her immense support.

We are indeed grateful to our Project Coordinator **Mr. S. SMYS, M.E.**

We specially express our heartfelt thanks to our internal guide, **Miss. K. PRESCILLA, M.E.,** for her inspirational guidance, motivation and strong support that she has provided throughout the project period.

We also thank **Mr. D. ARUL RAJ KUMAR, Mr. J. JEBAVARAM, Mr. B. JEYACHANDRAN, Mr. J. MANOHAR LIVINGSTONE**, Lab Technicians, ECE Dept., for helping us in the project.

We are also grateful to our friend **BENSINGH** for his help and guidance throughout the project.

We are blessed to have our dear parents for their prayers and support with invaluable encouragement. We also like to thank everyone who constantly helped us either directly or indirectly.

**ABSTRACT**

Automated line following vehicles has been under heavy research for sophisticated transport. They find their application in many domestic domains.

The objective of the project is to design and implement an autonomous line follower robot named Smart Conveyance Vehicle (SCV) which is to autonomously trace a path, avoid obstacles along the path, deliver the goods at the destination and come back to the loading point. The vehicle is primarily aimed at automating the vehicle used to transfer luggage in an Airport. The vehicle can transport goods to different destinations set by the user. The modified version of the vehicle can be used to transfer documents in an office setting.

The Smart Conveyance Vehicle is equipped with a Line Tracker sensor array, Junction sensor, Obstacle sensor, Actuators i.e. the DC motors and PIC16F877 microcontroller.

HelpMate was a line following robot designed as an automated mobile courier service in an hospital [1].Over 50 units are operational working in uncontrolled hospital environments up to 24 hours per day, 7 days per week.

Along the way, the report also discusses the various circuits, the hardware model, the failures underwent and also the rectifications done.

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO** | **TITLE** | **PAGE NO** |
|  | **ACKNOWLEDGEMENT** | **iii** |
|  | **ABSTRACT** | **iv** |
|  | **LIST OF FIGURES** | **ix** |
|  | **LIST OF TABLES** | **x** |
| **CHAPTER 1** | **OVERVIEW** | 1 |
|  | 1.1 Introduction |  |
|  | 1.2 Real time application |  |
|  | 1.3 Block diagram of the Smart Conveyance Vehicle. |  |
|  | 1.4 The navigation path of the Smart Conveyance Vehicle |  |
|  | 1.5 The objective |  |
|  | 1.6 Literature survey |  |
| **CHAPTER 2** | **MICROCONTROLLER PIC16F877** | 5 |
|  | 2.1 PIC Microcontroller |  |
|  | 2.2 Architecture of PIC16F877 |  |
|  | 2.3 Microcontroller core features |  |
|  | 2.4 Block diagram of PIC16F877 |  |
|  | 2.5 Pin configuration |  |
|  | 2.6 Port configuration |  |
|  | 2.6.1 Port A |  |
|  | 2.6.2 Port B |  |
|  | 2.6.3 Port C |  |
|  | 2.6.4 Port D |  |
|  | 2.7 List of pins used |  |
| **CHAPTER 3** | **SENSORS** | 11 |
|  | 3.1 A Basic line sensor |  |
|  | 3.1.1 Drawbacks |  |
|  | 3.2 Line detector using phototransistor |  |
|  | 3.2.1 Drawbacks |  |
|  | 3.3 Infrared line detector amplified signal circuit |  |
|  | 3.3.1 Drawbacks |  |
|  | 3.4 Line sensor employing TSOP 1738 |  |
|  | 3.4.1 Merits |  |
|  | 3.5 Junction sensor/obstacle sensor |  |
|  | 3.6 Analysis |  |
| **CHAPTER 4** | **MOTORS AND DRIVERS** | 14 |
|  | 4.1 Motor |  |
|  | 4.1.1 Working principle of motors |  |
|  | 4.2 Motor driver |  |
|  | 4.2.1 H-Bridge theory of operation |  |
|  | 4.3 Differential drive |  |
| **CHAPTER 5** | **THE PHYSICAL OUTLOOK** | 19 |
|  | 5.1 Chassis |  |
|  | 5.2 Motors/Wheels |  |
|  | 5.3 Power supply |  |
|  | 5.4 The final model |  |
|  |  |  |
|  |  |  |
| **CHAPTER 6** | **ALGORITHMS** | 20 |
|  | 6.1 Basic conditions |  |
|  | 6.2 Algorithms |  |
|  | 6.2.1 Line follow algorithm using three sensors |  |
|  | 6.2.2 Obstacle collision avoidance |  |
|  | 6.3 Algorithm for smart conveyance vehicle |  |
|  | 6.4 Analysis |  |
|  |  |  |
| **CHAPTER 7** | **BURNING THE CHIP** | 22 |
|  | 7.1 Picpgm LVISP |  |
|  | 7.2 Difficulties faced |  |
|  | 7.3 Alternative choice |  |
|  | 7.4 HI-TECH PICC-Lite™ V9.60PL2 compiler |  |
|  | 7.5 Embedded development environment |  |
|  | 7.6 Ponyprog PIC programmer |  |
|  |  |  |
| **CHAPTER 8** | **STRUCTURE OF SCV** | 26 |
|  | 8.1 Line tracking sensors |  |
|  | 8.2 The obstacle sensor |  |
|  | 8.3 The junction sensor |  |
|  | 8.4 The PIC board |  |
|  | 8.5 The hardware model |  |
| **CHAPTER 9** | **SMART CONVEYANCE VEHICLE** | 29 |
|  | 9.1 Testing the vehicle |  |
|  | 9.2 Problems faced and solutions |  |
|  | 9.3 Future scope |  |
| **CHAPTER 10** | **CONCLUSION** | 30 |
|  | **APPENDICES** | 32 |
|  | **REFERENCES** | 42 |

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **FIGURE NO** | **TITLE** | **PAGE NO** |
| 1.1 | Block diagram of the Smart Conveyance Vehicle | 2 |
| 1.2 | The Navigation Path of The Smart Conveyance Vehicle | 2 |
| 2.1 | Block diagram of PIC16F877 microcontroller | 7 |
| 2.2 | Pin diagram of PIC16F877 microcontroller | 8 |
| 3.1 | Basic line detector circuit | 11 |
| 3.2 | Line detector using phototransistor | 12 |
| 3.3 | Infrared line detector amplified signal circuit | 12 |
| 3.4 | Line sensor employing TSOP 1738 | 13 |
| 4.1 | A Simple 2-Pole Dc motor | 14 |
| 4.2 | H-Bridge and operation | 16 |
| 4.3 | Power amplifier | 16 |
| 4.4 | Driving and rotation of differential drive | 17 |
| 7.1 | Low voltage in system programmer | 22 |
| 7.2 | Compiling the code using HITECH C compiler for PIC16F877 microcontroller | 23 |
| 7.3 | Writing the chip using winpicprog | 24 |
| 8.1 | Line sensor array | 25 |
| 8.2 | The obstacle sensor | 25 |
| 8.3 | The junction sensor | 26 |
| 8.4 | The schematics of the pic board | 26 |
| 8.5 | The PIC board | 27 |
| 8.6 | The hardware model | 27 |
|  |  |  |
|  |  |  |

**LIST OF TABLES**

|  |  |  |
| --- | --- | --- |
| **TABLE NO** | **TITLE** | **PAGE NO** |
| 2.1 | List of port pins used in the project | 10 |

**CHAPTER 1**

**OVERVIEW**

**1.1 INTRODUCTION**

Machine intelligence means many things to many people, from the purely software side to the purely hardware side and everywhere in between. The focus of machine intelligence for this project is the use of sensors to intelligently interact with the environment to perform the desired task.

The Smart Conveyance Vehicle is an attempt to create an autonomous robot that uses sensors to interact with its environment and perform the following tasks i.e. (1) to follow a chosen track, (2) look for obstacles along the path, (3) choose an alternative path, (4) reach the destination and (5) follow back to the source.

The Smart Conveyance Vehicle uses the PIC16F877 microcontroller as the brain. The code is written in C language. The HI-TECH PICC-Lite compiler is used to write the code and it generates the necessary hex file. The Microcontroller is burnt using the Ponyprog Programmer. The software necessary for burning the chip is *winpicprog.*

The peripherals are the Line Tracking sensors, the Obstacle sensor and the Junction sensor. An array of line sensors are used to detect the track. The obstacle sensor indicates the presence or absence of an obstacle. The Junction sensor is used to detect junction.

The Smart Conveyance Vehicle is designed to follow a black line in a white background. The IR transmitter and receivers are used for line detection.

The actuators used are DC motors. The DC motors are controlled using the L293D Motor Driver.

**1.2 REAL TIME APPLICATION**

* The primary application aimed is automated luggage transfer in Airport.
* A small variant of the robot can be used to transfer documents in an office setting.
* HelpMate™ is used to deliver medical supplies in a hospital [2].

**1.3 BLOCK DIAGRAM OF THE SMART CONVEYANCE VEHICLE**

Line Sensor Array Input



DC Motors

PIC16F877 Microcontroller

User Input

L293D Motor Driver

Obstacle Sensor Input

Junction Sensor Input

Figure 1.1 Block diagram of the Smart Conveyance Vehicle

**1.4 THE NAVIGATION PATH OF THE SMART CONVEYANCE VEHICLE**

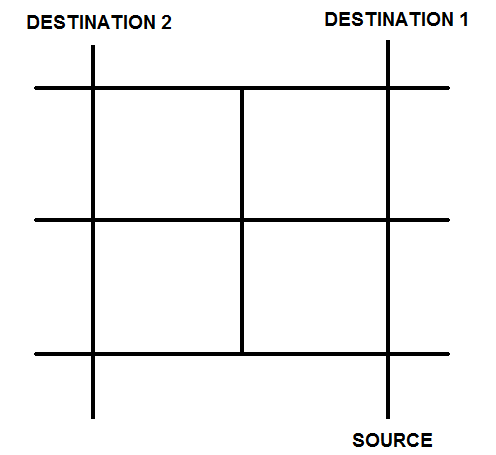


Figure 1.2 The navigation path of the Smart Conveyance Vehicle

**1.5 THE OBJECTIVE**

The Vehicle has to start from the source i.e. the loading point where the luggages are loaded manually onto the vehicle. Once the destination is set by the user the vehicle has to autonomously move towards the destination. As the vehicle proceeds it must also look for obstacles in the path. If there be any obstacles in the path, it must look for an alternative path. Once the vehicle reaches the destination it must stop still the goods are unloaded. Once the luggages are unloaded the user will set it to go back to the loading point. Then it must return to the loading point. The vehicle must also be programmed for different destinations. The source and the destinations can be clearly observed from the figure 1.2

**1.6 LITERATURE SURVEY**

[1] HelpMate is an autonomous robot specifically designed as a mobile robot courier for hospitals. Evans, J.M. published a paper in IEEE journal on the topic “*HelpMate: an autonomous mobile robot courier for hospitals*”. Over 50 units are operational working in uncontrolled hospital environments up to 24 hours per day, 7 days per week. The navigation and system technologies embodied in HelpMate are described, emphasizing the impact of real world requirements on the engineering design

[2] Mailbot is an autonomous robot that was created to retrieve outgoing mail from an office setting. This paper discusses the components of the mailbot, including the mobile platform itself, the actuation of the behaviours, description of the sensors, and summary of the behaviours. The paper also discusses the short comings and ways to increase performance of the robot.

