UNIVERSITY OF TORONTO

INSTITUTE FOR AEROSPACE STUDIES

4925 Dufferin Street, Toronto, Ontario, Canada, M3H 5T6

IRON KITTY

Traffic Cone Deployment Machine

prepared by

Team 14 – Monday

Alexander Hong (997584706)
Andrew Lin (997419538)
Teddy Lin (997689283)

prepared for

Prof. M.R. Emami

A technical report submitted for
AER201 – Engineering design

TA: Peter Szabo

April 11, 2011

From left to right: Andrew Lin, Alexander Hong, Teddy Lin
Iron Kitty robot is held by Alexander's left hand

Acknowledgement

We would like to acknowledge our brilliant professor, *Professor M. Reza Emami*, for his helpful lectures and experiences, our teaching assistant, *Peter Szabo,* for his abundant knowledge, the Creatron store owner, *Lawrence Chan,* for his supplies and expertise, a graduated EngSci (ECE) and now teaching students at the Institute of Optical Sciences, *Emanuel Istrate,* for his electrical engineering expertise, and our fellow classmates of the AER201 course.

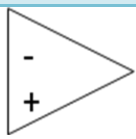
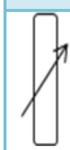
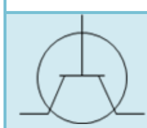
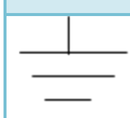
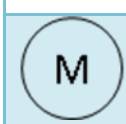
Abstract

The goal in this project was to design a machine that would autonomously deploy traffic cones along a traffic lane at pre-selected distances. The machine would also deploy a cone whenever a hole was detected, thus covering the hole. After completing this operation, it would then return to the start line, displaying the summary of the operation. The design that was chosen and implemented was one which used the concept of a skewed square coupled with a spring to deploy cones every time the square skewed in one direction. Simple DC motors were used to drive all parts and IR sensors were used to detect holes and assist the machine in travelling straight. The results of the final design were satisfactory. The machine always attempted to deploy cones when it was supposed to and succeeded in doing so 50% of the time. It also made a 180 degree turn at the end, returning to the start line and stopping, displaying the operation summary. The main problem was thus with the deployment mechanism and either a new mechanism needs to be designed or the current one must be improved until perfected.

Table of Contents

Symbols, Terms & Abbreviations

Symbols

Symbols	Description
<Keypad_Button>	The *Keypad_Button* button on the keypad
	Op Amp
	Diode
	Light Emitting Diode (LED)
	Resistor
	Variable Resistor
	Transistor
	Ground
	Motor
	Infrared Sensor Transistor

Terms

Black square and ***hole*** are used interchangeably
Cones and ***Traffic cones*** are used interchangeably
Iron Kitty name of prototype

General Abbreviations

Abbreviations	Description
CDN	Canadian dollars
DC	Direct Current
IO	Input / Output
IR	Infrared
LCD	Liquid Crystal Display
LED	Light Emitting Diode
PIC	Programmable Interface Controller
PWM	Pulse Width Modulation
RFP	Request for Proposal
RPM	Revolutions per Minute

Chapter 1:
INTRODUCTION

1.0 Statement of Need

Traditionally, traffic cone deployment has always been done manually by workers. The simplest method has always been using manual labour to carry the cones and placing them in their desired locations. Another alternative was to have stacks of cones lying on the back of a truck along with a worker. The truck would then drive along the lane, stopping every so often to allow the worker to get off and put a cone down. Historically, the traditional way of manually deploying cones has led to many problems. Foremost, there was the potential for injury. In 2004, $321,000 was paid out in lawsuits regarding injury due to traffic cone deployment [1]. Most of the injuries arose from workers being hit by oncoming traffic. For instance, in England, 2009, five workers were killed in this manner [2]. Another issue is that the cone deployment process is quite time consuming and often leads to long traffic delays. Also, they are usually weighted at the bottom, averaging a total weight of 10 kg [3]. This is difficult to manhandle and would expend a huge amount of energy just to place them all, especially in extreme weather [3]. You can also expect that it would take a great deal of time to manually deploy these cones, which is not ideal in the case of emergencies. However, not all is lost. With technology nowadays, engineers can design autonomous robots that will do the work for you and solve all the major concerns regarding traffic cone deployment.

1.1 Current Solutions

Many commercial solutions exist, such as using cone-carrying trucks that deploy cones by switches. A widely-known product that uses this idea is the AHMCT cone machine, which functions like a standard pickup truck with cones stacked, lying on their side (see Fig. 0-1). As the truck moves along the lane, the cones slide out, down a ramp, and lands on the road. It carries a maximum of 80 cones and also has the function of retrieving the cones [4]. More extreme solutions exist such as the Autocone 130, which carries a wheel of cones in a large cylindrical barrel (see Fig. 0-2). It then uses a robotic arm that swivels left and right to pick up the cone and place it on the road. It carries 130 cones and also has the retrieval function [5].

The current solutions have some problems associated with it. The AHMCT machine still requires an individual to operate it. With current technology, it is possible to make an autonomous machine that achieves the same task, and thus saving the city's money. The Autocone 130 has limitations such as the maximum number of cones it can carry and the space that it occupies. Furthermore, a driver must steer its path for cone deployment [5].

1.2 Goals & Objectives

1.2.1 Goals

1. A machine that deploys traffic cones smoothly and accurately
2. A fast and efficient machine
3. A light-weight, compact, and portable machine
4. A machine that is aesthetically pleasing
5. An easy-to-use user keypad interface and easily operable machine
6. A machine that is consistent and durable in cone deployment
7. A machine that is safe to the user and the public
8. An extendable prototype that represents realistic real world situations

1.2.2 Objectives

The machine should accomplish the following objectives:

1. Deploys cones correctly and accurately (± 5 cm), as initially specified by the user, along a lane marked by black tape.
2. Deploys a cone whenever a hole is detected, covering the hole.
3. Allows the user to initially input the desired distance between successive cones by incorporating a keypad and LCD screen
4. Returns to the starting line at the end of operation
5. Displays the number of cones deployed and the time of operation on return
6. Records the location of each cone within 10 cm of actual distance and allows the user to retrieve this information.
7. Records the location of each hole within 10 cm of actual distance and allows the user to retrieve this information

1.2.3 Constraints & Criteria

The machine should meet the following constraints and criteria:

1. Has an operational emergency stop button.
2. Has a carrying capacity of 10 cones.
3. Fits in a 50 cm x 50 cm x 50 cm envelope
4. Weighs less than 10 kg
5. Uses an onboard power supply during the operation
6. Returns to the starting line without knocking over any cones
7. Completes cone deployment operation under 3 minutes
8. Costs no more than $230 CND

1.3 Considerations

This section will reframe the problem, outline the assumptions made, and highlight potential problems that a cone deployment machine may run into.

This following will outline the assumptions made:

1. The terrain in which the machine will operate is a relatively smooth surface.
2. The terrain in which the machine will operate has a distinct colour other than the colour of the black hockey tape.
3. The demonstration will take place in a lit room.
4. The starting line is made of black tape
5. If a cone is deployed on a hole, the next cone is deployed in reference to the cone in the hole
6. The lane, if curved, will have a small curvature and no sharp turns.
7. In the case where the machine deploys cone right before a hole, thus making it difficult to deploy a cone on that hole, the following solution will be used: The machine will deploy the cone as instructed. When the hole is detected, the machine will move forward a certain distance such that the next cone will sit right beside the previous cone to avoid two cones being deployed on top of each other or hitting each other. In this case, the hole will be completely covered, but not centered by the cone

1.4 Initial Brain Storming

The team has came up with the following six designs for deployment of cones. (see **Table 0-1**).

1.4.1 Flap Holder

This solution incorporated using a tunnel with two sturdy yet flexible flaps that hold the stack of cone. A spring would be attached to the bottom of each flap. During deployment, these springs would pull back, pulling the flaps back, and allowing the first cone to fall through. The flaps would then recoil from the springs and catch the next cone.

1.4.2 Rotating Gears

This solution incorporated two gears sandwiching the stack of cones with predefined teeth that would hold the base of the stack of cones. As the gears turn inwards by one tooth, one cone is released and dropped to the road, allowing the next teeth in the gear to rotate in place and catch the next cone.

1.4.3 Twisting Coils

This solution incorporates two vertical columns of metal coils, held wide enough to fit the width of the cones. The cones rest as a stack on the grooves of the coils. The coils then twist, thus lowering the cones and eventually releasing them onto the ground below.

1.4.4 Claw Holder

This solution incorporated two U shaped claw holders to hold the base of the cones at the corners. The holders would have the base of the cone sandwiched between the two claws. The holders would then rotate 180 degrees along the axis of the top claw, releasing the bottom most cone and allowing it to fall onto the road. The top claw thus becomes the new bottom claw and catches the next cone.

1.4.5 Rotating Base

A square base with a square hole in the middle allows the cones to fall through. The cones are stacked at 45 degree angles to each other, thus preventing successive cones from falling through. When a cone is to be deployed, the base rotates, allowing the next cone to fall through. Since the following cone is at 45 degrees, it cannot fall through and remains rested at 45 degrees to the base until the next deployment.

1.4.6 Skewed Square

This solution incorporates a square base made of metal rods, with a square hole in the middle that allows the base of the cones to fall through, with the four sides connected to each other through loose screws, thus allowing rotation around the corners. The concept of our solutions pivots on the idea that since rotation can occur, the square base can deform into a skewed square or rhombus shape. In this shape, the base of the cones will be unable to fall through. This will be the initial condition of the base platform. A battering ram will then ram into the skewed square, pushing it back into a regular square shape, thus allowing the bottom most cone to fall through. A spring will then push the square shape back into a skewed square shape, preventing the next cone from falling through.

1.5 Division of Problem

The chosen solution was the skewed square concept. Each individual in the team were assigned a role to tackle a specific subsystem.

The circuit part was responsible for designing and constructing the electrical circuits that would be used to communicate between the microcontroller and the mechanical parts of the machine. This also involved acquiring power supplies, providing signal protection, and signal filtering.

The microcontroller part had the responsibility of programming the logic of the machine. The program was programmed in assembly language to maximize efficiency and minimize total memory used. Its main purpose was to tell the machine what to do by feeding the machine instructions and outputs and retrieving the inputs.

The mechanical part was responsible for designing and building the actual machine. This included designing the actual frame of the machine, designing the deployment mechanism, attaching all parts to the machine, and performing maintenance operations to ensure the functionality of the machine.

Chapter 2:
SUBSYSTEM: ELECTROMECHANICAL DESIGN

2.0 Introduction

The electromechanical subsystem is the most significant subsystem in the project. The proposed problem can be attacked in many different ways. How the machine will function and move has the greatest dependence on the electromechanical subsystem. The following chapter will described, in detail, the team's assessment of the problem, the solution chosen, background theories, supporting calculations, and potential problems with the subsystem, which can be further improved on.

2.1 Assessment of the Problem

The goal of the project is to design and manufacture a scale-down, proof-of-concept prototype of a mobile platform that can travel along a designated lane and deploy traffic cones according to specific instructions. The electromechanical aspect of the problem is to design a mobile unit and deploy cones effectively and correctly.

The machine is expected to deploy a number of small-size traffic cones within a designated lane. Dimensions and weight of the cone are shown in **Fig. A-1** and **Fig. A-2**. In addition, the machine is expected to carry at least 4 and not more than 10 traffic cones each weighing (60 ± 0.5) g.

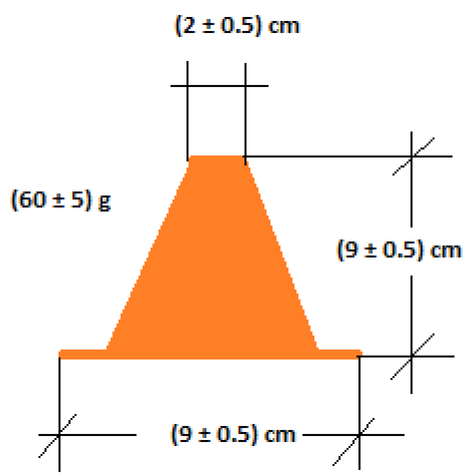


Fig. A-1 Dimensions and weight of cone

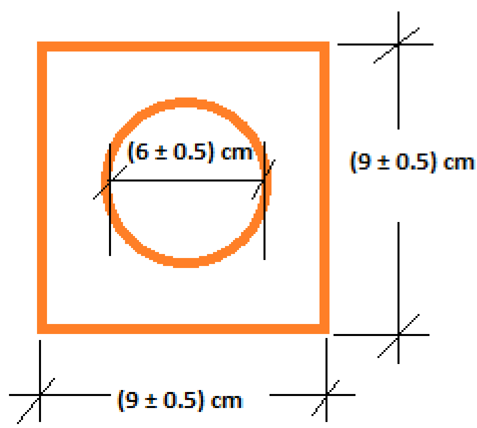


Fig. A-2 Bottom view of cone

Cones are deployed by user instruction and if the machine senses a (5 ± 0.5) cm black square. The black squares represent "holes" and are located on the lane centreline with ± 0.5 cm variance. The dimensions of the designated lane and an unknown location of a black square are shown in **Fig. A-3**.

The problem proposed gives a few requirements for the machine. First, the machine must be able to carry up to 10 cones and deploy them all correctly. Second, the machine must be able to operate with the load of circuit boards, power supply, PIC board, and the additional traffic cones. Third, the machine requires space to mount circuit boards, switches, actuators, sensors, power supply, PIC board, and a place for the cones to be ready to deploy. For the machine to be able to detect holes, a hole-sensor is required to tell the machine if there are holes in the environment. For the machine to be able to follow the designated lane, sensor(s) are required for the machine to swivel and follow the proposed path.

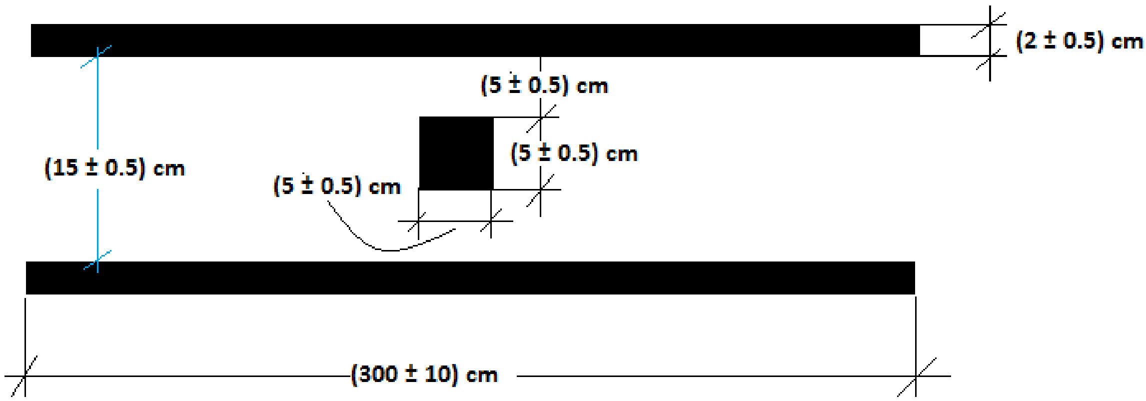


Fig. A-3 Lane dimensions with one black hole. Note that the lane does not have to be necessary straight.

One also has to take into account that the traffic cones must be deployed on top of the hole, thus covering it. This will affect the hole-sensor location, and the required speed of the machine. In order for the machine to deploy efficiently, there must be as little delay in deploying cones. Moreover, the machine itself must be durable and safe for the user to handle. Lastly, one of the major constraints is that the robot has to be built with a $230 CDN budget. The machine also has to weigh under 10 kg and fit within a 50x50x50 cm^3 envelope at all operation times. As a result, a machine that is compact and portable is preferred.

In order to design a successful machine, the team has identified the following key features and mechanism in order to accomplish the proposed task. The following list pinpoints the division of the problem into smaller, more manageable tasks:

1. **Light, cheap, durable and easy-to-work-with material:** The material has to be durable and strong enough to carry a heavy load.

2. **Cone holding:** A structure must be constructed in order to hold the cones that are on standby and ready to be deployed. The cone holding structure must hold at least 10 cones.

3. **Cone deployment mechanism**: A simple mechanism is required to both separate the cones and deploy the cones accurately and reliably.

4. **Hole-sensing mechanism**: The mechanism must be able to both sense holes and not interfere with the already deployed cones.

5. **Last cone detection mechanism**: The machine must somehow know that it is out of cones to deploy and must return back to the start line (refer to RFP for more details).

6. **Lane sensing mechanism**: Sensors must be placed in an optimal location on the machine in order for it to read the lane and swivel accordingly.

7. **Swivelling**: The machine must be able to swivel through the obstacle course and not spend too much time self correcting itself. Wheels must be chosen for optimal performance.

2.2 Solution

The following section will outline the solution to the proposed problem and address each individual electromechanical sub-problem addressed in section 2.1. The problem was attacked by identifying criteria for each constraint as summarize in **Table A-1**.

2.2.1 Choice of Materials

Materials that are relatively light, durable, cheap, and easy-to-work-with are preferred. A light machine will satisfy the constraint of the problem and also will have the ability to be transported easier from place to place. A durable and robust machine is absolutely required both for professionalism and safety purposes. Cheaper materials are preferred as it decreases manufacturing and fabrication cost of the overall machine. Given a fourteen week time period to build the machine, integrate it with other subsystems, and to debug it, one needs materials that are easy to handle and to process in a short amount of time. **Table A-2** summarizes which materials being used for the machine by comparing their advantages and disadvantages, while keeping in mind there is a $230 CDN budget for this prototype.

2.2.2 Cone Storage

The machine is required to hold up to 10 cones for the bare minimum, and can be extended to hold up to 20 cones. The cone storage system must be generic and should not be limited by any amount of cones. For instance, placing cones in a wheel, as shown in **Fig**. **0-1** is not a generic solution and is limited by the number of space on the wheel. In order to take advantage of the cones being already stackable, the cone storage chosen for the solution was a square-based rectangular prism with open ends, as shown in **Fig. A-4**. The open ends provide the user to feed cones into the machine and for the machine to deploy cones out of the other end. The storage itself is four acrylic piece attached by 1/16" flat screws. Acrylic was chosen as the material for the cone storage for aesthetic appeal.

2.2.3 Cone Deployment Mechanism

The cone deployment mechanism chosen for this project was the skewed square design as discussed in Chapter 1. The skewed square was constructed with four pieces of aluminum attached together by pivot screws on each corner, as shown in **Fig. A-5**. The pivot screws are used for the skewed square to transform from the skewed square state to the perfect square state easily, and vice versa. Aluminum was used, rather than plywood like the rest of the machine, because it provides a lower frictional property that allows the cones to be dropped without altering their orientation. This can be analogous as pulling a table cloth away from a table really fast and leaving everything that was on the table before unchanged.

The mechanism is attached to the bottom of the mobile base, as seen in **Fig. A-6,** by two 3/16" flat screws and are bolted on the other side of the mobile base. The attachment points are fixed points during any state the skewed square is in. In order for the square to transform from one state to the other, springs were used to keep the square in the skewed (neutral) position. In this state, the cones cannot drop pass the skewed-square due to simple geometry. A simple push is required to drive the skewed square mechanism and transform it into the perfect square state. The perfect square state will allow cones to drop through until the square returns to its original neutral state.

This solution by itself is not sufficient because more than one cone will be able to drop through in the perfect square state. In order to drop one cone at a time, it was required for the cones to be separated. **Fig. A-7** summarizes how the cone separation works. In **Fig. A-7 a),** the square is in its neutral state, viewing from below the mobile base. The yellow coloured triangles symbolize a thin piece of plywood, flaps, and are hidden from anyone viewing the mobile base from above. In the neutral position the skewed square will hold all the cones in the machine. When the square changes state into the perfect square state, the flaps appear, blocking the second cone from being deployed, as shown in **Fig A-7 b)**, and deploys the first cone. When the square changes back into its neutral state, the flaps will disappear and the 'second' cone will become the 'first' cone. This process is repeated for every deployment.

The skewed square mechanism is driven by one 12V DC gear-head motor with a piece of wood attached onto the driver. Every time the motor turns on, the motor will rotate the wood and the wood will provide the pushing force to push the skewed square from its neutral state to its perfect square state for an instance. After the instance it is a perfect square, the motor continues driving until the wood loses contact with the skewed square and the springing mechanism will retract forcing the skewed square back into its neutral position. As a result, one cone is dropped for each full revolution of the cone-deployment motor. This motor is attached to the side of the mobile base, utilising the plywood's thickness, by glue, as shown in **Fig A-8**.

2.2.4 Hole Sensor Mechanism

The machine was in need of a sensor for sensing black squares or 'holes'. The sensor should not interfere with the already deployed cones. The mobile base of the machine is said to be elevated 12 cm high (refer to section 2.2.7 for overall structure). Preferably, the hole sensor should be in the centreline of the lane for optimal sensing.

The hole sensor mechanism of the machine consists of having an arm attached to a 12V DC gear-head motor. The sensor is located at the end of the arm, as shown in **Fig A-9**. There are two states for this mechanism. The first state is the deployment phase. During this phase, the machine is running on the lane forward and the hole sensor mechanism is facing downwards, so that the sensors are facing perpendicular to the ground. The arm of the mechanism is long enough, 9 cm, for the sensor to differentiate between black and white on the floor. The second phase is when the robot is finished the lane and needs to return back to the user. The hole sensor is then raised by the motor close to 180 degrees. This allows the sensor not to interfere with the deployed cones when coming back to the user. The machine will return to the user backwards, following the lane again.

The arm of the hole sensor mechanism is plywood and the sensor was mounted on by glue. The attachment of the motor to the plywood is a simple drilled hole glued to the motor driver. The motor is attached to the centre of the edge of the mobile base, as shown in **Fig A-9 b)**.

2.2.5 Last Cone Detection

The RFP stated that it wants the machine to know when it is out of cones. Typically, a sensor for sensing if there are any cones left will suffice. However, this requires extra circuitry and also a drain in battery life for the machine.

The last cone detection for this machine uses the already available materials on the machine. By creating a substitute cone, one wrapped around aluminum foil and connected to the circuitry of the machine by a wire, it will act as the "out of cones" signal. The wire is in contact with the aluminum foil and extends to the circuitry of the machine. Another wire is attached from the skewed square to the circuitry of the machine. When the last cone is deployed, the substitute cone will hit the aluminum skewed square, creating a complete circuit and signalling the PIC that the machine is out of cones. This solution is discussed in full detail in Chapter 3.

This solution justify our use for a aluminum deployment mechanism as it is conductive and will allow the brain of the machine to know when it is out of cones.

2.2.6 Lane Sensing

Lane sensors are for the machine to be able to swivel properly and adjust to any lane presented. Mounting the sensors in the correct location is crucial and directly affects the performance of the machine. One does not want the machine to over swivel, causing constant swivelling throughout the course. The sensors must be a particular distance apart so that it is just right and the machine will swivel less throughout the course. This improves the performance of the machine.

The solution uses two sensors for sensing the black lane. For optimal performance, the two sensors should be about 2.5 cm apart from each other. The sensors are mounted on a small circuit board which is attached to a 2 cm wide plywood as shown in **Fig. A-10 a)**. The plywood is then attached to the bottom of the driving motor for the wheels, as shown in **Fig A-10 b)**. Since the mobile base is 20 cm wide, this is the optimal location for the sensors to be mounted as the machine will be centred on the lane. In addition, the sensors will be at optimal viewing distance, 3 – 7 mm from the ground. Attachments between circuit boards, plywood, and to motor are done with glue.

