1998

Computer-controlled autonomous model car: A mechatronics project

Aparna Sachidanand
West Virginia University

Computer Controlled Autonomous Model Car: A Mechatronics Project

Aparna Sachidanand

Thesis submitted to the College of Engineering and Mineral Resources, West Virginia University in partial fulfillment of the requirements for the degree of

Master of Science
In
Mechanical Engineering

Charles Stanley, Chair
B. Gopalakrishnan
Nithi Sivaneri

November 20, 1998
Morgantown, WV

Keywords: Mechatronics, Autonomous Car

Computer Controlled Autonomous Model Car: A Mechatronics Project

Aparna Sachidanand

(ABSTRACT)

Mechatronics is a synthesis of mechanical engineering and electronic engineering, and computer engineering, distinct areas that overlap in the design of systems. It represents the interdisciplinary nature of design and development of today's products.

The current research focuses on the design, construction and testing of a computer controlled autonomous model car which can exhibit intelligent behavior such as timed course execution, obstacle detection, and response to sensor inputs. The car is intended as a mechatronics design project that will be integrated into an existing one-semester mechanical engineering undergraduate instrumentation course.

The car was designed around a microprocessor board (Tern Analog Drive) controlled by a 16-bit microcontroller (Tern V104) and equipped with several sensor channels. Two stepper motors were used to propel and guide the car. Photocells were used to detect the path. The control program was written in Turbo C.

The car was tested on a path of reflective white tape about 2 inches wide. The path consists of a 36-inch straight portion followed by a 17-inch radius of curvature curved portion, and completed by a 6-inch straight section with an obstacle at the end. The autonomous car successfully traversed the path and stopped when it detected the obstacle.

It was concluded that a successful mechatronic design project could be developed around the construction and testing of an autonomous car.

ACKNOWLEDGEMENTS

I would like to thank my graduate committee member Dr. B.Gopalakrishnan for his encouragement and guidance on this project. I would like to thank my committee member and associate chairman of the department of Mechanical and Aerospace engineering, Dr. Nithi Sivaneri for his constant encouragement during my master's program. I would like to express my gratitude and appreciation to my advisor and committee chairman Dr. Charles Stanley for his guidance, support, and suggestions during this research project and his constant encouragement during the course of my graduate studies.

I would like to take this opportunity to thank Mr. Chuck Coleman and Mr. Cliff Judy at the mechanical engineering labs for their suggestions and help during this project.

I would like to thank my friend Henry for his valuable suggestions. Thanks are also due to friends Jill and John for their encouragement. Finally, I would like to thank my family for their support and inspiration.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Mechatronics is the term used for the integration of mechanical and electronic engineering disciplines to achieve the objective of developing efficient systems controlled intelligently. The International Federation for the Theory of Machines and Mechanisms defines Mechatronics as the synergistic integration of mechanical and electronic engineering, electronic controls, and software in the design of products and manufacturing processes.

Mechatronics is the totality of fundamentals, procedures and techniques for the service, production and development of future-oriented objects. Thus Mechatronics is an interdisciplinary technical discipline, built upon the basis of classical mechanical, electrical and electronic engineering, binding these sciences not only with one another, but also with computer science and software engineering. The central focus of Mechatronics is the integral development of systems from technical components ("Mecha"), which are to be intelligently controlled ("tronic"). Thus a system composed of mechanical and electrical parts, overlaid with sensors which record information, microprocessors which interpret, process and analyze the information and assemblies which then react upon this information, becomes a complete mechatronic system.

Chapter 2

Mechatronics

2.1. History and Evolution of Mechatronics

Mechatronics is a rapidly evolving field. The term 'Mechatronics' is said to have been coined by Japanese engineers about twenty years ago. However, the actual collection of subjects and ideas that make up Mechatronics is not new. These ideas have existed, for example in the aerospace industries and have been successful for many years. The development of economical computational power and intelligent power electronics is the main reason that this interdisciplinary point of view has been used for the development of new products outside the scope of aerospace engineering.

The term "electro-mechanical system" has been used by English speaking engineers to indicate a new category of electro-mechanical systems for many years. Since computerization, a technology changing rapidly is one of the technologies, which define the field; evolution of mechatronics has been rapid. Until about twenty years ago, the mechanical engineering field has employed traditional techniques for product development and manufacturing. With the advent of semiconductors, the integration of machines and manufacturing with electronics has developed rapidly. Microcomputers, as they influenced other technical disciplines, have rationalized the thinking process in the mechanical engineering field. The concept of 'intelligent machines' was popularized by the introduction of small and inexpensive microprocessors as integral parts of machines. These microprocessors, embedded in the machinery enable the machine to independently think and make decisions.

2.2. Scope of Mechatronics

It is evident that product development in mechanical engineering has undergone radical changes in the past few years. Most of these changes are due to the introduction of mechatronics in the process. The addition of some inexpensive electronics and a simple computer can radically change the functionality of a machine. Mechatronics can

help in achieving the objectives of concepts like concurrent engineering, and design for manufacturability for product design. Thus, mechatronics can also be thought of as an application of the concept of concurrent engineering to the design of electromechanical systems.

The scope of mechatronics ranges from consumer and commercial products, laboratory test instruments and equipment to industrial applications. A range of examples might include automatic video cameras, the Compact Disc player, photocopier, and a car engine with emission sensors and computer controlled injection. It can be expected that the majority of everyday consumer products will be computerized in the near future. Thus, mechatronics has applications in almost all technological fields.

2.3. Need for Mechatronics Education

Since the very definition of mechatronics suggests an integration of various engineering disciplines, educating engineers from various fields in mechatronics is important. With the spread of mechatronics to engineered products and systems of all kinds, it is imperative that engineering students be exposed to its principles and practices. Engineers will encounter mechatronic systems in their professional practice, regardless of their discipline. In order to participate and contribute in all stages of engineering design, from conceptualization to final product design, a working understanding of the capabilities and limitations of mechatronics is essential. The increasing complexity due to the use of microcontrollers and microprocessors in a wide variety of consumer and commercial products, laboratory equipment and industrial applications has created a need for mechatronics education in all fields of engineering education. Since a mechatronic system is a combination of mechanics, electronics and software, a mechanical engineer of today must be aware of the function of electronics and software in the system. Teaching mechatronics to non-mechanical engineers aims at familiarizing them with the design and practical aspects of mechanical engineering because mechatronics exemplifies an integrated design approach where electrical, electronic, computer and mechanical subsystems are simultaneously designed to function as an integrated single system. Thus mechatronics education is very important for various engineering disciplines.

The teaching method for mechatronics in various engineering disciplines is still an open area for discussion. The subject of mechatronics is broad, encompassing and interdisciplinary. It is generally agreed that the curriculum should be organized to give incentives to the students for mechatronics design and make up for lack of experience with mechanics and electronics. However, the design of mechatronic systems should not be viewed as a replacement of mechanical components by electronics and software.

To achieve mechatronics education in various engineering fields, educators have experimented with several approaches. One approach is to introduce mechatronics material into all four years of engineering studies. The second approach is to design a course in mechatronics, which is taught after the student has finished various basic prerequisite courses. A compromise is a course, which combines mechatronics with other topics like microprocessors, control engineering etc.

In the first approach, no new mechatronics courses are designed for general engineering students, and mechatronic material will be integrated into existing courses and programs. The other two methods work similarly, by introducing mechatronic material with other topics. For mechanical engineers, mechatronics could be integrated with either microprocessor or controls material.

The objective of mechatronics courses is to produce engineers capable of successfully employing electronics, microprocessors and software in the design of electro-mechanical systems. The goals, approaches and emphasis of mechatronics courses will depend on the discipline and level (graduate or undergraduate) of students, and duration for the course. Since computerization is an integral part of mechatronics, it is important to keep the curriculum current with the rapidly changing computer technology and on the leading edge of design methodology. The curriculum for a mechatronics course can consist of lecture periods and laboratory sessions, which generally also include design projects.

Chapter 3
Objective

The objective of this research is to design, construct and test a prototype of a mechatronics design project that will be integrated into an existing one-semester mechanical engineering undergraduate instrumentation course. The design project involves groups of students who design, construct, and test a small self-contained microprocessor controlled car. The cars are competitively evaluated.

The specifications were developed from the previous class projects. The prototype car developed in this research must comply with all of the constraints contained in the design project. Specifically,

1. The prototype must utilize the concepts of microprocessor interfacing, i.e., analog to digital, digital to analog, digital input and digital output.
2. The prototype must use specific microprocessor control boards.
3. The prototype must execute a timed course defined by reflective tape. The timed event will be started by momentarily turning off room lights.
4. The prototype must be capable of competing in a ‘tug of war’.
5. The prototype must execute a timed obstacle course. The prototype must be able to detect an obstacle and stop before the main body of the car contacts the obstacle.
6. The prototype must be less than 12 inches long and less than 6 inches wide.

These specifications were developed based on the previous class projects.

Chapter 4
Literature Review

The term mechatronics refers to a multidiscipline, integrated approach to product design and manufacturing. Mechatronics represents the synergistic integration of mechanical and electronic engineering, electronic controls, and software in the design of products and manufacturing processes. Mechatronics is the foundation for the next generation of machines, robots and intelligent systems in a variety of environments.

This review studies past work in the field of mechatronics education. Educators all over the world have recognized the need for mechatronics education in various fields of engineering. Since mechatronics is the integration of various engineering disciplines, objectives of mechatronics education can be achieved only with interdisciplinary communication and efforts. Educators have experimented with several approaches to mechatronics education. This study attempts to explain curriculum development, teaching methods, mechatronics system design, and performance studies carried out by several educators and researchers.

4.1. Mechatronics- What, Why, How?

Harashima [1] defines and answers questions about mechatronics in his editorial. The work discusses the history of mechatronics, a projection of mechatronics into the future, and the scope and direction of the new journal IEEE/ASME Transactions on Mechatronics.

Auslander [2] attempts a definition of mechatronics that will allow for its differentiation as an identifiable field of engineering and follows this direction to discuss some of the associated philosophy. The definition focuses on the centrality of computing in mechatronic systems.

Kyora and Oho [3] discuss mechatronics from an individual perspective. The term mechatronics is defined to provide a framework for technical and practical considerations. The importance of intimate and organic integration in mechatronic product designs is raised.

Mechatronics is also defined as the totality of fundamentals, procedures and techniques for the service, production and development of future-oriented machines, devices and installations. The work discusses integration of mechanical, electronic and software engineering disciplines to form mechatronics. The history of mechatronics is also discussed.

4.2. Mechatronic System Design

Isermann [4] discusses the integration of mechanical systems and microelectronics and its implications in process design and automatic functions. The mutual interrelations between the design of the mechanical system and the digital electronic system, and the different ways of integration within mechatronic systems and the resulting properties are described. Some examples highlighted are products with precision mechanics and integrated electronics such as devices for telecommunication, consumer electronics, data processing services, sensors and actuators, and optical devices. Two levels of integration namely hardware integration and software integration are discussed.

Youcef -Toumi [5] presents a modeling and control system perspective relevant to a systems approach and fast adaptive control systems. Topics such as graphical system representation, achievable performance and fast control algorithms are discussed.

Craig [12] discusses the issues of mechatronic system design at Rensselaer Polytechnic Institute. The work focuses on questions of what constitutes a winning design and what constitutes a successful designer. The work describes a two-course, senior elective, which addresses these two questions.

According to Craig, if winning designs are to be produced in today's world, it is imperative that electronics and computer control be included in the design process at the same time the basic functions and properties are defined. The benefits of mechatronic approach are shorter development cycles, lower costs, increased quality, reliability, functionality and performance. These are fundamental areas of technology on which successful mechatronic designs are based. The theme for the course is the integration of these key areas into a successful design. The question of what constitutes a winning design is addressed in the courses. It is clear that practicing design is an integral part of

teaching design. After a student is taught analytical techniques and component technology and studies elements of the design procedure, the student learns good design methodology through experience. The second course, mechatronic system design allows the student experience design.

The first course, Mechatronics, covers control sensors and actuators, interfacing sensors and actuators to a microcomputer, discrete and continuous controller design, analog and digital controller electronics, and real-time programming for control using the C programming language. Throughout the course, mechatronic design principles are stresses and are reinforced through the reverse engineering process of successful mechatronic products and systems. This course is heavily laboratory based, with the course material presented so that information discussed in lecture is immediately reinforced in the laboratory. A wide variety of analog and digital sensors and actuators are used in the laboratory exercises. The students learn basic electronic devices such as passive components, op amps, digital logic, etc. Several laboratory sessions early in the course are devoted to connecting and operating basic devices.

The second course, mechatronic system design, addresses the 'design and build' aspect of mechatronics. The major emphasis of this course is microprocessor-based detailed mechanical design from concept through construction and testing. This course, mechatronic system design, lets the student experience design. Real world design problems from industrial sponsors constitute the projects for the course. In this course, the students are divided into design teams for the laboratory exercises and design project. The first third of the course focuses on mechatronic design principles and microprocessor hardware, software, interfacing, and the lectures cover the systematic approach to mechatronic design. This approach is illustrated and reinforced with a step-by-step case study of an actual microprocessor based mechatronic product complete with detailed design drawings and a working prototype. At the same time that these steps are discussed and illustrated, the design teams are performing these steps through the conceptual design phase for their industry-sponsored design problem. The remaining two thirds of the course is devoted to the detailed design, construction, and testing of the team's conceptual design. The ultimate goal of the course is to have a working prototype

accompanied by a written design report including drawings, schematics and documentation.

4.3. Mechatronic Equipment

Kyura [6] describes the classification of mechatronics equipment into four groups. The first group is the addition of sophisticated controllers using electronics to provide higher performance or enhanced functionality. Examples are NC machine tools and industrial robots. The second category is the replacement of mechanical controllers with new controllers combining mechanical and electronic components as in electronic sewing and knitting machines. The third group is the replacement of mechanisms designed primarily for information handling with electronic systems, such as digital watches, desktop calculators. The last category covers products such as home appliances driven by microprocessors, items with extremely simple mechanisms and electronic components.

4.4. Microprocessors and Sensors

A microprocessor is described as the heart of a computing system. The microprocessor in a mechatronic system functions similarly. Sensors are as important in a mechatronic system as the senses are to the human being.

Luo [7] discusses the sensor technologies and microsensor issues for mechatronic systems. The role of sensors in the field of mechatronics is identified and different sensors are characterized. The work examines the surge of microsensors and the multisensor fusion. These principles are normally used in military applications and are not being applied to consumer products.

Prabhu and Wright [8] illustrate the use of microcontrollers in mechatronics education. The work describes experiences in the use of microcontrollers in the teaching of a laboratory driven undergraduate course in microcomputer based system design. The time related functionality of Motorola 68332 microcontroller, together with the opportunity to program in C aids in mechatronics education for non-computer students, thus not requiring the students to learn fine architectural details and assembly language programming.

Shoureshi [9] has highlighted the issues for teaching mechatronics in the 90's. He emphasizes usage of an 8-bit single board microcomputer as it simply and elegantly illustrates the basic elements of microprocessor architecture, assembly language programming and interrupt driven controls.

Ume and Timmerman [10] discuss how mechatronics is taught at Georgia Tech. The work discusses the course structure, which includes design at the microchip level, and assembly language programs for measurement. Laboratory experiments are designed to provide hands-on experience to the students in interfacing sensors, actuators, and passive and active devices with microprocessors and microcontrollers.

This course presents both software and hardware design simultaneously, and stresses the need for an integrated, multi-disciplinary approach in learning microprocessor design. Elements of the course include:

1. Lectures in digital arithmetic, analog and digital circuits, assembly language programming techniques, microprocessor structures, analog/digital conversion and advanced topics such as I/O design an interrupt driven processing.
2. Laboratory experiments designed to teach I/O between the microprocessor board and the host computer (usually a PC) and other experiments designed to explain topics like interrupts, data output, data input, and analog conversion.
3. Short tests on the lecture materials.
4. A final project that encourages the students to be creative and to integrate and to relate both software and hardware aspects of microprocessor design with knowledge acquired in more traditional areas of mechanical engineering analysis. The final project encourages students to learn to work together in a goal-oriented situation.
5. Case studies of industrial problems intended to convey the flavor of the role of microprocessors in professional practice.

The laboratory experiments are designed to illustrate particular topics in the curriculum instruction:

1. The first experiment involves writing a short program, assembling it, and downloading and running it on the Motorola 68HC11 EVB board. This teaches the basic operation of the EVB.
2. The second experiment teaches the use of the general I/O ports for data communications. The students write code to turn a bank of LEDs on and off through a data port in a certain sequence representing counting from hex 00 to hex FF in binary form. The experience of seeing the LEDs blink on and off in response to software physically reinforces the concept of a digital port and its use.
3. The third lab focuses on A/D conversion and data manipulation by software. The students use a strain gage to measure the stress in a cantilever beam assembly. This strain signal is amplified and digitized through the EVB's A/D converters. The strain signal is further manipulated in the EVB to correspond to the millivolt value of the strain signal. This signal is finally sent to the host PC as ASCII text. The use of a familiar mechanical sensor allows the students to understand the nature of a digitized analog signal better.
4. The fourth lab involves the construction of an input and output interface to measure the speed of a small DC motor and to control its speed through a pulse-width modulation scheme. This lab illustrates advanced capabilities of the EVB like input and output capture and event timing. Additionally, the programs for the lab illustrate the use of advanced program concepts like vector interrupts and complex branching. This lab challenges the students and presents them with a true closed-loop control situation. The motor with its visible and audible speed and stability and other concepts.