[3] Jonathan W. Valvano published a lab manual along with the book titled, “*Embedded Microcomputer Systems: Real Time Interfacing*”. The manual describes the complete hardware and software requirement of a line tracking robot.

[4] Plermjai Inchuay describes an award winner line follower robot from VingPeaw Competition 2543. The site hosts all the circuit schematics and the software required for the robot.

[5]Thomas Braunl authored this book which presents a unique combination of mobile robots and embedded systems, from introductory to intermediate level. It is structured in three parts, dealing with embedded systems (hardware and software design, actuators, sensors, PID control, multitasking), mobile robot design (driving, balancing, walking, and flying robots), and mobile robot applications (mapping, robot soccer, genetic algorithms, neural networks, behaviour-based systems, and simulation).

[6] This website hosted by TRI – The Robotics Institute describes various possible algorithm for a mobile robot from line following to grid tracing. The site also describes the constraints dealing with the robot body design.

[7] This page hosted by Society of Robots website explains a basic colour detector circuit. The page also briefs in detail the design of the circuit,

[8] This page hosted by Society of Robots website describes the differential drive method of controlling of robot with only two motorized wheels. The page also briefs the algorithm of the drive mechanism.

[9] PICPgm is a free PIC development programmer for Windows. This site gives an in-depth knowledge about the hardware and the software required for the programmer. The site also holds the programmer tutorials with schematics.

[10] This website describes how the baggage handling operation works in an

Airport.

**CHAPTER 2**

**MICROCONTROLLER PIC16F877**

**2.1 PIC MICROCONTROLLER**

**PIC** is a family of Harvard architecture microcontrollers made by Microchip Technology, derived from the PIC1640 originally developed by General Instrument's Microelectronics Division. The name PIC initially referred to "**Programmable Interface Controller**", but shortly thereafter was renamed "**Programmable Intelligent Computer**".

PICs are popular with developers and hobbyists alike due to their low cost, wide availability, large user base, extensive collection of application notes, availability of low cost or free development tools, and serial programming (and re-programming with flash memory) capability. Microchip recently announced the shipment of its five billionth PIC processor.

**PIC16F877** is one of the most commonly used microcontroller especially in automotive, industrial, appliances and consumer applications.

**2.2 ARCHITECTURE OF PIC16F877**

The PICmicro MCU family is based on RISC architecture, RISC standing for Reduced Instruction Set Computer. High performance is accomplished as a result of the Harvard architecture, which has separate data and address busses and two separate memory spaces.

Instruction pipelining gives the capability of higher speed while the register file concept allows you to do the same thing with the register as you would with a piece of RAM.

Single cycle instructions allow high speed code execution, and single word instructions improve the throughput and reduce the amount of program memory space required.

The long word instruction gives the capability of holding immediate data in the same line of code as the instruction itself, which leads to a reduced instruction set which is very orthogonal or symmetric. This means that for every Set instruction there is a Clear instruction, for every Increment there is a Decrement and so on.

All instructions can be performed with any of the registers, so there are no special functions or special instructions for doing operations on the carry or handling any I/O peripherals. They all use the same instruction set. Figure 2.1 shows the block diagram of PIC16F877 microcontroller.

**2.3 PIC MICROCONTROLLER CORE FEATURES**

* High performance RISC CPU
* Only 35 single word instructions to learn
* All single cycle instructions except for program branches which are two cycle
* Operating speed: DC - 20 MHz clock input

DC - 200 ns instruction cycle

* Up to 8K x 14 words of FLASH Program Memory,

Up to 368 x 8 bytes of Data Memory (RAM)

Up to 256 x 8 bytes of EEPROM Data Memory

* Interrupt capability (up to 14 sources)
* Eight level deep hardware stack
* Direct, indirect and relative addressing modes
* Power-on Reset (POR)
* Power-up Timer (PWRT) and
* Oscillator Start-up Timer (OST)
* Watchdog Timer (WDT) with its own on-chip RC oscillator for reliable operation
* Programmable code protection
* Power saving SLEEP mode
* Selectable oscillator options
* Low power, high speed CMOS FLASH/EEPROM

Technology

* Wide operating voltage range: 2.0V to 5.5V

**2.4 BLOCK DIAGRAM OF PIC16F877**

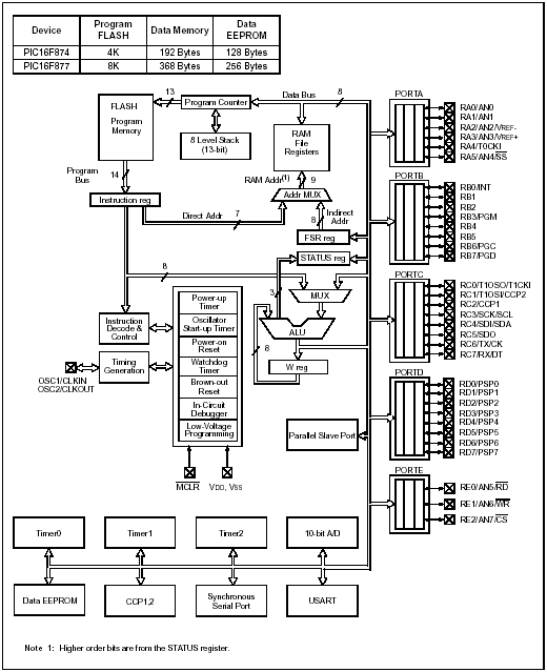


Figure 2.1 Block diagram of PIC16F877 microcontroller

**2.5 PIN CONFIGURATION**

The PIC16F877 is a 40 pin chip. There are four ports, which can be programmed as input/output. Most of the pins can be configured for a different function. The VDD supply is given to pins 11 and 32. VSS is given to pins 12 and 31. The pin configuration can be better understood from the figure 2.2

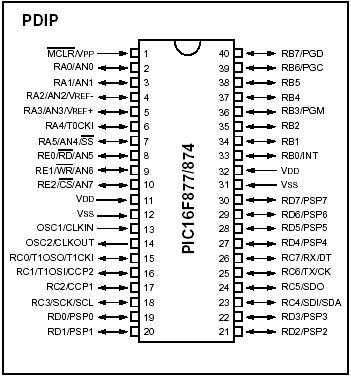


Figure 2.2 Pin diagram of PIC16F877 microcontroller

**2.6 PORT CONFIGURATION**

Some pins of the I/O ports are multiplexed with an alternate function for the peripheral features on the device. In general, when a peripheral is enabled, that pin may not be used as a general purpose I/O pin.

**2.6.1 PORT A**

PORTA is a 6-bit wide, bi-directional port. The corresponding data direction register is TRISA. Setting a TRISA bit (= 1) will make the corresponding PORTA pin an input (i.e., put the corresponding output driver in a Hi-Impedance mode). Clearing a TRISA bit (= 0) will make the corresponding PORTA pin an output (i.e., put the contents of the output latch on the selected pin).

Reading the PORTA register reads the status of the pins, whereas writing to it will write to the port latch. All write operations are read-modify-write operations.

Therefore, a write to a port implies that the port pins are read; the value is modified and then written to the port data latch.

In this project pins RA0 and RA1 of port A are used as destination selection inputs.

**2.6.2 PORT B**

PORTB is an 8-bit wide, bi-directional port. The corresponding data direction register is TRISB. Setting a TRISB bit (= 1) will make the corresponding PORTB pin an input (i.e., put the corresponding output driver in a Hi-Impedance mode). Clearing a TRISB bit (= 0) will make the corresponding PORTB pin an output (i.e., put the contents of the output latch on the selected pin).

Three pins of PORTB are multiplexed with the Low Voltage Programming function: RB3/PGM, RB6/PGC and RB7/PGD. The alternate functions of these pins are described in the Special Features Section.

Each of the PORTB pins has a weak internal pull-up. A single control bit can turn on all the pull-ups. This is performed by clearing bit RBPU (OPTION\_REG<7>). The weak pull-up is automatically turned off when the port pin is configured as an output. The pull-ups are disabled on a Power-on Reset.

In this project the pins RB0, RB1, RB2, RB3, RB4 and RB5 are used. These pins are used as inputs. The outputs of the line tracking sensors, Obstacle sensor and Junction sensors are connected to these pins. The internal pull-ups are disabled.

**2.6.3 PORT C**

PORTC is an 8-bit wide, bi-directional port. The corresponding data direction register is TRISC. Setting a TRISC bit (= 1) will make the corresponding PORTC pin an input (i.e., put the corresponding output driver in a Hi-Impedance mode). Clearing a TRISC bit (= 0) will make the corresponding PORTC pin an output (i.e., put the contents of the output latch on the selected pin).

PORTC is multiplexed with several peripheral functions. PORTC pins have Schmitt Trigger input buffers.

In this project the pins RC4, RC5, RC6 and RC7 are used as output. They provide as control signals for the L293D DC motor driver to control the motors. These leads are taken through a 100 ohm resistor to prevent the chip from getting damaged from any possible back emf generated from the motor.

**2.6.4 PORT D**

PORTD is an 8-bit port with Schmitt Trigger input buffers. Each pin is individually configurable as an input or output.

PORT D is not used in the project.

**2.7 LIST OF PINS USED**

|  |  |
| --- | --- |
| PORT PINS | CONNECTIONS |
| RA0 | Switch 1 |
| RA1 | Switch 2 |
| RB0 | Output of Front middle sensor |
| RB1 | Output of front left sensor |
| RB2 | Output of front right sensor |
| RB3 | Output of left junction sensor |
| RB4 | Output of right junction sensor |
| RB5 | Output of obstacle sensor |
| RC4 | Input 1 to motor driver |
| RC5 | Input 2 to motor driver |
| RC6 | Input 3 to motor driver |
| RC7 | Input 4 to motor driver |

Table 2.1 List of port pins used in the project

The table 2.1 shows the list of port pins used in the project and their respective leads.

**CHAPTER 3**

**SENSORS**

There are different sensors available. Each has its own advantages. Here the robot has to follow a white line, detect obstacles and also junctions. Hence arrays of sensors are needed for line tracing and one for obstacle sensing. The following circuits were designed for the project and the best was selected.

**3.1 A BASIC LINE SENSOR**

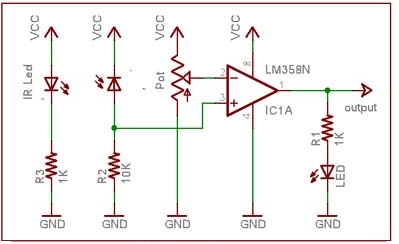


Figure 3.1 Basic line detector circuit

The LM358 OpAmp is used in the comparator mode. The IR photodiode (receiver) is used in a potential divider in a reverse bias mode. A threshold voltage is set at the inverting terminal of the OpAmp using a potentiometer. So when the IR light reflects from a lighter surface, say white, the resistance of the photodiode would decrease and this in turn when exceeds the threshold voltage will make the output of the OpAmp go high. LM 358 has two OpAmps in its 8 pin package, thus two sensors could be built out of 1 IC. The circuit can be seen from the figure 3.1. The use of LM324 comparator reduces the number of ICs.

**3.1.1 DRAWBACK**

The output of the circuit was too low to differentiate a 1 and 0.