2.2.7 Overall Structure

The overall structure of the machine is shaped like a table. Each wheel of the machine is attached to each 10 cm long, 2 cm wide leg. The free wheels are glued on the back legs of the machine, as shown in **Fig A-11 a)**. The driving wheels are attached to the gear-head motor and the motors are attached to the front legs of the machine by a U-shape mount, as shown in **Fig A-11 b)**. Two motors are required to drive the machine as it gives the machine the freedom to move forwards, backwards, and the ability to swivel (one wheel moves forward, the other wheel moves backwards). The table structure was chosen because it gives maximum stability and has a wide surface area for circuits to be mounted on. The mobile base has a 9.5 cm x 9.5 cm square hole centred on the base, as shown in **Fig. A-12**. The square hole is for cones to fit through and deploy. The cone storage is attached directly on top of the 9.5 cm 9.5 cm square hole by 1/16” screws. The skewed square is attached on the bottom of the machine as shown in **Fig A-9**. The skewed square is mounted onto the machine by 3/16” screws, washer and bolts. Springs are attached to two opposite corners of the skewed square and are mounted to the mobile base by 3/16” wood screws. The motor driving the skewed square is mounted on the sides of the table surface, as shown in **Fig A-8**, and the motor driving the hole sensor is mounted on the front edge, in the centre, of the mobile base, as shown in **Fig. A-9**. Moreover, the lane sensors are mounted as described in section 2.2.6.

2.3 Supporting Calculations & Experiments

The following sections show some of the few crucial experiments that guided the design decision and some supporting calculations.

2.3.1 Key Experiments

Experiment I: Cone Dropping Testing Height
The experiment's objective was to test for the maximum safe height to drop cones without the cones tipping. Since the cones have a height of 9 cm, the experiment seeks a distance greater than 9 cm so that the machine can drop the cones properly. The experiment was conducted by dropping cones from different heights and finding the optimal height for the cones to be dropped. The experiment concluded that one can drop from a safe height of 12 cm without the cone tipping. This allows the machine to be elevated about 12 cm high for it to be able to deploy cones without them tipping. This also allows some room for error when building the prototype.

Experiment II: Spring Testing
The skewed square mechanism uses a spring with unknown spring constant. The best way to approach this problem is trial and error for the best position of where the spring's ends have to be located to be able to have a functioning skewed square. The experiment was conducted by having spring attached to opposite corners of the skewed square and having the unused spring ends attached to the mobile base in different distances from the skewed square. This trial and error process was iterative and an optimal distance was found for the skewed square to function as planned. The distance is depicted in **Fig. A-9**.

Experiment III: Placement of Driving Motor for Skewed Square
The skewed square is driven by a push and an optimal location for the driving motor is necessary. This experiment was also done by trial and error. The experiment was conducted by placing the motor in different location on the side of the mobile base and seeing how the motor hits the skewed square for every revolution. The optimal location was found when cones were able to be deployed through the mechanism and is shown in **Fig A-9**.

2.3.2 Calculations

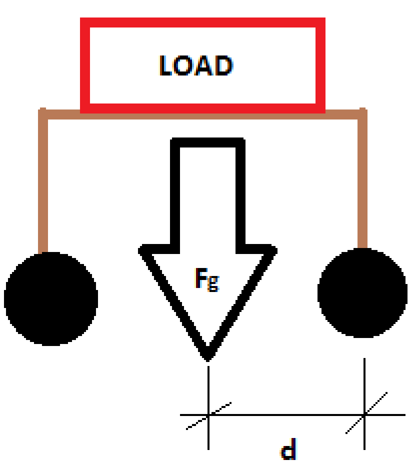


Fig. A-14 Torque calculation

Driving Motor for Mobile Base
The weight of the overall machine may be an issue for the wheels in order to drive it. There is a need to calculate how much torque is needed for the motor to drive the machine.

The machine weighs 2.7 kg in total (including circuit boards, power supply and 10 cones).
The centre of mass of the machine can be assumed to be the centre of the machine, as shown in **Fig. A-14**.

Each wheel needs a torque of:
$\tau = d \times F_g = (0.10\ m)\ (2.7\ kg / 2)\ (9.81\ m/s^2) =$ **2.6 Nm**

Driving Motor for Cone Deployment
Springs have a resistive force on the driving motor for cone deployment. Since the spring constant is unknown, one has to predict the amount of torque needed to drive the skewed-square.
This problem can be solved in reverse order. One can try a motor that drives the skewed square, and measure torque from there.

$\tau_m = K_t\, i_a = (0.5\ \text{Nm/A})(4\ \text{A}) = \mathbf{2\ Nm}$

Driving Motor for Hole Sensor Mechanism
There was a need to calculate the moment on the hole sensor mechanism because one would suspect to have the motor always on when the arm is raised due to its weight. This is a major disadvantage in our solution if the machine was to always have the motor on driving the arm to be constantly raised. If the moment is low enough, there will be no need to drive the arm after it has been raised, which is optimal.

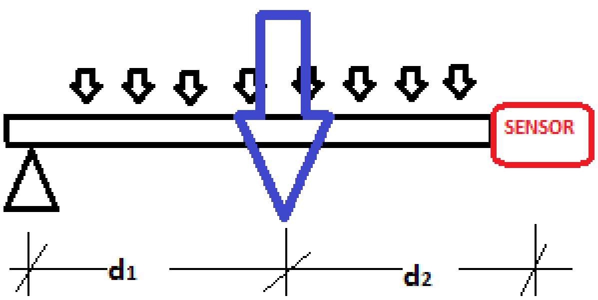


Fig A-15 Moment calculations

$$M = (d_1 \times F_g + d_2 \times F_{sensor})$$
$$= (9.81\ \text{m/s}^2)[(0.05\ \text{m})(0.0575\ \text{kg}) + (0.10\text{m})(0.05\ \text{kg})]$$
$$= \mathbf{0.07\ Nm}$$

=> An insignificant amount of torque, thus no need to have the motor always on.

2.4 Suggestions for Improvement of the Subsystem

The subsystem can be improved in a number of ways. The wheels chosen for the solution was not as effective as predicted because they were flat tires. This dramatically slows down the performance of the machine. A better option would to replace the wheels with rigid tires. This will improve both the speed and swivelling capabilities of the machine. Moreover, the machine was elevated a tad too high. Cones being dropped out of the machine usually bounce and do not completely cover holes or are not completely accurate with the user’s instructions. This problem can be solved by lowering the surface of the mobile base by 1 cm, still leaving enough room for cones to go through its legs. Another improvement that could be made is replacing the free wheels with office chair wheels. This will allow the machine to swivel more easily, instead of fight frictional force during swivelling. However, fighting some frictional force may be a good thing as it forces the machine not to over-swivel. Furthermore, the skewed square mechanism can be improved on by having thinner aluminum. There are cases where that aluminum is too thick and catches the cone, thus not deploying any cone at all.

Chapter 3:
SUBSYSTEM: CIRCUIT DESIGN

3.0 Introduction

The circuit portion of the project is of the utmost importance. It is the part with the most potential error and the highest potential for failure. Thus, it is of utmost importance to carry out the design of such circuits in an organized and knowledgeable manner in order to ensure that the machine will function properly. The circuit portion also requires intensive coordination with the programming and electromechanical sections to ensure that the logic matches the circuitry and that the circuitry fits on the machine.

3.1 Assessment of the Problem

In order to comply with the constraints of the competition, the circuitry would have to incorporate several components such as sensing and actuator driving. There would have to be drivers for four motors in total, two of which required H-bridges. It would also need to provide sensors with the ability to sense black and white. On top of these basic requirements, it would be needed to provide enough current via power supplies to all these mechanism and equip them with enough protection such that no circuit would burn or be destroyed.

The electronic component of the project dealt with the following challenges:

1) Direction & Speed control for the driving motors

- There was a choice between using an H-Bridge IC or manually building an H-Bridge with power transistors. Manually building an H-bridge would consume more time and take up more space. It would only be wise if not too many H-bridges would be needed. Using an IC would be more convenient, but would cost more and be less flexible since we would not have been able to control the maximum allowed current.
- There was also a choice of the amount of current to be used for driving the wheels. A large current would have meant a faster and stronger machine; however, it would also dissipate a lot of heat in the form of power and thus large ceramic resistors would have needed to be used.

2) Motor control for deployment and raising of sensor arm

- Initially, it needed to be decided whether or not we would need bidirectional control. There was a choice between using an H-bridge IC or simple transistor circuit. Using an H-bridge would have allowed bidirectional and speed control, but would be more difficult to build. It would also have limited the amount of current that we wanted to use due to the constraints of the IC itself. Using a transistor circuit would have sacrificed the bidirectional control, but would have increased the amount of current we could use to drive the motors. This would also require one less signal.
- It was also decided through thorough testing that a high current such around 4A would be needed to successfully operate the deployment mechanism. This thus would have required a more expensive H-bridge IC such as the LM339 which can take up to 10A of current. The alternative was to use a simple transistor circuit with power transistors, but would also sacrifice bidirectional control.

3) Sensors for detecting holes and swivelling

- With the various types and models of sensors around, we needed to decide which type would best suit our problem for distinguishing between black and white. The two most relevant sensors would have been colour sensors and IR sensors.
- Also, the distance between the sensors and the ground needed to be determined so that we could buy sensors with an adequate range capability.

4) Emergency Stop Button

- The emergency stop button would have to break the circuit to stop all mechanical parts such as the motors. The question revolved around whether the button would break the circuit and the positive end of the circuits, where the power supply connects or at the grounds of the circuits.
- There was also a choice to be made regarding the type of button that would be used as an emergency stop. With a variety of types of switches and buttons, it was narrowed down to a choice between a rocker switch or a button.

5) Signal for out of cones

- Detecting when the machine was out of cones was a difficult problem with various solutions. The most intuitive would have been to use a sensor to detect whether there are still cones inside the machine. This would however require another sensor and more circuitry.

6) Signal Interference

- There was a likely chance that some form of signal interference, especially from the electromagnetic radiation of the motors, would interfere with our circuitry and signals. Hence, a method of shielding would have been necessary in order to protect the quality of the signals in the circuit.
- There was also a possibility of interference between various electrical components. These would have had to been investigated as the testing progressed.
- The motors themselves were susceptible to a variety of signal interference such as voltage spikes. Hence, it would have been necessary to shield them with capacitors or zener diodes in order to prevent these spikes.

7) Grounding setup

- A great challenge was also deciding how to ground the different circuits. In order to function as a whole, all the grounds of the individual circuits would have needed to been tied down to one common ground in order to provide the correct reference voltages to all the circuits.

8) Power Supplies

- There was also a choice regarding the voltages of batteries to be used. A larger voltage would be needed to drive the wheels, while other devices such as the sensors could have used smaller voltages.
- There was also a choice regarding the type of batteries to be used. Regular carbon zinc batteries are cheap, but have a short battery life. Lead acid batteries are cheap and long-lasting, but are also heavy and potentially toxic. Alkaline batteries are long-lasting and small, but are quite expensive.

9) Circuitry Design

- There was also a decision to be made regarding the philosophy of circuit design. Did all the circuits need to be compact and small, or could the components be spread out and less dense?

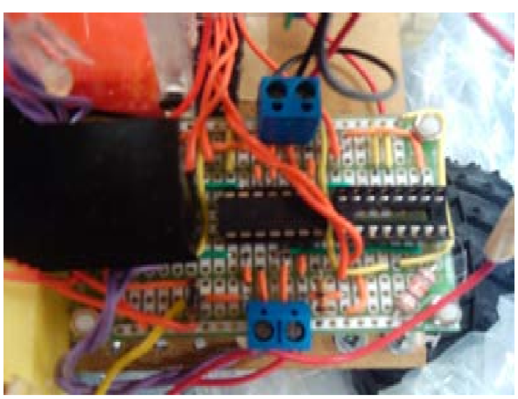
3.2 Solution

This section will describe the circuit design and how it contributes to the overall performance of the machine

3.2.1 Direction and speed control

It was decided that the best way to provide direction and speed control would be to use an H-bridge. Initially, H bridges were manually built with power transistors and diodes; however, due to the number of H bridges needed, the SN754410 H-bridge driver was used instead to save space and cost. See **Fig B-1, B-2** for the circuit diagram.

a)



b)

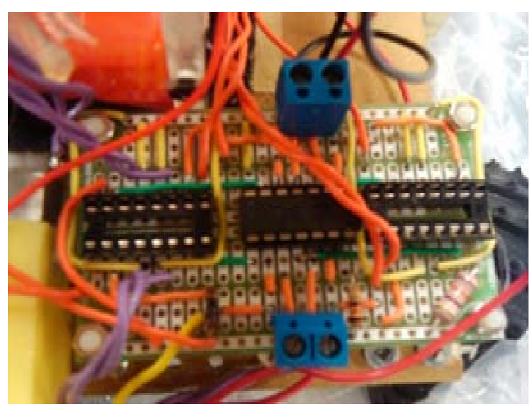


Fig B-2 H-bridge circuit **a)** with H-bridge **b)** without H-bridge

3.2.2 Motor Control for Cone Deployment and Arm Raising

It was decided that the 4A current from the power supply was sufficient to give the deployment necessary speed and torque. Due to power and cost issues, it was decided that the current we would use to drive the motor would be 3A. Though not as strong as using a 4A current, the 3A current would be sufficient to operate the deployment mechanism. See **Fig B-3, B-4** for the circuit diagram.

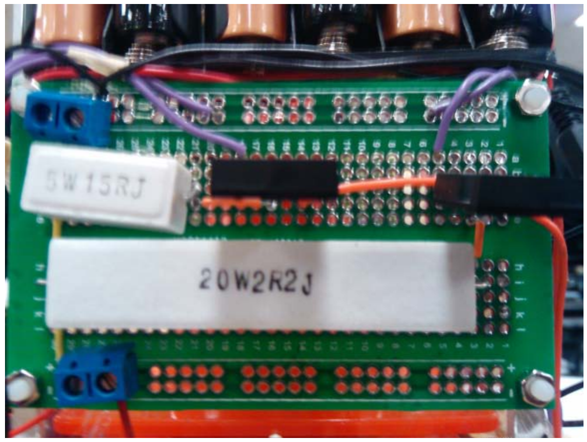


Fig B-4 Transistor circuit for motor control

3.2.3 Sensors

Since black and white were the only colours being sensed for, it was decided to use IR reflector sensors as opposed to other types of colour sensors. There was also a choice to be made regarding the type of sensor based on their ability to sense certain distances. With sensor ranges varying from 3 mm to 20 cm, it was chosen to use sensors of the 3-6mm range. In the end, the model chosen was the TCRT5000 due to its low price and compactness. See **Fig B-5, B-6** for the circuit diagram.

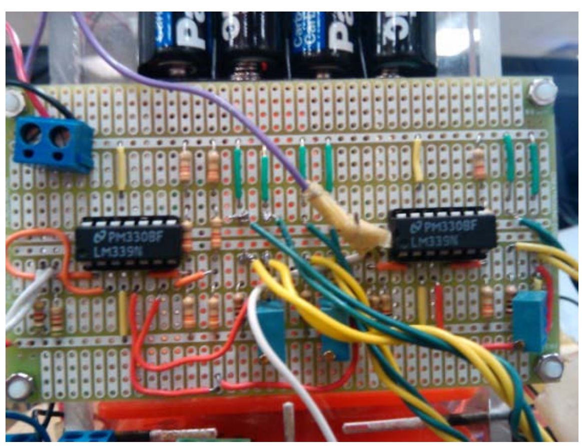


Fig B-6 Sensor circuit

3.2.4 Emergency Stop Button

A red rocker LED switch was decided to be used as the emergency stop button due for convenience sake. The rocker switch also served as an on button for the power supplies, thus serving a dual purpose. There was also the choice to make the rocker switch light up, thus indicating it as an emergency button switch. It was decided that the switch would break the circuit right at the ground of the power supplies. This would make it simple to turn off all circuits with two connections to one simple switch. The alternative was to break the circuit at the positive terminals of the power supplies; however, this would require two connections per power supply, thus requiring a switch with several terminals. We had three separate power supplies and hence a switch with 6 leads would have been needed. The first option was chosen due to convenience. See **Fig D-4** for location of emergency stop button.

3.2.5 Signal out of cones

A variety of options were available for sensing when the machine was out of cones. The most commonly heard option used was to utilize sensors to detect if cones were still present inside. The method chosen for our machine was quite unique and innovative. Taking advantage of the aluminum square in our machine, it was decided that there would be a cone coated with aluminum that would lay on top of the stack of cones to be deployed. When all the cones have been deployed, the aluminum cone would hit the aluminum square, thus completing a circuit and sending a circuit to the PIC, indicating that no more cones were left in the machine.

3.2.6 Signal Interference

The main concern in regards to signal interference was electromagnetic radiation from the various DC motors. Hence, we had to ensure that there was significant distance between the circuits and the motors. In some cases, a piece of aluminum sandwiched between two slices of paper was placed as a barrier to test if there was any improvement in quality of signals. There was also the concern of extraneous infrared radiation from sources such as sunlight affecting the readings of the sensors. Hence, the sensors had to be calibrated each time in different environments.

3.2.7 Grounding

In order to tie all the grounds of the circuits together, it was decided to use a grounding bus format, whereby the grounds of all circuits were first tied together and then to the ground of the power supplies. This allowed for the breaking of the power supply with a switch. See **Fig B-7** for the circuit diagram.

3.2.8 Power Supplies

In order to provide for the various circuits, an abundant power supply was needed. Initially, a 12V lead acid battery was chosen to supply the driving motors because of its long lifetime; however, it was soon switched to a 12V carbon zinc supply due to its light weight. A 6V carbon zinc supply was chosen to power the deployment motor, arm raising motor, and the NAND logic cage. A separate 6V supply was chosen to power the sensors. Near the end of the project, the carbon zinc batteries were replaced with Duracell alkaline batteries to extend battery life.

a) **b)**

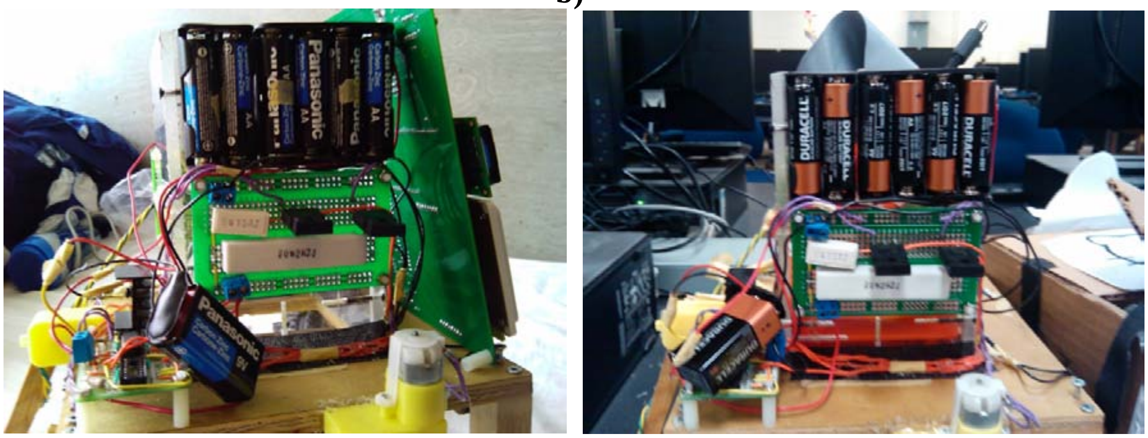


Fig B-8 Power supplies **a)** Panasonic **b)** Duracell

3.2.9 Circuit Design

In order to fit the necessary circuits on our small machine, it was decided to build very compact circuits. All the sensors were to fit on one circuit board. The H-bridge circuit was to fit on one circuit board. The deployment and arm raising circuits were to fit on one circuit board. Finally, the I/O bus was to fit on one circuit board. See **Fig B-9**, **B-10** for the circuit diagram.

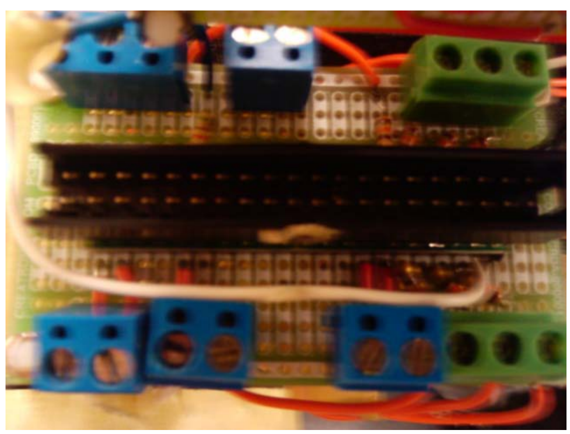


Fig B-10 IO hub circuit

3.3 Supporting Calculations

This section will show all works done for circuit calculations

SENSORS

$$P_{transistor} = \frac{V^2}{R} = \frac{36}{47000} = 0.0008\ W$$
$$P_{diode} = \frac{V^2}{R} = \frac{36}{100} = 0.36\ W$$
$$P_{side} = \frac{V^2}{R} = \frac{36}{9900} = 0.004\ W$$
$$P_{out} = \frac{V^2}{R} = \frac{36}{5100} = 0.007\ W$$
$$I_{transistor} = \frac{V}{R} = \frac{6}{47000} = 0.1\ mA$$
$$I_{diode} = \frac{V}{R} = \frac{6}{100} = 60\ mA$$
$$I_{side} = \frac{V}{R} = \frac{6}{9900} = 0.6\ mA$$
$$I_{out} = \frac{V}{R} = \frac{6}{5100} = 1.2\ mA$$

I/O HUB

$$P_{inputs} = I^2R = (0.012)^2(130) = 0.019$$
$$P_{CDT} = \frac{V^2}{R} = \frac{36}{5100} = 0.007\ W$$
$$I_{CDT} = \frac{V}{R} = \frac{6}{5100} = 1\ mA$$

H-BRIDGE

$$P_{supply} = \frac{V^2}{R} = \frac{144}{20} = 9.6\ W$$
$$P_{NAND} = \frac{V^2}{R} = \frac{36}{130} = 0.28\ W$$
$$I_{supply} = \frac{V}{R} = \frac{12}{20} = 0.8\ A$$
$$I_{NAND} = \frac{V}{R} = \frac{6}{130} = 46\ mA$$

TRANSISTORS

$$P_{DPL} = \frac{V^2}{R} = \frac{36}{2} = 18\ W$$
$$P_{RAISE} = \frac{V^2}{R} = \frac{36}{15} = 2.4\ W$$
$$I_{DPL} = \frac{V}{R} = \frac{6}{2} = 3\ A$$
$$I_{RAISE} = \frac{V}{R} = \frac{6}{15} = 0.4\ \text{A}$$

3.4 Suggested Improvements

Some of the following can be improved in the circuits:

1) Due to the lack of signal protection and filtering, there were likely some inconsistent signals and signal interference. Luckily, these did not affect the circuits too badly and there was no problem with the functionality of our signals. However, it was noticed that in some occasions, the PIC microcontroller would freeze and all the signals would turn on. It may have been possible that this was due to uncontrolled signals that were entering the PIC. Hence, it may have been worthwhile to implement signal protection.

2) The batteries were used directly as power supplies. It would have been ideal to incorporate a voltage regulator of some sort in order to maintain the voltage at a certain level. This may have reduced the voltage spikes that were being experienced by the machine.