   Each lab requires the construction of a circuit on a breadboard, the writing and debugging of some assembly code, an experimental demonstration, and a write-up.

Morin [11] describes evolution of a course oriented towards the use selection and design of microprocessor systems which stresses the modular nature of this technology. The course was built around the two-board Motorola D2 evaluation kit, which uses the 8-bit 6800 processor.

4.5. Curriculum Development for Mechatronics Courses

The increasing complexity of microcontrollers and microprocessors in a wide variety of consumer and commercial products, laboratory equipment and industrial applications has created a need for mechatronics education in various engineering fields, educators have experimented with several approaches. This section describes some of the mechatronics curriculum development approaches.

Hajnal [18] provides a brief survey on mechatronics education in Hungary, emphasizing the specialty of that region. The paper discusses the curriculum of the five-year informatics course. The course includes seven mechanical engineering related subjects and a review of these subjects is provided.

Memis [19] discusses mechatronics education in the United Kingdom. The mechatronic engineering courses have rapidly increased in undergraduate and postgraduate levels as well as vocational training courses in recent years. This paper provides brief reviews of the research and postgraduate activities in European and U.K higher education institutions. The paper compares the two approaches to engineering education: generalist engineering versus specialist engineering.

Carryer [20] discusses the design of laboratory experiments and projects for mechatronics courses. The paper describes a set of guidelines that have been developed and employed to aid in the design and review of laboratory experiments and open-ended design projects for use in both undergraduate and graduate levels. The context of the courses in which these guidelines were developed is described and some examples of the experiments and projects that have resulted are presented.

Sasaki [13] describes lectures, exercises and experiments related to mechatronics. This paper describes mechatronics education at the Department of Precision Machinery

Engineering at the University of Tokyo. The curriculum is organized to give incentives to the students for mechatronics design and make up for the lack of experience with mechanisms and electronics. The curriculum focuses on mechanical engineering students. The curriculum of the precision machinery engineering department is organized to cover the following fields: Mechatronics, Production Engineering, CAD/CAM, Metrology, Medical Engineering.

The curriculum for Mechatronics education includes lectures, design assignments and laboratory experiments. The 90-minute lectures given once a week cover the topics listed below:

- Mechanics: Statics and Dynamics of mechanisms
- Mechatronics 1: Mechanical components and mechanisms (link, gear, cam, belt, screw, etc)
- Mechatronics 2: Sensors for mechanical systems (position, velocity, acceleration, and force), Operational amplifier, Signal conditioning, AD/DA conversion
- Mechatronics 3: Electromagnetic actuators, Pneumatic actuators, Electrostatic actuators
- Robotics: Kinematics and dynamics of manipulators, Sensors for robots, Robot programming, Artificial intelligence
- Electrical Engineering 1, 2
- Control Theory 1, 2
- Optics and Image Technology
- Strength of Materials
- Vibration Theory

The design assignments provide the students with many practical aspects in mechatronics design. The assignments are:

Direct Drive Robot: design a two degree-of-freedom direct drive robot.

Artificial Respirator: the artificial respirator in this case is a simple pneumatic pump whose motion of the piston is computer controlled. This mechanism was chosen because of the course that is offered in medical engineering.

Analysis of Compact Disc Player Head: The class is divided into groups of four to five students. An optical head for a compact disk player is provided to each group. An instructor is invited from a compact disk player manufacturing company. The instructor gives lectures on the mechanism and control issues on tracking and focusing of an optical disk. Students are required to dissemble and reassemble the mechanism to analyze the structure and components.

Machines for competition: the assignment is to design a machine for the design competition at the end of the semester. The actual fabrication of the machine and the competition are non-credit activities. Only the design part will be evaluated as the course.

Single Axis Feed Table for NC Machine: this design assignment is based on real products, which involves many practical aspects of mechanical design. Instructors, who work in the design division of Toshiba Machine, are invited. Students are required to select motors, gears and a ball screw that satisfy the given specification.

Application of Optical Shaft Encoder: The assignment is to design any kind of machine that uses optical shaft encoder(s). No restriction is imposed on the function of the machine. The objective of this assignment is to learn how to break down the given function into detailed specification. An instructor from Nikon is invited.

The course includes three laboratory experiments, which address various aspects of mechatronics, for seniors. Each experiment consists of four small experiments conducted by four to five students

1. Control of an inverted pendulum:
    a. Basics: controlling I/O of PC, Up/down counter for optical shaft encoder, AD/DA converters, interrupt timers
    b. Measurement of parameters: moment of inertia of the pendulum, step response of the motor, viscous friction in the mechanism
    c. Simulation: formulation of state equation, calculation of feedback gains, design of observer, simulation using MatLab
    d. Control Equipment: writing control program in C, comparison of response and stability using the gains obtained in the previous experiments
2. Design of an approximate linear mechanism

a. Design and analysis: input parameters into Pro/ENGINEER, a CAE software for kinematic analysis of mechanisms, change parameters and observe the difference in trajectory, print out mechanical drawings of designed parts
b. Fabrication of designed links: program the NC machine, fabricate the links
c. Examination of the trajectory: assemble the mechanism using the fabricated parts, examine the trajectory of the mechanism

3. Control of an inverted pendulum:
   a. Operational amplifiers: inverting and non-inverting amplifier, comparator
   b. Fabrication of frequency counter
   c. Measurement of blood flow: measurement with a dummy, a mixture of tiny glass beads and water running inside a pipe, measurement of blood flow in the vein of one's arm

An industrial experience program is provided for the juniors during summer recess. The length of the program is two to three weeks. Visits to factories and laboratories are good opportunities for the students to gain a realistic image of what they are learning. One visit is usually a half-day-tour to factories in the vicinity of Tokyo.

Harwin and Foulds [14] describe mechatronics education at the University of Delaware. The work discusses the course syllabus for a course in mechatronic design. The course stresses the interdisciplinary nature of mechatronics as well as providing a hands-on design experience and lessons in resource management. The course emphasizes the need for student exposure to problem-based education which broadens the scope of traditional disciplines. The course not only does cross discipline teaching but also introduces new topics such as formal notations in software design, real-time computing systems, gait analysis and novel robotics. The work discusses projects, the introductory project is based on the six-legged walking insect and the primary design project is the design and prototype of a haptic display.

The students who enrolled for the course were from computer science, mechanical engineering, electrical engineering and other disciplines and claimed to have experience of varying degree with the disciplines. The wide background of students made teaching a

challenge. A primary reference was prescribed to familiarize the students with the electronic engineering concepts. There was also a strong emphasis on peer teaching, both informally and through a formal presentation each student is required to give on an aspect of the material. Usually students picked a presentation topic from their primary disciplines, thus enabling interdisciplinary information exchange. The course has three parts, a taught component, an introductory project and a more substantial design project.

The introductory design project is based on the Stiquito, the six legged walking insect developed by Jonathan Mills at Indiana University. The basic Stiquito is a six legged insect like walking machine using Nitinol wires as actuators and costing about $3 in parts. These kits are supplemented with a 8051 based microcontroller and the electronic components to interface this to the Nitinol actuators. The students are asked to design and build autonomous variations on the basic Stiquito that can compete without intervention in three 'Olympic' events. The events are a 30cm flat race, an S shaped obstacle course and a tug-o-war.

The primary design project lasted the remainder of the semester and reflected the interests of the Center in Applied Science and Engineering. In 1993, the students designed and built mobile robots based on the reactive principles outlined by Rodney Brooks, and the 1994 project was based on 'haptic' displays. The target applications of mobile robots such as a wheelchair, a robot vacuum cleaner or a robot in a public building to aid people with vision problems were identified. This lead to the design objective of identifying a mobile robot that could navigate to a recharging station without colliding into the environment.

In 1994 the primary project was to design and prototype a haptic display. The display specification was that it should be similar to a computer mouse but be able to allow the user to sense virtual edges and corners. Four design groups were established, two groups used passive magnetic particle brakes to impede the motion of the mouse and two groups used electric motors that allow the design to push back at the user. A standard virtual environment program was provided to evaluate the designs. This program read a shape from a file and when given the x, y position of the mouse, returned the distance to the nearest internal edge of that object.

Alciatore and Histand [15] discuss the mechatronics and measurement systems course at Colorado State University. The paper presents an innovative four credit junior level course which combines mechatronics, instrumentation, and computer control topics with a laboratory experience. The paper describes course objectives, a course description and syllabus, lab exercises, term projects and reference materials.

Course description: The course begins with an intensive review of passive circuit analysis, active semiconductor devices, and analog circuits with a focus on the operational amplifier, and digital devices. Next, the response of electromechanical systems is studied as a basis for the proper selection and/or design of a measurement system. The next topic treats the conversion from analog to digital signals and interfacing the analog world to computers and controllers. The course topics conclude with the theory and applications of sensors and actuators. All lecture topics are reinforced by the laboratory portion of the course where students gain an invaluable 'hands-on' experience which is crucial to understanding the course topics. This experience also provides a fundamental building block for later design courses, graduate education experiences, and 'real world' engineering. The course concludes with a group term project involving a mechanical system, instrumentation, sensors, actuators, and computer programming and control. The evaluation for the course includes homework, exams, and projects.

The objectives for students of this four-credit course are to:

- Further apply an understanding of passive electrical circuits and active electronics.
- Learn how to properly select and/or design measurement systems.
- Learn the theory and application of sensors
- Become proficient using standard laboratory instrumentation
- Learn how to interface sensors and electromechanical systems to computers

Laboratory schedule: The objective of the laboratory is to provide the experimental learning component whereby students reinforce understanding of abstract concepts from the lectures by executing small group exercises which are closely coordinated temporally with the lecture. The students are exposed to group problem solving situations, and they

are required to report on their work as groups. This serves as a good model for their future coursework and engineering experiences. The laboratory schedule includes topics such as demonstration of basic instruments, instrument familiarization and basic electrical relations, oscilloscope, instrument bandwidth, operational amplifier characteristics and circuits, digital circuits, data acquisition and control and sensors such as strain gage, accelerometer and thermocouple.

The term projects for the course include:

Three-dimensional surface digitizing laser scanner: this device consists of a laser, phototransistors, stepper motors, and limit switches which allow digitization of points on a 3-D surface. The laser illuminates a spot on the surface, the phototransistors are translated to detect the spot and the coordinates of the spot are calculated by triangulation. The object, laser, and sensors are independently controlled with stepper motors. The objective presented to the students is to recognize, as quickly as possible, an object placed on the digitizing table. The students must design software to distinguish between several different shapes such as cylinder, sphere, cone, box, and prism.

Cylindrical robot with optical tracking capabilities: The objective of this project is to control a cylindrical robotic arm to locate and transport an object in the robot's workspace. The robot's end-effector is equipped with an optical sensor head and a solenoid-operated gripper to enable object detection and transport. Four levels of difficulty were used to evaluate the student groups in a competitive format. The levels included following a specified trajectory, searching the workspace for the object, tracking a moving object once found, and picking up and safely transporting an object with payload to a specified location.

High speed flexible robot arm and delivery system: The challenge of this problem is to devise an algorithm to control a stepper motor-driven highly elastic beam with magnetic end-effector in order to repeatedly transport ball bearings from a dispenser to a bin. Speed is the competition criterion requiring both trial and error to determine the system response and astute motor driver design for acceleration and deceleration.

Acoustic room perimeter mapping system: An acoustic puse-echo system attached to a stepper-motor-driven mount is presented to the students with the objective to scan for the

perimeter of a room and graphically display the room geometry on the computer screen. Objects and impediments in the room must also be displayed.

Electro-optical distance scanner: The apparatus presented to students consists of a stepper-motor-driven belt which runs past an aluminum plate with two milled slots. Students must design and mount an optical sensor on the belt drive to pass over and accurately measure the separation of the slots. Various unknown slot separations are presented in the final competition.

Innovative methods applied in the course: An important innovation is the fact that the course is a non-traditional addition to the mechanical engineering curriculum reflecting the impact modern electronics has had on the practice of mechanical engineering. The course goes beyond basic electrical circuit theory and deals with the world of electronics and computers which is not covered in a traditional mechanical engineering curriculum. The authors have given a true mechatronics flavor to the traditional instrumentation, measurement system, and experimental methods type course.

Rajagopalan et al [17] present the methods developed to teach mechatronic concepts to mechanical engineering students at Concordia University. The paper discusses two undergraduate courses which train the students in controlling electro-mechanical systems using microprocessors. The objective of the courses is to provide a unique opportunity to the mechanical engineering students to learn electronics, internal architecture of microprocessors, and system interfacing to control an electro-mechanical system. The 'Industrial Electronics' course introduces the students to the basic and advanced concepts of analog and digital electronics. The second course 'Microprocessors and Applications' is designed for the control of analog and digital devices with projects involving microprocessor control of electro-mechanical systems and is an elective course for the undergraduate students registered primarily in the control option.

The course provides lectures on the internal structure, the instruction cycles of microprocessors, timing diagram, programming in low level language to understand how data handling is done by the microprocessors, types of busses, bus buffering techniques, interfacing input/output devices using memory-mapping, different types of interrupts,

direct memory access (DMA), methods of interfacing memory, techniques of interfacing the analog world with the digital computer, I/O techniques, and a variety of support systems. The course provides hands-on experience for the student in the form of a project work. The students are divided into groups of 2 or 3 and each group is assigned a project. Lab sessions are supervised by teaching assistants and a full-time departmental electronic technician. The objective of each project is to control a mechanical system actuated by a DC or stepper motor. The control action could be position or velocity control. The closed-loop or feedback control is carried out with the help of the feedback signal from a potentiometer or a tachometer or an encoder. The command signal to the amplifiers of the motors are either a DC signal or a pulse width modulated output. Control schemes implemented are P, PI, and PID. A microprocessor unit with a Z-80 processor as the host CPU, with Counter Timer Circuit (CTC) and Parallel I/O (PIO) device is used as the controller. Communication with the outside world is carried out using the bus of Z-80 processor or the PIO data lines. The CTC chip performs the important function of timing and counting in order to generate interrupts at regular intervals thereby serving to maintain the cycle time, to generate the PWM output, and also to count the pulses coming in from an encoder. The necessary hardware such as ADC, DAC, bi-directional amplifier circuitry to drive the motor, and encoder decoding are built by the students during the lab hours. Display routines to show in real-time the desired and the actual velocity or position and keyboard interface routines to modify the control parameters on the fly are developed by the students.

Host Microprocessor: The basic components of a typical microcomputer are the CPU, the memory, and the I/O devices. These units are connected by a number of parallel wires called the busses. For real-time control of mechanical systems from a teaching point-of-view, we can adopt the following computer hardware:

- Microprocessors with very basic processors and limited processing capabilities
- Microcontrollers with built-in input and output channels
- Microcomputers such as a 386 or 486 based
- Minicomputers controlling a number of work stations

In the first approach, the students are provided with an inexpensive device that can function well while providing the required knowledge and the hands-on experience

required. The second approach makes use of standard off-the-shelf microcontroller units not requiring a great deal of hardware knowledge. The third approach confirms with the recent trend in technology, i.e., more powerful desktop computers. This approach is very similar to the first approach, except that the students would be programming in 8088 assembler instead of Z80 assembler. The last approach is not recommended as it takes away the joy of probing into the hardware.

Software environment: Control software can be either high level or low level. High level languages such as C, C++, Pascal and Fortran can be used. Low level programming includes machine codes and assembly codes. While using the high level languages, the students do not get the opportunity to understand how the microprocessor performs data handling. Hence the Z-80 assembler is used for programming.

Student projects: The primary goal is to control an electromechanical system. Lectures on the use of Z80 and its supporting peripherals (CTC, PIO) runs parallel with the lab sessions. The programming instructions of Z80 are discussed in the class at length with examples that the students can try in the laboratory. Programs are designed not only to carry out simple arithmetic operations, but also to perform input and output functions using the system bus as well as the PIO. Some of the projects are:

- Analog driven DC motor velocity control using an encoder feedback
- Analog driven DC motor position control using an encoder feedback
- PWM driven DC motor velocity control using an encoder feedback
- PWM driven DC motor position control using an encoder feedback
- Analog driven DC motor velocity control using a tachometer feedback
- Analog driven DC motor position control using a potentiometer feedback
- PWM driven DC motor velocity control using a tachometer feedback
- PWM driven DC motor position control using a potentiometer feedback
- DC motor velocity/position control using a LM629 motion controller
- Stepper motor control with position feedback
- Control of a stepper motor driven XY-table

Rizzone and Keyhani [16] propose the development of an integrated interdepartmental curriculum for the design of mechatronic systems. The paper addresses

issues such as mechanical and electrical engineers being employed to design, test and manufacture electromechanical products, the need for engineers with a more in-depth knowledge of mechanical, electrical, and electronic devices and systems. Electro-mechanical engineers should be able to integrate the two disciplines so as to deal with the design and analysis of electromechanical engineering products. The paper describes the development of a electromechanical system design curriculum leading to a supplement to a BS in mechanical or electrical engineering. The principal goals of this project are:

1. To develop an innovative curriculum in Electro-mechanical system design that will serve as a national model.
2. To use this innovative curriculum as a means of integrating electrical, electronic, mechanical, and computational engineering education into a capstone design sequence.
3. To introduce the use of video and computers into the classroom and laboratory.
4. To emphasize the use of a variety of CAD tools in engineering design and analysis.
5. To provide a realistic engineering problem setting in the lab, enabling the students to address design and experimental issues in engineering.