**3.2 LINE DETECTOR USING PHOTOTRANSISTOR**

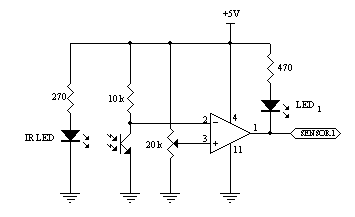


Figure 3.2 Line detector using phototransistor

The IR transmitter emits radiations. The body reflects it back. The IR Receiver diode receives the radiations and is compared with a threshold. The potentiometer is used to vary the threshold. The output is taken from the comparator as shown in Figure 3.2 [4].

**3.2.1 DRAWBACKS**

The receiver received not only IR radiation but even the ambient light.

**3.3 INFRARED LINE DETECTOR AMPLIFIED SIGNAL CIRCUIT**

|  |  |
| --- | --- |
|  | **R1 = 150 ohms**  **R2 = 220 Kohms**  **R3 = 4.7 Kohms**  **R4 = 10 Kohms**  **OP1** = Operational Amplifier **LM358** package includes two op amps.  **Vcc = +5 Volts** |

Figure 3.3 Infrared line detector amplified signal circuit

In the circuit shown in figure 3.3 the output is amplified and then compared with a threshold. R3 and R4 determine the amplification of the op amp, gain = 1 + R4/R3. An appropriate ratio can be determined by connecting up the circuit and measuring the voltage entering the op amp and knowing the threshold value needed at Vout [6].

**3.3.1 DRAWBACKS**

The sensitivity of the circuit is too low.

**3.4 LINE SENSOR EMPLOYING TSOP 1738**

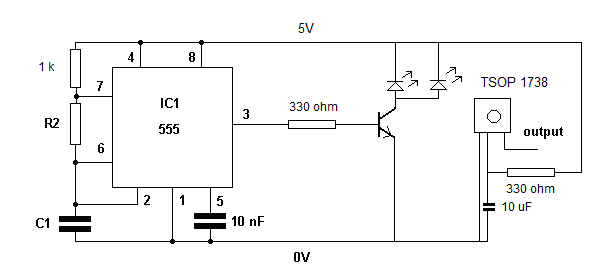


Figure 3.4 Line sensor employing TSOP 1738

The TSOP 1738 is a tuned Infrared receiver. It receives IR radiations which have a frequency of 38 KHz. For this purpose the 555 timer circuit seen in figure 3.4 produces a 38 KHz output which is used to drive the IR transmitters. The transmitters emit IR radiations at 38 KHz. These radiations are reflected by the body and received by the TSOP receiver. Since the receiver is tuned to 38 KHz it will pick up the reflected radiations while all other radiations get attenuated. The BC547 transistor amplifies the 555 timer output.

**3.4.1 MERITS**

As the circuit was tuned to receive only the specified frequency, the sensor is highly sensitive to the colour.

**3.5 JUNCTION SENSOR/OBSTACLE SENSOR**

The sensor circuit which is used to detect the line can also be used to detect junctions and obstacles.

**3.6 ANALYSIS**

From all the above circuits the line sensor employing TSOP 1738 was found to be most reliable and stable. The sensitivity of the receiver can be adjusted with the help of a variable resistor. It is found that the black surface must be perfectly as even a slight distortion in the black colour can cause false detections.

**CHAPTER 4**

**MOTORS AND DRIVERS**

**4.1 MOTOR**

The motors are essential parts in robotics as it provides locomotion. The choice of motors is important for an application. Since line follower robot requires very high speed, the dc motors are the best option.

Stepper motors are not used because of its low speed and servomotors too are not used because of high cost when compared to dc motors.

**4.1.1 WORKING PRINCIPLE OF MOTORS**

In any electric motor, operation is based on simple electromagnetism. A current carrying conductor generates a magnetic field; when it is placed in an external magnetic field. It is known for magnets unlike poles attract and like poles repel. The internal configuration of a dc motor is designed in such a way to carry the interaction between a current carrying conductor and an external magnetic field to generate rotational motion.

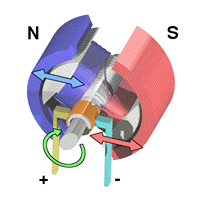


Figure 4.1 A simple 2-Pole DC Motor

Figure 4.1shows a simple 2-pole dc electric motor. Every DC motor has six basic parts namely axle, rotor, stator, commutator, field magnets and brushes. In most common DC motors, the external magnetic field is produced by high strength permanent magnets. The stator is the stationary part of the motor; this includes the motor casing, as well as two or more permanent magnet pole pieces. The rotor rotates with respect to the stator. The rotor consists of windings which are being electrically connected to the commutator. The figure 4.1 shows a common motor layout, with rotor inside the stator magnets.

The geometry of the brushes, commutator contacts, and rotor windings are such that when power is applied, the polarities of the energized winding and the stator magnets are misaligned, and the rotor will rotate until it is almost aligned with the stator’s field magnets.

In real life, dc motors will always have more than two poles. If a rotor is exactly at the middle of its rotation, it will get stuck.

Meanwhile, with a two-pole motor, there is a moment where the commutator shorts out the power supply. This results in energy wastage, and damage motor components as well.Yet another disadvantage of such a simple motor is that it would exhibit a high amount of torque “ripple”. So since small dc motors are of a three-pole design.

**4.2 MOTOR DRIVER**

The signal which is passed by the microcontroller cannot be directly driven by the dc motor due to current loading. Hence there is a need for a driver circuit. **L293D** is a H-bridge motor driver IC which is an interface between PIC 16F877 and motor. The 16F877 passes the signal to L293D and it drives the motor.

**4.2.1 H-BRIDGE THEORY OF OPERATION**

For this application, the vehicle must be able to perform two operation

1. Run it in forward and backward directions.

2. Turn left and right.

An H-bridge is what is needed to enable a motor to run forward/backward. Figure 4.2 demonstrates the H-bridge setup, which received its name from its resemblance to the letter “H” [5]. There is a motor with two terminals a and b and the power supply “+” and “–”. Closing switches 1 and 2 will connect *a* with “+” and *b* with “–”: the motor runs forward. In the same way, closing 3 and 4 instead will connect a with “–” and b with “+”: the motor runs backward.

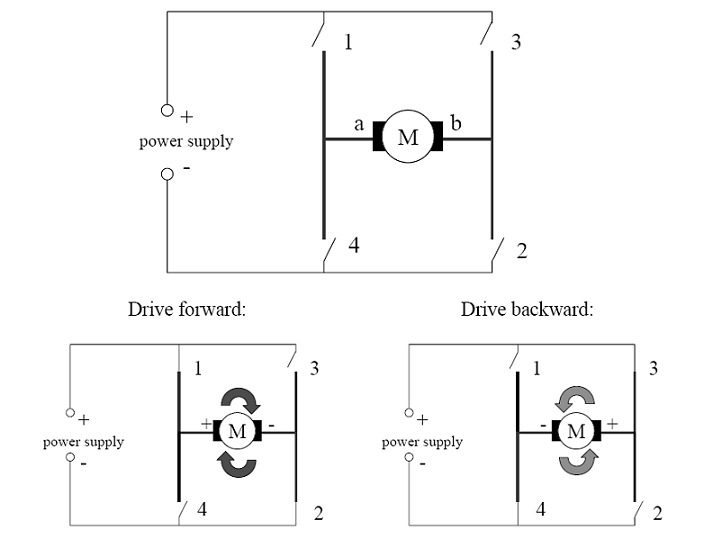


Fig 4.2 H-Bridge and operation

The way to implement an H-bridge when using a microcontroller is to use a power amplifier chip in combination with the digital output pins of the controller or an additional latch. This is required because the digital outputs of a microcontroller has very severe output power restrictions. They can only be used to drive other logic chips, but *never* a motor directly. Since a motor can draw a lot of power (for example 1A or more), connecting digital outputs directly to a motor can destroy the microcontroller.

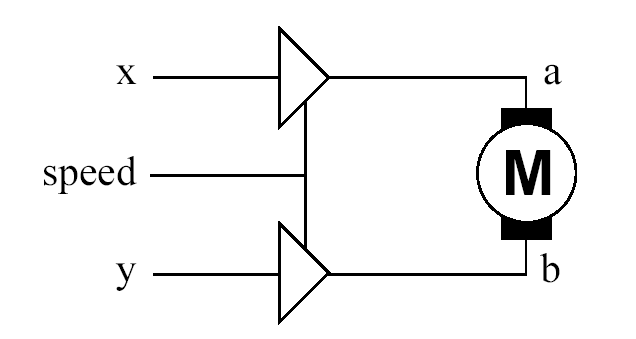


Fig 4.3 Power amplifier

A typical power amplifier chip containing two separate amplifiers is L293D. Figure 4.3 demonstrates the schematics. The two inputs x and y are needed to switch the input voltage, so one of them has to be “+”, the other has to be “–”. Since they are electrically decoupled from the motor, x and y can be directly linked to digital outputs of the microcontroller. So the direction of the motor can then be specified by software, for example setting output x to logic 1 and output y to logic 0. Since x and y are always the opposite of each other, they can also be substituted by a single output port and a negator.

**4.3 DIFFERENTIAL DRIVE**

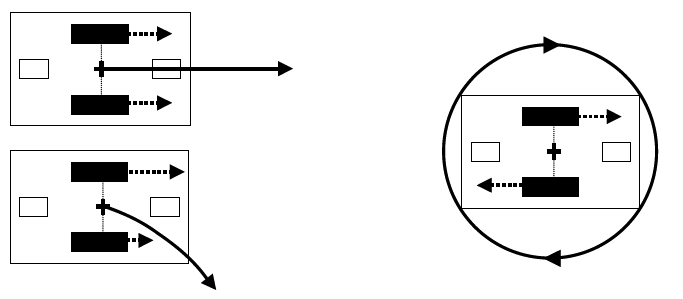


Figure 4.4 Driving and rotation of differential drive

The differential drive design has two motors mounted in fixed positions on the left and right side of the robot, independently driving one wheel each. Since three ground contact points are necessary, this design requires one or two additional passive caster wheels or sliders, depending on the location of the driven wheels. Differential drive is mechanically simpler than the single wheel drive, because it does not require rotation of a driven axis. However, driving control for differential drive is more complex than for single wheel drive, because it requires the coordination of two driven wheels. The minimal differential drive design with only a single passive wheel cannot have the driving wheels in the middle of the robot, for stability reasons. So when turning on the spot, the robot will rotate about the off-center midpoint between the two driven wheels. The design with two passive wheels or sliders, one each in the front and at the back of the robot, allows rotation about the

center of the robot. However, this design can introduce surface contact problems, because it is using four contact points. Figure 4.4 demonstrates the driving actions of a differential drive robot. If both motors run at the same speed, the robot drives straight forward or backward, if one motor is running faster than the other, the robot drives in a curve along the arc of a circle, and if both motors are run at the same speed in opposite directions, the robot turns on the spot [8].

**CHAPTER 5**

**THE PHYSICAL OUTLOOK**

The vehicle was designed with two rear drive wheels and two supporting omni - directional free running wheels for stability. The three line detection sensors are placed in the front side of the robot. The Junction sensors are placed just in front of the real wheels. The obstacle sensor is placed again on the front side but on top of the line detection sensors.

**5.1 CHASSIS**

The Chassis was made using Aluminium frames and plastic rulers. The chassis was designed for 4 wheels for greater stability.

**5.2 MOTORS/WHEELS**

The DC motors with a power rating of 6V and maximum speed of 80 RPM were used. The readymade wheels with shaft holes exactly fitting those of the motors were used. Two ball castor wheels were used in the front for frictionless turns.