Chapter 4:

SUBSYSTEM: PROCESS AND CONTROL

4.0 Introduction

According to the RFP, the robot has to be fully autonomous. The programming part makes sure this can be accomplished. Its main functionality is to figure out the logic so that the robot knows what to do and how to respond to the environment.

4.1 Assessment of the Problem

As specified in the proposal, a fully autonomous robot was required to complete the traffic cone deployment operation. The robot required a processing unit to control the entire operation by communicating with the circuits and the mechanical components via the input and output ports. The processing unit was required to do the following:

1. Execute instructions to set up the robot.
2. Include an easily understandable interface to communicate with the user.
   a. Ask for user inputs at the beginning of the program.
   b. Update status and location of robot as well as other information during the program.
   c. Display summary at the end of the program.
3. Constantly read inputs and send outputs during the operation to respond to the environment and carry out operation specified by the proposal.

4.2 Solution

Process and control of the robot were accomplished by programming a microcontroller with I/O ports that could communicate with the circuits. A microcontroller was chosen over a microprocessor because the robot did not require as much speed and memory as a computer, thus minimizing the cost of the chip to under $10. The microcontroller chosen for this project was a Peripheral Interface Controller (PIC) chip manufactured by Microchip Technology; specifically, the part number is PIC16F877, which is one of the recommended ones for this project.

4.2.1 Software

The microcontroller (which is interchangeably used with PIC) was in charge of all processing and control of the robot. It was constantly executing instructions to communicate with the environment via the circuits. The program was written in assembly language because it saved memory space and it was easier to deal with binary inputs and outputs. Below is a summary of the program logic and procedure. See **Fig C-1** in the appendix.

There were several features in the program that made it more compact. Constants were used at the very beginning for two reasons. Firstly, it made debugging and system integration easier since only one number (e.g. Num_Cycles_U_Turn) needed to be changed to affect the program. Also, it made the program more readable as Num_Cycles_U_Turn tells us that it is the number of cycles required to make a complete U-turn. Macros were also used for this program; for instance, the display macro, allows instruction such as “Display Welcome_Msg1” instead of writing character by character into the LCD screen. Polling was used for this program whenever an input was needed from the user or the circuits to the microcontroller because it was necessary to wait for an input signal for the microcontroller to decide what to do next. Because the robot was never in standby mode for a long time or waiting for an input for a long time, it was not necessary to use interrupt over polling, both of which would do the job.

4.2.2 Hardware

The hardware used for process and control was facilitated by using a PIC DevBugger board that included various features for the operation. Here are the ones that were used. Refer **Fig. C-2**.

1. HD 44780 LCD screen (display)
2. 4 x 4 Keypad (user input)
3. 40-pin bus (I/O signals)
4. Power supply module (connection to power source)
5. 10 MHz Oscillating clock (PWM)

There were other features that are really useful as well. Most of debugging was done using the PIC DevBugger board that included switches and LED lights to simulate inputs and outputs. Other features, such as Real Time Clock, A2D (analog to digital converter), and RS232 (PC interface) were also very useful but were not incorporated in the program.

4.3 Computer Program

The program was written in the order of the operation of the robot. It is as follows: setting up the robot, asking for input, carrying out the operation by moving the robot and deploying cones, returning and displaying the summary. See appendix G.

4.3.1 Programming Initialization

There were several common practices used when coding the PIC at the beginning of the program for initialization and clarity of code. At the start of the program, there is a comment that summarizes what the program does (refer to SUMMARY section of code). It is then followed by the hierarchy (HIERARCHY), or structure of the program, which is similar to a table of content to allow the reader to understand the structure. The four lines of assembler directives (ASSEMBLER DIRECTIVES) follow the hierarchy. They specify which chip is being used and the settings required to program the chip. After that the convention is to put constants or the “equates section” where numbers that won’t get changed (CONSTANTS) can be defined and assigned. The numbers that will get changed (VARIABLES), are right after constants. Next are the vectors that assign the addresses of start of program and interrupt (VECTORS); tables, macros, and helper functions come after those (TABLES, MACROS, and HELPER FUNCTIONS). After the set up, the main program (MAIN CODE) can finally begin (usually starting with an initialization section). Below is the section up to the main program for this robot.

4.3.2 Settings for Operation of Robot

The main program will start by displaying welcome messages and asking for inputs. It will then display the inputs and poll for the start button to start the cone deployment operation (refer to Input_Location subroutine of code).

4.3.3 Cone Deployment Operation

The length of the lane was 300 cm and the robot needed to know when it reached the end. This was done by calibrating the speed of motor controlling the wheels. This turned out to be fairly consistent because the robot did not need to stop when detecting a hole, deploying cones, or swivelling. Upon reaching the end of lane, the machine needed to make a U-turn and return. The time it took for each run was also fairly constant, making the operation time easy to calculate, even without a real time clock.

During the operation, there were several things that needed to be checked continuously; thus it made sense to write a big loop so that after checking everything once, the microcontroller could do it again to achieve effective communication with circuits and mechanical part of the robot (LOOP subroutine). The program was coded in order of priority so that more important parts would run first for the robot operation.

4.3.3.1 Emergency Stop

Emergency stop: if the machine needed to be mechanically shut down immediately (for instance, safety concern), it would be done so without any other delay. There was a switch that turned all circuits off, prompting a signal to be sent to the PIC to display a “Emergency Stop” message.

4.3.3.2 Swivelling

Swivelling was a technique used to adjust the direction of the robot by adjusting the speed and direction of each wheel powered by a motor. There were two sensors mounted on the front right wheel of the 4-wheel machine. One was inside the lane (sensing white) and one was outside the lane (sensing white) while the boundary of the lane was black. If the left one sensed black, the machine needs to turn left. Hence the voltage of the left motor was reduced to 50% using PWM. If the right one sensed black, the machine needs to turn right. Hence the voltage of the right motor was reduced to 50% using PWM.

4.3.3.3 Updating location and display location

When the machine was travelling in the lane, it updated the location every centimetre to tell the user where exactly it was at. It also updated the number of cones deployed thus far as well as the number of holes detected. Had every updated location been displayed (every centimetre), the LCD screen would refresh so fast that it would be really hard to read. Thus the LCD display was only updated every 10 centimetres or when a cone was deployed.

4.3.3.4 Checking the motor for deploying cones

Check if the deployment was on (which would be turned on in section 4.3.3.6 or 4.3.3.7 of previous cycles). If it was on, check if it was time to turn it off and loop again because when the motor was on, the robot can’t deploy another cone. If it was off, then proceed to the next step and check if a cone should be deployed.

4.3.3.5 Checking the end of lane

This was used to check if robot reached the end of the lane. If the cumulative location reached 300 cm (length of lane), the machine would display message and make a U-turn.

4.3.3.6 Detecting hole

To check if there was a hole:

1. If there is no hole detected, then skip.
2. If a hole is already detected, then skip.
3. If a hole is newly detected, do the following:
   a. Turn the motor and set a timer to turn the motor off afterwards.
   b. Increment the number of cones deployed and the number of holes detected.
   c. Record the location in both hole detected and cone deployed.
   d. The distance between subsequent cones is set to zero since a cone has just been deployed.
   e. If the machine is out of cones, display a message and make a U-turn.

4.3.3.7 Deploying a cone at regular interval of distance

To check if the distance to deploy a cone was reached:

1. If the input of first cone has not been deployed, check if the distance of first cone is reached. If no, then continue.
   a. Turn the motor and set a timer to turn the motor off afterwards.
   b. Increment the number of cones deployed.
   c. Record the location of cone deployed.
   d. The distance between subsequent cones is set to zero since a cone has just been deployed.
   e. If the machine is out of cones, display a message and make a U-turn.
2. If the input of first cone has been deployed, check if the distance between subsequent cones is reached. If not, then continue. If yes, do the 5 steps (a to e) above.

4.3.4 Returning to Start Line

Once the machine ran out of cones or reached the end of the lane. It would make a U-turn and return to the start line (U_Turn and Returning subroutine). As mentioned in 4.3.3, the speed of the motors was fairly consistent so the machine was able to accurately return to start line in a short time with minimum calibration.

4.4 Suggestions for Improvement

The program for this project was efficient, functional, and optimized. However, there were still areas for improvement. The DevBugger facilitates addition features such as real time clock and PC interface that made the overall program better. One problem was that the I/O pins were not optimally grouped together due to the number of pins available and small circuit board. Using the PIC18 would solve the problem and would offer more additional features that could have been incorporated into the project.

Chapter 5:
INTEGRATION

5.0 Introduction

This section will discuss the integration and debugging process. It will highlight key problems during the integration phase and the team's corresponding solutions. Moreover, suggested improvements of the overall machine will be discussed. Finally, this section will summarize the physical properties and key features of the machine.

5.1 Integration

Phase I: Mutual Component Testing

The IR sensors were tested with the PIC to see if signals could be sent to the PIC. The motor was tested with the power supply to see if cones could be deployed.

Phase II: Mutual Subsystem Testing

The three subsystems were combined to test if they could all communicate. The first test was to see if the PIC could make the motors run and then reverse via the H-bridge after 100 seconds. This initially did not work, which was due to a signal not being connected to power. This lack of circuitry understanding led to a long delay, but was solved and the test succeeded.

The second test was to see if the sensors would send a signal to the PIC, which would then send a signal to the deployment motor to turn on. This worked very well on the first try except there were some spikes where the signal went on without black.

The next step was to determine whether the PIC would drive the deployment motor with the H-bridge. After attaching the necessary connections, the procedure failed and the motor did not function. Also, the resistor to the H-bridge burned, also causing the H-bridge to burn as well. It was then discovered that power considerations had not been taken into account, thus prompting the resistors to be replaced by resistors of higher rating.

Phase III: Unity

The next step was to test the full functionality of the machine by testing all the components to see if they worked in unison. The grounds of the separate circuit boards were tied together and the full cycle of the machine was tested. The machine ran and detected holes properly. Swivelling was also tested and after some programming adjustments, the wheels adjusted properly to accommodate swivelling of the machine.

Phase IV: Mounting

This step consisted of mounting all the circuits onto the machine itself. This task in itself was not an easy one, as many wires needed to be re-soldered and extended to accommodate the distance between circuit boards. Tape and glue gun were used to attach the circuits themselves to the machine. The grounding of the circuit was also rewired to allow the installation of the emergency stop button. After all the mounting was complete, the circuits were then tested again and failed to function properly. At first, it was thought that it was a grounding problem due to the rewiring that had occurred; however, it was discovered afterwards that it was merely a soldering issue in which a pin for the H-bridge had become loose. After soldering it back on, the machine ran successfully.

Phase V: Debugging

This step was the most critical and difficult step, which was to debug the machine and make it perform perfectly.

The first step was to see if the machine would actually run on land and return after 300 seconds. This went well. The next step was to test whether the machine would swivel with one wheel spinning forward and the other backwards. This somewhat worked; however, the machine kept getting stuck every time it tried to swivel due to the high friction between the rubber wheels and the ground. Consequently, a programming adjustment was made to make one wheel spin slower (25% PWM) and the other wheel full speed (100% PWM). This made the swivelling work and though the machine still got stuck on some occasions, it was speculated that with fresh batteries, it would work fine. The machine was then placed on a lane and tested to see if it could travel to the end without veering off the path. This test was successful.

The next step was to test the cone deployment with the movement. This worked well on the first try; however, the mechanism itself was not consistent and constant adjustments were made to the skewed square. The most major addition was lines of rubber strips on the sides to slow down the cones so that only one cone would deploy instead of two or three. After adding this, there were no more cases of multiple cones dropping at once; however, a new problem arose. The cones kept getting stuck because of the high friction rubber so that on some occasions, no cones deployed at all. Constant adjustments to the rubber were then made with the goal of producing enough friction so that only one cone drops at a time while preventing the cones from getting stuck. This was a constant and tedious struggle that never truly succeeded. This problem was thus temporarily put aside to concentrate on the other problems.

The next step was to debug the arm raising sensor. It was discovered that a pin had become loose and needed to be resoldered. After resoldering, the arm rose and worked perfectly.

At around this point, a major and consistently frustrating problem arose. At very random times, particularly when the machine would try to deploy a cone, the whole PIC board would freeze, causing all the signals to turn on and the machine would cease to function. It is stressed that this problem occurred at very random times, thus making it very difficult to debug. This problem never occurred when testing the program itself with the debugging module on the PIC DevBugger board; thus the problem fell into the pins connecting to the circuit or the circuit itself. At first, it was discovered that by covering the right sensor with black and thus forcing the signal on all the time, the problem would disappear. After analyzing the circuit for a long period of time and performing some resoldering operations, the problem still could not be solved. It was then decided that the machine could function without the right sensor, using only the left sensor to swivel and travel straight. This worked for quite awhile; however, eventually the problem returned. It was then decided that turning off the reversing of the machine might solve the problem. After turning off the reversing capabilities, the problem disappeared again for some time so the right sensor was put back into operation. Another possibility was brought up when debugging with the DevBugger module. The pins chosen might have already been in use in other parts of the program (e.g. keypad) thus receiving unwanted and unexpected signals. That was not a strong argument but it was eventually tested by changing I/O pins. The problem still occurred sometimes and was never completely resolved.

The next step was to test the backwards swivelling of the machine to see if it could return to the start line, while travelling in a straight line so as to avoid bumping into cones.

One problem arose, when the machine tried to swivel to adjust its path: the machine: it went off the lane in a way that looked like over-swivelling, causing it to make a really wide turn and veering off the track. It was then discovered that this was because while travelling backwards, the sensors were in the back and thus by the time the sensors detected the black lane, the machine was already too far off the track. It was then decided that the machine would use a 180 degree turn to return to the start line. This also removed the risk of bumping into any cones, the biggest penalty in the competition.

The next step was to implement the out-of-cone detection system. A stranded wire was attached to the dummy cone and coated with aluminum foil to provide more surface area. It was then decided to attach the wire to a screw that was in contact with the skewed square rather than the skewed square itself. The system was then tested and the signal worked, causing the out of cones signal to come on. It was soon noticed, however, that there was a lot of signal interference and the signal would turn on randomly, causing the machine to return. It was then decided that a programming adjustment would be made to make the machine return to the start line only if it sensed the out-of-cone signal for more than 5 seconds so that it would ignore other interference signals. This worked for awhile; however, the out-of-cone signal was turned off at the end due to the problem of excess interference signals that persistently caused the machine to return due to the out-of-cones signal.

At this point, the most important issue had yet to be dealt with: cone deployment. An innovative idea was implemented in which the skewed square would have flaps that would come out during deployment to trap the cones above so as to prevent more than one cone from falling through. This was first attempted with cardboard flaps and once it worked, it was switched to wooden flaps. This solved a large portion of our deployment problem by making it deploy consistently. On occasion, however, a cone would still get stuck. This time, however, it would not get stuck vertically, it would get stuck horizontally due to the cone being out of place.

The last step was to calibrate the machine so that it would travel 300 cm and U-turn perfectly 180 degrees. This took quite a few cycles and many problems arose, particularly with the U-turn. The wheels kept getting stuck and delaying the U-turn. Eventually, the machine travelled around 300 cm and the U-turn, though still slanted, was sufficient for the machine to return to the start line.

5.2 System Improvement Suggestions

1) The wheels of the machine had a very high friction factor, which limited its ability to perform certain functions such as swivelling and U-turning. This was the factor that prevented our machine from completing the U-turn and returning to the start line. This also severely slowed down the machine's performance. The wheels themselves were also empty and thus acted as a flat tire, further slowing the machine. If possible, it would have been ideal to find wheels that were firm and sturdy.
2) The circuit design had various imperfections in regards to neatness and organization. Due to the lack of time, there was no time to clean up and perfect the circuit boards. Consequently, there was a lot of excess solder, extra parts, and messily soldered parts on the circuit board. It would have been ideal to have removed the unused parts and to have cleaned up the solder. This messiness factor could have played a big part in the signal interference problems that were occurring.

3) The PIC board, though useful, had many extraneous and unnecessary parts. If more time were available, it would have been a good idea to build a PIC board manually. This would have reduced the size of the board itself and could have also reduced the signal interference problems we were having due to the ribbon cable and ports.
4) The cone holder tunnel caused many problems due to its high friction. The holder was designed to match the dimensions of the cones exactly with very little room for error, hence causing some cones to get stuck. Though grease was applied to decrease this friction, cones would still get stuck once in awhile. It would have been better had the cone holder been made with at least 3-5 mm of error so that cones would not get stuck within the cone holder.
5) The main problem with the deployment mechanism after the flaps were added was that the cones kept shifting out of place. It would have been better had there been some kind of mechanism added in order to retain the positions of the cones such that they would drop through without getting stuck. While the cone deployment was already functional, a mechanism such as this would have made it more consistent.
6) Using plywood as the material for the machine greatly reduced the cost of the machine; however, it also greatly decreased the aesthetic appeal of the machine. Had the budget been higher and more reasonable, it would have been ideal to have the machine made of plexiglass or aluminum.

The motors were left quite exposed and, in some cases, were in close proximity to the circuits. It was assessed that there was no electromagnetic interference with the circuit; however, it would have been safer to add some conductor to block the electromagnetic field of the motor.

5.3 Accomplished Schedule

Over the past 14 weeks, the team has been working diligently on their separate subsystems and collaborating with one another for integration. The proposed and accomplished schedules for the 14 weeks are shown in **Appendix E**.

5.4 Description of Overall Machine

This section will provide an overview of the completed machine. It will summarize the placements of different circuitry on the machine and provide figures of the finished product.

5.4.1 Actuators

The machine uses four motors in total, as shown in **Fig. A-8, Fig. A-9, Fig. A-11**. Two 12V DC 50 rpm gear-head motor drive the wheels of the machine. Another motor is used to drive the hole-sensor mechanism on return of the robot. Finally, the last motor is used to drive the skewed-square mechanism.

5.4.2 Sensors

The machine uses a total of three IR sensors, as shown in **Fig. A-9, Fig. A-10**. Two sensors are mounted next to the wheels and are used to detect the lane. These sensors will tell the machine to swivel appropriately to get back on track. The last sensor is used for sensing black squares, or holes.

5.4.3 Circuit Boards

There are in total 4 circuit boards, excluding the PIC: H-bridge circuit, transistor circuit, IO hub, and sensor circuit. The locations of these circuits are shown in **Fig D-3**. Wires are neatly connected from circuit to circuit in a straight pattern.

5.4.4 PIC board

The PIC board is the essentially the brain of the machine and contains the PIC16 microcontroller. It is located at the back of the machine for the user to easily enter inputs for the machine to go forward as shown in **Fig D-4**. The PIC is mounted on the machine by temporary mounts, such as Velcro, for debugging purposes. It can easily be removed. The PIC is connected to the rest of the machine by a ribbon cable.

5.4.5 Iron Kitty Encasing & LED

The Iron Kitty encasing is made from cardboard and provides a barrier between the circuits and the user, as shown in **Fig D-1**. Cardboard was chosen because it was relatively light and affordable. The encasing allows the top of the case to be opened or closed for the user to store cones. In addition, the encasing consists of holes for the user to interact with the keypad and the LCD. There is also a hole for a LED light for the user to see when the machine detects a hole, as shown in **Fig D-2**.

5.4.6 Key Features

One of the major key features of the machine is that it weighs 27% of the 10kg weight constraint and is 20% of the volume constraints, 50x50x50 cm^3. The compactness is what makes this machine unique from its other competitors. This feature allows the machine to be very portable. In addition, the machine is extendible – it can work for non-straight lanes and can deploy more than 10 cones (despite the small size) if needed to. Moreover, parts of the machine are modular, so they can be replaced easily if a defect is found. The machine is also neatly wired so technicians will have very little trouble debugging, if there comes a need to.

5.5 Standard Operating Procedure

This section will have instructions for the user to operate the machine.

5.5.1 Interface

The robot has been designed for user-friendly interface. There is no instruction written on the robot. However, once turned on with a on/off switch, the LCD display explains every step clearly so first-time users will be able to understand the procedure and operation easily. The keypad used is a 4 x 4 key buttons and we put stickers on three of the buttons that say <Enter>, <Back>, and <Start> so that the user can easily understand what each button is. In addition, it reduces an unnecessary level of conversion when displaying “Press <A> to enter” instead of “Press <Enter>”.

The first step in operating this machine is to turn the power on for both the PIC and the circuits by their respective switches. The LCD then will display the welcome message and asking for location of first cone. The cursor is moved to the right of the message in the first line to allow user input from the keypad and the second line tells the user to press <Enter> to continue. The LCD screen then asks for the distance between subsequent cones the same way. Once both numbers are entered, the LCD screen will display the summary of the two inputs so the user can verify the numbers. If the user makes a mistake at any point, the reset button on the PIC will allow the user to restart and erase previous inputs. Once the summary is displayed, the LCD screen then displays a message to ask the user to press <Start> to start the operation.

5.5.2 Cone Deployment Operation

Once the fully autonomous robot has been started by pressing <Start>, the set up interface is complete the robot will carry out the task fully autonomously. The robot will then operate autonomously deploying cones until the task is completed. The robot will return back to the user when the machine is done deploying cones.

Once the operation is complete, the user will be able to see the summary with the menu display on the LCD screen. There are three components of the summary specified by the proposal: operation time, number of cones deployed and their location with reference to Start Line, number of holes detected and their location with reference to Start Line. This completes the entire operation.

5.5.3 Summary

Once the operation is complete, the user will be able to see the summary with the menu display on the LCD screen. There are three components of the summary specified by the proposal: operation time, number of cones deployed and their location with reference to Start Line, number of holes detected and their location with reference to Start Line. This concludes the whole operation.

Closing Words

The Iron Kitty cone deployment machine was successfully constructed and implemented. The machine worked well and as planned; however, there were a few issues with the mechanical aspect of the machine. Foremost was the deployment mechanism, which was inconsistent with deployment due to the cone getting stuck. Still, by implementing a groove that would keep the cones in place and with a few other minor adjustments, this problem could be easily solved. Secondly, the wheels had high friction and thus caused problems for functions such as swiveling and U-turning. By choosing sturdier and smaller wheels, the surface area would be reduced and the friction would have decreased, thus solving the problem and simultaneously, increasing the speed of the machine. Though the machine was quite satisfactory in design, there would be a few limitations, specifically with regard to real-life implementation. Firstly, by using a column to hold the cones, the number of cones that the machine could hold would be quite limited compared with that of other methods. Secondly, the idea of using a U-turn to return to the start line would be often impractical due to the fact that there may be traffic/pedestrians in the surrounding environment. Thirdly, due to the weight of real-sized cones, the idea o f using the skewed square could be impractical due to the degree of strain/stress resistance that would be needed. More study is still required, mainly to perfect the deployment mechanism in terms of consistency. Further study is also needed to determine whether such a design is sustainable long-term given the large amounts of stress and strain on the square and the associated springs due to the weight of the cones.

References

[1] "Cone Machine." Internet: http://www.dot.ca.gov/newtech/maintenance/ahmct/cone_machine/index.htm, 2007 [Jan. 17, 2011].
[2] "Traffic Cone Machine could signal end of the road for delays." Internet: http://thetransportationgazette.com/2010/09/traffic-cone-machine-could-signal-end-of-the-road-for-delays, Sept. 19, 2010 [Nov. 29, 2003].
[3] "Cone Laying in Safety". Internet: http://www.safetcone.com, [Jan. 21, 201].
[4] "AHMCT Cone Machine." Internet: http://www.ahmct.ucdavis.edu/index.php?title=Cones, 2007 [Jan. 17, 2011].
[5] "Autocone 130." Internet: http://www.innovativequip.com, [Jan. 17, 2003].
[6] Collins, Michael P. (2009) CIV102 Lecture Notes.
[7] Thomas, Rosa, Toussant. (2009). The Analysis and Design of Linear Circuits. 6th ed. pp. 124-129.