The means through which these objectives will be accomplished are a sequence of three courses and an open laboratory focusing on the design of electro-mechanical systems. The laboratory will consist of an Electro-Mechanical Systems Laboratory, in which the students will be exposed to the electrical, electronic, mechanical, hydraulic and computational aspects of hybrid systems. Applications that will be studies in this laboratory will include an electro-magneto-mechanical proportional effort steering system; an anti-lock braking system; a hydraulic load; a flexible shaft with inertial load; and friction loads; hydraulic systems; and electric and hybrid-electric vehicle applications. The students will have a number of experiment stations at their disposal, including a small internal combustion engine, and various electric machines, including induction, synchronous, SC, and switched reluctance motors. Each station will be equipped with a Motorola DSP 56000 digital signal processor that will serve as a general purpose controller. Workstations will also be available for simulation studies and data processing.

The three courses that will form the core sequence of the program are: Introduction to Mechatronics, Dynamics and Simulation of Electro-mechanical systems, Control of Electro-Mechanical Systems. These courses and the laboratory will be integrated into the existing Electrical and Mechanical engineering curricula, and will culminate in a capstone design project centered around the facilities available in the laboratory. This setting will permit the development of a curriculum in electro-mechanical systems with a focus on real engineering problems for electrical and mechanical engineering students. Through this curriculum the authors seek to lead the undergraduate students to creatively identify capstone design projects in electro-mechanical systems. The design projects will be realized through the use of open laboratories and with the participation of industrial sponsors, who will assist in defining projects and will provide in-kind and cash support for individual projects.

Chapter 5
System Development

The main component of the computer-controlled car is the microprocessor or a programmable controller which controls the movements of the car. The objective was to choose a controller, which would be C/C++ programmable and come equipped with Analog Input/Output, Digital Input/Output, and signal conditioning. Controller boards from different manufacturers were studied before making a decision to use the controller board from Tern, Inc. The first board considered was the Handy Board, but it did not come with the necessary signal conditioning features. Controllers from Z-World were also studied. None of the controllers from Z-World featured the signal conditioning and the boards used a customized version of the C language, Dynamic C, for programming. The Analog Drive from Tern, Inc was chosen because it provides six channels of high gain instrumentation operational amplifiers. The Analog Drive is controlled by Tern's V104 controller.

5.1. Analog Drive

The computer monitoring and control functions are performed by two interconnected boards: the V104 and the Analog Drive (A-drive). Both boards are manufactured by Tern, Inc.

The V104 is installed on top of the A-Drive board via 60 pin headers. The A-Drive motherboard provides regulated power and RS-232/485 drivers. The V104 features a 16-bit 8 MHz V25 CPU, three 16-bit timers, three serial ports, additional 24 bi-directional I/O lines of an 82C55, up to 512K battery backed-up SRAM, up to 512K EPROM or Flash EEPROM, 512 bytes of EEPROM, an LCD interface, a real-time clock, a 64-pin PC/104 bus, an LED, and a supervisor chip providing power failure detection, watchdog timer, and reset. All analog inputs and output signals are connected via screw terminals.

Overview

The A-Drive is designed for industrial process control, automatic test equipment, and data acquisition applications that require many channels of analog signal I/Os. The A-Drive costs about $300 and features 22 channels of 12-bit ADC inputs and 8 channels of 12-bit DAC outputs, including 11 ADC inputs and four DAC outputs from the V104. There are 10 channels of signal conditioning circuits for the ADC inputs on the A-Drive board, including six instrumentation operational amplifiers providing high input impedance, high gain, and low noise to low-voltage output sensors, such as EKG electrodes, thermocouples, or strain gauges. The circuits provide user-configurable gain, filter, and offset settings. Resistors buffer the 11 ADC inputs to the V104. A precision temperature sensor (10 mV per degree C) is located next to the screw terminals for thermocouple cold junction compensation. A precision, 3 ppm per degree C, 2.5V reference supports 12-bit ADCs and DACs. The 12-bit ADC features sample-and-hold and isolation of analog circuitry from logic and supply noise. A switched-capacitor design allows low-error conversion over the full operating temperature ranger up to approximate 2500 Hz sample rate. The 8 channels of 12-bit DAC can output either voltage or current via 8 operational amplifiers, 0-2.5V or 4-20 mA. The DAC features power-on reset, 3us output setting time and 5 V/us slew rate. User may customize analog input and output ranges by setting different gain resistors.

Seven channels of solenoid drivers can sink/source 350 mA at 50V. Two 16-bit counters and three external interrupt inputs are buffered by Schmitt-trigger inverters, in order to increase noise immunity and transform slowly changing input signals to fast changing and jitter-free signals. There are two RS-232 serial ports and an RS-485 serial port.

Specifications for the A-drive:

- Measures 6.2 x 4.2 inches.
- Power consumption: 170 mA, includes V104
- Low power version: 85 mA full speed, mA standby.
- Temperature: -40 degrees C to +85 degrees C.
- Driven by a V104, 16-bit V25, and 8 MHz.

- 22 channels of 12-bit ADC(11 on V104), 2500Hz
- Six channels of high gain(1000) instrumentation ops.
- Directly interfaces to thermocouples, RTD, or strain gauges.
- 8 channels of 12-bit DAC output 0-2.5 V or 4-20 mA
- Two RS-232 channels and one networking RS-485 channel.
- On-board temperature sensor for cold-junction compensation.
- 2.5V, 3 ppm/Celsius precision reference voltage.
- Up to 512K Flash/EPROM, 512K battery-backed SRAM.
- 24 bi-directional I/O lines from 82C55 on the V104.
- 7 comparators inputs or 7 high voltage drivers.
- 3 16-bit timers, 2 external event counter up to 500KHz.
- Real-time clock RTC72421, 512 byte EEPROM
- LCD interface and 64 pin PC/104 bus interface.
- 3 external interrupt inputs with Schmit trigger buffers.
- On-board power supply for +5V, +7V, -12V, -5V, and 2.5V.

Features:

The A-drive can be programmed from the PC via RS-232 serial link, using either Borland or Microsoft C/C++ compilers. The C/C++ program can be remotely downloaded and debugged at 115,000 βαυδ rate. The A-drive comes with a complete C/C++ evaluation or development kit with everything necessary to start developing the application software. A-drive is driven by V104 module. The A-drive interfaces with the V104 via three 10x2 headers.

Hardware and Software requirements:

- A-drive plus V104 with Debug EPROM
- A serial cable (PC-V25) with a DB9 connector and an 5x2 IDC connector
- A center negative wall transformer (+9 V, 500 mA)
- A PC or PC compatible computer

- Microsoft Visual C/C++ and TASM611 or Borland C/C++ 3.1/4.0/4.5, Turbo C/C++ 3.0 and TASM
- Paradigm DEBUG/RT-V25 and Paradigm LOCATE
- TERN System Disk

Hardware

Instrumentation Operational Amplifiers and analog signal conditioning

The analog signal conditioning circuit provides six channels of instrumentation operational amplifiers with high gain (>1000) and high input impedance ($<10^{12}$ ohms). The instrumentation ops allow direct interfacing to low-voltage output sensors, such as EKG electrodes, thermocouples, or strain gauges. The high input impedance adjustable-gain differential amplifier is constructed with three operational amplifiers.

Analog to Digital Converter (ADC)

The analog drive contains two ADC chips: U10 V104-ADC and U12 A-Drive-ADC. It is a 12-bit, switched-capacitor, successive approximation, 11 channels, serial interface, analog-to-digital converter. The converter has an on-chip 14 channel multiplexer that can select any one of the 11 inputs or any one of three internal self-test voltages. The sample-and-hold function is automatic. At the end of conversion, the end-of-conversion output goes high to indicate that conversion is complete. The chip features differential high-impedance inputs that facilitate ratiometric conversion, scaling, and isolation of analog circuitry from logic and supply noise. A switched-capacitor design allows low-error conversion over the full operating temperature range. While at full samples rate, the analog input signal source impedance should be less than 50 ohms and capable of slewing the analog input voltage into a 60-pF capacitor.

The ADC can be read with the function provided in the library *ad_ad12(ch)*. An analog input is provided to the function and the function outputs its equivalent digital output, which is a number. The full scale ADC reading 0-4095 is valid in the voltage range 0-2.5V. The power supply was used in combination with resistors to generate different voltages, which serve as analog input voltages. The converter was tested for

these different input voltages using the function provided. The A-drive features 22 channels of 12-bit ADC inputs. Most of these channels were tested with the function, which enabled to determine the individual gain on those channels.

Digital to Analog Converter (DAC)

The A-drive contains two DAC chips, one is on the V104 and one is on the A-drive. The 12-bit DAC combines four 12-bit, voltage output digital to analog converters and four precision output amplifiers in a 16-pin chip. The chip operates with +-5V power supply. Each DAC has a double-buffered input. A 16-bit serial word is used to load data into input/DAC register. The DAC features power-on reset, 3 μs output setting time and 5V/μs slew rate. The DAC requires –5V to operate. The valid DAC output voltage range is 0-2.5V. There are operational amplifiers for conditioning DAC voltage output to either current or voltage. The DAC can be written with a function provided in the library: *ad_da12(ch,dat*).

Temperature Sensor and Reference Voltage

A precision temperature sensor is located to the screw terminals for thermocouple cold junction compensation. The sensor outputs a voltage linear to degree C. While the board temperature is 25^{o}C, the sensor outputs V0=250 mV. V0 is routed to the V104 ADC channel 10. A 2.5V precision voltage reference is on board providing 2.5V reference for all ADC and DACs.

RS-232 and RS-485 Serial Ports

There are two RS-232 channels routed at sSER1 and SER2 ports. SER0 is the default debug port. The SCC2691 UART is routed to the RS-485 driver.

5.2 V104: A C/C++ Programmable 16-bit Controller with PC/104 Bus

Overview

The V104 from TERN is a low cost (about $300), high performance, Borland/Microsoft C/C++ programmable 16-bit industrial microcontroller. It is designed for embedded applications that require compactness, low power consumption, and high

reliability. The V104 can be integrated into an original equipment manufactured (OEM) product as a processor core component, or as a stand-alone controller in an application system. By building the product around the V104, a manufacturer can reduce the time from design to market introduction, cut development costs, minimize technical risks, and deliver a more reliable product.

Measuring 3.5x4.0 inches, the V104 offers a complete C/C++ programmable computer system. The V104 features a 16-bit V25 CPU, 48 bi-directional user programmable I/O lines, eight comparator inputs, 11 channels of 12-bit ADC inputs, 4 channels of 12-bit DAC voltage outputs, three serial ports, three 16-bit timers, two external event counters, 3 external interrupts, an additional 24 bi-directional I/O lines from the PPI 82C55, up to 512K of battery backed-up SRAM, up to 512K EPROM or Flash EEPROM, 512 byte EEPROM, LCD interface, real time clock, LED, and a supervisor chip providing power failure detection, watchdog timer, and reset.

The eight comparator inputs can be used to measure either digital or analog inputs. Three external interrupt inputs can trigger the immediate execution of user interrupt service routes. Two counters can count input pulse rising edges up to 500KHz. The 12-bit ADC has a serial interface, successive approximation, and sample-and-hold. The ADC features differential high-impedance reference inputs that facilitate ratiometric conversion, scaling, and isolation of analog circuitry from logic and supply noise, with low-error (*1LSB) over the full operating temperature range at 2500 Hz sample rate. The ADC input voltage range is 2.5V to 5V. The 4 channel 12-bit DAC voltage outputs (0V to 2.5V) are internally buffered by precision unity-gain followers with a typical slew rate of 3V/us with 5Kohm load. The V104 is powered by a regulated 5V. Only the DAC requires a -5V power supply. A VE232 may be used to provide +-5V power and RS-232 drivers. A 16-pin header LCD interface supports various types of LCD modules.

The V104 can be programmed from the PC via serial link with a RS232 interface. Microsoft or Borland C/C++ compiler can be used. The C/C++ program can be remotely debugged with remote debugger over the serial link at 115000 baud rate. A C/C++ evaluation/development kit from TERN includes LOCATE, DEBUG, I/O driver libraries, sample programs, EPROM, configuration files, batch files, and all the necessary hardware to start developing the application software. After debugging a program, by

changing a single jumper, the program can be tested to run the controller in the field, away from the PC, since the application program will reside in the battery-packed SRAM.

Specifications for V104

- Measures 3.5 x 4.0 inches, 16-bit CPU (NEC V25), 8 MHz.
- 5V, 75 mA full speed, 30 mA standby, 3 mA low power.
- Operating Temperature: -40 Celsius to +85 Celsius
- Upto 1MB Flash/EPROM and battery-backed SRAM
- 11 channels of 12-bit ADC and 4 channels of 12-bit DAC
- 24 bi-directional I/O lines (multiplexed functions) in the V25.
- 8 comparators in the V25 for analog or digital inputs
- 3 serial ports: SER0, SER1 and UART for 9-bit network
- Additional 24 bi-directional I/O lines from a 82C55.
- 3 16-bit timers. 2 external event counters up to 500K Hz.
- Real-time clock RTC72421, coin battery, 512 byte EEPROM
- Supervisor chip (691) for power failure, reset and watchdog.
- LCD interface and 64 pin PC/104 bus interface.

Hardware and software requirements

- A PC or PC compatible computer
- V104 with Debug EPROM
- A VE232 interface board, PC-V25 serial cable and a center negative wall transformer (+9 V, 500 mA)
- Microsoft Visual C/C++ , or Borland C/C++ 3.1/4.0/4.5, or Turbo C/C++ 3.0, DOS
- Paradigm LOCATE or LOCATE-EV
- Paradigm DEBUG/RT-V25 or DEBUG/RT-EV for debugging application programs
- TERN’s C libraries

Hardware

V25 I/O Ports

V25 is the CPU of the V104. The V25 CPU has 32 I/O lines which are basically organized as three bi-directional I/O ports (P0-2) and a comparator input port T. The 24 bi-directional I/O lines are multiplexed with different functions. One I/O line can be specified as an input, output, or a control line. There are three Special Function Registers (SFR) associated with each port: Port Mode Control Register (PMC0, PMC1,PMC2), Port Mode Register (PM0, PM1, PM2), and Port Data Register (P0, P1, P2). The SFRs are memory mapped. These registers can be read or written via two functions: *peekb(0xfff0, 0x??)* and *pokeb(0xfff0, 0x??, 0x!!),* where ?? is the register offset address, !! is the control/data byte.

Programmable Peripheral Interface or Parallel Interface Unit

This device is a low-power CMOS programmable parallel interface unit for use in microcomputer systems. It provides 24 I/O pins, which may be individually programmed in two groups of 12 and used in three major modes of operation.

In MODE 0, the two groups of 12 pins can be programmed in sets of 4 and 8 pins to be inputs or outputs. In MODE 1, each of the two groups of 12 pins can be programmed to have 8 lines of input or output. Out of the remaining pins, 3 are used for handshaking and interrupt control signals. MODE 2 is a strobed bi-directional bus configuration. The functions *outportb(ch,dat)* and *inportb(ch)* are used to set the ports to output or input mode.

I/O Space Mapped Devices

External I/O devices use I/O mapping. The I/O can be accessed with the two functions *inportb(port)* or *outportb(port,dat).* The external I/O space is 64K, ranging from 0x0000 to 0xffff. In the I/O space of 0x0000-0x7fff, the I/O access time is 500 ns. In the I/O space of 0x8000-0xffff, the I/O access time is 250 ns.

To test the digital input lines on the V104 controller, a circuit emulating a stepper motor is used. Out of the 24 bi-directional I/O lines available, 8 lines are used. Port0 was initialized to function in the output mode. The function *outportb(port,data)* is used to set Port0 to output mode. One of the ADC input channels was used for the input signal. An average of 100 input samplings was taken to compute the average digital output value, which determines the digital input sent through Port0. Once the program starts running, first the average output is computed and the loop which runs the stepper motor starts executing. Each time through the loop the stepper motor is fired in the correct sequence. This is accomplished by sending out the correct binary number to the function *outportb(port,data).* For example,
outportb(PPI0, 0x30) sends a value of 1 on lines 5 and 6 of Port0. A delay of few milliseconds is provided between consecutive digital inputs.