**5.3 POWER SUPPLY**

A 9 Volt battery provides power to the microcontroller board. To prevent the chip from getting damaged due to high voltage the supply is first given to a 7805 voltage regulator, stepped down to 5 Volt and then given to the board.

A 7.2 volt battery supplies power to the L293D motor driver through which the motor is driven.

A 6 Volt battery provides power to all the sensor circuits.

**5.4 THE FINAL MODEL**

The vehicle was earlier designed with two drive wheels in the back and one castor wheel in the front. The stability was found to be week. Hence the vehicle was later designed with four wheels with greater stability.

**CHAPTER 6**

**ALGORITHMS**

**6.1 BASIC CONDITIONS**

* The vehicle must follow a black line as fast and unwavering as possible.
* The vehicle must make 90 degree turns.
* The vehicle must reach the specified destination.
* The vehicle must avoid collisions and take alternative paths.

**6.2 ALGORITHMS**

The vehicle employs line follow algorithm and obstacle collision avoidance algorithm.

**6.2.1 LINE FOLLOW ALGORITHM USING THREE SENSORS**

The three sensors are placed in front of the vehicle with the middle sensor exactly facing the line and the other two facing the other two sides of the line [6].

* Start the vehicle.
* Check for line.
* If middle sensor detects the line and the other two senses the surface, move forward.
* If left sensor detects the line, turn left till the middle sensor senses the line.
* Once the middle sensor detects the line, move forward.
* If right sensor detects the line, turn right till the middle sensor senses the line.
* Once again when the middle sensor detects the line, move forward.

**6.2.2 COLLISION AVOIDANCE**

The obstacle sensor is placed in front of the vehicle facing forwards. The algorithm is as follows.

* Start the vehicle.
* Follow the line.
* Check for obstacle along the path.
* If obstacle detected, stop the vehicle.
* Take a 90 degree right turn.
* Move forward for a delay.
* Take a 90 degree left turn move for a considerable delay.
* Take a 90 degree left turn and move till the line is detected.
* Follow the line.

**6.3 ALGORITHM FOR SMART CONVEYANCE VEHICLE**

* Start the vehicle.
* Wait for destination selection.
* Take the appropriate 90 degree turn.
* Move forward and follow the line.
* Look for obstacles along the path.
* Adopt the collision avoidance algorithm if obstacle detected.
* Reach the destination and wait till the luggages are unloaded.
* Once the user sets it to return back to the source, make the appropriate 90 degree turn.
* Move forward and follow the line.
* Look for obstacles along the path.
* Adopt the collision avoidance algorithm if obstacle detected.
* Reach the source and wait for the next destination.

**6.3 ANALYSIS**

The line follow algorithm has been tested and found to be effective.

**CHAPTER 7**

**BURNING THE CHIP**

**7.1 PICPgm LVISP**

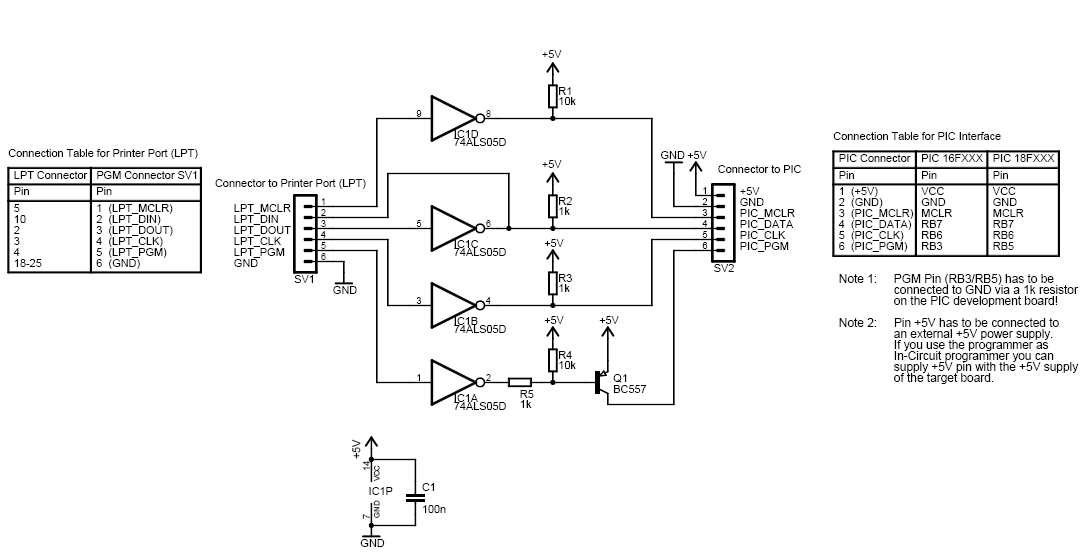


Figure 7.1 Low voltage in system programmer circuit

The PICPgm LVISP is low voltage in system programmer that supports the burning of the chip PIC16F873A wherein there is no need for a separate programming voltage. Figure 7.1 shows the circuit of the PICPgm LVISP. This feature is mentioned in the datasheet as *Single supply 5V In-Circuit Serial Programming (ICSP) via two pins*. Winpicpgm is software that aids in burning the chip [9].

**7.2 DIFFICULTIES FACED**

The programmer could detect the chip. But the software didn’t support the proper burning of the chip PIC16F873A.

The last few lines in the hex file could not be downloaded into the flash memory of the chip.

**7.3 ALTERNATIVE CHOICE**

Due to the above failure of the programmer and non availability of programmer for the chip PIC16F73A, the microcontroller PIC16F877 was chosen and the programmer for the same was available in the Department of Electronics and Communication, Karunya University. The source code for the project was written in C language using HI-TECH PICC-Lite™ v9.60PL2 Compiler. The hex file generated was burnt into the chip using ponyprog PIC programmer. The software necessary to do so is winpicprog.

### 7.4 HI-TECH PICC-LITE™ V9.60PL2 COMPILER

HI-TECH Software has provided this freeware HI-TECH PICC-Lite compiler as a tool for hobbyists and students, but the licence allows its use for commercial purposes as well. It is ideal as a teaching tool for an introduction into the 'C' language and embedded programming on a Microchip device.

### 7.5 EMBEDDED DEVELOPMENT ENVIRONMENT

New to this release of HI-TECH PICC-Lite is the powerful integrated development environment, HI-TIDE™ 3. HI-TIDE 3 has been designed to seamlessly integrate with HI-TECH PICC-Lite and provides a total development system complete with project manager, editor, code creation tool and debugger. HI-TECH Software provides HI-TIDE 3 free of charge for use with HI-TECH PICC-Lite. Figure 7.2 shows the compilation of the code.

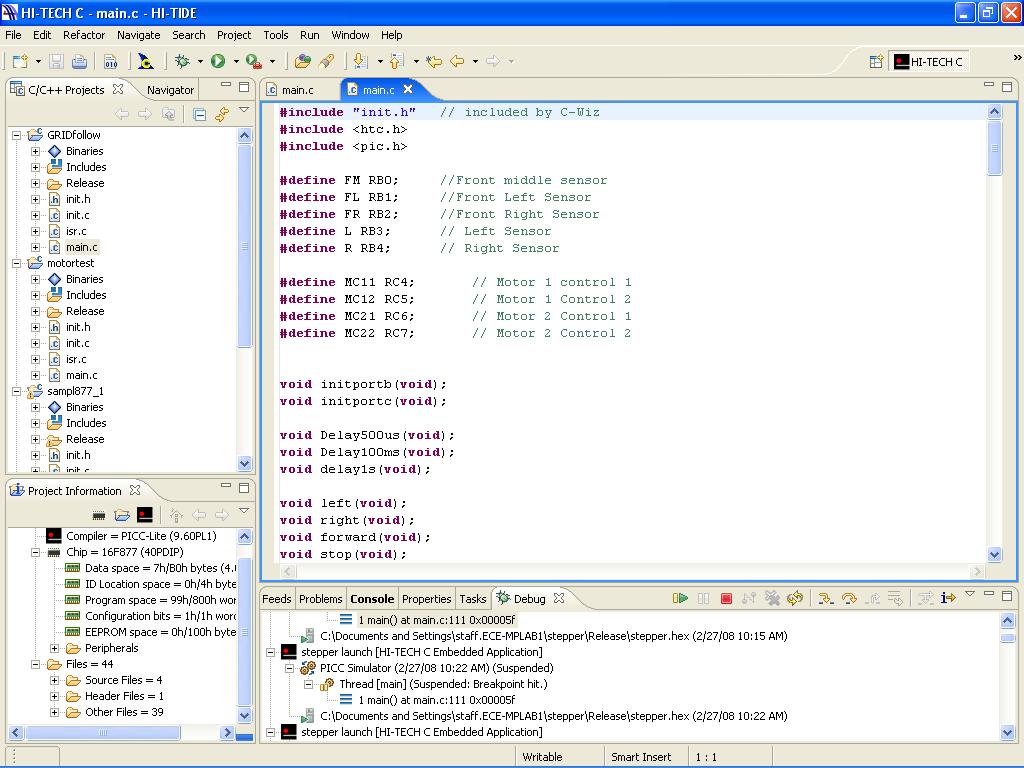


Figure 7.2 Compiling the code using HITECH C compiler for PIC16F877

**7.6 PONYPROG PIC PROGRAMMER**

The software winpicprog is used to burn the hex file generated by the compiler into the chip with the help of the ponyprog programmer. The hardware settings are made as per the manual. The programmer has two modes – programming mode and experiment mode. The programming mode is used to burn the chip. The experiment mode is used to execute the chip. Figure 7.3 shows the burning of the hex file into the chip using winpicprog.

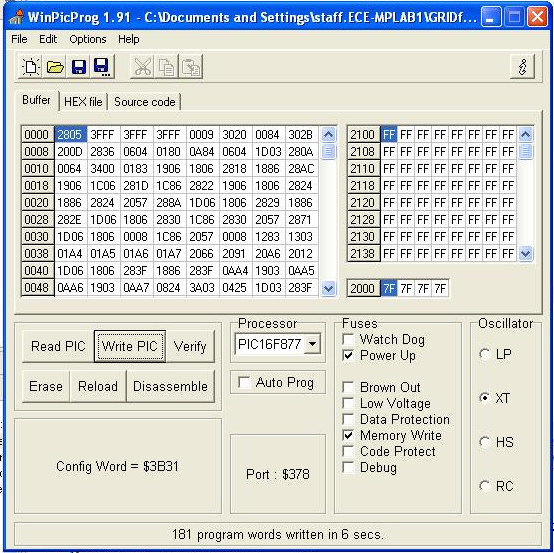


Figure 7.3 Writing the chip using winpicprog

**CHAPTER 8**

**STRUCTURE OF SCV**

**8.1 LINE TRACKING SENSORS**

Figure 8.1 shows the line tracking sensor array facing downwards.