Suppliers

Suppliers	**Address**	**Phone**
Above All Electronic Surplus	602 Bloor ST W, Toronto	(416) 588-8119
Active Surplus	347 Queen St. W., Toronto, Ontario M5V 2A4	(416) 593-0909
Canadian Tire	65 Dundas Street W, Toronto, ON M5G 2C3	(416) 979-9056
Creatron Inc.	255 College Street, Toronto, ON M5T 1R5	(416) 977-9258
Home Depot	2121 St. Clair Avenue West, Toronto, ON M6N 5A8	(416) 766-2800
Home Hardware	306 College Street, Toronto, ON M5T 1S3	(416) 922-1158

Bibliography

W.L. Chapman. A.T. Bahill, A.W. Wymore, *Engineering Modeling and Design,* FL: CRC Press, 1992.

B. Davies, *Electric Motors and Mechanical Devices for Hobbyists and Engineers,* Toronto: WERD Technology Inc., 1997.

B. Davies, *Practical Robotics*, Toronto: WERD Technology Inc., 1997.

A. Kusiac, *Engineering Design: Products, Process, and Systems,* London: Academic, 1999.

J.B. Peatman, *Design with PIC Microcontrollers,* NJ: Prentice Hall, 1997.

P.Scherz, *Pratical Electronics for Invertors,* New York: McGraw-Hill, 2000

N. Sclater, N. P. Chironis, *Mechanisms and Mechanical Devices Sourcebook*, McGraw-Hill, 3rd ed., 2001.

J.Walton, *Engineering Design: From Art to Practice,* MN: West Publishing Company, 1991.

APPENDIX 0: CONCEPTS, THEORIES & DESIGNS

0.1 Figures

Fig. 0-1: Autocone 130 cone deployment machine

Fig. 0-2 Non-autonomous traffic cone deployment

0.2 Tables

PUGH CHART									
Designs	**Construction Feasibility**	**Potential Functionality**	**Durability**	**Safety**	**Cost**	**Speed Predictions**	**Simplicity**	**Reliability**	**SUM**
Flap Holders	0	0	0	0	0	0	0	0	0
Rotating Gears	-	+	+	0	-	-	0	-	-2
Twisting Coils	+	-	0	0	0	-	+	0	0
Rotating Claw Holder	+	-	0	0	-	-	-	+	-2
Rotating Base	+	0	-	0	+	+	0	-	+1
Skewed Square	+	0	0	0	0	+	0	0	+2

Table 0-1: Pugh chart comparison of cone deployment designs

Pros	Cons
• Simple to manufacture • Usage of spring allows fast recoil and thus faster deployment • Driving mechanism is a simple pull/push motion driven by motor • Low power consumption • High chance of successful functionality	• Requires experimentation in getting exact motor speed • Stacking of cones limits the number of cones that can be stored

Table 0-2: Pros and cons of skewed square mechanism

APPENDIX A: ELECTROMECHANICAL SYSTEM

The following appendix will show both figures and tables of the electromechanical system of the design.

A.1 Figures

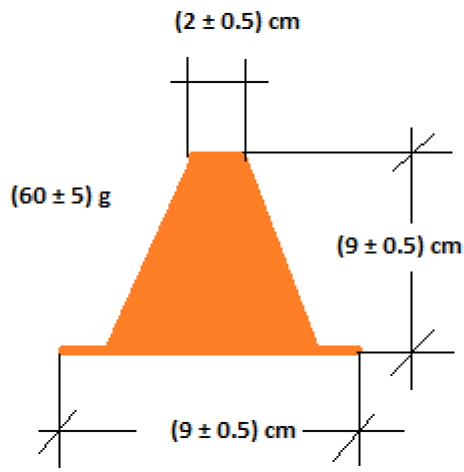


Fig. A-1 Dimensions and weight of cone

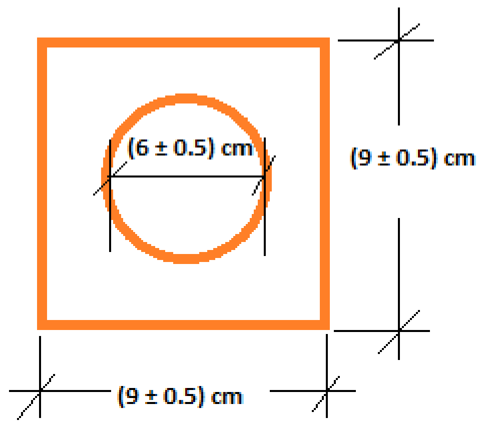


Fig. A-2 Bottom view of cone

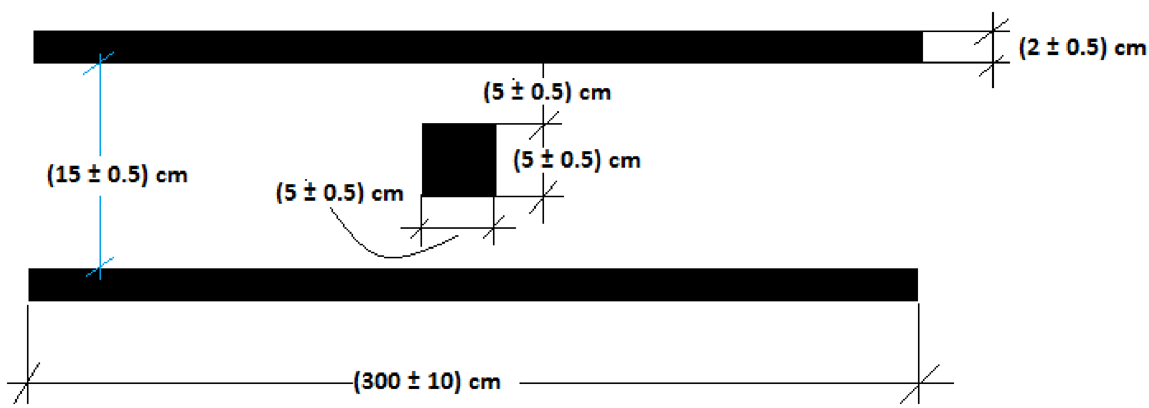


Fig. A-3 Lane dimensions with one black hole

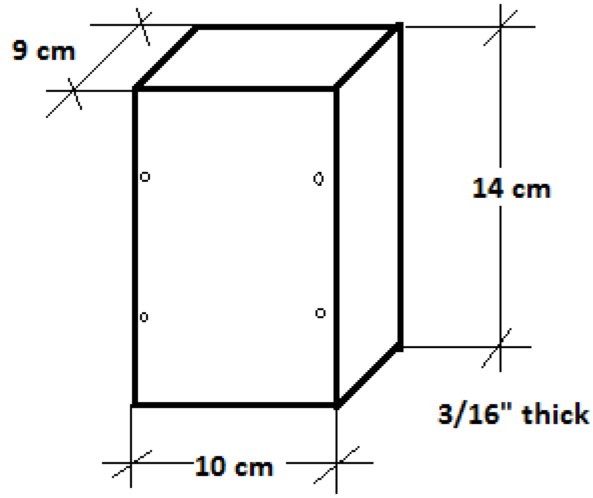


Fig. A-4 Cone storage system

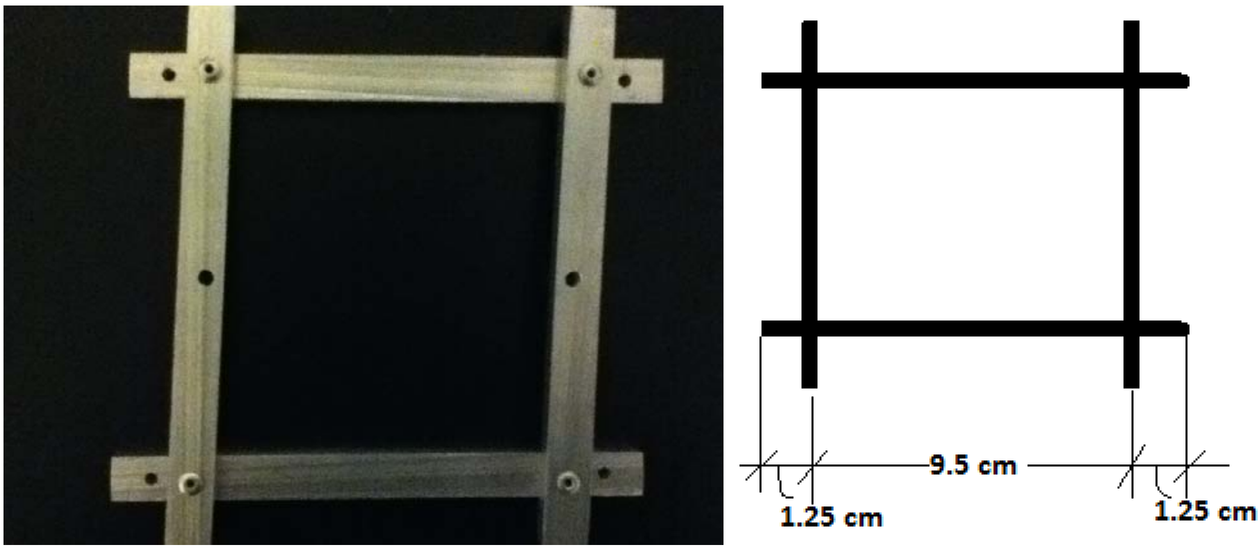
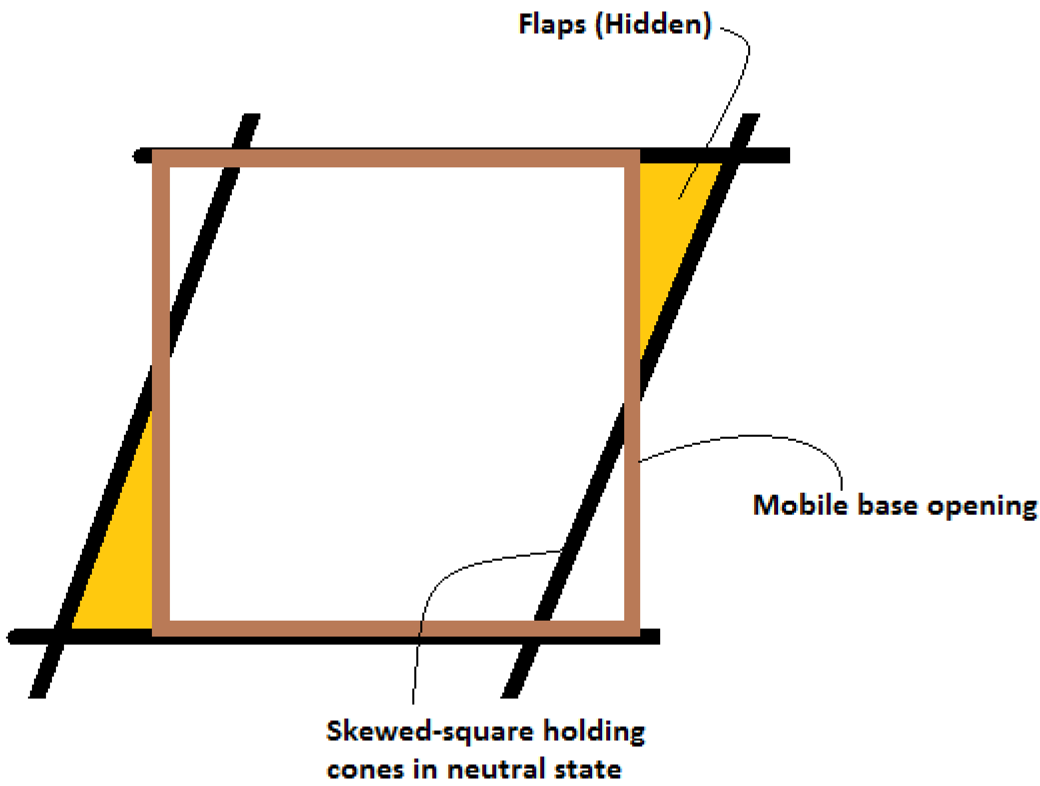


Fig. A-5 The skewed square mechanism

Fig. A-6 Bottom view of the machine and attachment site of skewed square a)



b)

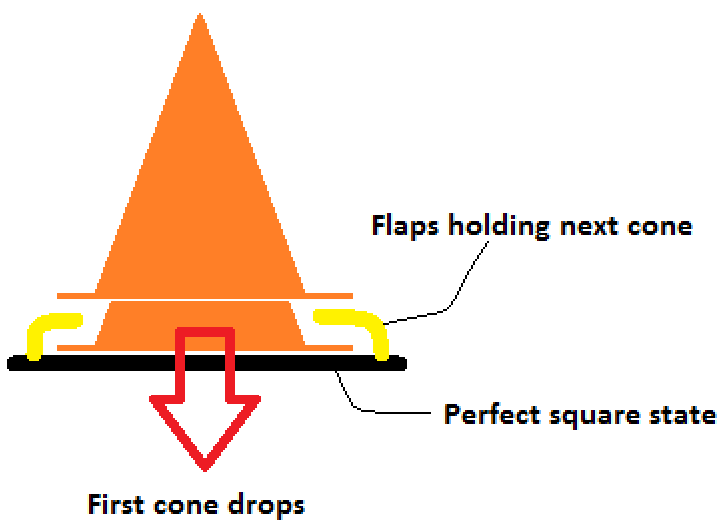


Fig. A-7 Cone separation mechanism **a)** neutral state **b)** perfect square state

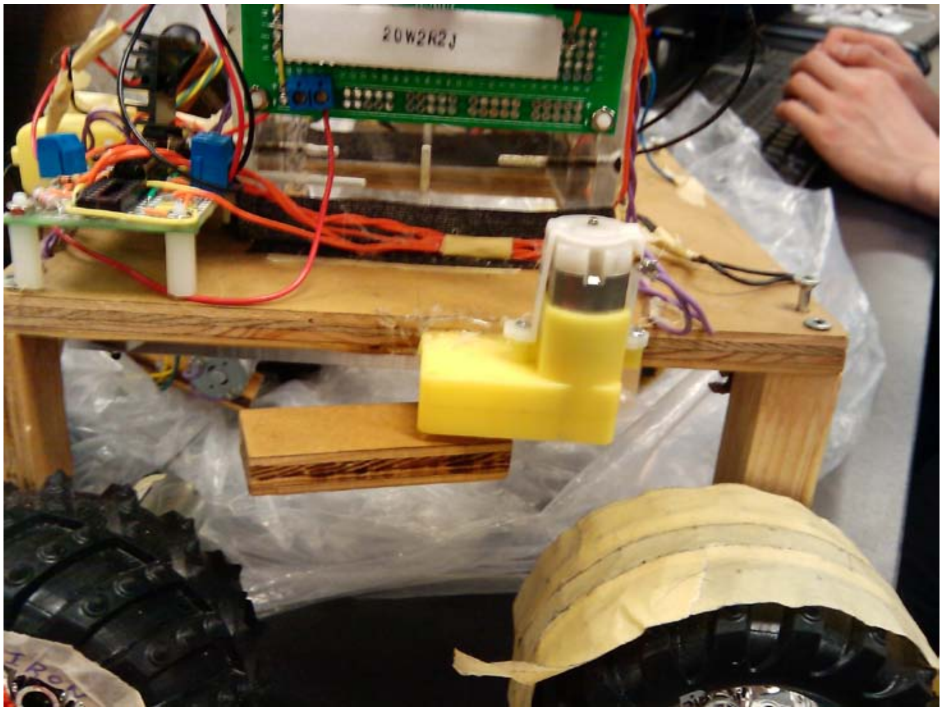


Fig. A-8 Cone-deployment motor attachment

a) **b)**

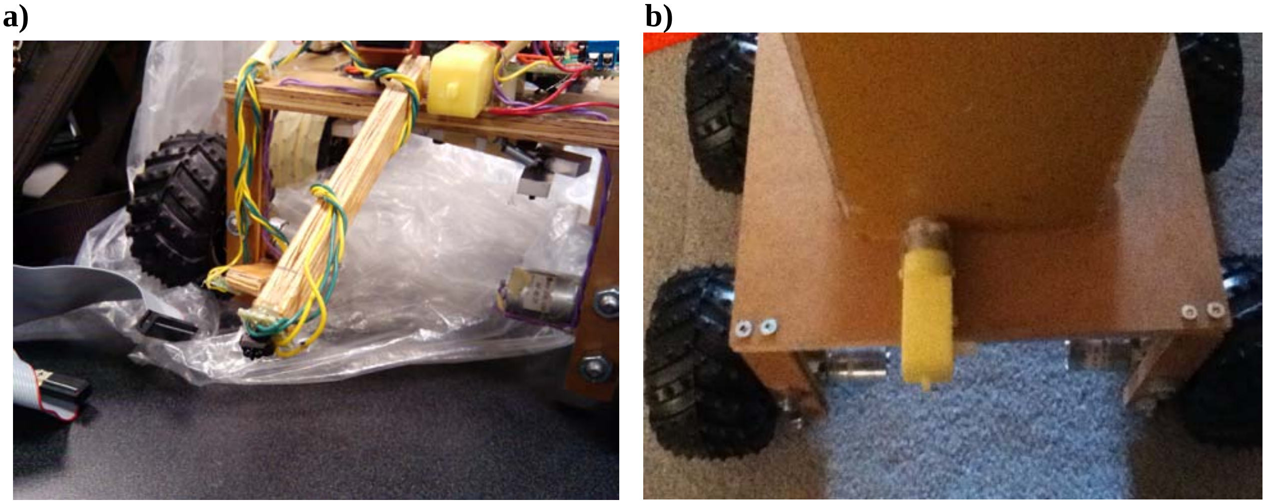


Fig. A-9 Hole sensor **a)** mechanism **b)** hole sensor motor attached to base

a) b)

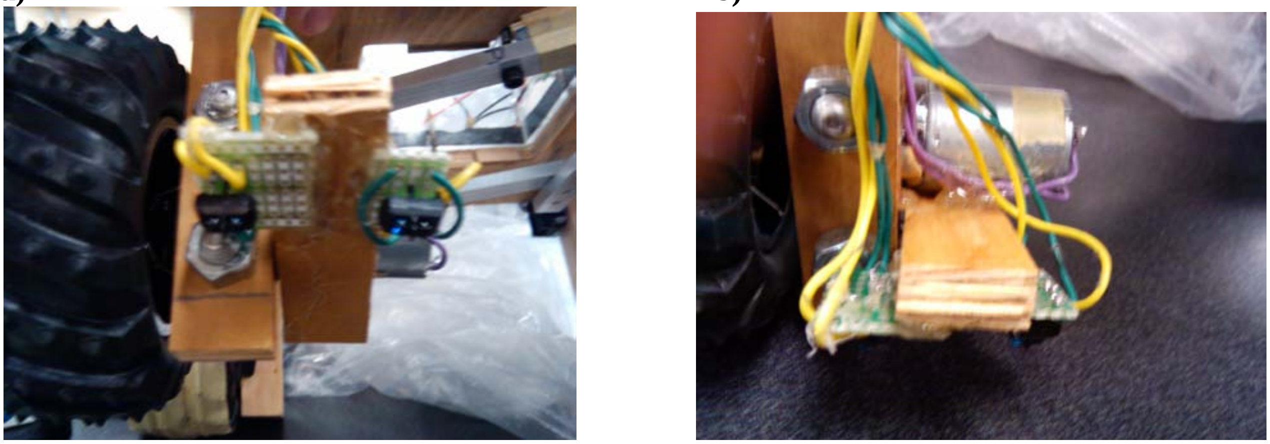


Fig. A-10 Lane sensor attachment **a)** sensor to plywood **b)** plywood to motor

a) b)