Analog to Digital Converter (ADC)

The ADC is a 12-bit, switched-capacitor, successive approximation, 11 channels, serial interface, and analog-to-digital converter. It has three control inputs and is designed for communication with a host through a serial tri-state output. The converter has an on-chip 14 channel multiplexer that can select any one of the 11 inputs or any one of three internal self-test voltages. The sample-and-hold function is automatic. At the end of conversion, the end-of-conversion output goes high to indicate that conversion is complete. The chip features differential high-impedance inputs that facilitate ratiometric conversion, scaling, and isolation of analog circuitry from logic and supply noise. A switched-capacitor design allows low-error conversion over the full operating temperature range. While at full samples rate, the analog input signal source impedance should be less than 50 ohms and capable of slewing the analog input voltage into a 60-pF capacitor. The ADC can be read with the function provided in the library *ce_ad12(ch).*

Digital to Analog Converter (DAC)

The 12-bit DAC combines four 12-bit, voltage output digital to analog converters and four precision output amplifiers in a 16-pin chip. The chip operates with +-5V power supply. Each DAC has a double-buffered input. A 16-bit serial word is used to load data

into input/DAC register. The DAC features power-on reset, 3 μs output setting time and 5V/μs slew rate. The DAC requires –5V to operate. The valid DAC output voltage range is 0-2.5V. There are Ops for conditioning DAC voltage output to either current or voltage. The DAC can be written with a function provided in the library: *v104_da12(ch,dat).*

5.3. Startup Circuit of the Car

One of the objectives of this project, specifically objective 3A, is to start the timed event by momentarily turning off the room lights. Each car must be designed to respond, under its own control, to the decrease in light. This technique eliminates the human element in starting the race. To achieve the start, a startup circuit is designed. The startup circuit proposed contains a photocell, which responds to the intensity of light. A program is designed to start the car when the photocell senses a reduction in the light intensity.

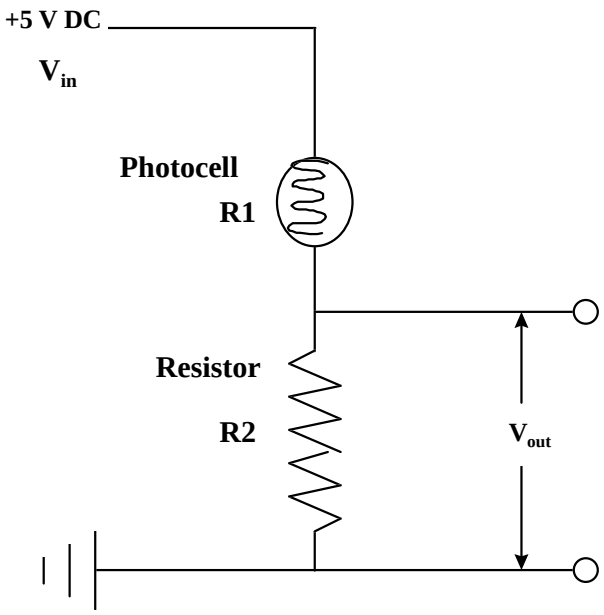
5.3.1. Analysis for the Startup Circuit of the Car

The startup circuit is a voltage divider circuit that is used to initially start the car. The circuit consists of a photocell and a resistor in series(voltage divider circuit) as shown in Figure 5.1. The resistance of the photocell decreases with the intensity of light incident on it. The resistor chosen has its resistance equal to or slightly less than the resistance of the photocell under room light conditions. The photocell is connected to a 5V DC power supply, and the free end of the resistor is connected to the ground. If the resistance of the photocell and the resistor are equal, the voltage drop across the photocell should be 50% of the voltage supply to the circuit. The voltage drop across the resistor is used as the input to channel 6 of ADC on the A-drive and the analog to digital conversion program will output a number in the range 0-4095 as the input varies from 0 to 2.5 V DC. The photocell used is a Cadmium-Sulphide, model 276-1657 photocell from Radio Shack and the resistor has a measured resistance of approximately 1105Ω.

Tests were carried out to determine the resistance as the voltage drop across the photocell changed as a function of light. The resistance and voltage were measured

under three different light intensities: dark, standard room light, and intense light. Standard testing conditions for the dark configuration was with the room door and windows closed, the room lights switched off, and the surface of the photocell covered with black paper. Standard testing conditions for room light configuration was with the normal room lights on. The intense light configuration was achieved by focusing a 10amp, Arrow H&H light beam directly onto the surface of the photocell. The light was held about 3.5 inches away from the photocell.

For each of the above mentioned test conditions, voltage across the resistor and resistance of the photocell were recorded. It was observed that with the increase in the intensity of light, the resistance of the photocell decreased. The voltage dropped across the resistor increased with the increase in the light intensity. The voltage and resistance readings are plotted against the intensity of light in Figure 5.2 and Figure 5.3 respectively.



$$V_{out} = V_{in} * R_2/(R_1+R_2)$$

Figure 5.1: Startup Circuit

The circuit output is given by

$$V_{out}=V_{in} * R_2/(R_2+R_1) \qquad \textit{Equation 1}$$

Using Equation 1, the output for the dark condition is:

V_{out} = 5 VDC 1105/(1105+2362.6) = 1.6 Volts DC

For room light condition:

V_{out} = 5 VDC 1105/(1105+1080) = 2.53 Volts DC

For intense light:

V_{out} = 5 VDC 1105/(1105+45) = 4.8 Volts DC

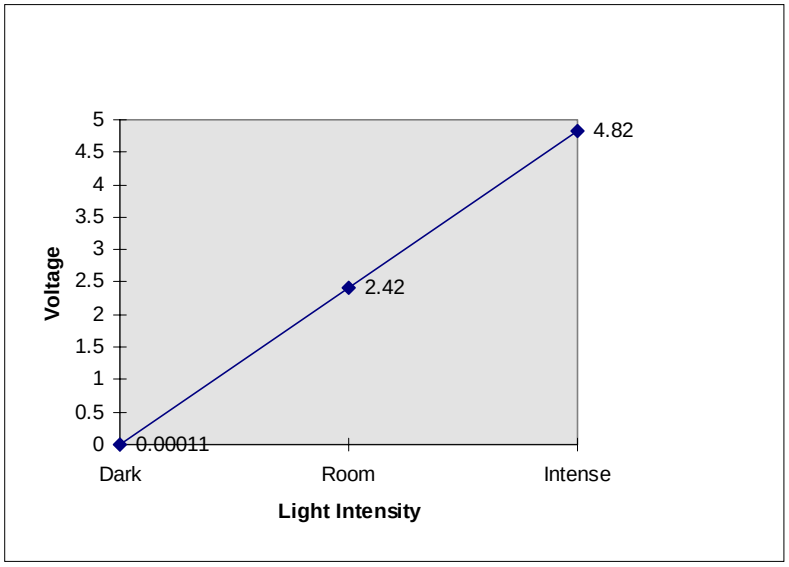


Figure 5.2: Output Voltage Variation for Startup Circuit

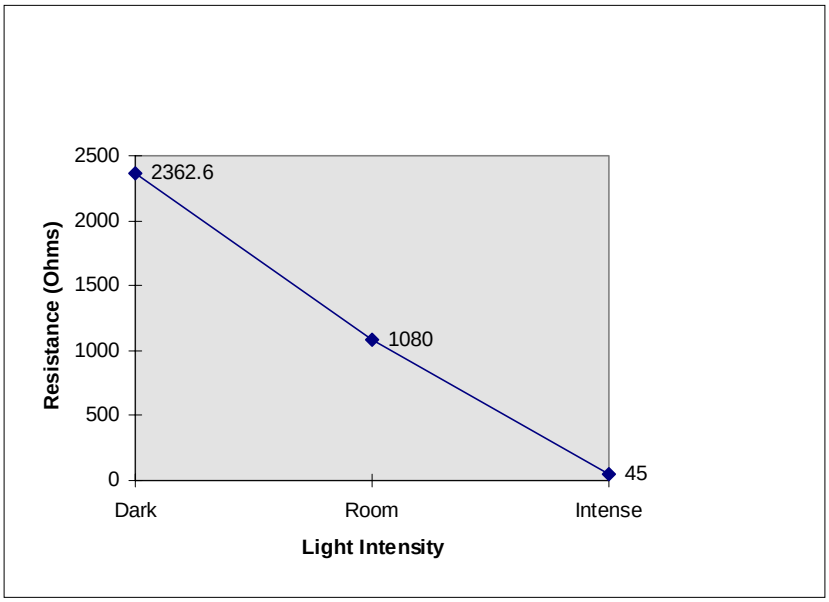


Figure 5.3: Resistance Variation of the Photocell for the Startup Circuit

5.3.1b. Software

The 12 bit A/D converter senses the output voltage and linearly converts the voltage to a digital number between 0 and 4095. The program used to detect the digital numbers and run the motor contains a section, which deals with the startup circuit. The program listing that demonstrates the startup circuit and a brief description is given below.

The values of avg, sum and I are initialized. I is initialized to a value of 6 since 6 is the ADC channel which is used in the startup circuit. Then a for loop is used to sample the ADC channel output. An average of 10 values is taken. If the average output from the ADC is less than 1000, the program assumes that the room is dark and the motor should start running. The program checks the avg values and starts the motor. This check is made only in the beginning of the program, and once the motor starts running, the room light intensity does not have any effect on the motor.

Program statements:	**Description:**
sum=0; avg=0; I=6;	The values of sum, avg, and I are initialized.

	Sum and Avg are used to sample channel 6 ten times and average the results
for(k=0;k<10;k++){	Beginning of the for loop
ad_ad12((unsigned char)I);	This call to the program flushes the old value
adat= ad_ad12((unsigned char)I);	Here the digital output from channel 6 is stored in adat.
sum+=adat;	The adat value obtained is added to sum
}	
avg=sum/10;	Avg is calculated
if(avg<1000){	This statement tests the condition to start running the motor.
	The code to run the stepper motor goes here. It is illustrated in the following section
}	

5.4. Analog to Digital Converter (ADC) chip

The analog drive contains two ADC chips: U10 V104-ADC and U12 A-Drive-ADC. Both are 12-bit, switched-capacitor, successive approximation, 11 channels, serial interface, analog-to-digital converters. The converter has an on-chip 14 channel multiplexer that can select any one of the 11 inputs or any one of three internal self-test voltages. The sample-and-hold function is automatic. At the end of conversion, the end-of-conversion output goes high to indicate that conversion is complete. The chip features differential high-impedance inputs that facilitate ratiometric conversion, scaling, and isolation of analog circuitry from logic and supply noise. A switched-capacitor design allows low-error conversion over the full operating temperature range. While at full sampling rate, the analog input signal source impedance should be less than 50 ohms and capable of slewing the analog input voltage into a 60-pF capacitor. The ADC on the A-

drive can be read with the function *ad_ad12(ch)*, and the ADC on the V-104 can be read with the function *ce_ad12(ch).*

5.4.1. Testing the Gain of the Analog to Digital Converter

The eleven channels of analog input have different fixed gains. To determine the gain of each channel, a test circuit was devised. The circuit used to test the ADC gain consisted of two resistors in series as shown in Figure 5.4. The resistors R1 and R2 were chosen such that the resistance of R1was 4 times the resistance of R2. The voltage across R2 was used as the input to each of the ADC channels. The resistors were chosen this way to facilitate a sufficient voltage drop across the first resistor, which in turn keeps the voltage input to the ADC channel low since the channel has gain. Free end of R1 is connected to a 5V DC power supply, and the free end of R2 is connected to the ground. The voltage across R2 was chosen as the input for one of the ADC channels on the A-drive and the analog to digital conversion program will output a number in the range 0-4095 corresponding to 0 to 2.5 V DC. This setup is used to test the gain on the 11 ADC channels available.

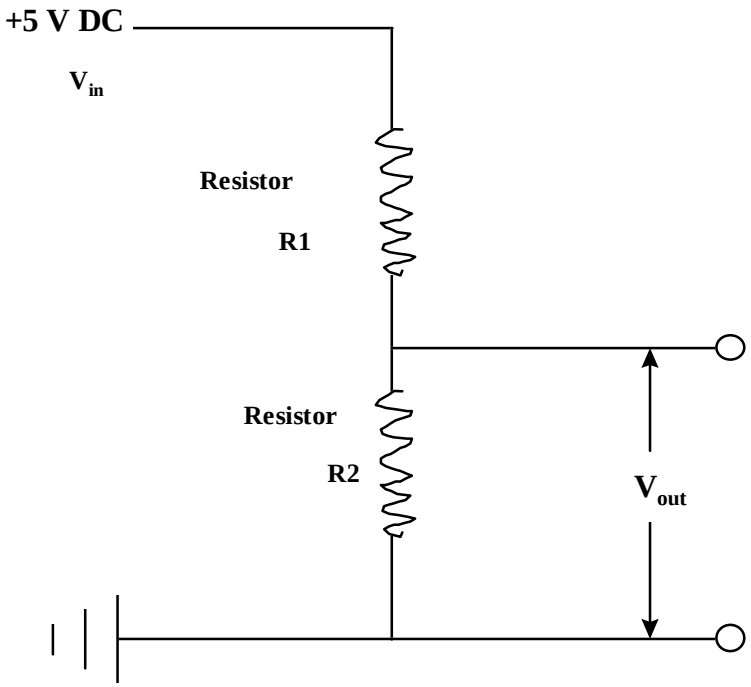


Figure 5.4: ADC Channel Gain Test Circuit

A low voltage of about 4 millivolts was put into each of the channels and the digital output was recorded. An average of six readings was taken and used to calculate the gain. Channels 0-5 have gains ranging from 115 to 185. Channels 6 and 8-10 exhibit very low or no gain. Channel 7 did not respond to the input voltage and the output was always 0. A chart of the channels and their respective gains is illustrated in Table 5.1 and a graph of the ADC channels gain is illustrated in Figure 5.5.

Channel	Gain
0	115
1	162
2	143
3	185
4	162
5	145
6	1
7	*N/A
8	1
9	1
10	1

Table 5.1: ADC Channel Gains

* Channel 7 did not respond to the input voltage. The digital output was always 0.

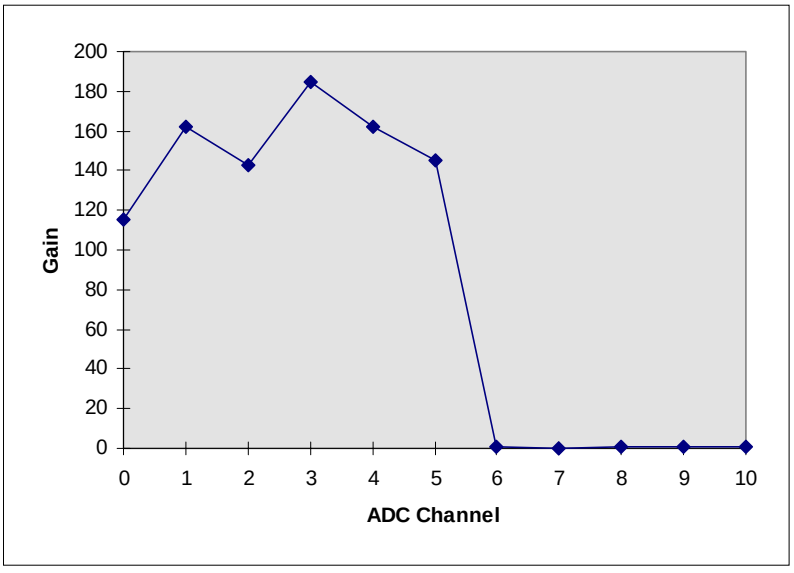


Figure 5.5: Gain for the ADC Channels

5.5. Stepper Motor

Stepper motors are used to drive the wheels of the computer-controlled car. The stepper motor used is a 4 Phase Unipolar model SM3401 stepper motor manufactured by Jamenco Electronic Components. Popular applications of the stepper motor are automation, robotic control and precision mechanical control.

5.5.1. Hardware

Specifications:

Unipolar (4 Phase)

24 VDC, 210mA

Coil: 117 Ohm, 131mH

1.8 degrees/step

Shaft: 0.25”D x 0.69”L

Mounting Hole Diameter: 0.20”

Mounting Hole Spacing: 2.62”

Motor: 2.22”D x 1.5”H

Detent Torque: 432 g-cm

Holding Torque: 5185 g-cm

Weight: 0.80Lbs

The schematic diagrams for the stepper motor are illustrated in Figure 5.6, Figure 5.7, and Table 5.2.