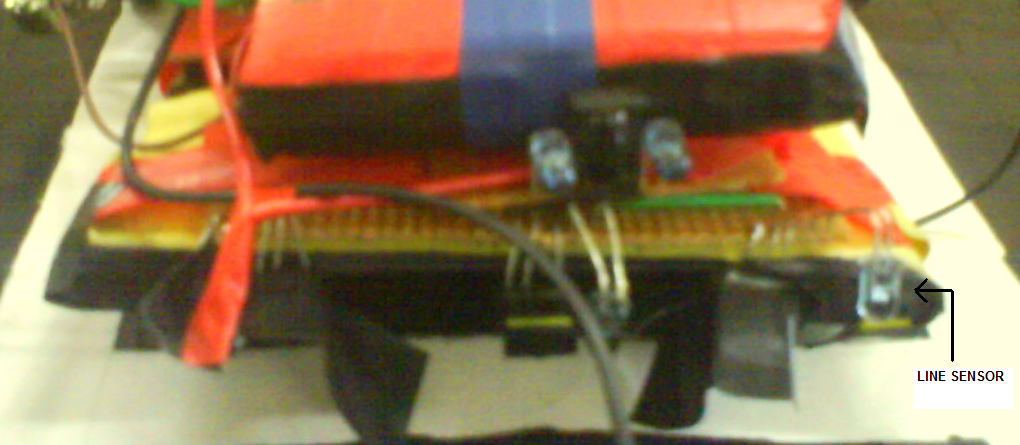


Figure 8.1 Line sensor array

**8.2 THE OBSTACLE SENSOR**

Figure 8.2 shows the obstacle sensor facing forward.

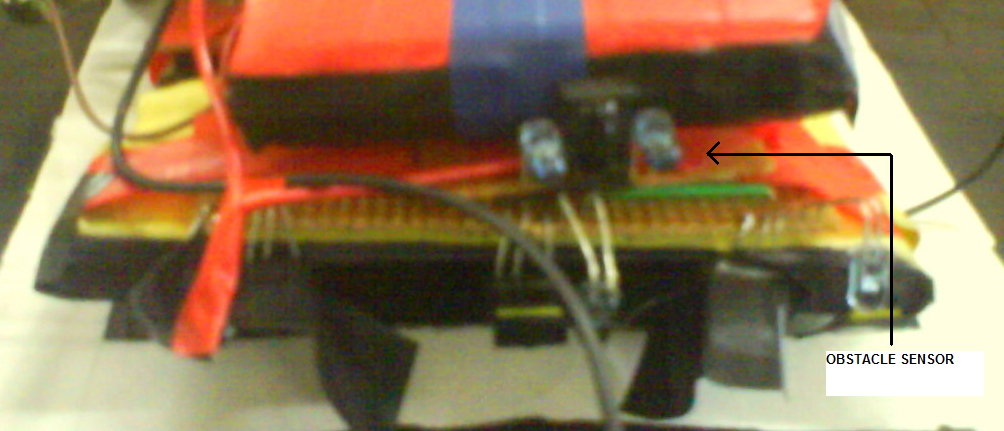


Figure 8.2 The obstacle sensor

**8.3 THE JUNCTION SENSOR**

Figure 8.3 shows the junction sensors located at the two sides of the vehicle.



Fig 8.3 The junction sensor

**8.4 THE PIC BOARD**

Figure 8.4 shows the schematics of the PIC board.

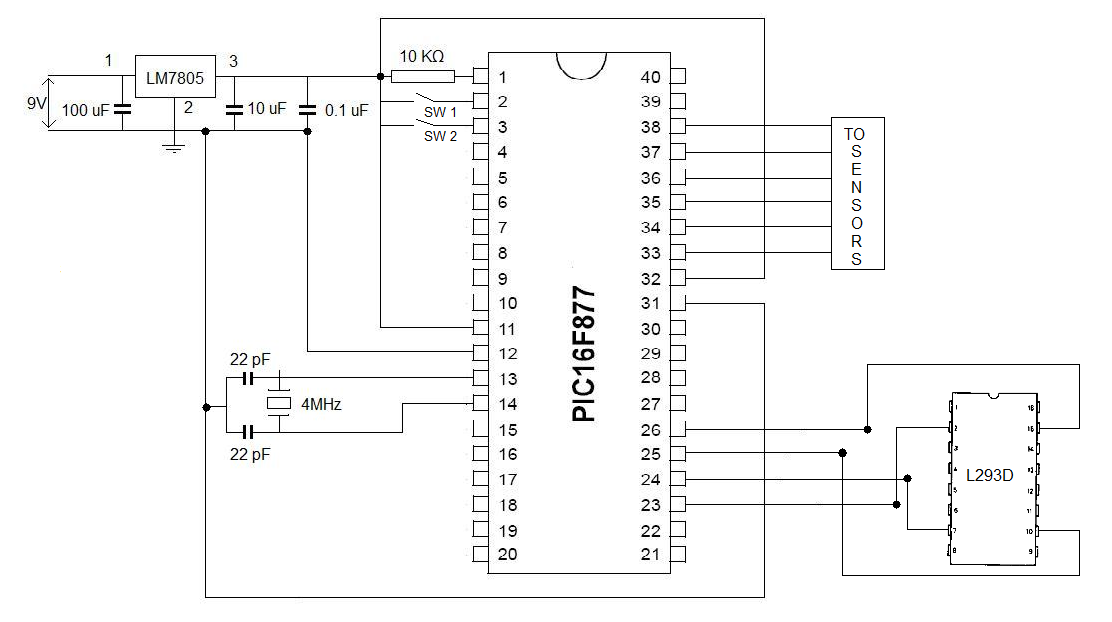


Figure 8.4 The schematics of the PIC board

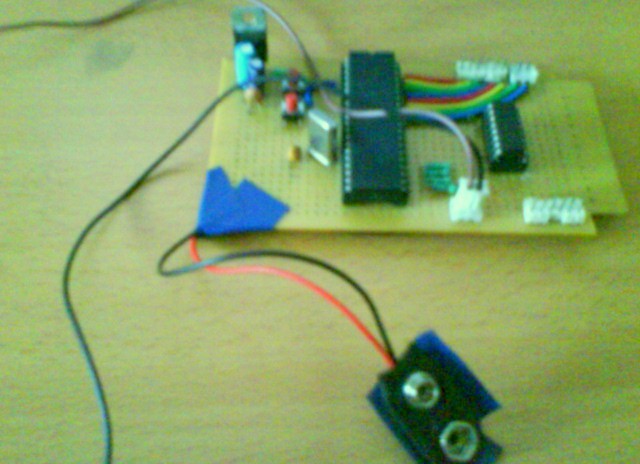


Figure 8.5 The PIC board

**8.5 THE HARDWARE MODEL**

The Figure 8.6 shows the basic hardware model of the Smart Conveyance Vehicle.

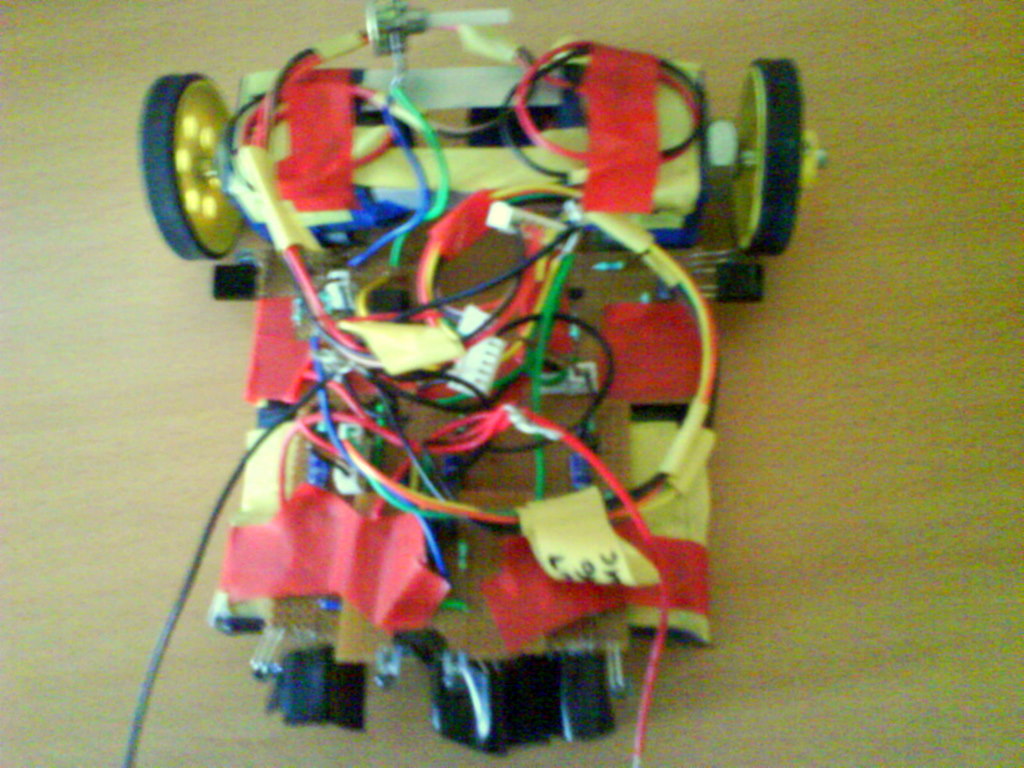


Figure 8.6 The hardware model

**CHAPTER 9**

**SMART CONVEYANCE VEHICLE**

**9.1 TESTING THE VEHICLE:**

The vehicle was tested on the path. The vehicle initially had problems. When the problems were rectified, the vehicle chose the path and reached the specified destination. When an obstacle was placed in front, the vehicle chose the alternative path and reached the specified destination.

**9.2 PROBLEMS FACED AND SOLUTIONS:**

* The vehicle had problems with decision making at junctions. The vehicle was made to run continuously, to check where exactly the error lied and later was rectified in code.
* The line sensors picked up unnecessary inputs. It is rectified with proper shielding.
* Loading of the power supply is an issue. The requirement of the current by the sensors, motors and microcontrollers is more. It is rectified by the use of separate power source for each circuit.
* The microcontroller is incapable of driving dc motor. The L293D motor driver IC is used as the interface to drive the 2 motors.

**9.3 FUTURE SCOPE**

The vehicle can be enhanced by the following features.