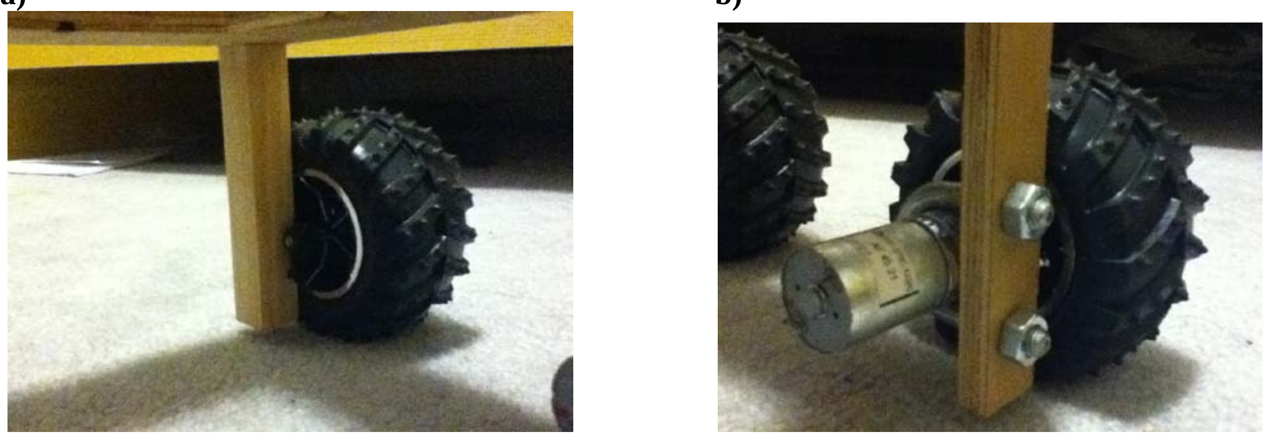


Fig. A-11 Wheel attachment **a)** free wheels **b)** swivelling wheels

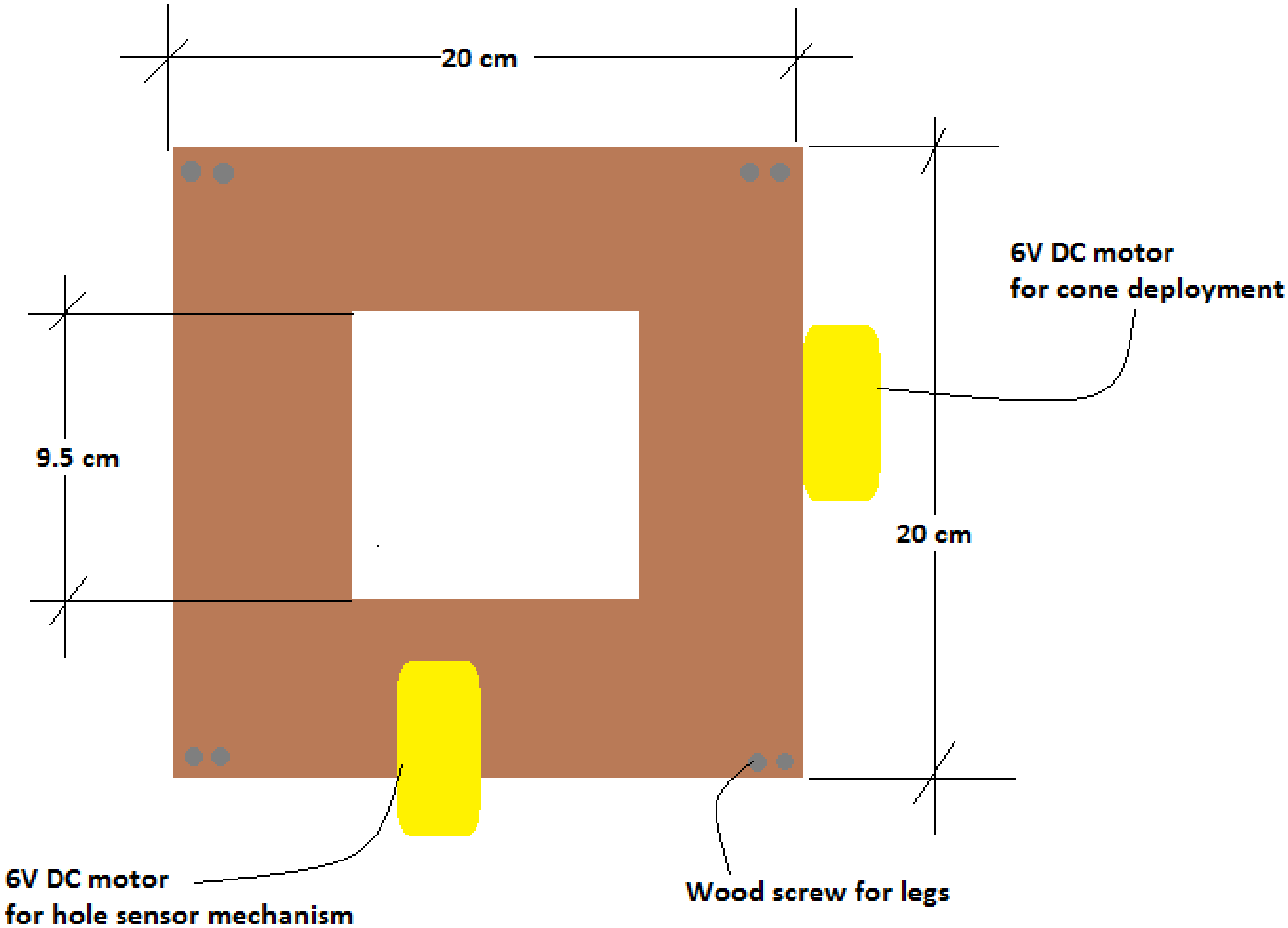
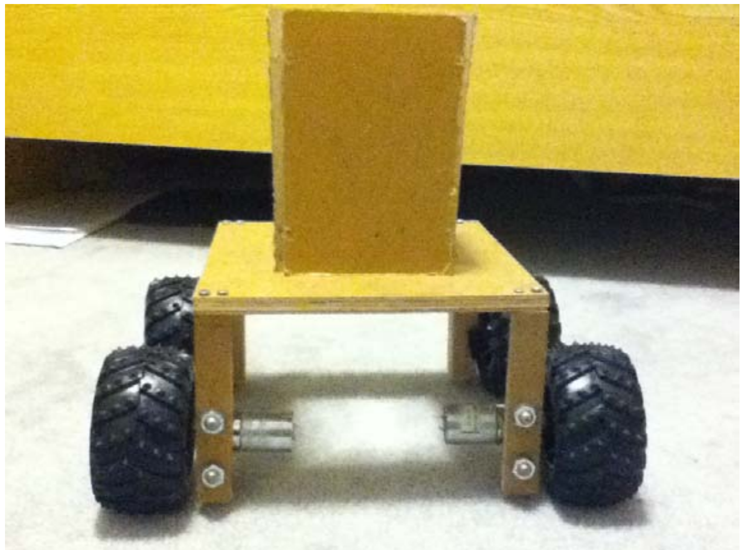
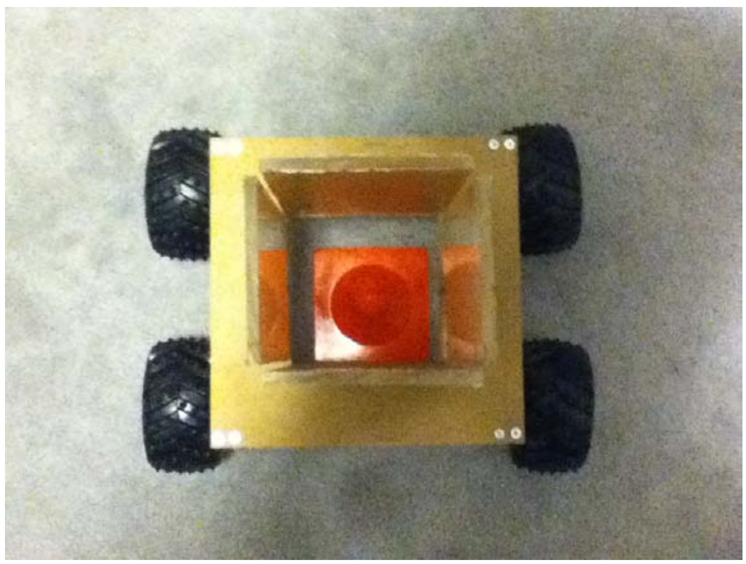


Fig. A-12 Mobile base

a)



b)



c)

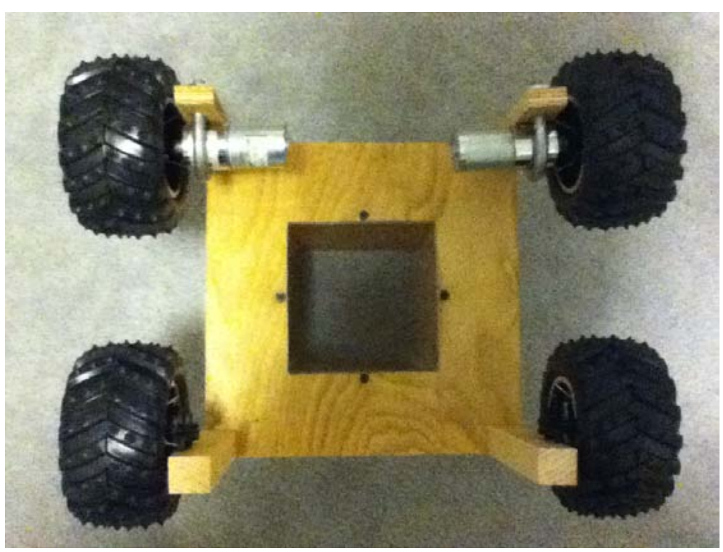


Fig. A-13 Mobile base with storage system **a)** frontal view **b)** top view **c)** bottom view

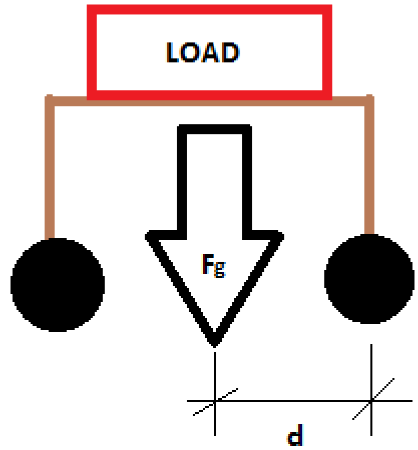


Fig. A-14 Torque calculations

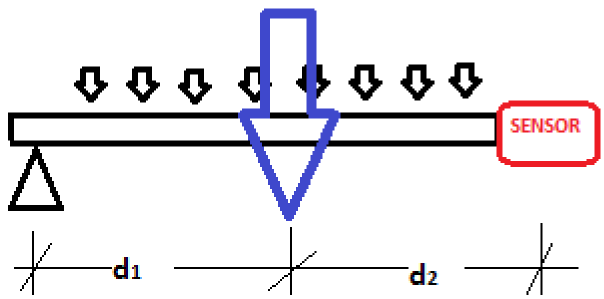


Fig A-15 Moment calculations

A.2 Tables

Constraint	Criteria
Maximum size: 50x50x50 cm^3	The smaller, the better
Maximum weight: 10 kg	The lighter, the better
Budget: $230 CDN	The cheaper, the better
Each run must not exceed 3 minutes	The faster, the better

Table A-1: Constraints and criteria

Material	Advantages	Disadvantages & Limitations
Plywood	◆ Affordable ◆ Light ◆ Strong ◆ Various thickness ◆ Easy-to-work-with	◆ Wood tend to chip off if not careful ◆ Not a clean finish as aluminum plate or acrylic
Aluminum	◆ Strong ◆ Conductive ◆ Durable ◆ Lightest & cheapest metal	◆ Harder to machine with than wood
Acrylic	◆ Very nice finish ◆ Aesthetically appealing	◆ Easy to crack or break when not careful ◆ Expensive ◆ Not as structurally strong as plywood or metal
Cardboard	◆ Light ◆ Cheap casing for robot	◆ Cannot be under excess force
Glue Gun	◆ Great for calibration ◆ Easy for adjustment ◆ Reasonably strong ◆ Cheaper than screws and bolts ◆ Saves time	◆ Cannot be under excess force ◆ Not a fine-finish if used excessively
Screws, bolts, nuts	◆ Strong joint ◆ Durable ◆ Great resistance against shear stress ◆ More professional	◆ Requires routine tightening during debugging phase ◆ More expensive than just glue

Table A-2: Material advantages and disadvantages

APPENDIX B: CIRCUIT DESIGN

B.1 Figures

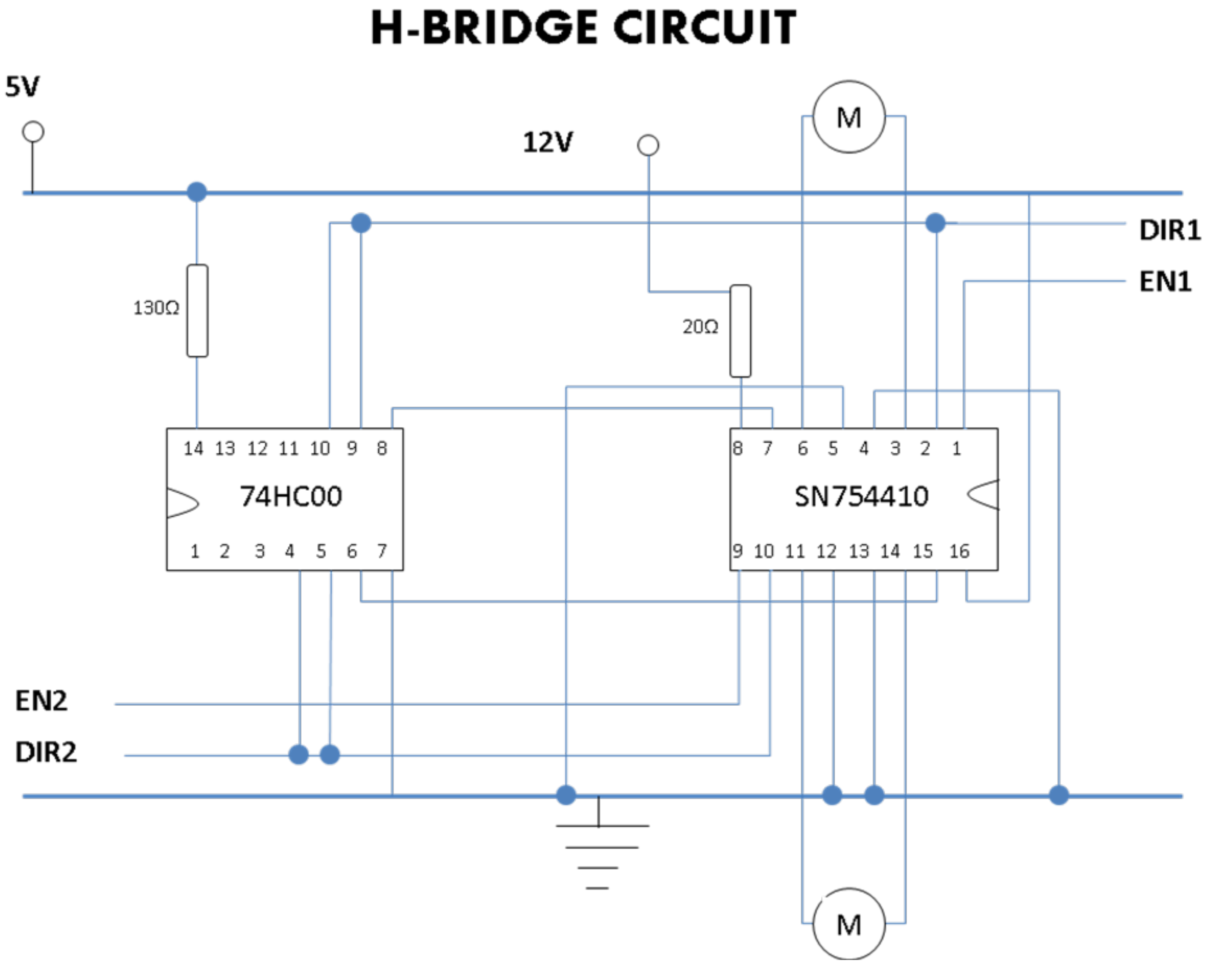
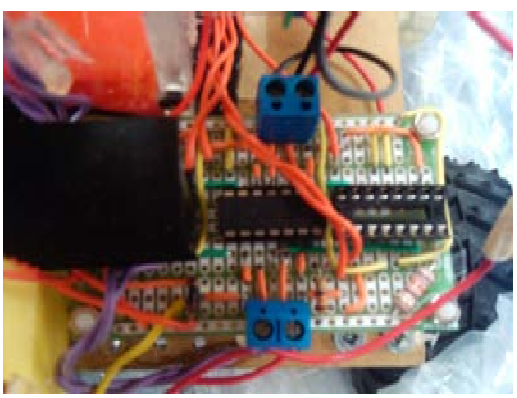


Fig. B-1 H-bridge circuit

a)



b)