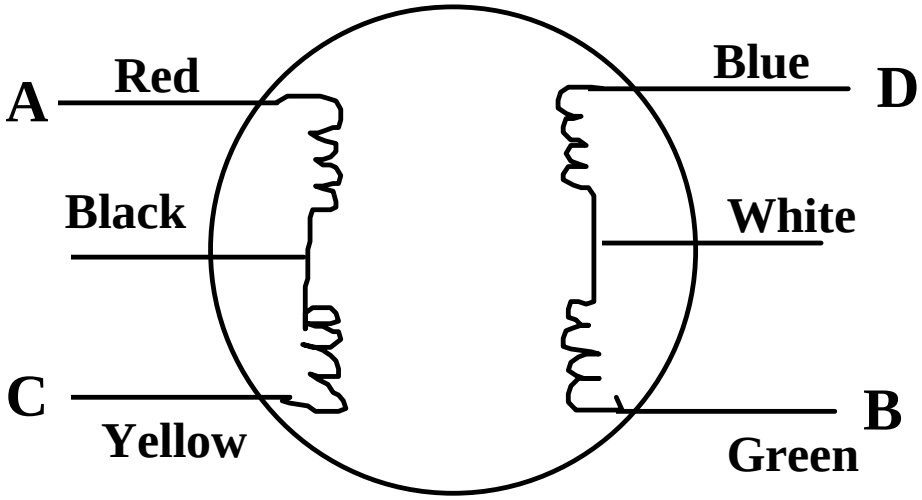


Figure 5.6: Wire Connections for the Stepper Motor

Black and White wires are common

Step	A	B	C	D	Common
1	**	**			+
2		**	**		+
3			**	**	+
4	**			**	+
5	**	**			+

Table 5.2: Firing Order for the Stepper Motor

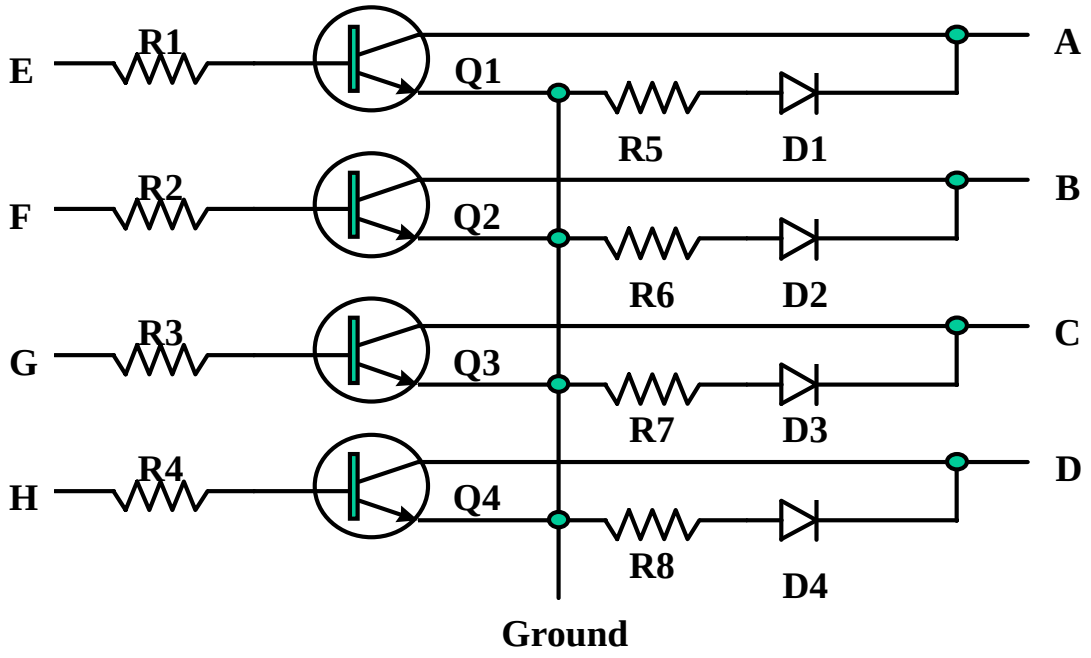


Figure 5.7: Driver Schematic for the Stepper Motor

As shown in Figure 5.8, stepper motors have a number of separate windings. The figure illustrates four separate windings, A,B,C,D and their connections. Each of these windings can be energized independently in steps. The motor can be made to rotate by energizing each coil in a cyclic fashion; each time the next coil is energized the motor rotates by a fixed amount depending upon the design of the motor. The stepper motor used turns by 1.8 degrees per step. Each step is accomplished by sending a signal through the appropriate part of the stepper motor circuit.

The program that accomplishes this is described in detail in the next section. If the sequence is monitored by the microcomputer, a count can be kept of the cumulative angle through which the motor shaft has turned. The direction of the motor's rotation can be reversed simply by reversing the sequence of signals to the motor. The two motors are programmed to rotate in opposite directions because of their physical placement in the car.

The black and white wires of Figure 5.6 are connected to +12 VDC. An individual motor is energized by completing the circuit through A, B, C, or D. Each circuit is energized by turning on the proper switching transistor (Q1, Q2 ,Q3 or Q4) in Figure 5.8. These transistors are controlled by four lines (E, F, G, and H) from the digital output port. Voltage spikes are reduced by diodes D1, D2, D3 and D4.

5.5.2. Speed Analysis for the Stepper Motor

The stepper motor driver circuit consists of transistors, resistors and diodes connected to the power supply as illustrated in Figure 5.8. Depending on the firing order, the input voltage flows in a specific path within the circuit, which translates into the movement of the motor.

The speed of the stepper motor is a function of input voltage and the delay variable in the driver program for the motor. A delay variable is needed because the mechanical motor cannot operate as fast as the computer. A model 5313 analog oscilloscope manufactured by Tektronix is used to measure the speed of the motor. A permanent magnet transducer is placed beneath the wheel attached to the motor shaft. A metal object is taped to the moving wheel. When the moving metal passes through the magnetic field of the magnet, the magnetic field is disturbed which induces a voltage in a coil of wire on top of the permanent magnet. The storage oscilloscope records the voltage spikes as a function of time. Speed in RPM of the motor is calculated with the known rotations and the time.

Input voltages ranging from 6.5-15 VDC are used for the analysis. The motor will not turn below 6.5 volts and the circuit is designed for a maximum voltage of about 15 volts. Delay values ranging from 3-14 milliseconds are used. A delay of less than 3 milliseconds is too fast for the stepper motor to recognize. The values of voltages and speed are illustrated in Table 5.3. A graph showing the variation in speed with the change in voltage is illustrated in Figure 5.8.

Delay(ms)	2	3	4	6	8	10	12	14
	Speed of the motor in RPM							
Voltage								
6.5		12.50	15.79	64.29	50.60	40.91	36.59	33.33
7		18.75	26.09	62.50	48.98	40.91	36.59	26.09
8		31.58	92.31	64.29	50.00	34.88	35.71	26.09
9		63.16	92.31	63.16	51.43	41.86	30.00	25.00
10	7.50	55.81	83.87	56.84	43.64	34.88	29.27	24.66
11	27.27	111.11	84.00	57.14	43.48	34.29	28.92	25.00
12	50.00	111.86	84.51	56.47	42.86	34.88	28.92	24.66
13	80.00	111.11	84.38	58.06	40.00	34.48	28.24	25.00
14	72.00	111.43	85.71	57.53	42.86	34.48	28.57	25.00
15	64.29	111.43	85.71	56.84	43.37	34.48	29.03	25.00

Table 5.3: Stepper motor speed as a function of supply voltage and delay

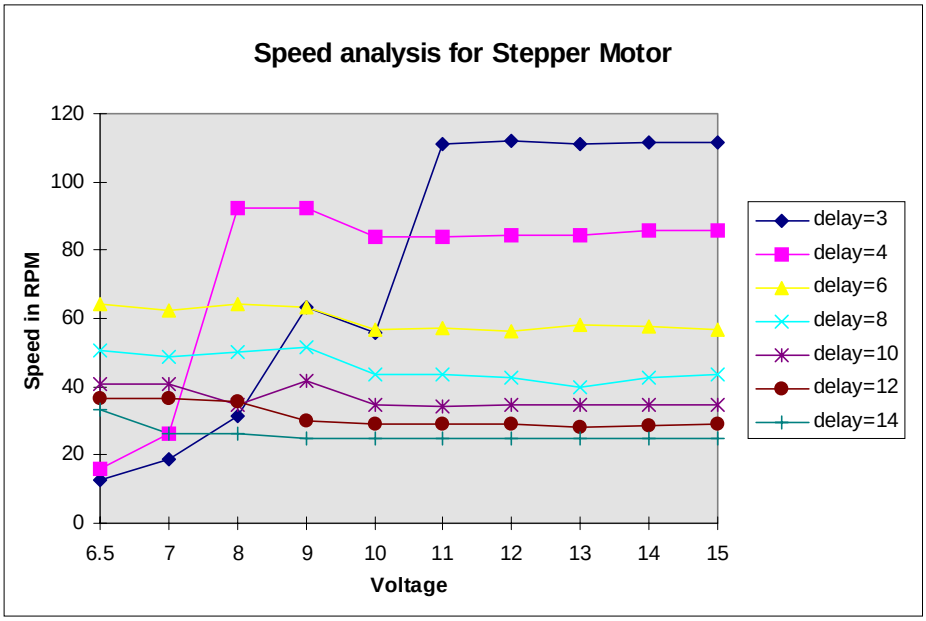


Figure 5.8: Stepper motor speed change with voltage

5.5.3. Software

The stepper motor is run by a program written in C. The program is briefly described below.

Statement	Explanation
#include <stdio.h>	The include statements include the library files
#include <dos.h>	The library files include all the standard input and output functions, and dos standard functions.
#include “ad.h”	ad.h and ve.h are header files which contain
#include “ve.h”	predefined functions for the analog drive and V104.
#define PPI0 0xC100	The define statements define variables
#define PPI1 0xC101	PPI0 is digital port 0, PPI1 is digital port 1 and so on..
#define PPI2 0xC102	Each of these ports is assigned a specific hexadecimal
#define PPI3 0xC103	address such as 0xC100.
int adat, adat1, avg, avg1;	These statements declare variables
unsigned int I,k,dat1,dat2;	These statements declare variables
unsigned char p0, p1, p2;	These statements declare variables
long sum, sum1;	These statements declare variables
void main(void){	Beginning of the main program
ve_init();	Initializing the boards
ad_init();	Initializing the boards
pokeb(0xfff0,0x11,0x40);	Initializing the ports to perform as output
pokeb(0xfff0,0x10,0xf7);	ports
outportb(PPIC,8x80);	Initializing port 2

```
sec=4;                                  Setting the delay variable to 4 milliseconds
outportb(PPI2, 0);                      Initializing port 2

while(1){                   The whole loop accounts for one complete rotation
for(k=0;k<20;k++){
        outportb(PPI2,0xCC);     This statement writes a particular hex number to the port
        delay_ms(sec);           The delay statement is used to slow down the computer.
        outportb(PPI2,0x66);  .
        delay_ms(sec);
        outportb(PPI2,0x33);  This statement writes a particular hex number to the port
        delay_ms(sec);
        outportb(PPI2,0x99);  This statement writes a particular hex number to the port
        delay_ms(sec);
}
}
}
```

The first part of the program with the include and define statements does the initialization for the program. The pokeb statements set the ports to be input or output ports. this program uses port 2. The part of the code which causes the motor to run is in the while loop. The digital output port is connected to the stepper motor circuit. The outportb statements are used to put out a specific number on the port, which in turn will energize the appropriate part of the stepper motor circuit. Depending on the number that is output on the port, the stepper motor will turn in a specific direction. The numbers input to the digital port are chosen such that the stepper motor circuit is energized in order.

5.6. Guidance Circuit for the Car

The guidance circuit is the circuit responsible for guiding the car along a specified path. The path is a length of reflective white tape. The guidance circuit consists of two photocells, which are used to control the individual stepper motors that drive the car. The

photocells are connected to resistors and to the power supply in the same way as the startup circuit. The resistance of the photocell changes depending on the amount of light incident on the photocell. Since the car is to be guided along a path defined by reflective tape on black paper, there will be a significant change in the resistance of the photocell as the car approaches the reflective tape. Depending on the value of the resistance, a decision is made about the direction in which the car is supposed to move. Turning the car can be accomplished by slowing down one of the wheels. A test was carried out to determine the optimum height from the reflective tape where the photocells for the guidance circuit should be mounted. The results are illustrated in Table 5.4 and Figure 5.9. If the photocells are mounted closer to the tape, a more distinct variation in resistance occurs.

Distance on tape in inches	-1	-0.75	-0.5	-0.25	center	0.25	0.5	0.75	1	1.25
Distance from the floor										
0.5in	3.53	3.15	2.2	1.59	1.49	1.63	2.23	3.15	3.6	3.55
1.0in	3.18	2.79	2.34	1.83	1.69	1.87	2.42	3.04	3.45	3.66
1.25in	3.27	3.12	2.77	2.47	2.44	2.53	2.74	2.96	3.23	3.38

Table 5.4: Resistance (Kilo Ohms) variation of the photocell on reflective tape

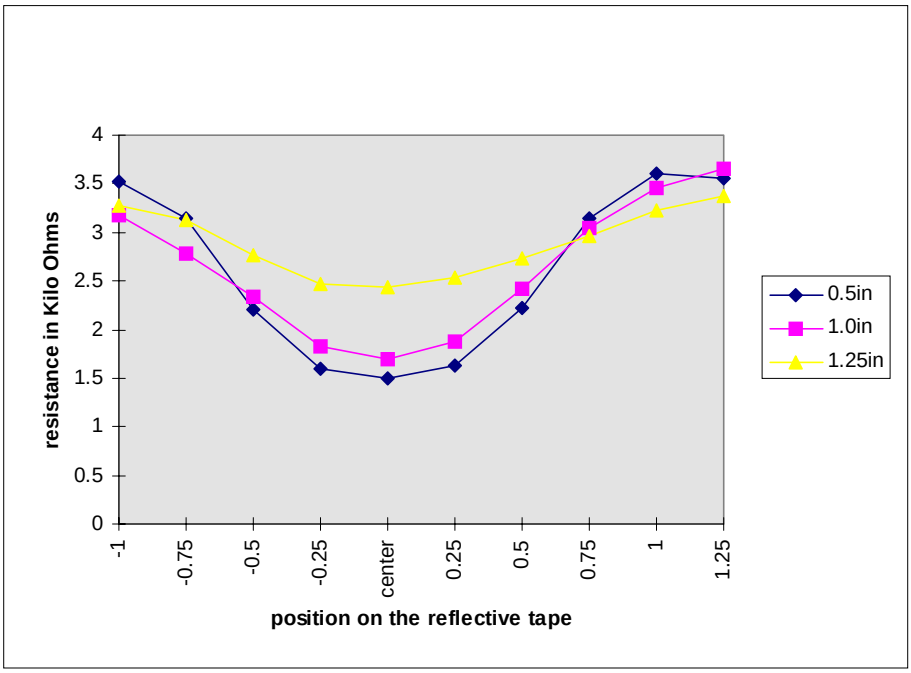


Figure 5.9: Resistance variation of the photocell on reflective tape

5.7. Steering and Movement Algorithms for the Car

Several algorithms were designed to test the steering and running of the car, in order for the car to follow a path defined by reflective tape. As mentioned in section 5.6, the guidance circuit determines the movement of the car and consists of two photocells, which act as 'eyes' for the car. The two photocells are used to control the stepper motors. Resistance of the photocells is a direct function of the light seen by the photocells. The resistance in turn is converted into a digital number by the guidance circuit. Therefore the photocells generate digital numbers depending on the amount of light. These numbers are used as the variables that determine the speed of the stepper motor. Tests were conducted to determine the range of digital numbers generated by the photocells when they are looking at the white reflective tape and the black paper. The photocells were tested at two different locations. The first location was when the car was directly

under the room light where both photocells received equal lighting. The second location was the testing area where the right photocell is located nearer to the overhead light than the left photocell. The results show that the numbers generated by the photocells are consistent with the change in the incident light. Figure 5.10 illustrates the testing points with reference to the photocells and the reflective tape. The photocell readings were recorded at twelve different positions, six each for each photocell. The left and right photocells are marked 'L' and 'R' respectively. The positions are named A through L.