* The vehicle can be programmed for multiple destinations.
* Different colour sensors can be added to implement a traffic control system.
* With some modifications, the vehicle can be used as an automated waiter in restaurants, documents supply wagon in offices, medical supplies vehicle in hospitals, etc.,

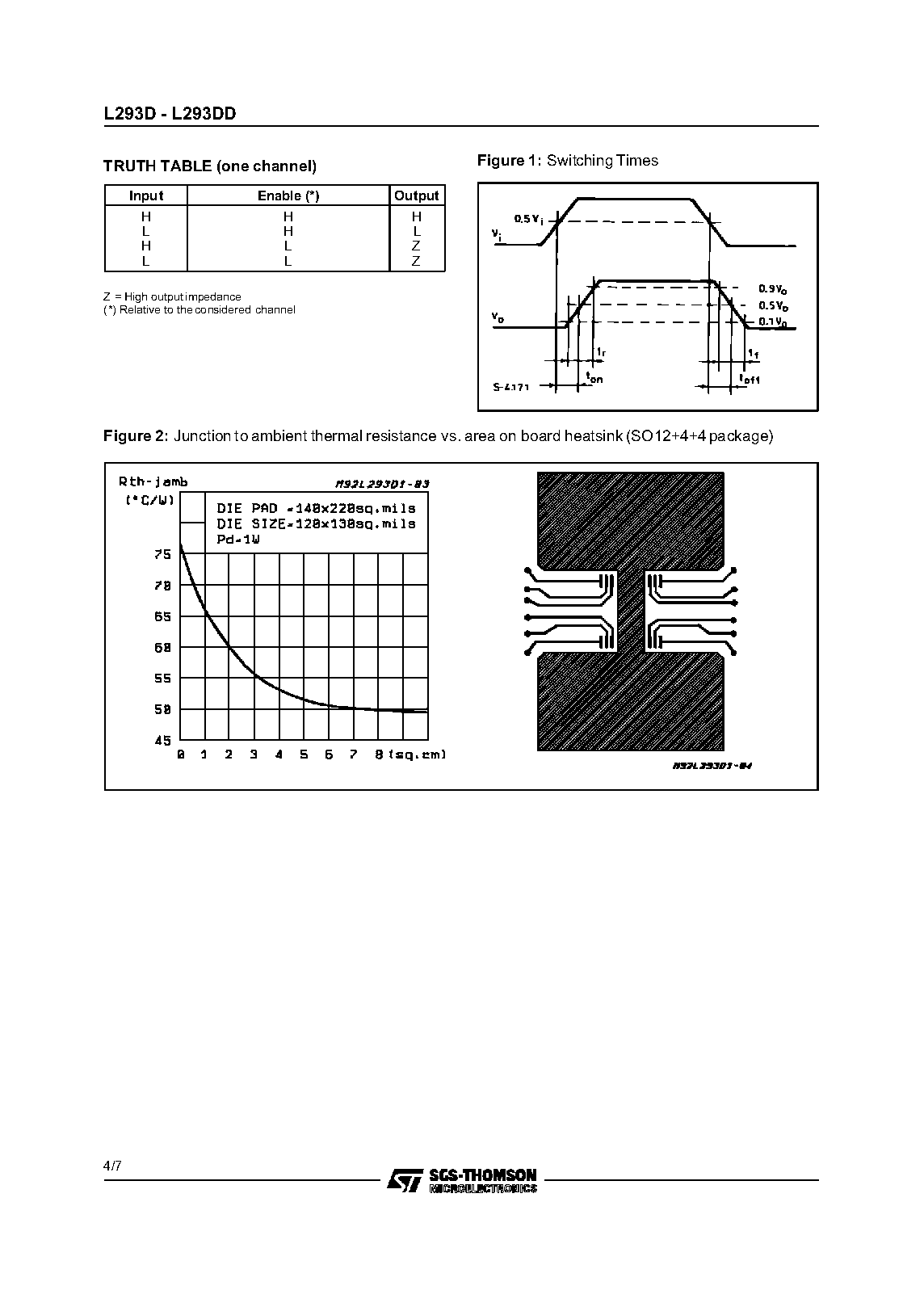
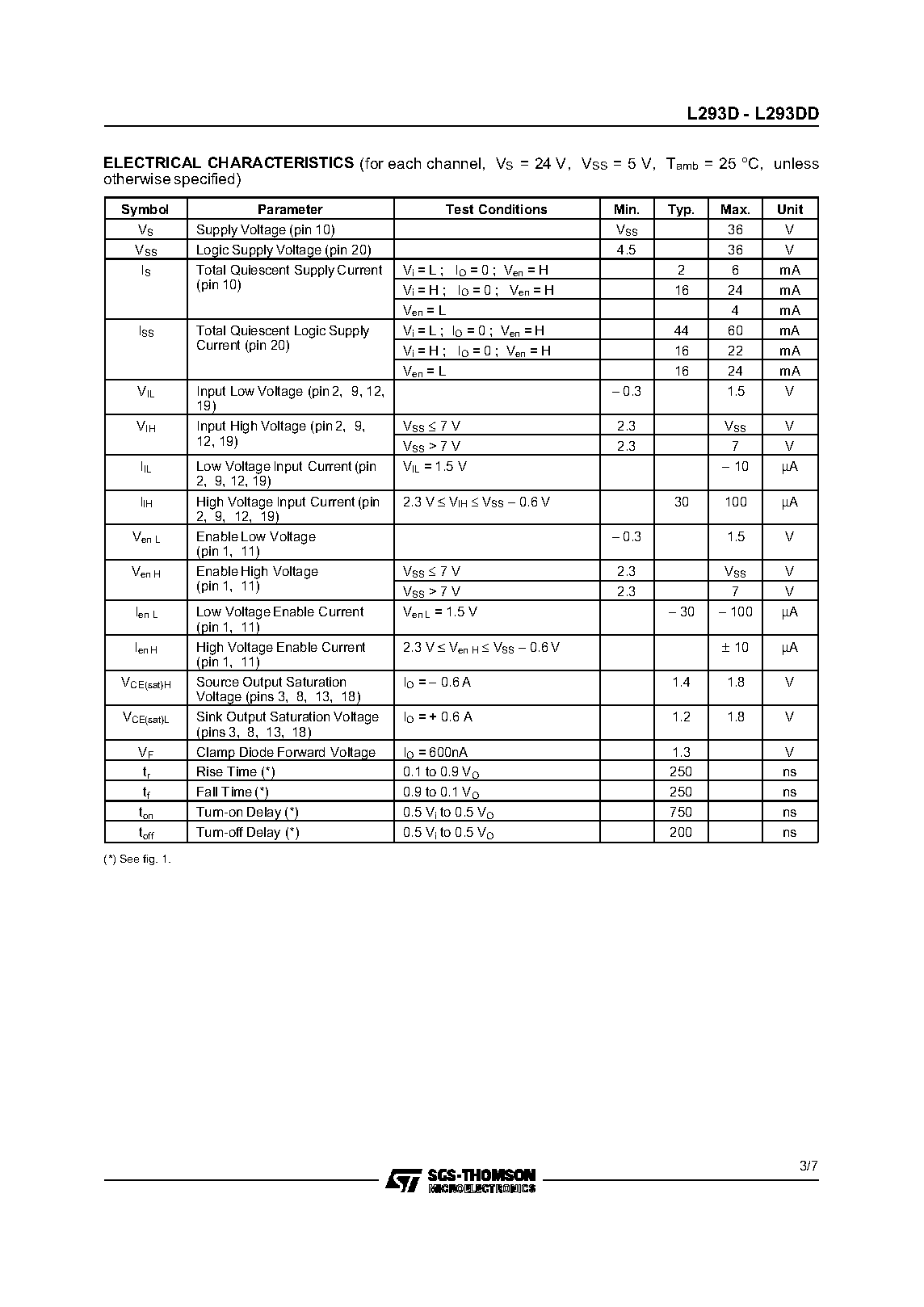
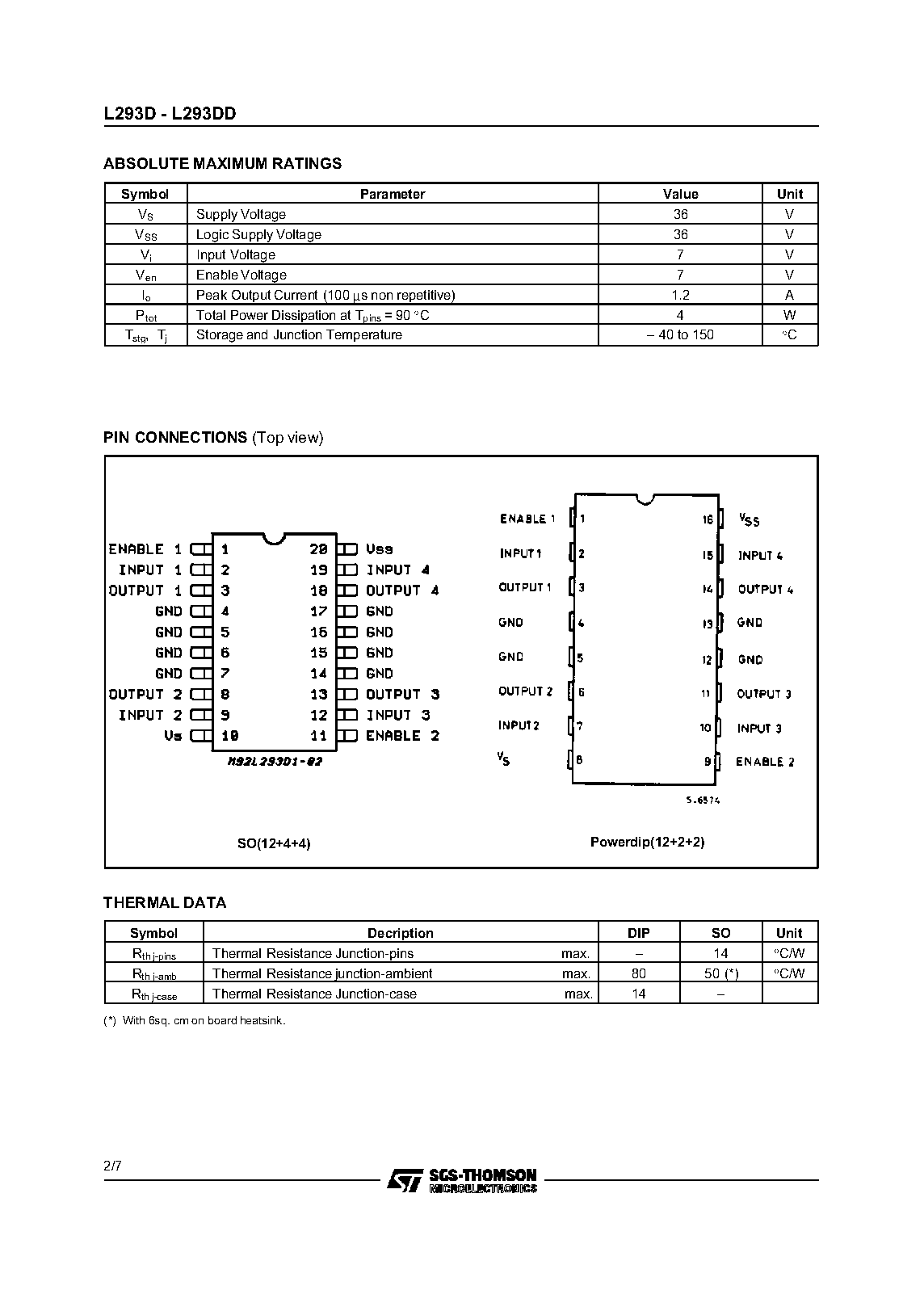
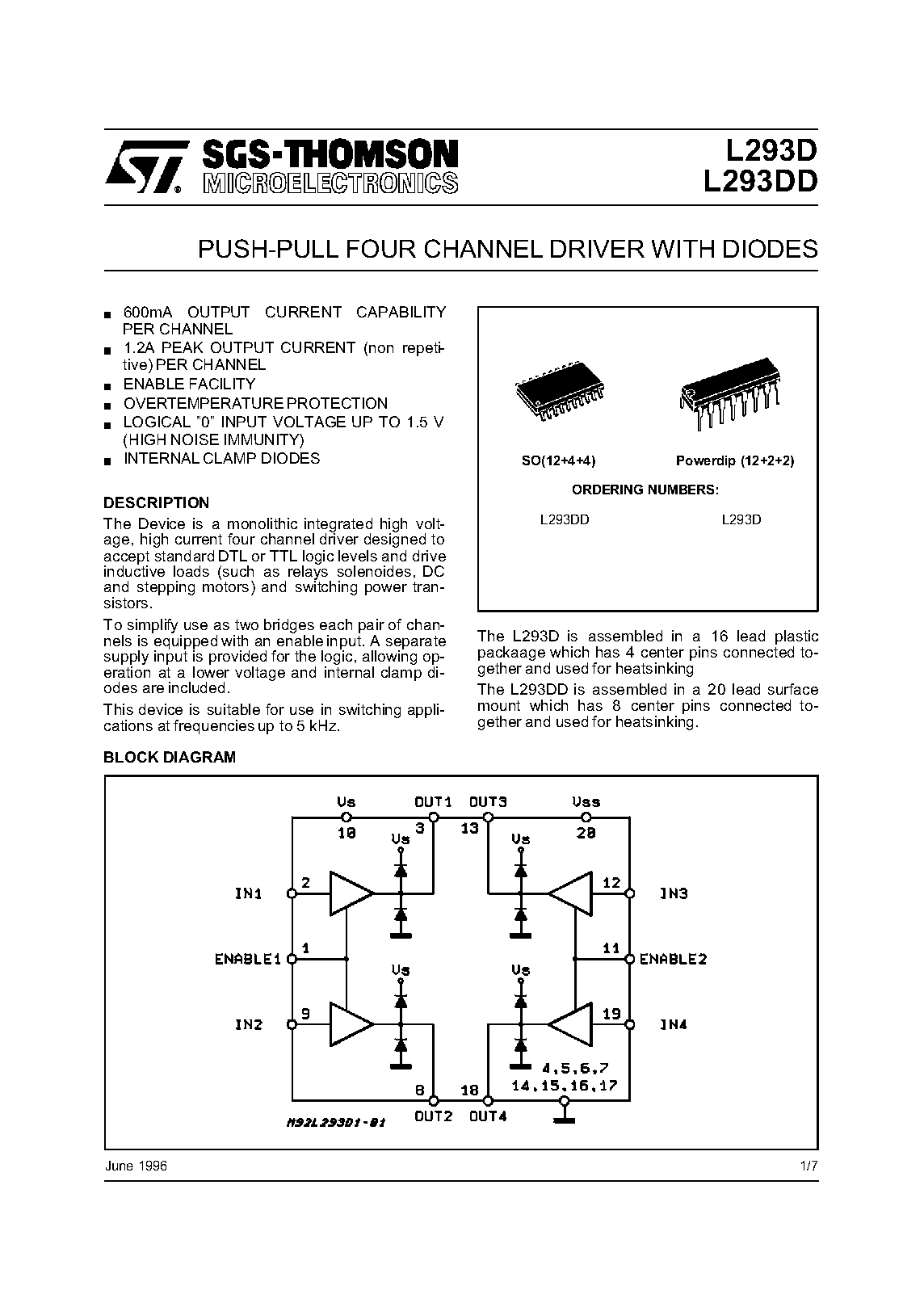
**CHAPTER 10**