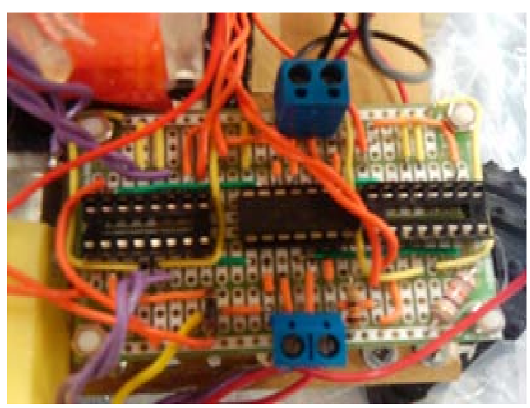


Fig B-2 H-bridge circuit **a)** with H-bridge **b)** without H-bridge

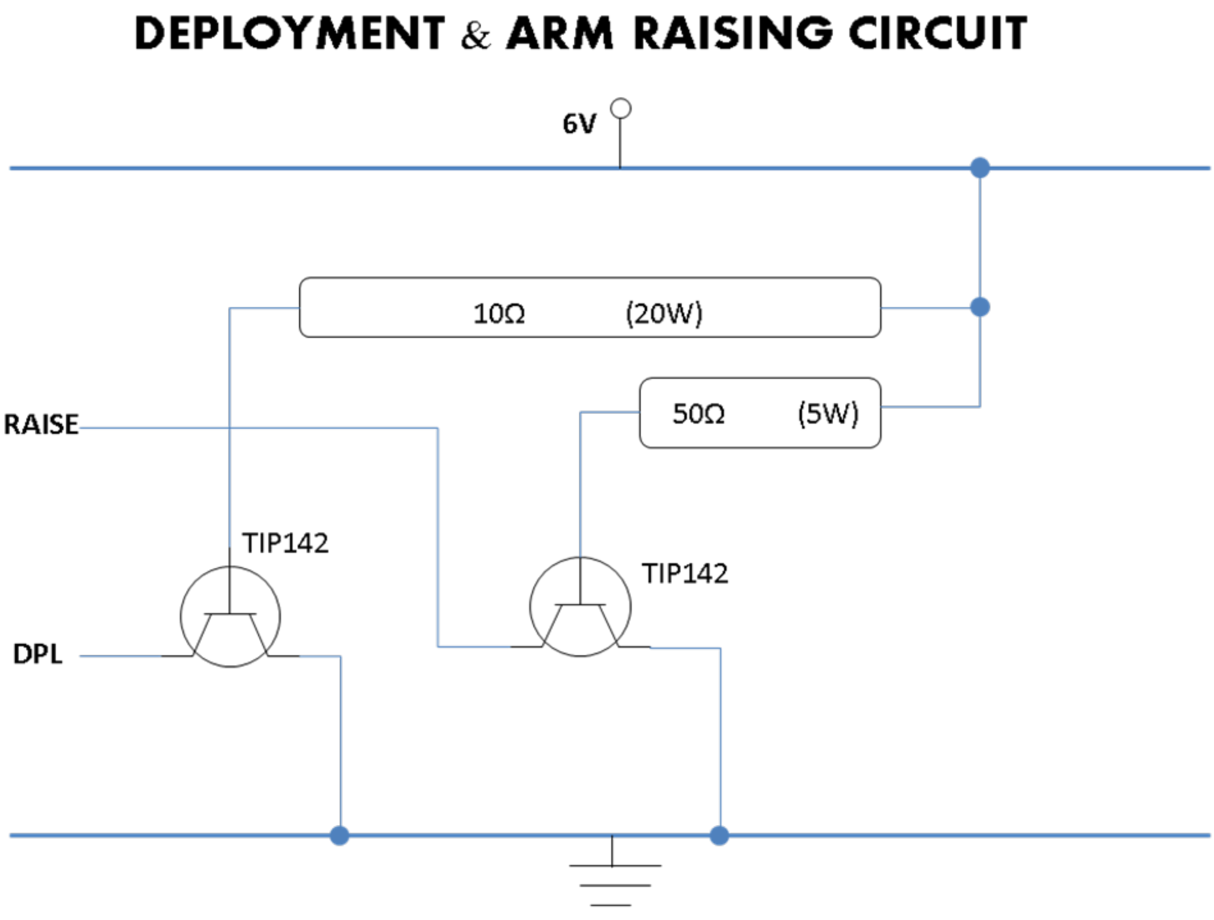


Fig. B-3 Deployment and arm raising circuit

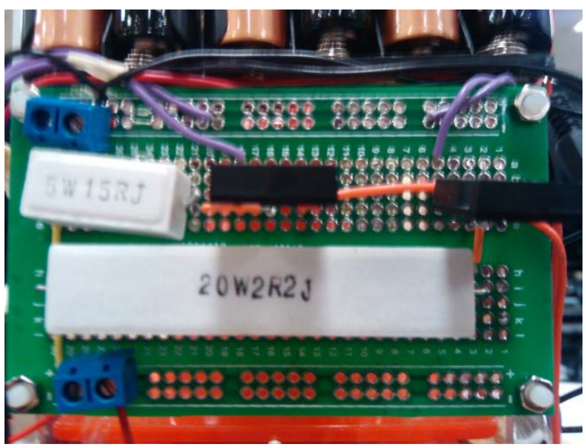


Fig B-4 Transistor circuit for motor control

SENSORS CIRCUIT

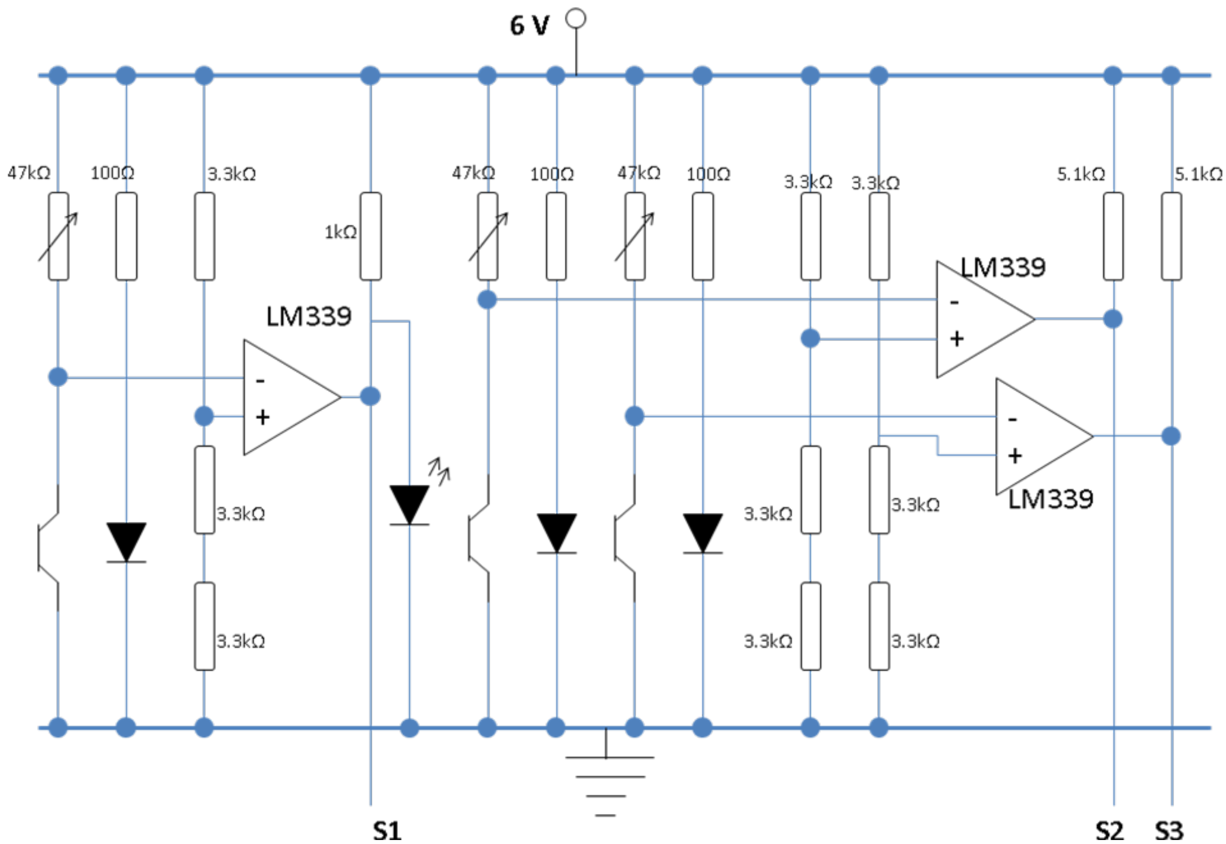


Fig. B-5 Sensors circuit

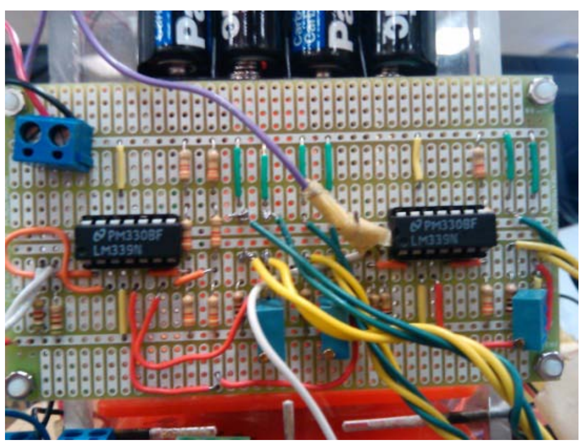


Fig B-6 Sensor circuit

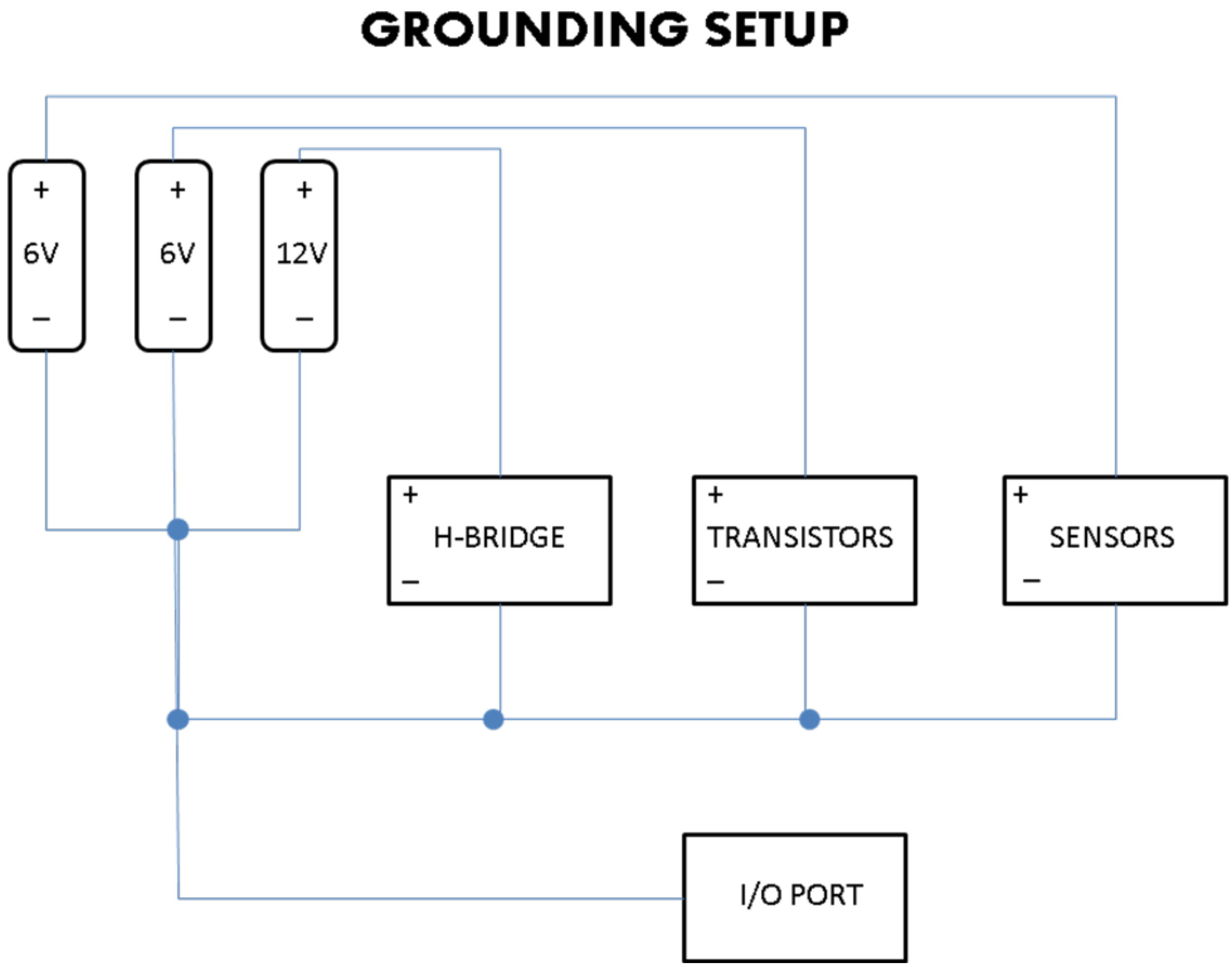


Fig B-7 Grounding all the circuit

a) **b)**

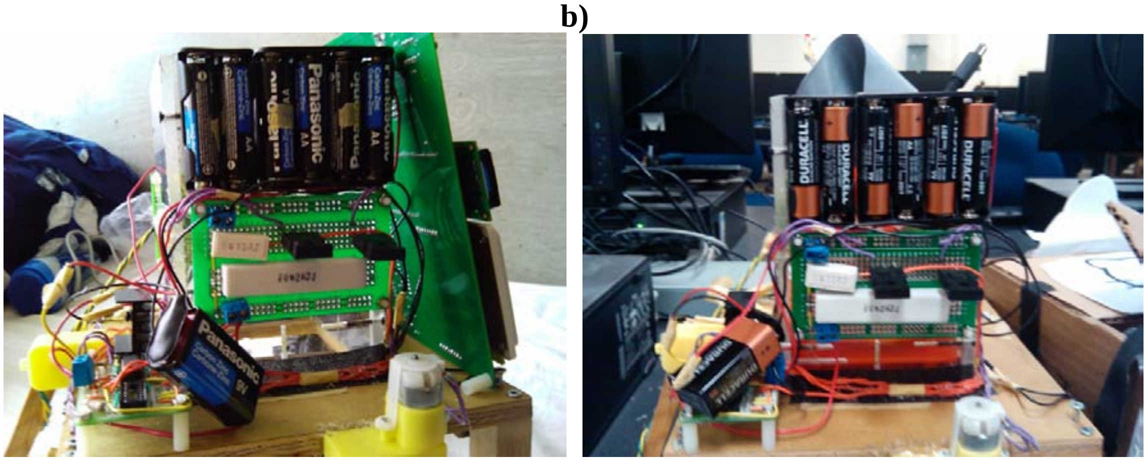


Fig B-8 Power supplies **a)** Panasonic **b)** Duracell

INPUT & OUTPUT BUS

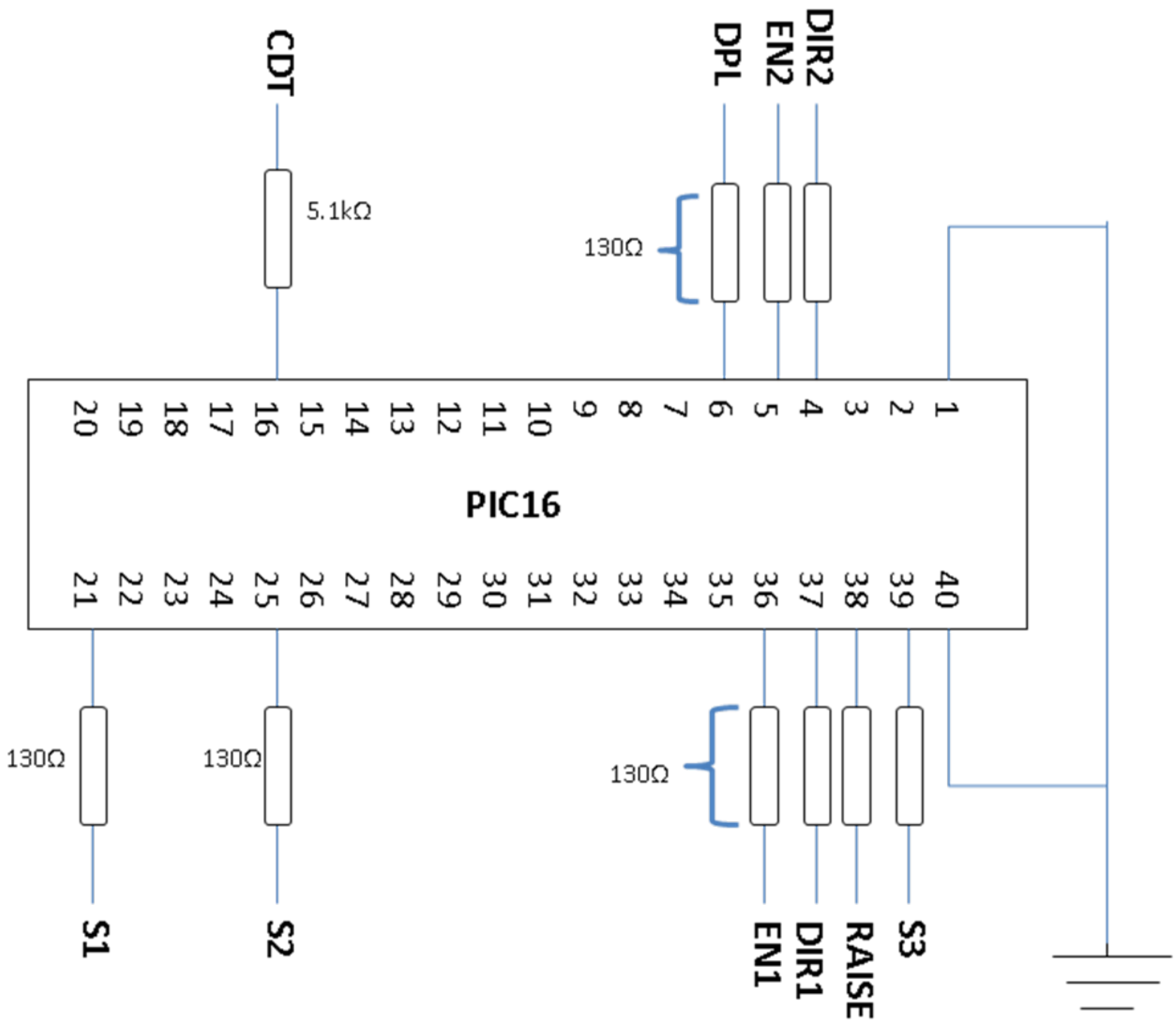


Fig B-9 IO Bus

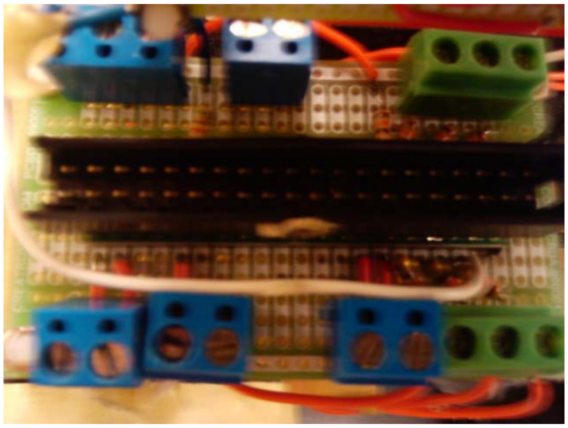


Fig B-10 IO hub circuit

B.2 Tables

ABBREVIATION	SIGNAL	PORT ON PIC
EN1	Enable 1	RC1
EN2	Enable 2	RC2
DIR1	Direction 1	RC3
DIR2	Direction 2	RC4
DPL	Deployment Motor	RC6
RAISE	Raise Arm Motor	RC5
CDT	Cone Detection	RD1
S1	Sensor 1 (hole detect)	RD0
S2	Sensor 2 (swivelling)	RB0
S3	Sensor 3 (swivelling)	RC7

Table B-1 The signal abbreviations

APPENDIX C: PIC LOGIC

C.1 Figures

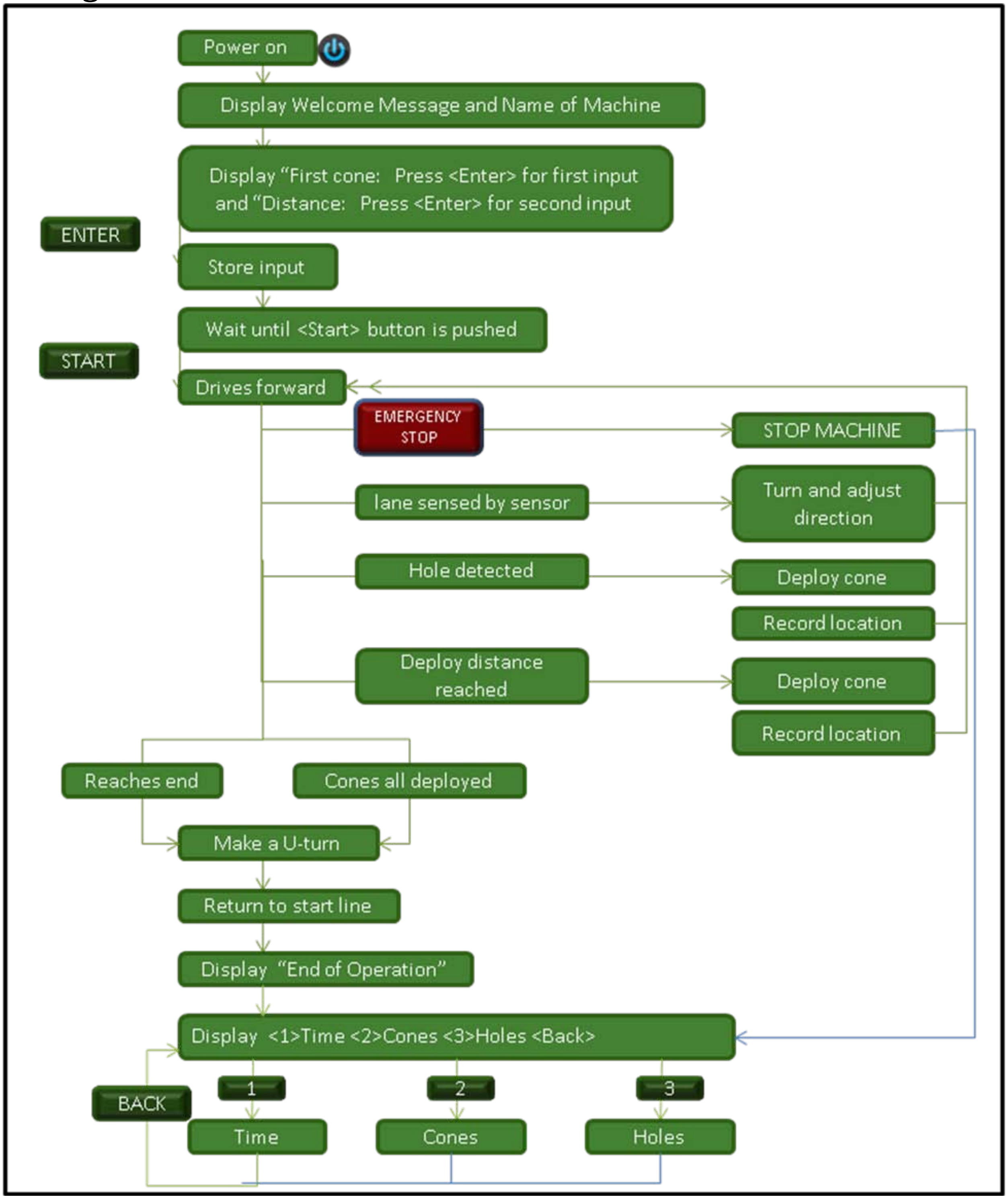


Fig C-1 Final Operating Procedures

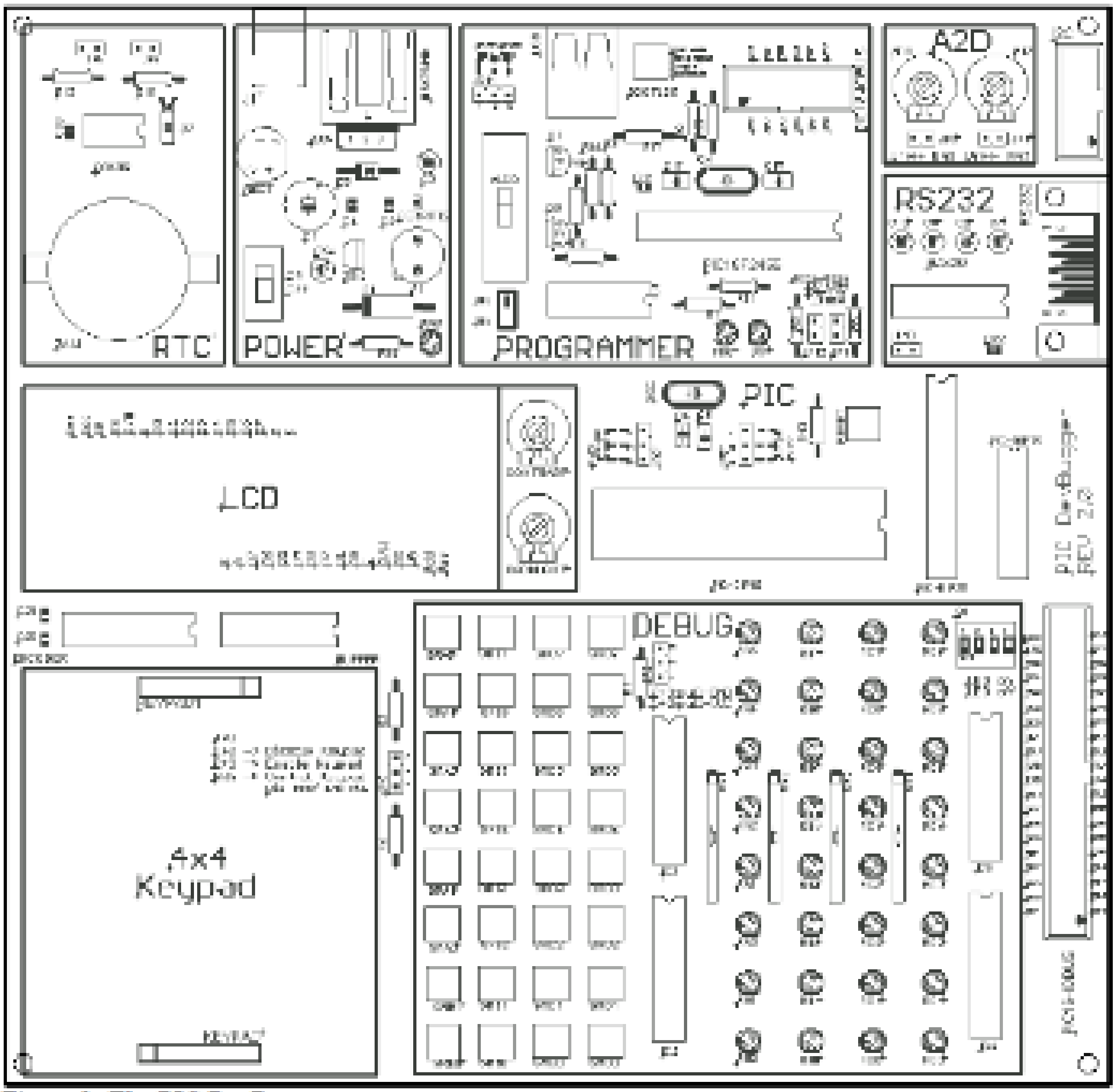


Fig C-2 PIC devbugger board

C.2 Tables

Constants	Values (binary)	Values (Decimal)	Note
Enter_Const	B'00000011'	3	code for keypad A
Back_Const	B'00000111'	7	code for keypad B
Start_Const	B'00001011'	11	code for keypad C
Num_Cycles_U_Turn	H'16'	22	Number of cycles of 0.5 second
Num_Cycles_Return	H'38'	56	Number of cycles of 0.5 second
Dec_Carry	B'00001010'	10	Check if there is a carry (decimal)
Duty_Cycle_100	B'00111111'	n/a	PWM duty cycle for motors for wheels
Duty_Cycle_75	B'00101111'	n/a	PWM duty cycle for motors for wheels
Duty_Cycle_50	B'00011111'	n/a	PWM duty cycle for motors for wheels

Table C-1 Table of constants

Input_Number	check if the current input is 1st or 2nd input
Num_Cones_Deployed	number of cones deployed
TEMP_Num_Cones_Deployed	temp variable
Num_Holes_Detected	number of holes detected
TEMP_Num_Holes_Detected	temp variable
Emergency	check if emergency button is pressed
Emergency_Temp	temp variable
Enter_Key	keypad code for Enter key
Back_Key	keypad code for Back key
Start_Key	keypad code for Startkey
TEMP_Digit	to store keypad num and other temp num
Bin_Digit	temp to convert keypad code to binary code
Display_Num	check if LCD should update number display
TEMP_Num_Cycles_U_Turn	temp variable
TEMP_Num_Cycles_Return	temp variable
TEMP_Dec_Carry	temp variable
Delay	if Delay = 1, hole already detected so ignore detection
TEMP_Delay	temp variable
Delay_12_Cycles	time delay before turning cone deployment motor off
Check_First	check if the input is for 1st cone (or subsequent cones)
Time_S	keep track of time of operation
Location0_Hundreds	location of first cone
Location0_Tens_Ones	
Distance_Hundreds	distance between subsequent adjacent cones
Distance_Tens_Ones	

TEMP_Location TEMP_Cumul_Location	temp variable
Cumul_Location_Hundreds Cumul_Location_Tens_Ones	Location tracker relative to start line
Location_Hundreds Location_Tens_Ones	location tracker relative to last cone
Cumul_Location0_Hundreds Cumul_Location0_Tens_Ones Cumul_Location1_Hundreds Cumul_Location1_Tens_Ones Cumul_Location2_Hundreds Cumul_Location2_Tens_Ones Cumul_Location3_Hundreds Cumul_Location3_Tens_Ones Cumul_Location4_Hundreds Cumul_Location4_Tens_Ones Cumul_Location5_Hundreds Cumul_Location5_Tens_Ones Cumul_Location6_Hundreds Cumul_Location6_Tens_Ones Cumul_Location7_Hundreds Cumul_Location7_Tens_Ones Cumul_Location8_Hundreds Cumul_Location8_Tens_Ones Cumul_Location9_Hundreds Cumul_Location9_Tens_Ones	store recorded cone location with reference to start line
Cumul_Location0_Hole_Hundreds Cumul_Location0_Hole_Tens_Ones Cumul_Location1_Hole_Hundreds Cumul_Location1_Hole_Tens_Ones Cumul_Location2_Hole_Hundreds Cumul_Location2_Hole_Tens_Ones	store recorded hole location with reference to start line
Digit100 Digit10 Digit1	for recording user inputs (up to 3 digits)
Digit_Count	count the number of digits
COUNTH	const used in delay
COUNTM	const used in delay
COUNTL	const used in delay
Table_Counter	for Display macro