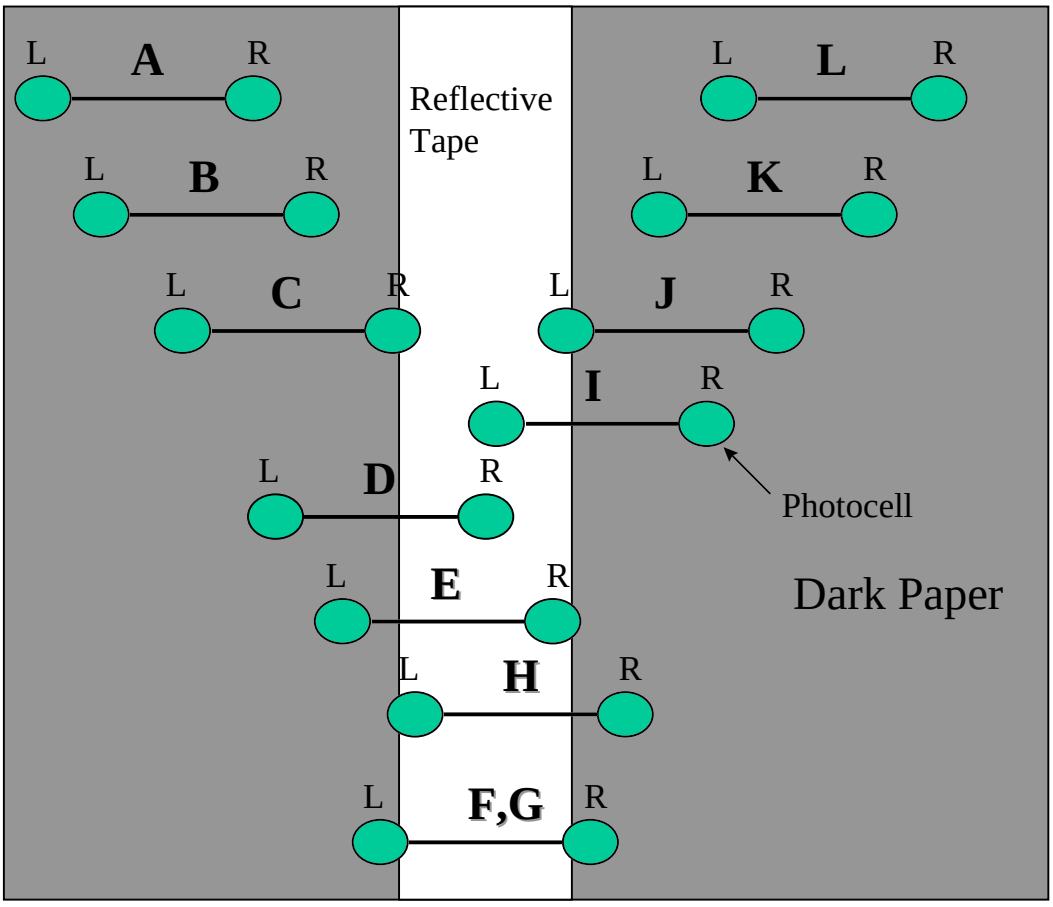


Figure 5.10: Testing points for the photocells with reference to the reflective tape

Table 5.5 contains the numbers read by the photocells at each position. Figure 5.11 illustrates a plot of numbers generated by left and right photocells. In this graph, the

area in which a specific reading is located determines the steering action for the car at that point.

Point	A	B	C	D	E	F	G	H	I	J	K	L
Left	1373	1379	1444	1523	1797	2275	2231	2986	3566	2903	1390	1255
Right	1643	1948	3366	3633	2798	2052	2065	1732	1614	1585	1567	1566

Table 5.5a: Location 1: Below the Overhead Light

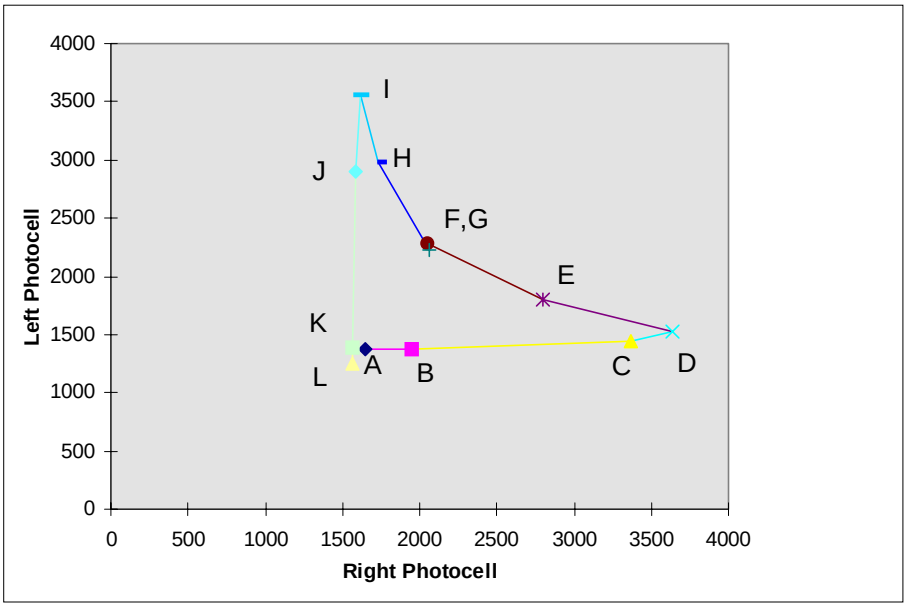


Figure 5.11a: Digital output of the photocells on the reflective tape for location 1

Point	A	B	C	D	E	F	G	H	I	J	K	L
Left	687	705	733	775	864	1169	1186	1966	2259	1545	732	734
Right	1184	1393	2757	2994	2018	1253	1253	1157	1159	1142	1113	1128

Table 5.5b: Location 2: Testing area

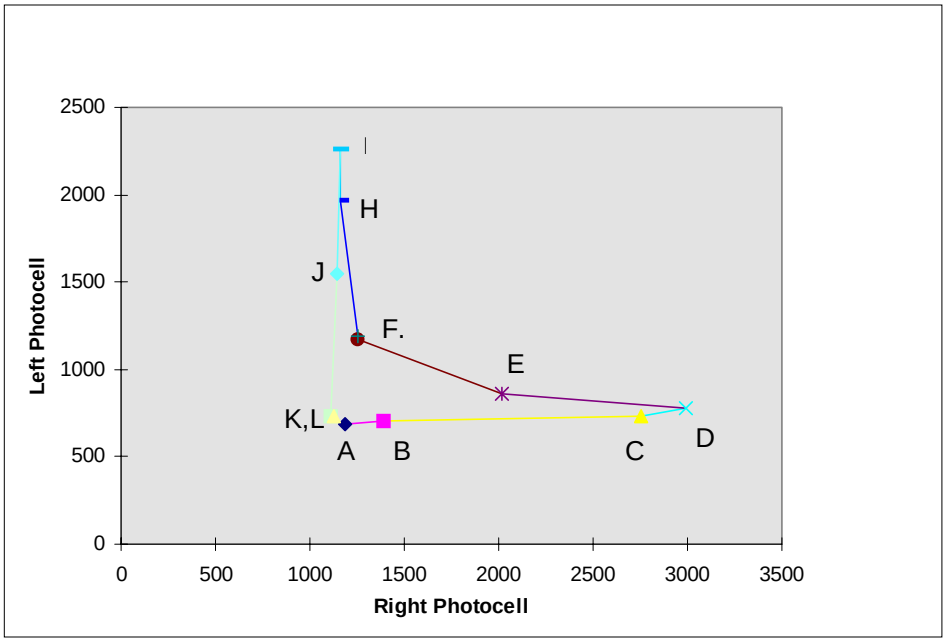


Figure 5.11b: Digital output of the photocells on the reflective tape for location 2

5.7.1. Algorithms

This section describes the different algorithms considered to accomplish the steering and running of the car. The variables that decide the direction of the car are the photocell readings.

5.7.1A. Difference Algorithm

This algorithm is based on the difference between the light seen by the two photocells. Since the resistance of the photocell changes with light, the photocell nearer to the reflective tape will record a higher reading. The algorithm periodically checks the difference between the two photocell readings and steers the car towards the side which records a higher reading. The steering is accomplished by having one stepper motor run slower than the other. This algorithm works reasonably well if the amount of room light incident on the dark paper and reflective tape is evenly distributed. It cannot be ensured that this even distribution will occur since it would depend on the room light and the location of the setup. Furthermore, the relative positions of the two photocells will often result in one photocell receiving less light than the other. This difference is indicative of

the relative location of the two wheels of the car, but does not tell much about the absolute location of the car with reference to the dark paper and reflective tape.

5.7.1B. Region Algorithm

This algorithm uses the information about the location of the car with reference to the reflective tape to make decisions about the speed of the stepper motors. This algorithm relies upon the photocell readings and assumes that the photocell recording a higher reading is nearer to the reflective tape. Like the difference algorithm, this works reasonably well if even distribution of light and consistent photocell readings occur.

5.7.1C. Combination Algorithm

This algorithm combines the working principles of the difference and region algorithm. This algorithm uses the result of the photocell tests to determine the dark and light regions. The algorithm periodically checks the two photocell readings. It uses the sum of the readings to determine the region. Using the sum of the readings in addition to the difference indicates the absolute location of the car. The photocells record higher readings around the reflective tape and if the sum of the readings exceeds a limit the car is near the tape as opposed to the dark region. This algorithm is used to drive the car.

The photocell readings can be measured using different techniques, e.g. the average of several values or a fraction of the previous reading in combination with the current reading to retain some history.

5.7.2. Software

This section describes the programs that decide the steering and running of the car. Different functions are written to accomplish the sequence of actions needed for the movement of the car. The car goes through three main actions while it is running. Accordingly, the functions can be broadly classified into sense or look, think, and act categories.

5.7.2A. Sensing Function

This function is intended to periodically check the reading of the sensors. This function reads the analog channels which record intensity of light sensed by the photocells. The sensing function is described below. The analog channels can be read using the library function ad_ad12(chan), where chan is the channel number being read.

Program statements:	**Description:**
sum=0; avg=0; I=6;	The values of sum, avg, and I are initialized. Sum and Avg are used to sample channel I ten times and average the results
for(k=0;k<10;k++){	Beginning of the for loop
ad_ad12((unsigned char)I);	This call to the program flushes the old value
adat= ad_ad12((unsigned char)I);	Here the digital output from channel 6 is stored in adat.
sum+=adat;	The adat value obtained is added to sum
}	
light=sum/10;	Average sensor reading obtained by dividing sum by the number of readings, 10.

5.7.2B. Thinking Functions

These functions take the sensor readings as input and make a decision about the relative speed of the two stepper motors. The general assumption is that the program compares the two sensor readings to decide which wheel is closer to the reflective tape and steer towards the tape. The sum of the two readings indicates the absolute location of the car on the dark paper, which helps in deciding the speed and depth of the turns.

Program Statement	**Description**
Void speed(int ll, int rl){	program is declared as taking two integer inputs and returning no value.

```
If(ll>1750){                    check if left light is greater than 1750
  If(ll>1750){                  check if right light is greater than 1750
     Left_speed=2;              set left motor speed to 2
     Right_speed=2;             set right motor speed to 2
  }
else{
        Left_speed=1;           set left motor speed to 1
        Right_speed=2;          set right motor speed to 2
  }
}  else  {                      if left light is less than 1750
   If(rl>1750){                 check if right light is greater than 1750
     Left_speed=2;              set left motor speed to 2
     Right_speed=1;             set right motor speed to 1
  } else If(rl>ll){             check if right light is greater than left light
     Left_speed=2;              set left motor speed to 2
     Right_speed=1;             set right motor speed to 2
  } else{                       if right light is less than left light
     Left_speed=1;              set left motor speed to 2
     Right_speed=2;             set right motor speed to 2
  }
}                               end function
```

5.7.2C. Acting functions

These functions take the input from the thinking program and execute the sequence of actions required to physically move the car. Different functions are used to execute each action.

STEP: This function takes an integer as input and updates the value and returns the new value. This function establishes the sequence of numbers that are used to fire the stepper motor in the required order.

Program statement	Description
Int step(int oldstate){	The function definition which indicates that the function requires an integer as input and returns an integer.
Int newstate;	This declares an integer variable newstate.
Switch(oldstate){	Case statement begins
Case 9: newstate=3;	If oldstate = 9, then assign newstate=3.
Break;	exit the case statement.
Case 3: newstate=6;	If oldstate = 3, then assign newstate=6.
Break;	exit the case statement.
Case 6: newstate=12;	If oldstate = 6, then assign newstate=12.
Break;	exit the case statement.
Case 12: newstate=9;	If oldstate = 12, then assign newstate=9.
Break;	exit the case statement.
}	The case statement ends.
Return newstate;	The program returns newstate as an integer.
}	//Program end.

TURN: This function requires two integers as input and returns no value. The two input integers represent the hexadecimal digital input numbers for the digital port, which controls the stepper motors. The function converts these integers into hexadecimal numbers and invokes the digital output writing function with the numbers.

Program statement	Description
Void turn(int pos_left, int pos_right){	Function header accepts two integers and returns no value.
Int sec=5;	declare an integer value sec which is the delay in milliseconds between two rotations.
Outportb(PPI2, (pos_left<<4)|(pos_right));	This statement converts pos_left and pos_right into hexadecimal numbers and invokes function outportb. PPI2 is digital port 2 which is initialized for output.

Delay_ms(sec);	invoke the delay function which delays the program for a few milliseconds.
}	// End program

SPIN: This function requires two integer values as input. The two variables are the speed of the two stepper motors, which are decided by the thinking program. The function uses two other functions namely STEP and TURN functions. The SPIN function turns the wheels as directed by the stepper motor speeds.

Program statement	**Description**
Void SPIN(int left, int right){	This statement is the function definition which indicates that the function requires two integer inputs and the returns no value.
Static posl=3, posr=3;	This statement declares variables posl and posr and initializes them to a value of 3. These are starting points for the rotation of the stepper motor.
If(left>=1){	if the speed of the left stepper motor is greater than 1 the program
Posl=step(posl);	invokes the STEP program to update the value of
}	posl.
If(left>=1){	if the speed of the right stepper motor is greater than 1 the program
Posr=step(posr);	invokes the STEP program to update the value of
}	posr.
Turn(posl,posr);	Invoke the TURN function with the updated values of posl and posr.
If(left>=3){	if the speed of the left stepper motor is greater than 3 the program
Posl=step(posl);	invokes the STEP program to update the value of

}	posl.
If(right>=3){	if the speed of the right stepper motor is greater than 3 the program
Posr=step(posr);	invokes the STEP program to update the value of
}	posr.
Turn(posl,posr);	Invoke the TURN function with the updated values of posl and posr.
If(left>=2){	if the speed of the left stepper motor is greater than 2, the program
Posl=step(posl);	invokes the STEP program to update the value of
}	posl.
If(right>=2){	if the speed of the right stepper motor is greater than 2, the program
Posr=step(posr);	invokes the STEP program to update the value of
}	posr.
Turn(posl,posr);	Invoke the TURN function with the updated values of posl and posr.
If(left>=4){	if the speed of the left stepper motor is greater than 4 the program
Posl=step(posl);	invokes the STEP program to update the value of
}	posl.
If(right>=4){	if the speed of the right stepper motor is greater than 3 the program
Posr=step(posr);	invokes the STEP program to update the value of
}	posr.
Turn(posl,posr);	Invoke the TURN function with the updated values of posl and posr.
}	// End of the function

5.8. Obstacle Detection

One of the specifications for the car is that the car must execute a timed course. At the end of the course, the car must be able to detect an obstacle and stop before the main body of the car contacts the obstacle. This is accomplished by equipping the car with bumper like structures, that are interfaced to the processor through a digital input port.

5.8.1. Obstacle detection setup

Two flexible thin strips of metal which act as bumpers are attached to the front of the car. This setup is illustrated in Figure 5.12.

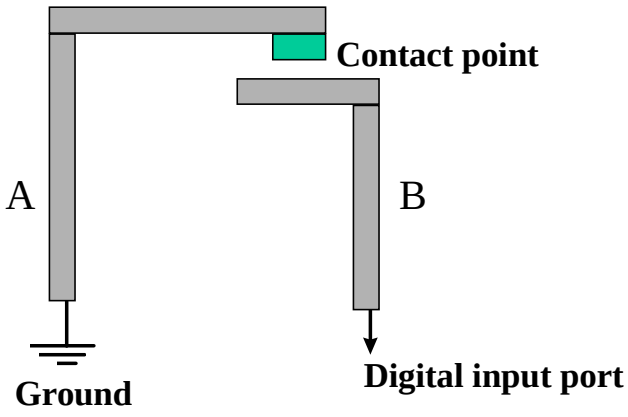


Figure 5.12: Bumper setup for obstacle detection

Two metal strips A and B are attached to the front of the car as shown in Figure 5.12. Strip A overlaps strip B and has a contact point on its inside surface. One end of strip A is connected to ground and strip B is connected to the most significant bit of a digital port. The digital port being used is port1. Under normal conditions, the digital port output is the number 255. When the car nears an obstacle, strip A makes contact with the obstacle. The force on strip A causes it to bend and touch strip B through the contact point. This causes the circuit to close and since strip A is connected to ground, this will also ground out the most significant bit on port 1 which is connected to strip B.

When one of the bits is grounded, the output of the port will be less than 255, in this case 255 minus 2^7 = 255-128=127. The section of the program which checks port1 is described below. When the output of port1 equals 127, the operating software program stops the car by interrupting the stepper motors on the drive wheels.

5.8.2. Software

The program to accomplish obstacle detection consists of a simple test to check the input of the digital port and send interrupt signals for the motors. This section of the program is described below.