**CONCLUSION**

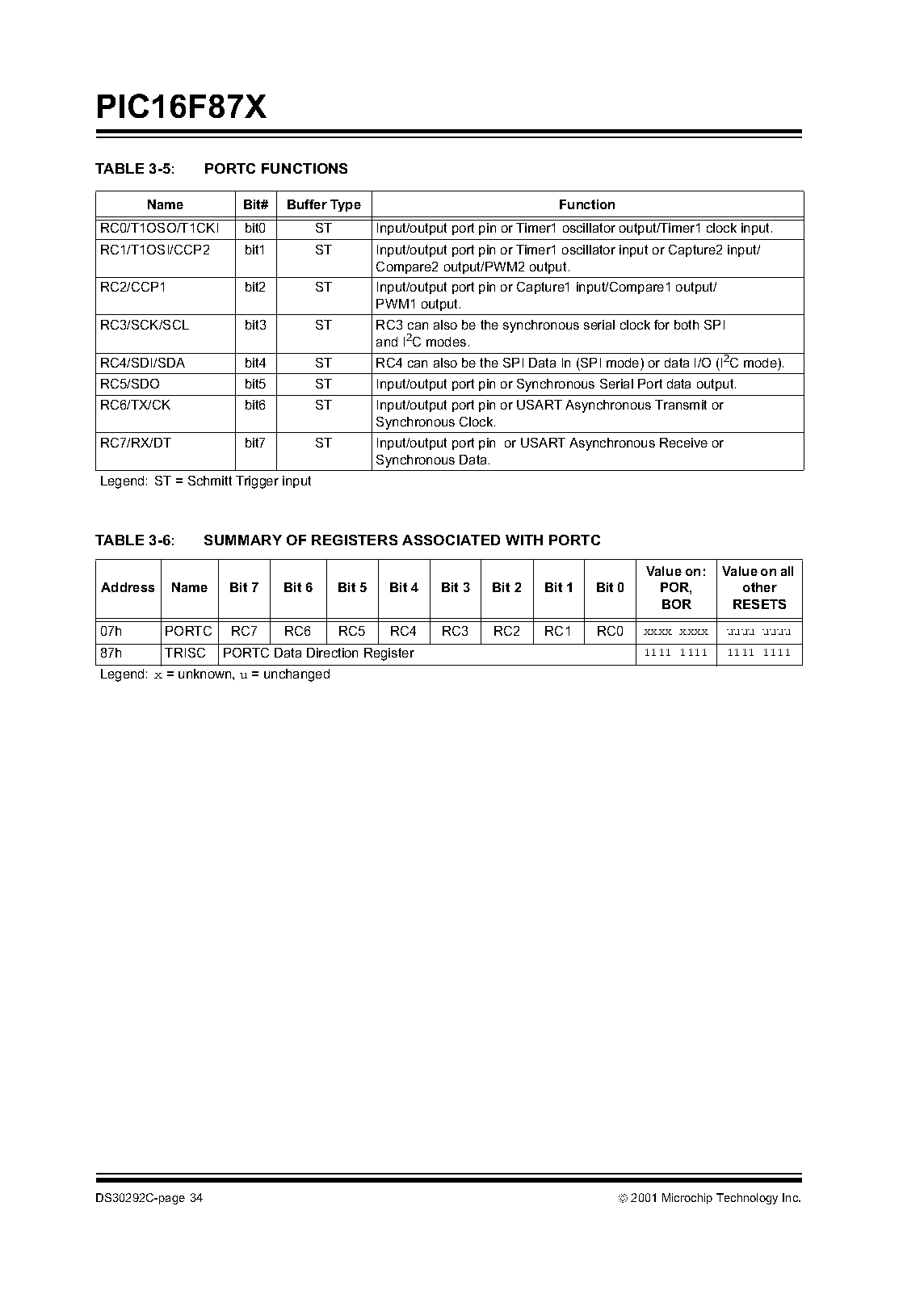
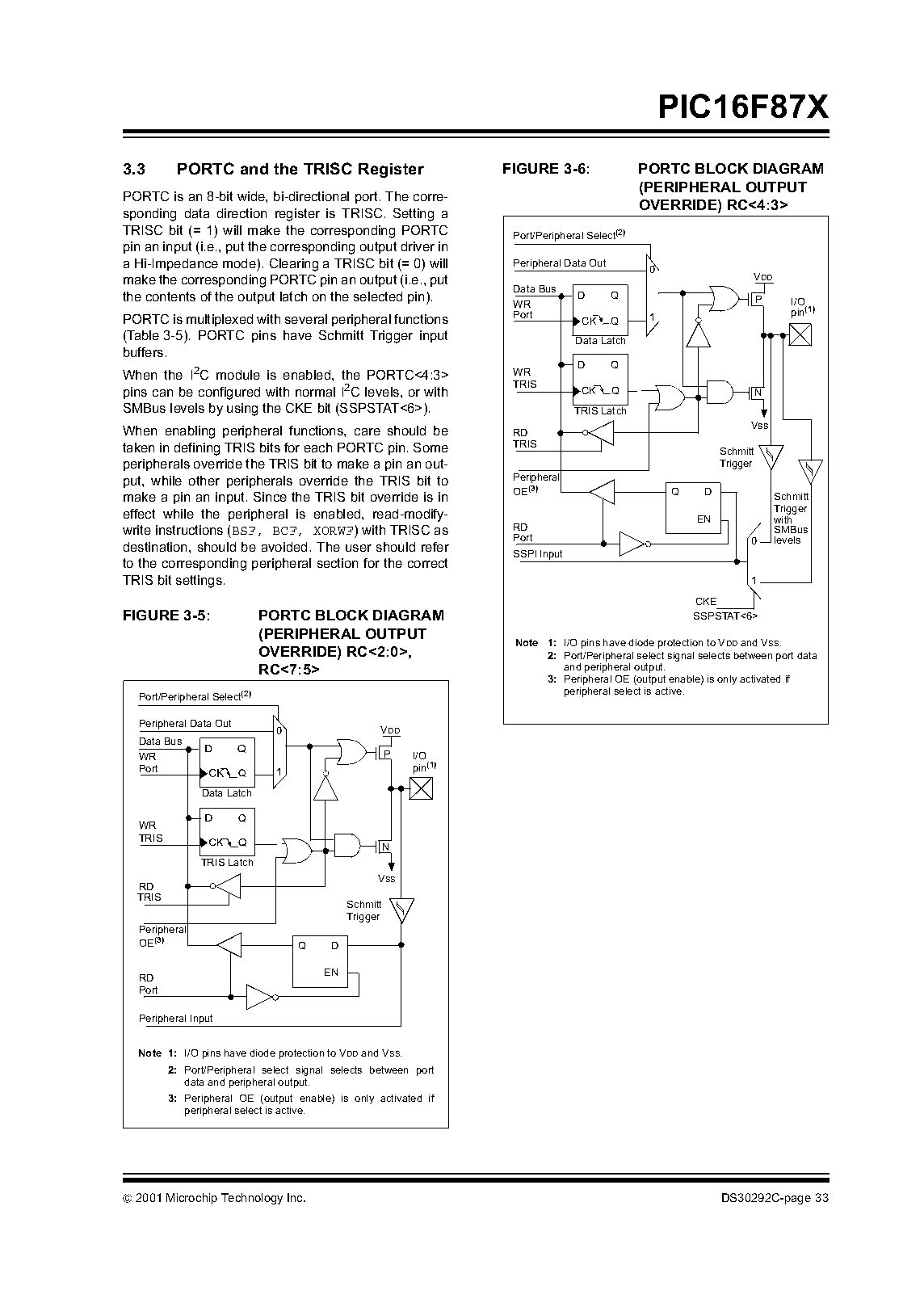
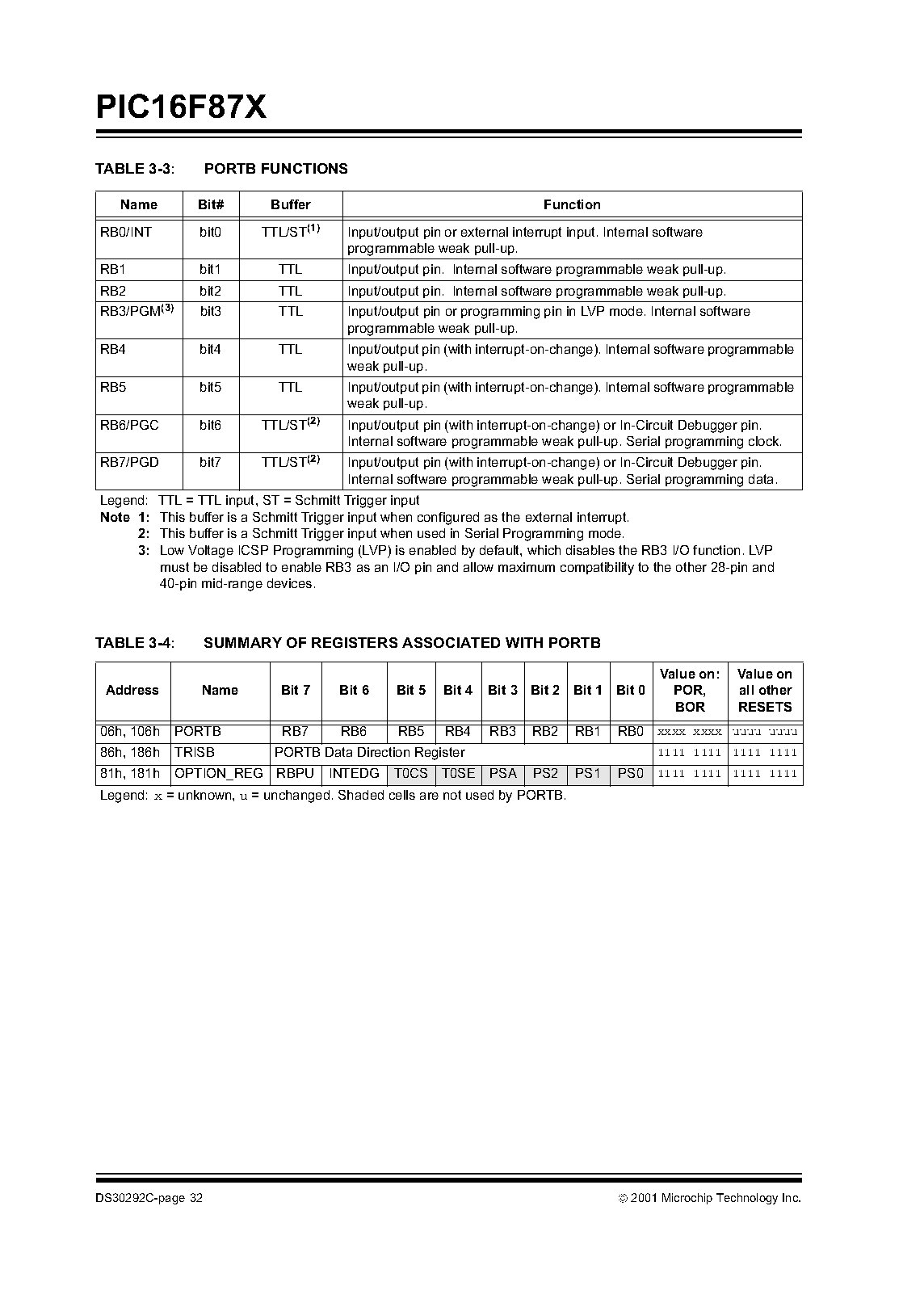
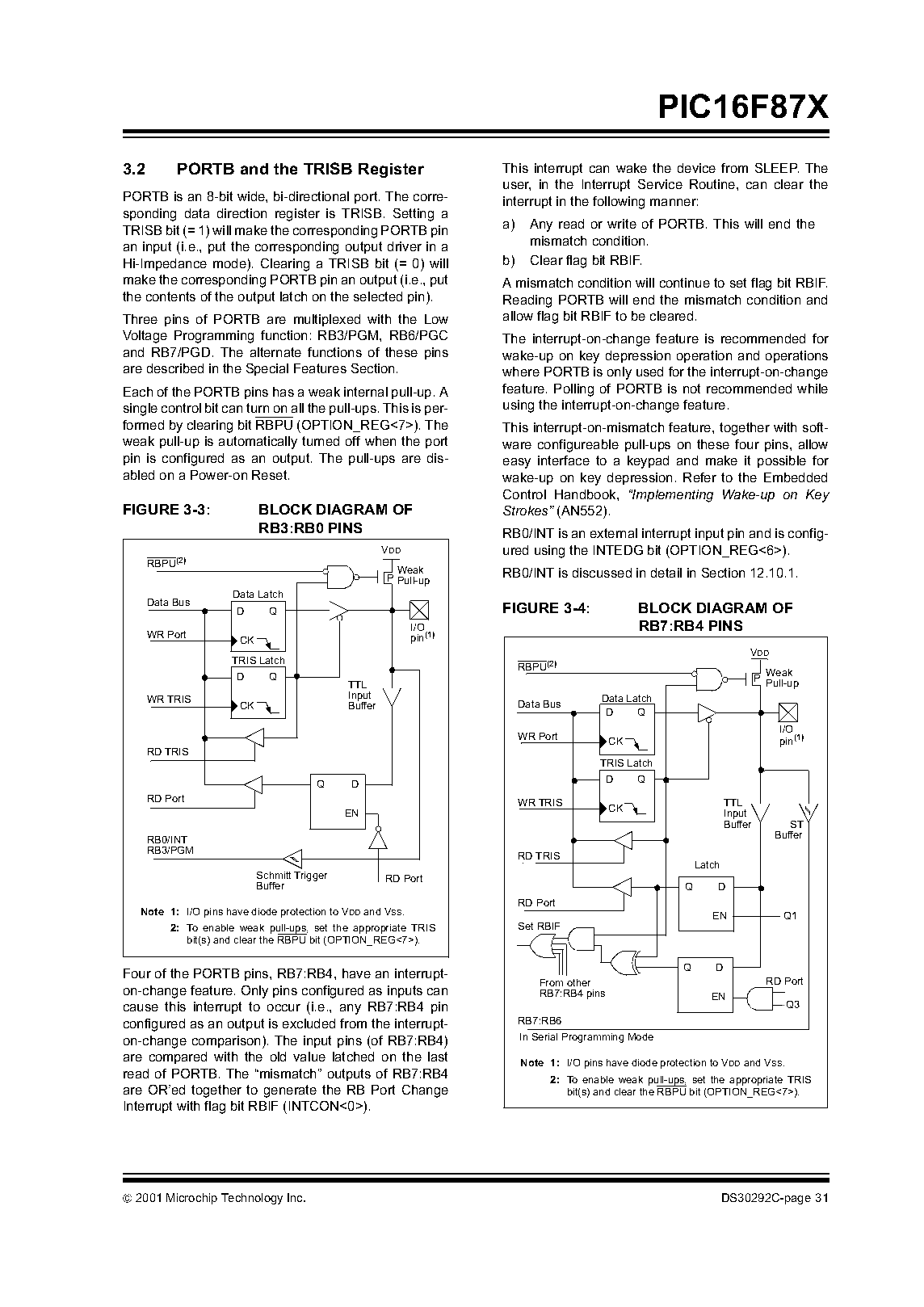
Automated path following vehicles have many potential applications in the disciplines of manufacturing, medicine and commercial settings. The need is to design highly mobile path following vehicles for a known environment. An automated goods transport vehicle in an airport can reduce the chaos and confusion that surrounds the baggage handling factor. The modernisation of every airport requires automated baggage handling as an important factor.