Table C-2: Table of variables

APPENDIX D: INTEGRATION

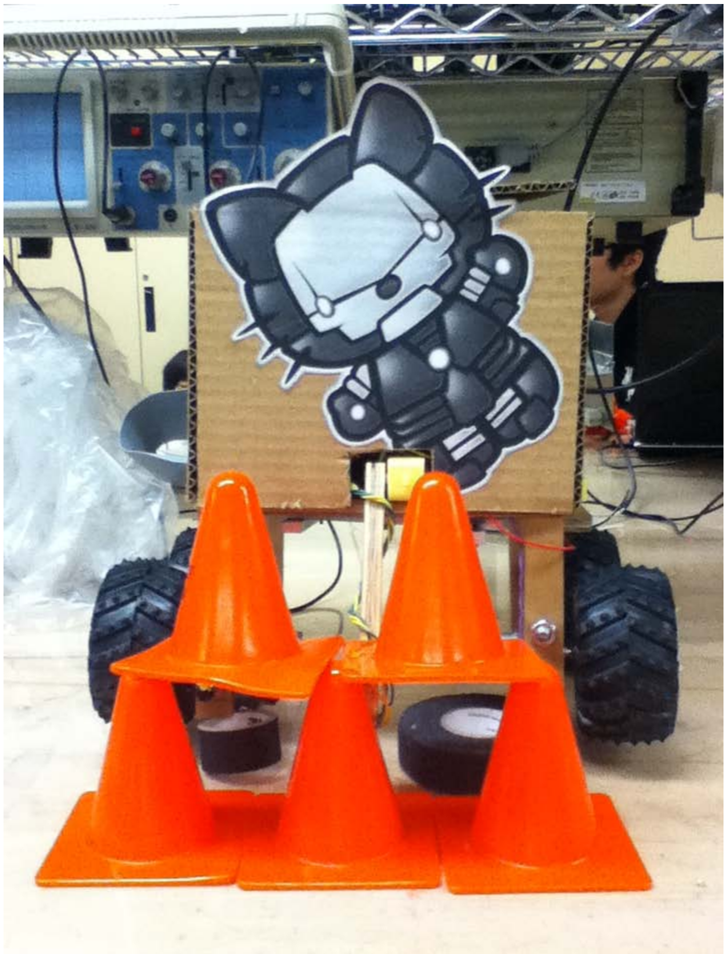


Fig. D-1 Iron Kitty encasing

Fig. D-2 LED for signalling detected cones

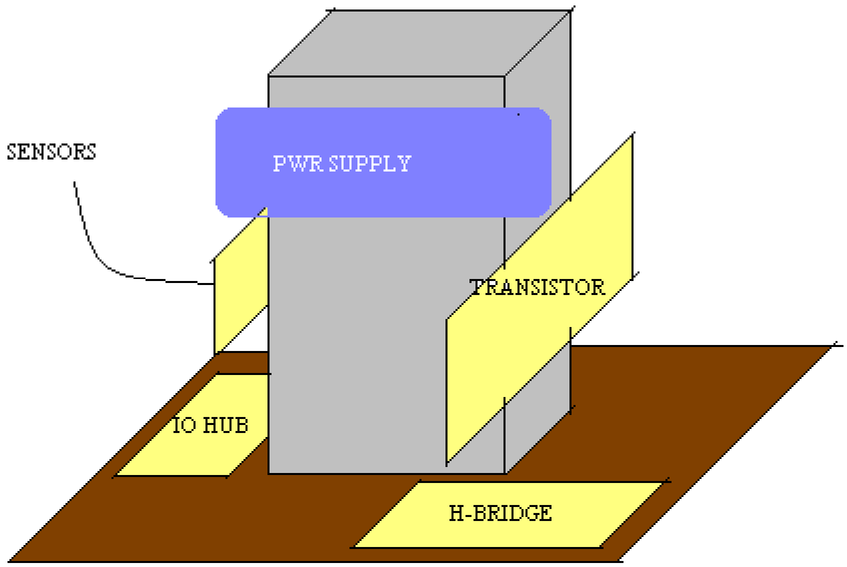


Fig D-3 Circuit placements

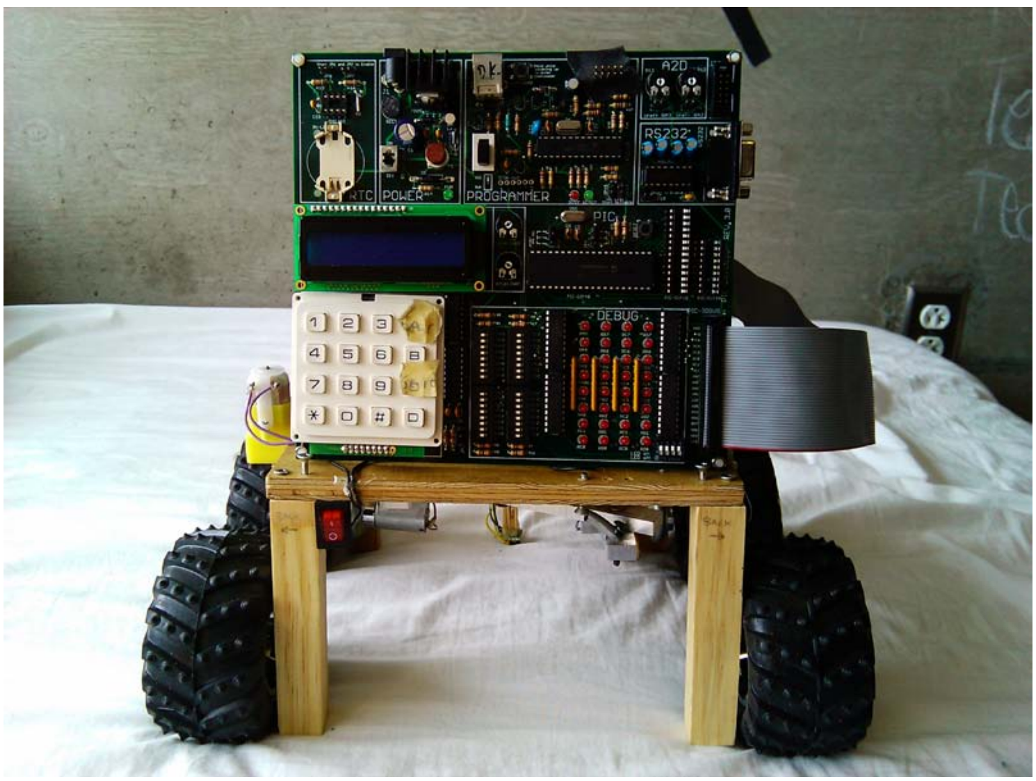


Fig D-4 PIC board mount

APPENDIX E: SCHEDULES

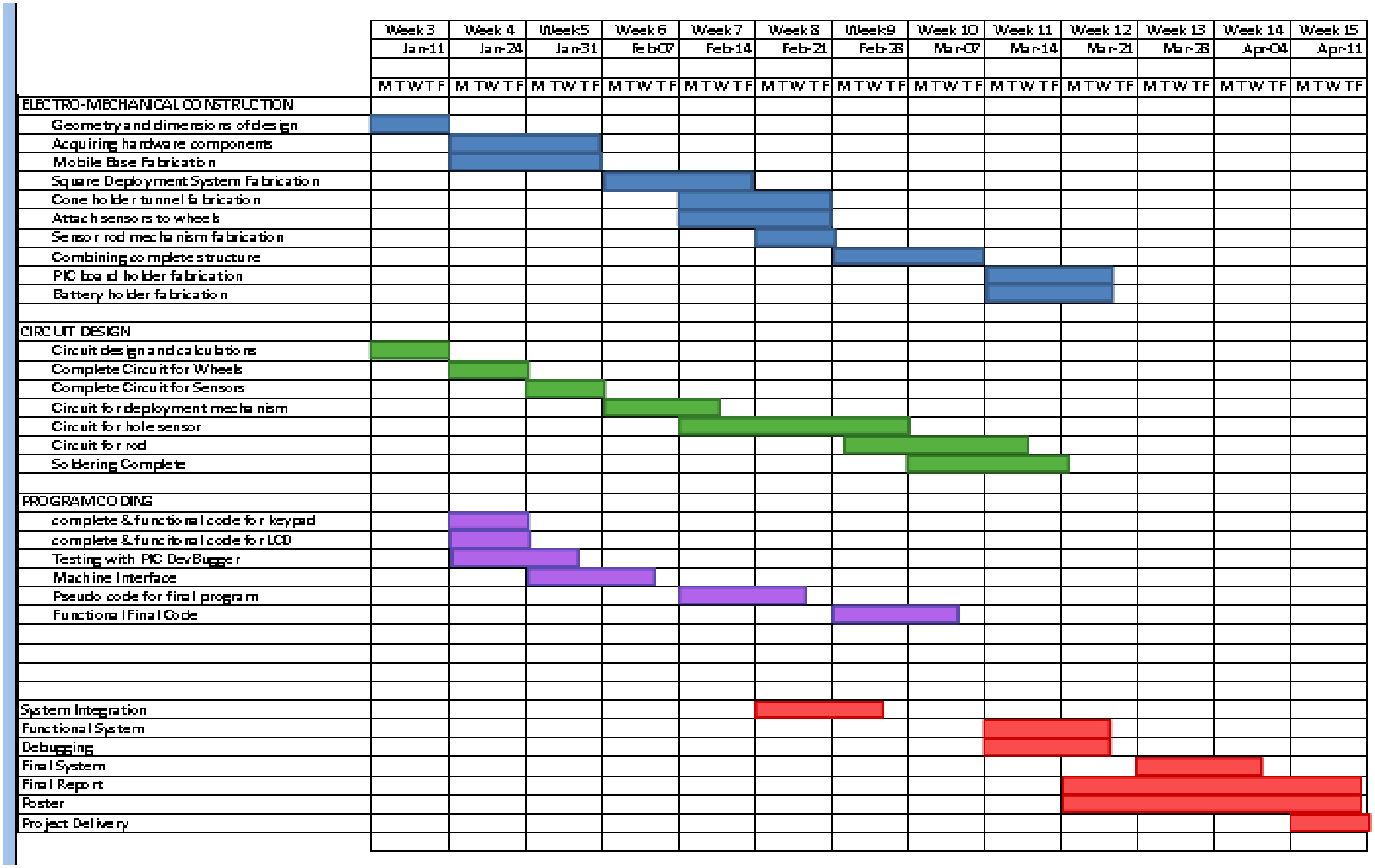


Fig E-1: Proposed Schedule

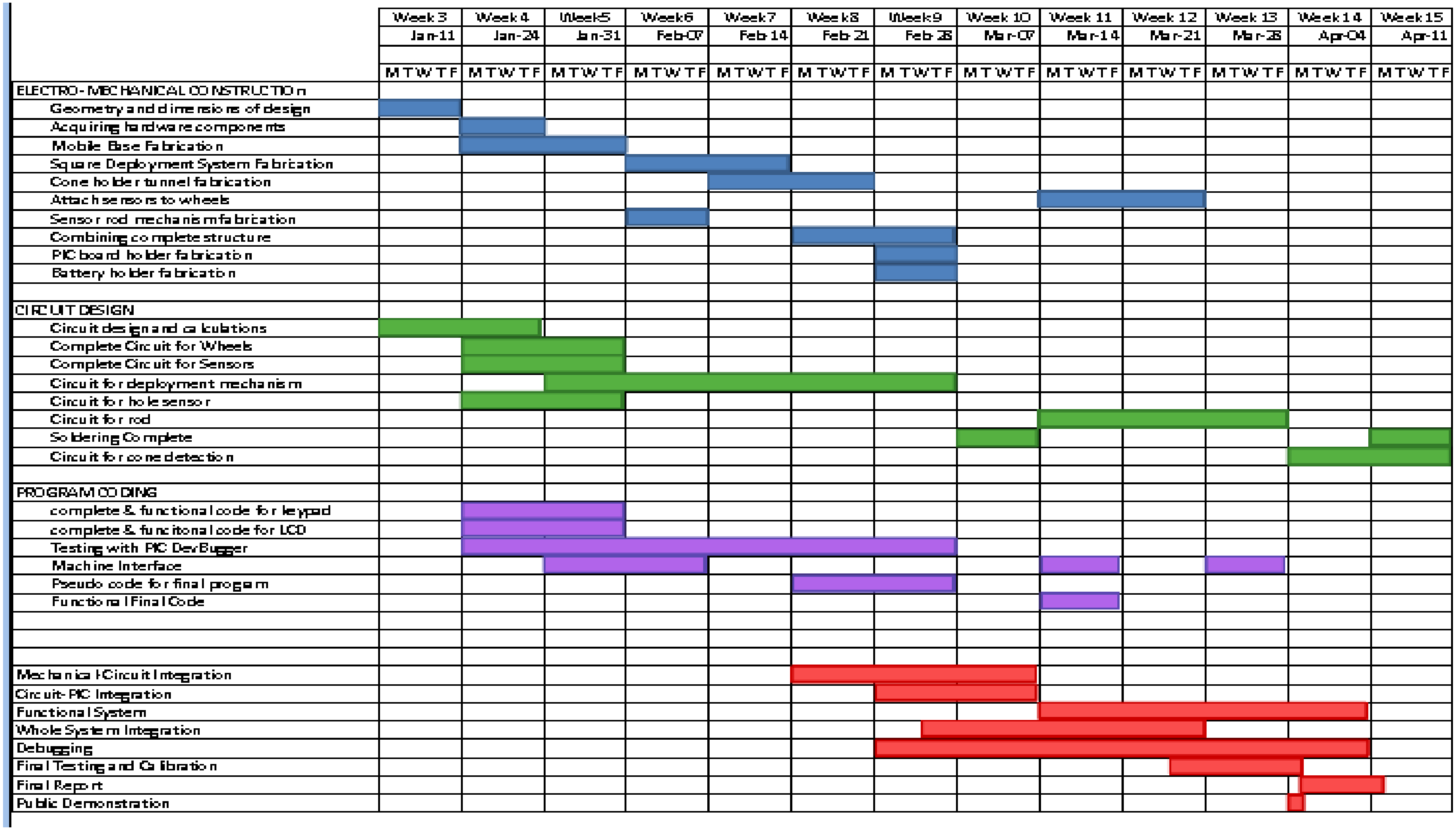


Fig. E-2 Accomplished Schedule

APPENDIX F: BUDGET

ITEM	QUANTITY	COST per ITEM	TOTAL
PROGRAMMING			
PIC Microcontroller	1	50.00	$50.00
CIRCUITRY			
H Bridge (SN754410)	1	2.95	$2.95
Power Transistors (TIP142)	2	1.77	$3.54
NAND gate (74HC00)	1	0.90	$0.90
Comparator (LM339)	2	0.80	$1.60
IR sensors (TCRT5000)	3	1.06	$3.18
40 pin ribbon cable	1	3.00	$3.00
Red LED rocker switch	1	2.45	$2.45
40 pin port ribbon cable port	1	7.66	$7.66
AA battery panasonic	4	0.55	$2.20
AA battery duracell	6	1.02	$6.12
9 V battery duracell	2	2.53	$5.06
4 AA battery holder	2	1.11	$2.22
2 AA battery holder	1	0.77	$0.77
9V battery holder	1	0.85	$0.85
9V battery holder with PIC adaptor	1	1.24	$1.24
mini-solder board (green & white)	2	1.82	$3.64
solder board (green & white)	1	2.67	$2.67
solder board (large holes)	1	4.18	$4.18
2P blue block terminals	9	0.66	$5.94
3P blue block terminals	1	0.67	$0.67
3P green block terminals	2	0.67	$1.34
solid core wires (ft)	5	0.13	$0.65
24 AWG stranded wire (ft)	25	0.12	$3.00
1/4 W resistors	26	0.05	$1.30
20 W 2 ohm resistor	1	2.12	$2.12
5 W 15 ohm resistor	1	1.23	$1.23
14 pin IC socket	3	1.18	$3.54
16 pin IC socket	2	1.18	$2.36
lead 60-40 solder	1	0.83	$0.83
heat sink	1	1.00	$1.00
thermal tape	1	0.10	$0.10

MECHANICAL

20" x 20" x 3/8" birch plywood	1	20.99	$20.99
4.5" x 3/4" x 3/4" Hardwood	1	4.99	$4.99
10" x 3" x 1/2" Aluminum	1	5.58	$5.58
6" x 15" x 3/16" Acrylic	1	3.78	$3.78
Tire wheels	4	4.90	$19.60
Shenzhen 6V DC Gearhead (Straight)	1	5.00	$5.00
Shenzhen 6V DC Gearhead (Angle)	1	5.00	$5.00
Zheng 12V DC Gearhead Motor	2	8.00	$16.00
U-shaped motor mount	2	0.50	$1.00
Mini-spring (fits in 1/16" screw)	2	0.50	$1.00
3/16" wood screw	10	0.29	$2.90
3/16" flat screw	2	0.50	$1.00
1/16" flat screw	10	0.25	$2.50
3/8" pivot screw	4	0.58	$2.32
1/6" bolt	26	0.10	$2.60
3/16" bolt	2	0.10	$0.20
3/16" washer	2	0.15	$0.30
20" x 25" cardboard	1	4.00	$4.00
Masking tape	1	0.20	$0.20
Mastercraft heavy duty glue stick	1	0.50	$0.50