Program Statement	**Description**
Int inport1;	This statement declares an integer variable inport1.
Outportb(PPIC, 0x82)	The outportb statement sets the command status word to a value, hexadecimal number 82, which enables Port1 to work in the input mode and Port2 to work in the output mode. Port1 is used for obstacle detection and Port2 is used to drive the stepper motors.
Inport1=inportb(PPI1);	This statement invokes inportb procedure. Inportb is the function used to read output of port1. The value is assigned to the variable inport1.
If(inport1>127){	This if statement checks if the output of port1 is greater than 127. If the check is true, the obstacle is not detected and the program to run the stepper motors is executed.
// Run Stepper motors	
else{	This part of the program is executed if inport1 reads a value of 127 or less. This means that an obstacle is detected and the motors have to stop running.
Spin(0,0);	To stop the stepper motors, the SPIN function is invoked with left and right speed values of 0. This will stop the

motors, which in turn will stop the car. The car will resume running once the obstacle is removed from its path.

5.9. Testing the pulling force of the car

To simulate a 'tractor pull' or 'tug of war', the pulling force of the vehicle is measured using a strain gage load cell. The car is positioned on a 9" x 16" piece of wood which has the load cell attached to one end. The setup is illustrated in Figure 5.13.

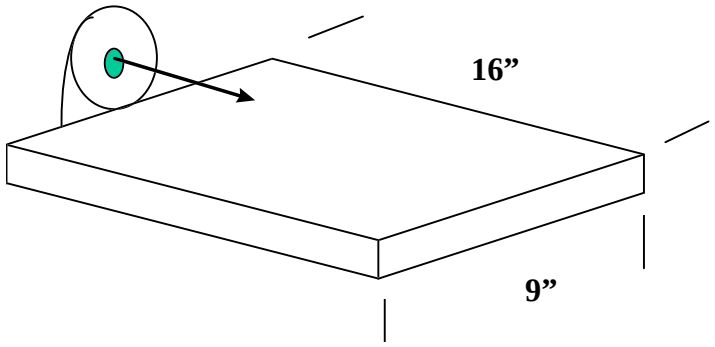


Figure 5.13: Setup for Tug of War test

The car is connected to load cell as the stepper motors are energized. The pulling force is measured by the load cell.

The load cell is a Sensotec model 41 strain gage transducer with a range of -/+50 lbf. A Sensotec model 450D signal conditioning unit displays the force to two decimal places.

The equipment is used to measure the force corresponding to the load pulled by the car. The force is measured against a change in the supply voltage for the car and the delay value in the driving program for the car. The delay variable is a specific number of milliseconds for which the program pauses between successive turns of the stepper motor. Delay values ranging between 8ms-22ms are used for the test. Voltages ranging

between 4V-15V are used for the test. Force is measured when the program is using each delay and the supply voltage increases from 4V to 15V. The results are illustrated in Table 5.6 and Figure 5.14.

Voltage	delay=8	delay=10	delay=12	delay=14	delay=16	delay=18	delay=20	delay=22
4	0.07	0	0	0.11	0.14	0.08	0.08	0.08
5	0.08	0.07	0.13	0.12	0.16	0.1	0.1	0.09
6	0.09	0.08	0.16	0.13	0.17	0.12	0.11	0.1
7	0.08	0.09	0.18	0.19	0.18	0.12	0.11	0.1
8	0.12	0.1	0.2	0.22	0.2	0.13	0.17	0.11
9	0.13	0.12	0.26	0.18	0.22	0.13	0.22	0.12
10	0.14	0.13	0.37	0.24	0.25	0.14	0.32	0.13
11	0.14	0.14	0.44	0.27	0.28	0.16	0.36	0.14
12	0.15	0.15	0.47	0.32	0.28	0.19	0.43	0.16
13	0.19	0.16	0.54	0.4	0.26	0.23	0.44	0.33
14	0.2	0.17	0.51	0.44	0.28	0.28	0.47	0.37
15	0.22	0.18	0.5	0.5	0.3	0.39	0.5	0.43

Table 5.6:Force(lbf) measured with reference to changes in supply voltage and delay

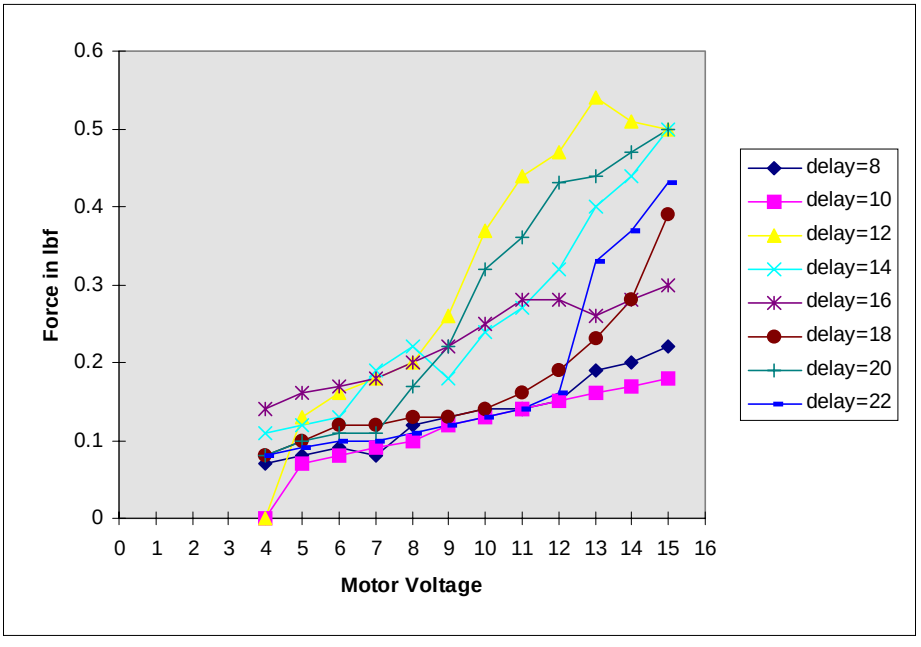


Figure 5.14: Tug of war test results

Chapter 6
Results

This section explains the results of the tests conducted on the computer controlled autonomous model car. Several tests were conducted to determine the optimum running conditions for the car. Using these optimum settings, the car was tested on the timed course and the results described.

6.1. Computer Controlled Autonomous Model Car

The model car had to comply with all the requirements mentioned in the project specifications. Components of the model car were mounted on an aluminum frame. The overall dimensions of the car are measures 10.5 inches long, 4.25 inches wide and 5 inches high. The car includes a microprocessor board, two stepper motors each of which drive one wheel of the car, and assorted electronic components. The frame made out of aluminum consists of a flat bed which supports the microprocessor board and a rectangular enclosure which houses the two stepper motors. Four inch diameter rubber wheels are mounted on the two stepper motors that provide independent front wheel drive. The stepper motors are connected to the microprocessor board through a circuit. Digital port port2 is used to run the motors. The stepper motor schematic and test are illustrated in Figure 5.6, Figure 5.7, and Table 5.2. The startup and guidance circuits use ADC channels 6,8 and 9 for input. These ADC channels were tested for gain provided by each of them. Channels with a gain of 1 are used. The setup and results from ADC channel gain test are illustrated in Figure 5.4, Figure 5.5, and Table 5.1. The startup circuit for the car consists of a photocell and a resistor in series, which starts the car only when the room is essentially dark. The startup circuit uses the ADC channel 6. The startup circuit is illustrated in Figure 5.1, Figure 5.2, and Figure 5.3. The guidance circuit is mounted at the front of the car on a plastic holder. Each of the two guidance circuits consists of a photocell and resistor connected in series. A software program monitors the circuits and controls the movement of each wheel. The photocells are mounted facing the roadway at a height of 0.5inches. Tests were conducted to determine the optimum

distance of the photocells from the ground so as to get best range of readings. Figure 5.9 and Table 5.4 illustrate that the resistance is the lowest and hence the readings are the highest when the photocells are 0.5inches from the ground. This circuit uses ADC channel 8 to control the left wheel, and channel 9 to control the right wheel. The bumper setup for obstacle detection is attached to the front of the car. This setup is illustrated in Figure 5.12. Digital port port1 is used for obstacle detection.

6.2. Performance Results of the Car

One of the specifications for the car is that the car has to execute a timed course. The car has to follow a path defined by white reflective tape on dark paper. At the end of this course defined by reflective tape, an obstacle is placed. The car has to detect the obstacle and stop before the main body of the car makes contact with the obstacle. The setup for this test is illustrated in Figure 6.1.

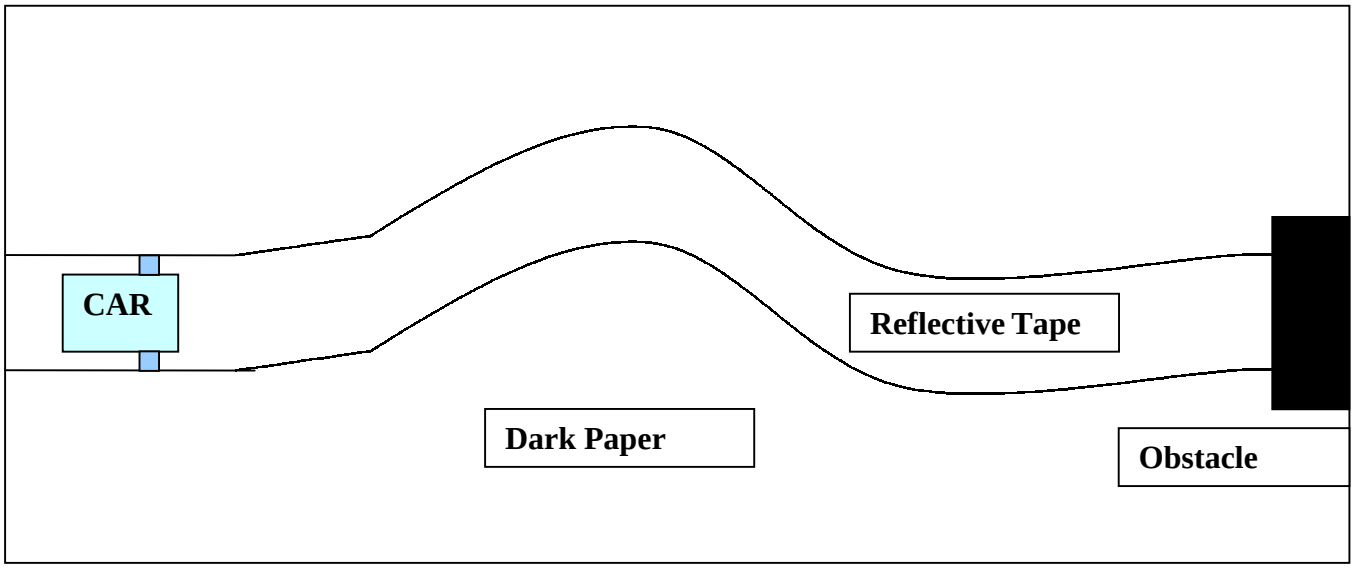


Figure 6.1: Setup for the timed course test for the car

The Figure illustrates the course defined by reflective tape. The reflective tape is about 2 inches wide and is taped on dark construction paper to define a path. The path consists of

about 32 inches of straight tape followed by an arc of radius of curvature about 17 inches, which is followed by 6 inches of tape at the end leading to the obstacle. An obstacle is placed at the end of the tape. The car is placed at the beginning of the tape.

The driving program for the car is executed and the start time is recorded. The driving program tries to follow the reflective tape faithfully and tries to accomplish this as fast as possible. When the car reaches the end of the tape and it stops when it detects the obstacle, and the end time is recorded. The speed of the car is determined by the voltage supplied to the stepper motors and the delay variable in the driver program of the car. The delay variable is a specific number of milliseconds the program pauses between two successive turns of the stepper motors. Delay variables ranging from 6ms to 16ms were used to determine optimum running conditions, at a supply voltage of 12V. The delay variable to be used depends on the speed of the motors. A lower delay seems to work better for higher speeds whereas at lower speeds a slightly higher delay works favorably. The car seems to run smoothly at a combination of lower speeds, medium delay and a voltage of 12VDC. The delay values and the corresponding time to finish the course are illustrated in Table 6.1.. A supply voltage of 12VDC was used drive the car.

Delay in milliseconds	Time in seconds
6	26
8	25
10	27
12	33
14	35
16	41

Table 6.1: Time taken to execute a timed course

As the results show, a delay of about 8 milliseconds combined with a supply voltage of 12VDC yields the fastest times.

Chapter 7

Discussion

7.1. Microprocessor Board

As described in Section 5, the Tern, Inc. models V104 and A-drive were chosen as the standard interfacing boards for this project. After working with the boards over the length of this project, several areas need to be discussed.

The specifications listed for the boards are ambiguous. Specifically, the boards are described as featuring 12 channels of digital to analog conversion capability. However, in reality the two digital to analog converter chips do not come with the package. Sockets are provided, but the chips are missing. Similarly, thirty-three channels of analog to digital conversion are implied, but only ten channels can be used simultaneously. Adding the options to meet the original specifications would increase the price to nearly $1000, compared to the already expensive $600 price. Based upon capabilities, the board seems very expensive.

7.2. Startup Circuit

The startup circuit consisted of a relatively simple voltage divider circuit with the voltage output sensed by an analog to digital converter channel. The Radio Shack model 276-1657 photocell in series with a 1105 ohm carbon resistor provided good reliable results. It was noticed however that the variations in light in a room with strip lighting and a large window produced a greater variation in circuit output than expected. Since the circuit only had to distinguish “lights on” versus “lights off”, the system functioned well.

7.3. Stepper Motor Motion Control

Front wheel drive, independently controlled stepper motors were used in this car. The Jamenco model SM3401 motors were rated for 24 volts DC, but were supplied with only 12 volts DC for this project due to limited space for the batteries plus the desirability

of using one battery to supply the motors and the other electronics on the car. The other electronics required 9 volts DC but would accept 12 volts. This limited the torque of the motors and the resulting speed of the car.

Other modes of motion control were considered. Using continuous rear wheel drive with stepper motor steering was rejected due to complexity. Using variable speed motors would have required the digital to analog converters which were not supplied with the boards. These chips could have varied the voltage to the motor which would have controlled the speed.

The software program that provided a sequence of pulses to the stepper motors had to contain a delay variable. The electro-mechanical motors could not respond as fast as the computer. As shown in Figure 5.8, as the delay became smaller, an increase in motor speed was possible but only if the supply voltage was increased. This limited the motor speed since the supply voltage was limited to 12 volts DC for a delay of 3 milliseconds.

The motor speed also was affected by the 'tug of war' constraints which required more torque than speed. As shown in Figure 5.14, a delay of 12 milliseconds produced the maximum pulling force of about 0.5 pounds. The compromise between torque and speed was also evident in the timed circuit test, where the stepper motor needs speed in the straight sections but torque in the turning sections. According to Table 5.7, a delay of 8 milliseconds produced the fastest time. This could change if a different path is chosen or if different sized wheels are used. The delay time finally chosen was 8 milliseconds to produce the fastest timed circuit times.

7.4. Obstacle Detection

Several methods of obstacle detection were considered. An ultrasonic transmitter and receiver could bounce a wave off of an obstacle. The technique is sensitive enough to actually yield the distance from the car to an obstacle. However, it is expensive.

A photocell could detect light reflected off of a nearby obstacle, but is not reliable in a room with variable lighting.

The method chosen of using a mechanical contact (Section 5.8.1) is cheap and reliable. However, the thin spring steel used in the device vibrated excessively during

operation of the stepper motors. This produced some spurious signals or contacts even when the obstacle was not near the car.

Chapter 8

Conclusions, Contributions and Recommendations

8.1. Conclusions

Based upon the objectives, results, and discussion of this research, the following conclusions can be made.

1. A computer controlled autonomous model car can be designed as a successful mechatronics project.
2. The Tern, Inc model V104 and Analog Drive control boards can successfully control a model car. These boards in standard configuration do not incorporate digital to analog interface capability.
3. The variable reaction time inherent in using a human to start the car can be eliminated with a voltage divider circuit containing a photocell that responds to room lights being momentarily turned off.
4. Stepper motors can be used provide movement and guidance for the car to execute a predetermined path.
5. Photocells incorporated in voltage divider circuits can provide input to an analog to digital conversion channel that can be used to guide the car along a reflective path.
6. A mechanical switch connected to a digital input channel can be used to detect an obstacle in the path of the car.
7. A 12-volt DC powered autonomous car can develop a pulling force of at least 0.5 pounds.

8.2. Contributions

Mechatronics course structures developed in other universities were studied in the beginning of this design project. Several approaches have been attempted to achieve mechatronics education in engineering studies. The courses contain lecture and laboratory components and emphasize the interdisciplinary nature of mechatronics. Several design projects developed in other schools were studied. Some of these projects are discussed in detail in the literature review. The main objective of this design exercise

is for students to use the principles of mechatronics in the design projects. Most of the lab projects require the students to use the elementary microprocessor concepts such as assembly level programming, analog-to-digital and digital-to-analog conversions, analog and digital input and output. Some typical mechatronics design projects include three-dimensional surface digitizing laser scanner, cylindrical robot with optical tracking capabilities, high speed flexible robot arm and delivery system, electro-optical distance scanner, design of an approximate linear mechanism, control of an inverted pendulum, control of an inverted pendulum, and single axis feed table for NC machine.