Smart Conveyance Vehicle gives a better baggage handling system with the good selection of motors and sensors. With the knowledge of the project implementation one can acquire the idea or plan to work on any robotic project very easily. This project gives a clear idea about the PIC microcontroller usage, the sensors and the driver circuits.

**APPENDIX A**



**APPENDIX B**



APPENDIX C

SAMPLE SOURCE CODE

#include "init.h" // included by C-Wiz

#include <htc.h>

#include <pic.h>

void initportb(void);

void initportc(void);

void Delay500us(void);

void Delay100ms(void);

void delay1s(void);

void left(void);

void right(void);

void forward(void);

void stop(void);

void reverse(void);

void linefollow(void);

static int i=0;

void initportb(void)

{

TRISB = 0b11111111; //setting portb as input

PORTB = 0b00000000; //initializing portb values to zero

}

void initportc(void)

{

TRISC = 0b00000000; // setting port c as output

PORTC = 0b00000000; // initializing port c values to zero

}

void Delay500Us(void)

{

unsigned char cnt500Us = 165; // Delay Cycle to achieve //500Us delay

while(--cnt500Us != 0) // Delay Timing is //approximately 3 usec per //loop

continue; // Note this routine is for 4MHz crystal //frequency

}

void Delay100ms()

{

unsigned char cnt100ms = 200; // 200 \* 500 Usec = 100 msec

do

{

Delay500Us();

}while(--cnt100ms != 0);

}

void delay1s()

{

unsigned char cnt1ms = 10; //10 \* 100 msec = 1 sec

do

{

Delay100ms();

}while(--cnt1ms != 0);

}

void left(void)

{

RC4 = 0b0; // right motor on

RC5 = 0b1;

RC6 = 0b0; // left motor off

RC7 = 0b0;

}

void right(void)

{

RC4 = 0; // right motor off

RC5 = 0;

RC6 = 0; // left motor on

RC7 = 1;

}

void forward(void)

{

RC4 = 0; //both

RC5 = 1; //motors

RC6 = 0; // on

RC7 = 1;

}

void stop(void)

{

RC4 = 0; // stop both motors

RC5 = 0;

RC6 = 0;

RC7 = 0;

Delay100ms();

RC4 = 1; // reverse motor for a cycle

RC5 = 0;

RC6 = 1;

RC7 = 0;

RC4 = 0; // stop both motors

RC5 = 0;

RC6 = 0;

RC7 = 0;

delay1s();

}

void linefollow(void)

{

int x=0;

while (i==0)

{

if (RB2 == 0 && RB0 == 1 && RB1 == 0)

{

forward();

}

else if(RB2 == 0 && RB0 == 0 && RB1 == 1)

{

while(RB0!=1)

left();

}

else if(RB2 == 0 && RB0 == 0 && RB1 == 0)

{

forward();

}

else if(RB2 == 1 && RB0 == 0 && RB1 == 0)

{

while(RB0!=1)

right();

}

else if(RB2 == 1 && RB0 == 1 && RB1 == 1)

{

stop();

i=1;

}

if (RB5 ==0)

{

stop();

i=2;

}

}

}

void

main(void)

{

init(); // Function call inserted by C-Wiz

while (1){

linefollow();

//TODO Auto-generated main function

}

}

**REFERENCES**

[1] Evans, J.M., “**Advanced *Robotic Systems and the Real World*”, published in IEEE transactions on** [**Intelligent Robots and Systems**](http://ieeexplore.ieee.org/xpl/RecentCon.jsp?punumber=3221), Volume: 3, on page(s): 1695-1700, Sep 1994.

[2] Frank Bond, “Mailbot*: An Autonomous Mail Collecting Robot*”, EEL 5666, IMDL, Spring 1997.

[3] Jonathan W. Valvano, Lab Manual , Page 24.1, Line Tracking Robot**,** “*Embedded Microcomputer Systems: Real Time Interfacing*”, Brooks-Cole Publishers, ISBN 0-534-36642-2, copyright © 2000.

[4] *http://www.kmitl.ac.th/~kswichit%20/LFrobot/LFrobot.htm*

[5] Thomas Braunl, “*Embedded Robotics: Mobile Robot Design and Applications with Embedded Systems*”, 2nd Edition, Springer-Verlag, Berlin, Heidelberg, 2006, pp. (XIV, 458)

[6] *http://www.triindia.co.in/resources/?p=39*

[7] *http://www.societyofrobots.com/schematics\_infraredemitdet.shtml*

[8] *http://www.societyofrobots.com/programming\_differentialdrive.shtml*

[9] *http://www.members.aon.at/electronics/pic/picpgm/*

[10]  *http://travel.howstuffworks.com/baggage-handling.htm*