The objective of this project was to develop a mechatronics design project around the Tern, Inc microprocessor boards, which could be integrated into an existing undergraduate instrumentation-engineering course.

This design project included exclusive testing of the capabilities of microprocessor boards to determine the optimum conditions that yield the best results. These tests can help other students in their lab projects in understanding the features and limitations of these boards. This project also contributed significantly to the upgrading of the existing undergraduate engineering course to include the concepts of mechatronics.

8.3. Recommendations

It is recommended that:

1. Other cheaper boards be evaluated that contain analog to digital, digital to analog, digital input, and digital output interfacing.
2. Other means of propulsion, such as variable speed DC motors be evaluated to increase the speed during the timed circuit.
3. Other means of obstacle detection such as ultrasonic sensors be tested.

References

1. D.M. Auslander, University of California, Berkeley, "What is Mechatronics?" IEEE/ASME Transactions on Mechatronics.
2. F. Harashima, M. Tomizuka, and T.Fukuda, editorial IEEE/ASME Transactions on Mechatronics.
3. N. Kyura and H. Oho, "Mechatronics- An Industrial Perspective", IEEE/ASME Transactions on Mechatronics, March 1996, Pp. 10-15.
4. R. Isermann, "Modeling and Design Methodology for Mechatronic Systems", IEEE/ASME Transactions on Mechatronics, March 1996, Pp. 16-28.
5. K. Youcef-Toumi, "Modeling, Design and Control Integration: A Necessary Step in Mechatronics", IEEE/ASME Transactions on Mechatronics, March 1996, Pp.29-38.
6. N. Kyura, "The Development of a Controller for Mechatronics Equipment", IEEE/ASME Transactions on Mechatronics, March 1996, Pp. 30-37.
7. C.R. Luo, "Sensor Technologies and Microsensor Issues for Mechatronic Systems", IEEE/ASME Transactions on Mechatronics, March 1996, Pp. 39-49.
8. G.M. Prabhu, C.T. Wright, "The use of Microcontrollers in Mechatronics education", Proceedings of the Workshop on Mechatronics Education, organized by The Synthesis Coalition of the National Science Foundation, Motorola Inc, Stanford University, July 21, 1994.
9. R. Shourshi, "Foundations for Mechatronics Education", Proceedings of the Workshop on Mechatronics Education, organized by The Synthesis Coalition of the National Science Foundation, Motorola Inc, Stanford University, July 21, 1994.
10. C. Ume, M. Timmerman, "Mechatronics Instruction in Mechanical Engineering Curriculum at Georgia Tech", Proceedings of the Workshop on Mechatronics Education, organized by The Synthesis Coalition of the National Science Foundation, Motorola Inc, Stanford University, July 21, 1994.
11. D.G. Morin, "The Design and Implementation of a First Course in Microprocessor Technology for Mechanical Engineers", Proceedings of the Workshop on

Mechatronics Education, organized by The Synthesis Coalition of the National Science Foundation, Motorola Inc, Stanford University, July 21, 1994.

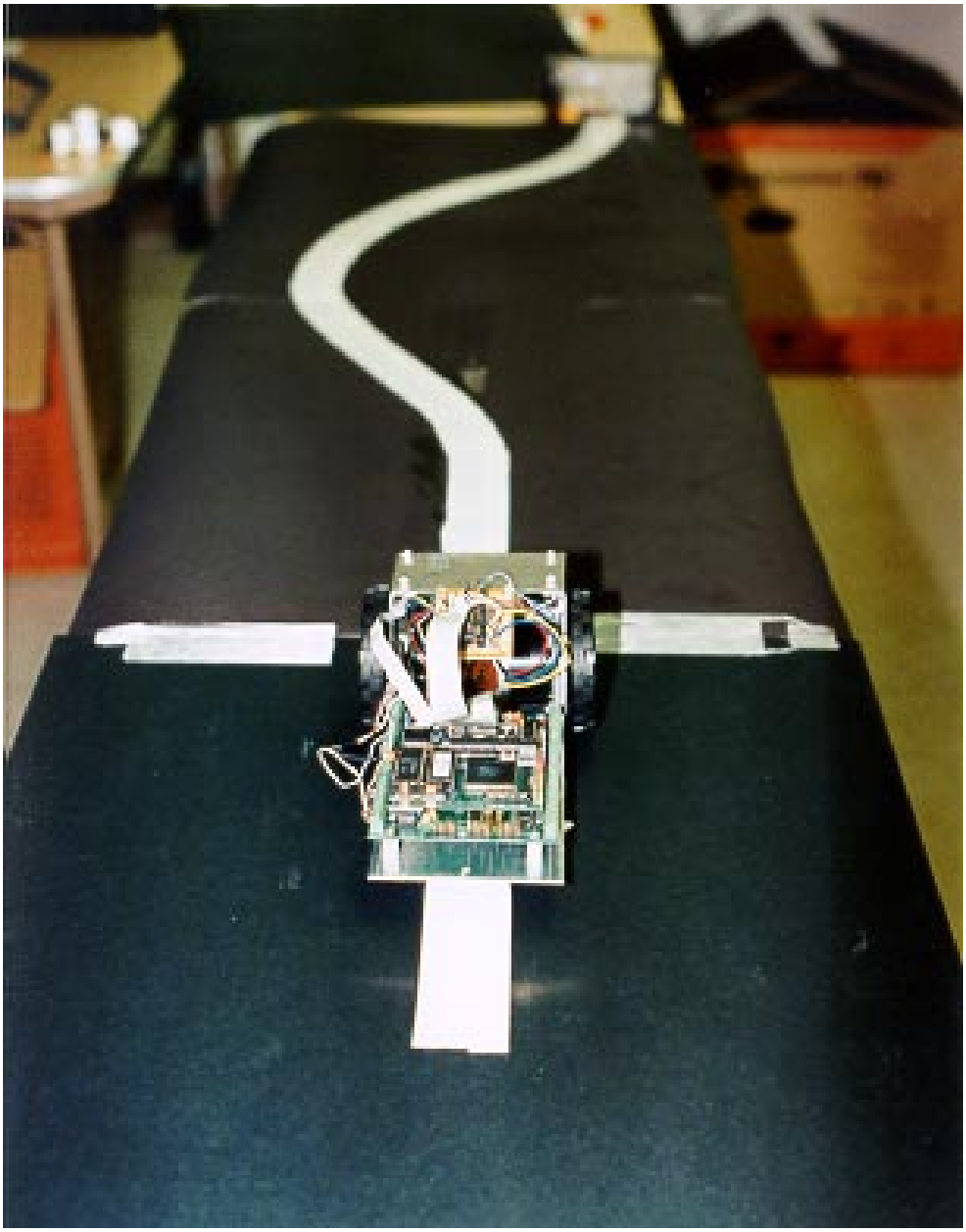
12. K. Craig, "Mechatronic System Design at Rensselaer", Proceedings of the Workshop on Mechatronics Education, organized by The Synthesis Coalition of the National Science Foundation, Motorola Inc, Stanford University, July 21, 1994.
13. K. Sasaki, "Getting started from the Physical World", Proceedings of the Workshop on Mechatronics Education, organized by The Synthesis Coalition of the National Science Foundation, Motorola Inc, Stanford University, July 21, 1994.
14. W. Harwin, R. Foulds, "Teaching Mechatronics at the University of Delaware", Proceedings of the Workshop on Mechatronics Education, organized by The Synthesis Coalition of the National Science Foundation, Motorola Inc, Stanford University, July 21, 1994.
15. D.G. Alciatore, M.B. Histand, "Mechatronics and Measurement Course at Colorado State University", Proceedings of the Workshop on Mechatronics Education, organized by The Synthesis Coalition of the National Science Foundation, Motorola Inc, Stanford University, July 21, 1994.
16. G. Rizzoni, A. Keyhani, "Design of Mechatronic Systems: an Integrated Inter-Departmental Curriculum", Proceedings of the Workshop on Mechatronics Education, organized by The Synthesis Coalition of the National Science Foundation, Motorola Inc, Stanford University, July 21, 1994.
17. R. Rajagopalan, R. Cheng, H. Hong, G. Huard, "Teaching Mechatronics to Mechanical Engineering Students at Concordia University", Proceedings of the Workshop on Mechatronics Education, organized by The Synthesis Coalition of the National Science Foundation, Motorola Inc, Stanford University, July 21, 1994.
18. M.P. Hajnal, "Teaching Mechatronics to Information Engineering Students", Proceedings of the Workshop on Mechatronics Education, organized by The Synthesis Coalition of the National Science Foundation, Motorola Inc, Stanford University, July 21, 1994.
19. A. Memis, "Mechatronics Engineering Education in the U.K", Proceedings of the Workshop on Mechatronics Education, organized by The Synthesis Coalition of the National Science Foundation, Motorola Inc, Stanford University, July 21, 1994.

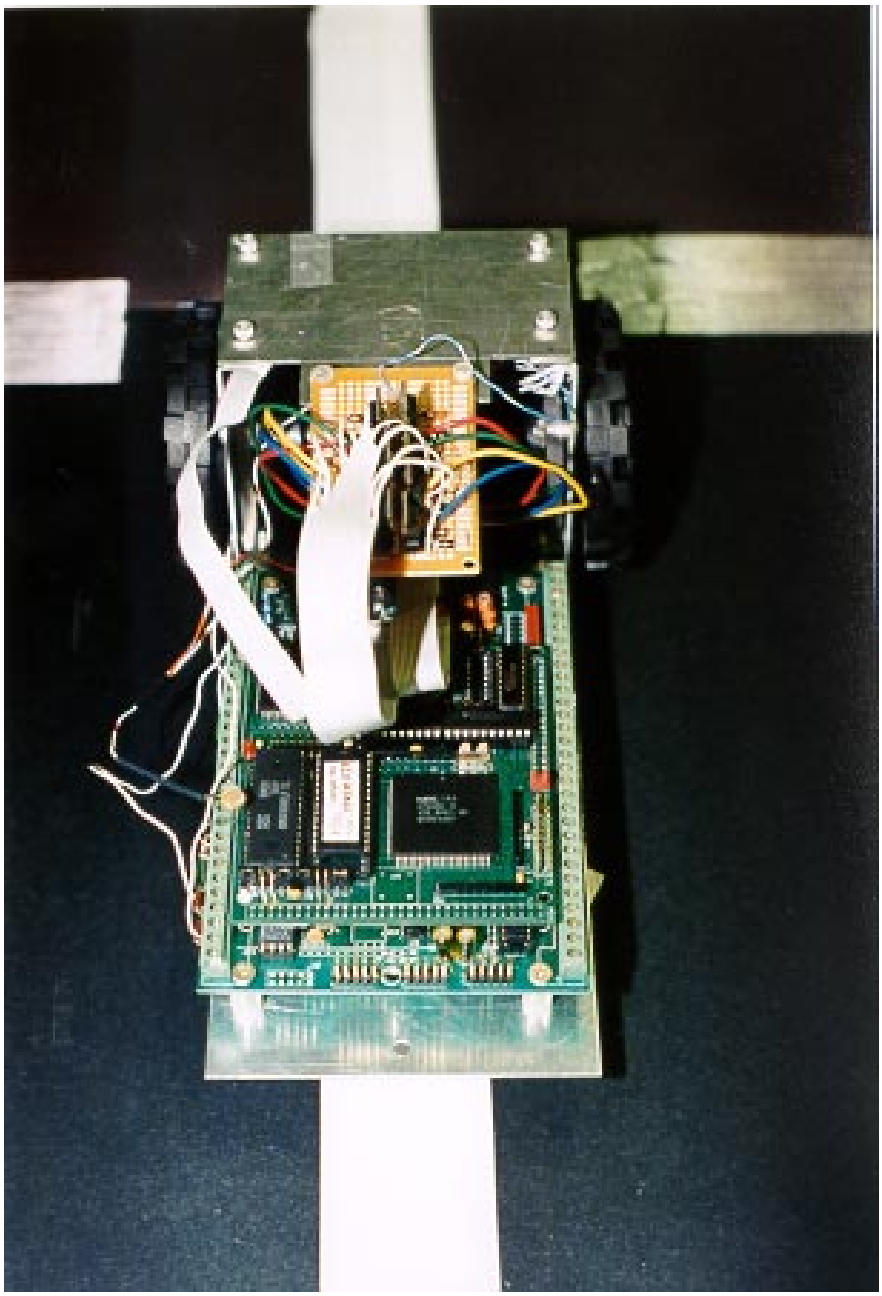
20. J.E. Carryer, “The design of Laboratory Experiments and Projects for Mechatronics Courses”, Proceedings of the Workshop on Mechatronics Education, organized by The Synthesis Coalition of the National Science Foundation, Motorola Inc, Stanford University, July 21, 1994.
21. U.R. Shashikiran, “A Computer Based System for Operating A Motorola 68HC11 Miniboard: A Mechatronics Application”, Thesis, Department of Industrial and Management Systems Engineering, West Virginia University, 1996.
22. T. Sriram, “Design and Development of a Mobile Smart Vehicle: A Mechatronics Application”, Thesis, Department of Industrial and Management Systems Engineering, West Virginia University, 1997.

Appendix

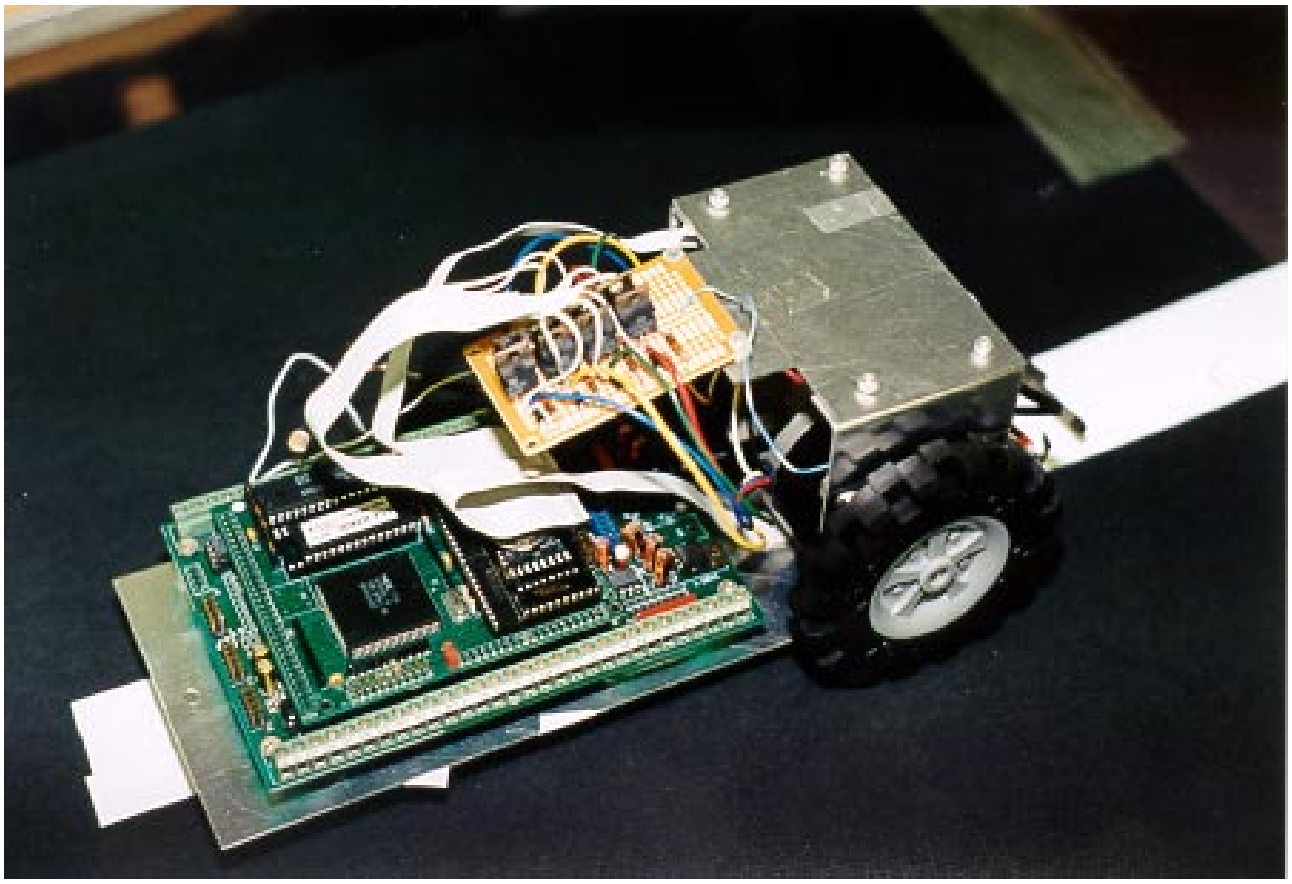
Pictures of the model car

1. Timed course setup for the model car





2. Front view of the model car



3. Profile view of the model car