TOTAL	**$227.77 CDN**

APPENDIX G: PIC PROGRAM

```
;********************************************************************************
; Complete code for traffic cone deployment machine
; File name: main.asm
; Assembler : mpasm.exe
; Linker    : mplink.exe
; Written By: Teddy Lin
;********************************************************************************
; SUMMARY:
;       This is the program for the traffic cone deployment machine. It travels on a lane and deploys cones
;       based on the following inputs: regular interval of distance specified by the user input through a
;       keypad and a sensor detecting a hole in the lane. Once completing the lane, the machine will make
;       a U-turn and return along the side of the lane to the starting line. The machine then will display a
;       summary menu to display operation time, number of cones deployed and their location with
;       reference to the starting line, number of holes detected and their location with reference to the
;       starting line.
;********************************************************************************
; HIERARCHY
;   Main
;   Input_Location
;   Poll_Start
;   Loop
;   U_turn
;   Returning
;   Summary
;********************************************************************************
; ASSEMBLER DIRECTIVES
    list p=16f877               ; list directive to define processor
    #include <p16f877.inc>       ; processor specific variable definitions
    __CONFIG _CP_OFF & _WDT_OFF & _BODEN_ON & _PWRTE_ON & _HS_OSC & _WRT_ENABLE_ON &
_CPD_OFF & _LVP_ON
    #include <lcd.inc>                          ;Import LCD control functions from lcd.asm
;********************************************************************************
; CONSTANTS
Enter_Const             EQU     B'00000011'             ;code for keypad A
Back_Const              EQU     B'00000111'             ;code for keypad B
Start_Const             EQU     B'00001011'             ;code for keypad C
Num_Cycles_U_Turn       EQU     H'16'                   ;22
Num_Cycles_Return       EQU     H'38'                   ;56
Dec_Carry               EQU     B'00001010'             ;10
Duty_Cycle_100  EQU     B'00111111'             ;duty cycle for motors for wheels
```

```
Duty_Cycle_75           EQU     B'00101111'
Duty_Cycle_50           EQU     B'00011111'
#define  RS         PORTD,2
#define  E          PORTD,3
;*******************************************************************************
; VARIABLES
 udata
Input_Number                res 1           ;check if the current input is 1st or 2nd input
Num_Cones_Deployed          res 1
TEMP_Num_Cones_Deployed     res 1
Num_Holes_Detected          res 1
TEMP_Num_Holes_Detected     res 1
Emergency                   res 1           ;check if emergency button is pressed
Emergency_Temp              res 1
Enter_Key                   res 1           ;keypad code for Enter key
Back_Key                    res 1
Start_Key                   res 1
TEMP_Digit                  res 1           ;to store keypad num and other temp num
Bin_Digit               res 1           ;temp to convert keypad code to binary code
Display_Num                 res 1           ;check if LCD should update number display
TEMP_Num_Cycles_U_Turn      res 1
TEMP_Num_Cycles_Return      res 1
TEMP_Dec_Carry              res 1
Delay                       res 1       ;if Delay = 1, hole already detected so ignore detection
TEMP_Delay                  res 1
Delay_12_Cycles         res 1       ;time delay before turning cone deployment motor off
Check_First                 res 1
Time_S                      res 1           ;keep track of time of operation
Location0_Hundreds          res 1           ;location of first cone
Location0_Tens_Ones         res 1
Distance_Hundreds           res 1           ;distance between subsequent adjacent cones
Distance_Tens_Ones          res 1
TEMP_Location               res 1
TEMP_Cumul_Location         res 1
Cumul_Location_Hundreds     res 1           ;location tracker relative to starting line
Cumul_Location_Tens_Ones    res 1
Location_Hundreds           res 1           ;location tracker relative to last cone
Location_Tens_Ones          res 1
Cumul_Location0_Hundreds    res 1           ;stores recorded cone location with reference
Cumul_Location0_Tens_Ones   res 1           ; to starting line
Cumul_Location1_Hundreds    res 1
Cumul_Location1_Tens_Ones   res 1
Cumul_Location2_Hundreds    res 1
```

```
Cumul_Location2_Tens_Ones     res 1
Cumul_Location3_Hundreds      res 1
Cumul_Location3_Tens_Ones     res 1
Cumul_Location4_Hundreds      res 1
Cumul_Location4_Tens_Ones     res 1
Cumul_Location5_Hundreds      res 1
Cumul_Location5_Tens_Ones     res 1
Cumul_Location6_Hundreds      res 1
Cumul_Location6_Tens_Ones     res 1
Cumul_Location7_Hundreds      res 1
Cumul_Location7_Tens_Ones     res 1
Cumul_Location8_Hundreds      res 1
Cumul_Location8_Tens_Ones     res 1
Cumul_Location9_Hundreds      res 1
Cumul_Location9_Tens_Ones     res 1
Cumul_Location0_Hole_Hundreds      res 1       ;stores recorded hole location with reference
Cumul_Location0_Hole_Tens_Ones     res 1       ; to starting line
Cumul_Location1_Hole_Hundreds      res 1
Cumul_Location1_Hole_Tens_Ones     res 1
Cumul_Location2_Hole_Hundreds      res 1
Cumul_Location2_Hole_Tens_Ones     res 1
Digit100            res 1           ;for converting 3 decimal digits to 1 hex value
Digit10                  res 1
Digit1                   res 1
Digit_Count              res 1           ;count the number of digits
COUNTH              res 1           ;const used in delay
COUNTM                   res 1           ;const used in delay
COUNTL              res 1           ;const used in delay
Table_Counter            res 1           ;for Display macro
; VECTORS for start of program and interrupt
    ORG    0x0000          ;RESET vector must always be at 0x00
    goto   init        ;Just jump to the main code section.
    ORG    0x0004          ;interrupt vector location
;*******************************************************************************
; LOOK UP TABLE (for display messages on LCD screen)
KPHexToChar
Addwf         PCL,f
 Dt           "123A456B789C*0#D"
Welcome_Msg1
        addwf        PCL,F
        dt           "***Welcome to***", 0
Welcome_Msg2
        addwf        PCL,F
```

```
            dt              "***Iron Kitty***", 0
First_Input_Msg1
            addwf           PCL,F
            dt              "First cone: ", 0
First_Input_Msg2
            addwf           PCL,F
            dt              "Press <Enter> ", 0
Next_Input_Msg1
            addwf           PCL,F
            dt              "Distance: ", 0
Start_Msg1
            addwf           PCL,F
            dt              "Press <Start> ", 0
Emergency_Stop_Msg
            addwf           PCL,F
            dt              "Emergency Stop", 0
Reach_End_Msg1
            addwf           PCL,F
            dt              "End of Lane", 0
All_Cones_Deployed_Msg
            addwf           PCL,F
            dt              "Out of Cones", 0
U_Turn_Msg
            addwf           PCL,F
            dt              "U-turn", 0
Returning_Msg
            addwf           PCL,F
            dt              "Returning", 0
End_Of_Operation_Msg1
            addwf           PCL, F
            dt              "End Of Operation", 0
End_Of_Operation_Msg2
            addwf           PCL,F
            dt              "See Summary", 0
Summary_Msg1
            addwf           PCL,F
            dt              "<1>Time <2>Cones", 0
Summary_Msg2
            addwf           PCL,F
            dt              "<3>Holes <Back>", 0
Summary_Time_Msg
            addwf           PCL,F
            dt              "Operation time ", 0
```

```
Summary_Cones_Msg
        addwf       PCL,F
        dt          "Cones(", 0
Summary_Holes_Msg
        Addwf       PCL,F
        dt          "Holes(", 0
;**********************************************************************************
; MACROS
;***************************************
; Display macro
;***************************************
Display macro   Message
        local   loop_
        local   end_
        clrf    Table_Counter
        clrw
loop_   movf    Table_Counter,W
        call    Message
        xorlw   B'00000000' ;check WORK reg to see if 0 is returned
        btfsc   STATUS,Z
        goto    end_
        call    WR_DATA
        incf    Table_Counter,F
        goto    loop_
end_
        endm
;***************************************
; movff macro
;***************************************
movff   macro   source, destination
        movf        source, W
        movwf       destination
        endm
;**********************************************************************************
; HELPER FUNCTIONS
;***************************************
; keypad code to binary code
;***************************************
Keypad_To_Binary
        movf        TEMP_Digit, W
        andlw       b'00001111'
        movwf       TEMP_Digit
        btfss       TEMP_Digit, 3
```

```
        goto            K3_0
K3_1
        btfss           TEMP_Digit, 2
        goto            K3_1_K2_0
K3_1_K2_1
        clrf            Bin_Digit
        goto            Done_Keypad_To_Binary
K3_1_K2_0
        movlw           b'1'
        subwf           TEMP_Digit, F
        movf            TEMP_Digit, W
        movwf           Bin_Digit
        goto            Done_Keypad_To_Binary
K3_0
        btfss           TEMP_Digit, 2
        goto            K3_0_K2_0
K3_0_K2_1
        movf            TEMP_Digit, W
        movwf           Bin_Digit
        goto            Done_Keypad_To_Binary
K3_0_K2_0
        movf            TEMP_Digit, W
        addlw           b'1'
        movwf           Bin_Digit
Done_Keypad_To_Binary
        return
;**************************************
; LCD control
;**************************************
Switch_Lines
        movlw  B'11000000'
        call   WR_INS
        return
Clear_Display
        movlw  B'00000001'
        call   WR_INS
        return
;**************************************
; Delay 0.5s
;**************************************
HalfS
local   HalfS_0
   movlw 0x88           ;136
```

```
    movwf COUNTH
    movlw 0xBD            ;189
    movwf COUNTM
    movlw 0x03            ;3
    movwf COUNTL

HalfS_0
    decfsz COUNTH, f
    goto   $+2
    decfsz COUNTM, f
    goto   $+2
    decfsz COUNTL, f
    goto   HalfS_0

    goto $+1
    nop
    nop
return
;*************************************
; Delay 0.1s
;*************************************
HundredMS
local       HundredMS_0
    movlw 0x3C            ;60
    movwf COUNTH
    movlw 0xC4            ;196
    movwf COUNTL
HundredMS_0
    decfsz COUNTH, f
    goto   $+2
    decfsz COUNTL, f
    goto   HundredMS_0
    goto $+1
    nop
    nop
return
;*************************************
; Delay 0.01s
;*************************************
TenMS
 local      TenMS_0
    movlw 0x86            ;134
    movwf COUNTH
```

```
    movlw 0x14              ;20
    movwf COUNTL
TenMS_0
    decfsz COUNTH, f
    goto   $+2
    decfsz COUNTL, f
    goto   TenMS_0
    goto $+1
    nop
    nop
return
;*******************************************************************************
; INITIALIZATION
 code     ;'code' lets the linker decide where in program memory to put these instructions.
init
        clrf        INTCON              ;No interrupts
        bcf         STATUS, RP1
        bsf         STATUS,RP0          ; select bank 1
        movlw       b'11111111'         ; Set required keypad inputs
        movwf       TRISA               ; All port A is input
        movwf       TRISB               ; All port B is input
        movlw       b'10000000'
        movwf       TRISC               ; All PORTC is output except for RC7
        movlw       b'00000011'         ;
        movwf       TRISD               ; All PORTC is output except for RD0, RD1
        clrf        TRISE               ; All port E is output
         bcf         STATUS,RP0          ; select bank 0
        clrf        PORTA               ; good practice to clear all ports after setting I/O
        clrf        PORTB
        clrf        PORTC
        clrf        PORTD
        clrf        PORTE
        clrf        Digit_Count         ;set the following variables to 0
        clrf        Digit100
        clrf        Digit10
        clrf        Digit1
        movlw       b'00001101'         ;keypad code for '0' (last 4 bits)
        movwf       Location0_Hundreds
        movwf       Distance_Hundreds
        movlw       b'11011101'         ;keypad code for '00'
        movwf       Location0_Tens_Ones
        movwf       Distance_Tens_Ones
        call         InitLCD        ;Initialize the LCD (code in lcd.asm; imported by lcd.inc)
```

```
;******************************************************************************
; MAIN CODE
Main
Display         Welcome_Msg1        ;Display welcome message for 2 seconds
        call            Switch_Lines
        Display         Welcome_Msg2
        call            HalfS
        call            HalfS
        call            HalfS
        call            HalfS
        call            Clear_Display
        Display         First_Input_Msg1        ;Display 1st input message and wait for input
        call            Switch_Lines
        Display         First_Input_Msg2
        bcf             RS
        movlw           b'10001100'             ;move cursor to 13th position of first row
        call            WR_INS
        bsf             RS
Input_Location
        movlw           Enter_Const
        movwf           Enter_Key
        btfss           PORTB,1                 ;Wait until data is available from the keypad
         goto           $-1
         swapf          PORTB,W                 ;Read PortB<7:4> into W<3:0>
         andlw          0x0F
        movwf           TEMP_Digit
         btfsc          PORTB,1                 ;Wait until key is released
        goto            $-1
        subwf            Enter_Key, F           ;check if <Enter> is pressed
        decf            Enter_Key, F
        incfsz          Enter_Key, F
        goto            Continue_Input          ;<Enter> not pressed
        goto            Store_Input             ;<Enter> pressed
Continue_Input
        call    KPHexToChar ;Convert keypad value to LCD character (value is still held in W)
        call    WR_DATA     ;Write the value in W to LCD
        incf            Digit_Count, F
        decfsz          Digit_Count, F
        goto            Multiple_Digits         ;Digit Count > 1
         movf           TEMP_Digit, W           ;Digit Count = 1, convert to binary
        call            Keypad_To_Binary
        movf            Bin_Digit, W
        movwf           Digit1
```

```
        call        Done_Digits
Multiple_Digits
        decfsz      Digit_Count, F
        goto        Three_Digits        ;Digit Count > 2
        movf        Digit1, W           ;Digit Count = 2, push digit down and convert
        movwf       Digit10             ; newest incoming digit
        movf        TEMP_Digit, W
        call        Keypad_To_Binary
        movf        Bin_Digit, W
        movwf       Digit1
        incf        Digit_Count, F
        call        Done_Digits
Three_Digits
        movf        Digit10, W          ;Push digit down and convert
        movwf       Digit100            ; newest incoming digit
        movf        Digit1, W
        movwf       Digit10
        movf        TEMP_Digit, W
        call        Keypad_To_Binary
        movf        Bin_Digit, W
        movwf       Digit1
        incf        Digit_Count, F
        call        Done_Digits
Done_Digits                                     ;Done converting digit, get next digit input
        incf        Digit_Count, F
        goto        Input_Location
Store_Input
        btfsc       Input_Number, 0
        goto        Second_Input        ;current number is for second input
First_Input                                     ;Stores current number into 1st input
        movf        Digit100, W         ;store 3-digit value in decimal in 2 variables
        movwf       Location0_Hundreds
        movf        Digit10, W
        movwf       Location0_Tens_Ones
        swapf       Location0_Tens_Ones, F
        movf        Digit1, W
        addwf       Location0_Tens_Ones, F
        clrf        Digit_Count         ;clear these variables for second input
        clrf        Digit100
        clrf        Digit10
        clrf        Digit1
        call        Clear_Display
        Display     Next_Input_Msg1     ;Display 2nd input message and wait for input
```

```
        call        Switch_Lines
        Display     First_Input_Msg2
        bcf         RS
        movlw       b'10001010'           ;move cursor to 11th position of first row
        call        WR_INS
        bsf         RS
        incf        Input_Number, F       ;move on to second input
        goto        Input_Location        ;go get next (2nd) input
Second_Input                                      ;Stores current number into 2nd input
        movf        Digit100, W
        movwf       Distance_Hundreds
        movf        Digit10, W
        movwf       Distance_Tens_Ones
        swapf       Distance_Tens_Ones, F
        movf        Digit1, W
        addwf       Distance_Tens_Ones, F
Done_Input                                        ;Display input summary
        call        Clear_Display         ;Display 1st input
        Display     First_Input_Msg1
        movf        Location0_Hundreds, W
        addlw       b'00110000'
        call        WR_DATA
        movff       Location0_Tens_Ones, TEMP_Location
        swapf       Location0_Tens_Ones, W
        andlw       b'00001111'
        addlw       b'00110000'
        call        WR_DATA
        movf        TEMP_Location, W
        andlw       b'00001111'
        addlw       b'00110000'
        call        WR_DATA

        call        Switch_Lines          ;Display 2nd input
        Display     Next_Input_Msg1
        movf        Distance_Hundreds, W
        addlw       b'00110000'
        call        WR_DATA
        movff       Distance_Tens_Ones, TEMP_Location
        swapf       Distance_Tens_Ones, W
        andlw       b'00001111'
        addlw       b'00110000'
        call        WR_DATA
        movf        TEMP_Location, W
```

```
        andlw       b'00001111'
        addlw       b'00110000'
        call        WR_DATA
        call        HalfS                   ;Display for 2 seconds
        call        HalfS
        call        HalfS
        call        HalfS
        call        Clear_Display
        Display     Start_Msg1              ;Display "Press <Start> message
Poll_Start_Button
        movlw       Start_Const
        movwf       Start_Key
        btfss       PORTB,1                  ;Wait until data is available from the keypad
         goto       $-1
        swapf       PORTB,W                  ;Read PortB<7:4> into W<3:0>
        andlw       0x0F
        movwf       TEMP_Digit
        btfsc       PORTB,1                   ;Wait until key is released
         goto       $-1
        subwf       Start_Key, F
        decf        Start_Key, F
        incfsz      Start_Key, F
        goto        Poll_Start_Button       ;Wrong key, try again
Start_Machine                                       ;<Start> pressed, machine starting
        call        Clear_Display
        clrf        Cumul_Location_Hundreds       ;set these variables to 0
        clrf        Cumul_Location_Tens_Ones
        clrf        Location_Hundreds
        clrf        Location_Tens_Ones
        clrf        TEMP_Location
        clrf        Check_First
        clrf        Display_Num
        clrf        Time_S
        clrf        Num_Cones_Deployed
        clrf        Num_Holes_Detected
        clrf        Delay_12_Cycles
Start_Motor         ;set both motors for wheels at 100% duty cycle and forward direction
        bsf         STATUS, RP0             ;select bank 1
        bcf         INTCON, GIE             ;disable global interrupt
        bcf         INTCON, PEIE
        movlw       B'00111111'             ;configure PR2 and CCP1CON
        movwf       PR2
        bcf         STATUS, RP0             ;select bank 0
```

```
        movwf       CCP1CON
        movwf       CCP2CON
        movlw       Duty_Cycle_100      ;configure CCPR1L, 100% duty cycle
        movwf       CCPR1L
        movwf       CCPR2L
        bsf         STATUS, RP0         ;select bank 1
        movlw       b'10000000'
        movwf       TRISC               ;configure PORTC as output except for RC7
        bcf         STATUS, RP0         ;select bank 0
        movlw       B'00000100'         ;configure T2CON
        movwf       T2CON
        ;bcf        PORTC, 3            ;set to forward rotation (0)
        ;bcf        PORTC, 4            ;set to forward rotation (0)
Loop
Check_Emergency     ;check if emergency switch is turned off by check if inputs are active
        btfsc       PORTB, 0
        goto        No_Emergency
        btfsc       PORTC, 7
        goto        No_Emergency
        btfsc       PORTD, 0
        goto        No_Emergency
        incf        Emergency, F
        movf        Emergency, W
        sublw       h'15'   ;21
        movwf       Emergency_Temp
        decfsz      Emergency_Temp, F
        goto        No_Emergency_2
        call        Clear_Display
        Display     Emergency_Stop_Msg
        call        HalfS
        call        HalfS
        incf        Time_S, F
        goto        Summary             ;Emergency stop, go to summary
No_Emergency        ;Emergency button not pressed so continue
        clrf        Emergency
No_Emergency_2      ;Inputs inactive but machine not turned off so continue
        ;check if the lane is curved or if machine is off the lane
        bsf         STATUS, RP0         ;select bank 1
        bcf         INTCON, GIE         ;disable global interrupt
        bcf         INTCON, PEIE
        movlw       B'00111111'         ;configure PR2 and CCP1CON
        movwf       PR2
        bcf         STATUS, RP0         ;select bank 0
```

```
        movwf       CCP1CON
        movwf       CCP2CON
        movlw       Duty_Cycle_100      ;configure CCPR2L, 100% duty cycle, for RC1
        btfss       PORTB, 0
        movlw       Duty_Cycle_50       ;RB0 clear (black): configure CCPR1L to 50%
        movwf       CCPR2L              ;RC1
        movlw       Duty_Cycle_100      ;configure CCPR1L, 100% duty cycle, for RC2
        btfss       PORTC, 7
        movlw       Duty_Cycle_50       ;RB2 clear (black): configure CCPR1L to 50%
        movwf       CCPR1L              ;RC2
        bsf         STATUS, RP0         ;select bank 1
        movlw       b'10000000'
        movwf       TRISC               ;configure PORTC as output except for RC7
        bcf         STATUS, RP0         ;select bank 0
        movlw       B'00000100'         ;configure T2CON
        movwf       T2CON
Incrememt_Location  ;Update location of machine relative to last cone
        movlw       Dec_Carry
        movwf       TEMP_Dec_Carry
        movf        Location_Tens_Ones, W
        andlw       b'00001111'
        subwf       TEMP_Dec_Carry, F
        decfsz      TEMP_Dec_Carry, F
        goto        No_Carry_To_Tens
        goto        Carry_To_Tens
No_Carry_To_Tens
        incf        Location_Tens_Ones, F
        goto        Done_Location
Carry_To_Tens
        movf        Location_Tens_Ones, W;clear location ones
        andlw       b'11110000'
        movwf       Location_Tens_Ones
        movlw       Dec_Carry
        movwf       TEMP_Dec_Carry
        swapf       Location_Tens_Ones, W
        andlw       b'00001111'
        subwf       TEMP_Dec_Carry, F
        decfsz      TEMP_Dec_Carry, F
        goto        No_Carry_To_Hundreds
        goto        Carry_To_Hundreds
No_Carry_To_Hundreds
        movlw       b'00010000'         ;BCD 10
        addwf       Location_Tens_Ones, F
```

```
        goto        Done_Location
Carry_To_Hundreds
        clrf        Location_Tens_Ones
        incf        Location_Hundreds, F
Done_Location       ;Finished updating location
Increment_Cumul_Location     ; Update location of machine relative to starting line
        movlw       Dec_Carry
        movwf       TEMP_Dec_Carry
        movf        Cumul_Location_Tens_Ones, W
        andlw       b'00001111'
        subwf       TEMP_Dec_Carry, F
        decfsz      TEMP_Dec_Carry, F
        goto        No_Carry_To_Tens_Cumul
        goto        Carry_To_Tens_Cumul
No_Carry_To_Tens_Cumul
        incf        Cumul_Location_Tens_Ones, F
        goto        Done_Cumul_Location
Carry_To_Tens_Cumul
        movf        Cumul_Location_Tens_Ones, W ;clear cumul location ones
        andlw       b'11110000'
        movwf       Cumul_Location_Tens_Ones
        movlw       Dec_Carry
        movwf       TEMP_Dec_Carry
        swapf       Cumul_Location_Tens_Ones, W
        andlw       b'00001111'
        subwf       TEMP_Dec_Carry, F
        decfsz      TEMP_Dec_Carry, F
        goto        No_Carry_To_Hundreds_Cumul
        goto        Carry_To_Hundreds_Cumul
No_Carry_To_Hundreds_Cumul
        movlw       b'00010000'         ;BCD 10
        addwf       Cumul_Location_Tens_Ones, F
        goto        Done_Cumul_Location
Carry_To_Hundreds_Cumul
        clrf        Cumul_Location_Tens_Ones
        incf        Cumul_Location_Hundreds, F
Done_Cumul_Location
Check_Update_Display_I ;update display every 10 cm
        movf        Cumul_Location_Tens_Ones, W
        andlw       b'00001111'
        movwf       TEMP_Location
        incf        TEMP_Location
        decfsz      TEMP_Location, F
```

```
            goto            Check_Update_Display_II
            goto            Update_Display
Check_Update_Display_II         ;update display every time a cone is deployed
            btfss           Display_Num, 0
            goto            No_Update_Display
Update_Display              ;Update display every 10 cm or when a cone is deployed
 ;display current location (with reference to last cone)
            call            Clear_Display
            movf            Location_Hundreds, W
            addlw           b'00110000'
            call            WR_DATA
            movff           Location_Tens_Ones, TEMP_Location
            swapf           Location_Tens_Ones, W
            andlw           b'00001111'
            addlw           b'00110000'
            call            WR_DATA
            movf            TEMP_Location, W
            andlw           b'00001111'
            addlw           b'00110000'
            call            WR_DATA
            movlw           " "
            call            WR_DATA
;display current cumul location
            movf            Cumul_Location_Hundreds, W
            addlw           b'00110000'
            call            WR_DATA
            movff           Cumul_Location_Tens_Ones, TEMP_Location
            swapf           Cumul_Location_Tens_Ones, W
            andlw           b'00001111'
            addlw           b'00110000'
            call            WR_DATA
            movf            TEMP_Location, W
            andlw           b'00001111'
            addlw           b'00110000'
            call            WR_DATA
            movlw           " "
            call            WR_DATA
            movf            Num_Cones_Deployed, W       ;display num cones deployed
            addlw           b'00110000'
            call            WR_DATA
            movlw           " "
            call            WR_DATA
            movf            Num_Holes_Detected, W       ;display num holes detected
```

```
        addlw       b'00110000'
        call        WR_DATA
        clrf        Display_Num
No_Update_Display
        call        TenMS                   ;time to travel 1cm (0.06s delay + 0.04s to go
        call        TenMS                   ;    through the big loop = 0.1s)
        call        TenMS
        call        TenMS
        call        TenMS
        call        TenMS
Not_Out_Of_Cones
        btfss       PORTC, 5                ; if motor is on, check if it should be turned off
        goto        Deployment_Motor_Off
        decfsz      Delay_12_Cycles, F
        goto        Loop
Deployment_Motor_Off
        bcf         PORTC, 5                ; turn motor off (if not already off)
        clrf        Delay_12_Cycles
        movlw       b'100'                  ;4
        movwf       TEMP_Cumul_Location
        movf        Cumul_Location_Hundreds, W
        subwf       TEMP_Cumul_Location, F
        decfsz      TEMP_Cumul_Location, F
        goto        Not_End                 ;Has not reached the end
        goto        Reach_End               ; if Cumul_Location_Hundreds = 3, reached end
Not_End     ;Has not reached the end
        btfsc       PORTD, 0                ; check  if there is a hole
        goto        No_Hole_Detected
Hole_Detected       ; hole detected, check if it is newly detected
        incf        Delay, F
        decfsz      Delay, F
        goto        Hole_Already_Detected
        bsf         PORTC, 5                ;new hole detected, activate motor for 1.2s
        movlw       b'00001100'             ;    to deploy a cone
        movwf       Delay_12_Cycles
        incf        Delay, F
        incf        Num_Holes_Detected, F           ;increment num holes detected
        movf        Num_Holes_Detected, W
        movwf       TEMP_Num_Holes_Detected
        decfsz      TEMP_Num_Holes_Detected, F  ;record location to appropriate memory
        goto        Second_Cone_Hole_I
        movff       Cumul_Location_Hundreds, Cumul_Location0_Hole_Hundreds
        movff       Cumul_Location_Tens_Ones, Cumul_Location0_Hole_Tens_Ones
```

```
        goto        Done_Hole_I
Second_Cone_Hole_I
        decfsz      TEMP_Num_Holes_Detected, F
        goto        Third_Cone_Hole_I
        movff       Cumul_Location_Hundreds, Cumul_Location1_Hole_Hundreds
        movff       Cumul_Location_Tens_Ones, Cumul_Location1_Hole_Tens_Ones
        goto        Done_Hole_I
Third_Cone_Hole_I
        decfsz      TEMP_Num_Holes_Detected, F
        goto        Done_Hole_I
        movff       Cumul_Location_Hundreds, Cumul_Location2_Hole_Hundreds
        movff       Cumul_Location_Tens_Ones, Cumul_Location2_Hole_Tens_Ones
Done_Hole_I
        incf        Num_Cones_Deployed, F       ;increment num cones deployed
        movf        Num_Cones_Deployed, W
        movwf       TEMP_Num_Cones_Deployed
        decfsz      TEMP_Num_Cones_Deployed, F ;record location to appropriate memory
        goto        Second_Cone_Hole_II
        movff       Cumul_Location_Hundreds, Cumul_Location0_Hundreds
        movff       Cumul_Location_Tens_Ones, Cumul_Location0_Tens_Ones
        goto        Done_Hole
Second_Cone_Hole_II
        decfsz      TEMP_Num_Cones_Deployed, F
        goto        Third_Cone_Hole_II
        movff       Cumul_Location_Hundreds, Cumul_Location1_Hundreds
        movff       Cumul_Location_Tens_Ones, Cumul_Location1_Tens_Ones
        goto        Done_Hole
Third_Cone_Hole_II
        decfsz      TEMP_Num_Cones_Deployed, F
        goto        Fourth_Cone_Hole_II
        movff       Cumul_Location_Hundreds, Cumul_Location2_Hundreds
        movff       Cumul_Location_Tens_Ones, Cumul_Location2_Tens_Ones
        goto        Done_Hole
Fourth_Cone_Hole_II
        decfsz      TEMP_Num_Cones_Deployed, F
        goto        Fifth_Cone_Hole_II
        movff       Cumul_Location_Hundreds, Cumul_Location3_Hundreds
        movff       Cumul_Location_Tens_Ones, Cumul_Location3_Tens_Ones
        goto        Done_Hole
Fifth_Cone_Hole_II
        decfsz      TEMP_Num_Cones_Deployed, F
        goto        Sixth_Cone_Hole_II
        movff       Cumul_Location_Hundreds, Cumul_Location4_Hundreds
```

```
            movff       Cumul_Location_Tens_Ones, Cumul_Location4_Tens_Ones
            goto        Done_Hole
Sixth_Cone_Hole_II
            decfsz      TEMP_Num_Cones_Deployed, F
            goto        Seventh_Cone_Hole_II
            movff       Cumul_Location_Hundreds, Cumul_Location5_Hundreds
            movff       Cumul_Location_Tens_Ones, Cumul_Location5_Tens_Ones
            goto        Done_Hole
Seventh_Cone_Hole_II
            decfsz      TEMP_Num_Cones_Deployed, F
            goto        Eighth_Cone_Hole_II
            movff       Cumul_Location_Hundreds, Cumul_Location6_Hundreds
            movff       Cumul_Location_Tens_Ones, Cumul_Location6_Tens_Ones
            goto        Done_Hole
Eighth_Cone_Hole_II
            decfsz      TEMP_Num_Cones_Deployed, F
            goto        Nineth_Cone_Hole_II
            movff       Cumul_Location_Hundreds, Cumul_Location7_Hundreds
            movff       Cumul_Location_Tens_Ones, Cumul_Location7_Tens_Ones
            goto        Done_Hole
Nineth_Cone_Hole_II
            decfsz      TEMP_Num_Cones_Deployed, F
            goto        Tenth_Cone_Hole_II
            movff       Cumul_Location_Hundreds, Cumul_Location8_Hundreds
            movff       Cumul_Location_Tens_Ones, Cumul_Location8_Tens_Ones
            goto        Done_Hole
Tenth_Cone_Hole_II
            movff       Cumul_Location_Hundreds, Cumul_Location9_Hundreds
            movff       Cumul_Location_Tens_Ones, Cumul_Location9_Tens_Ones
            goto        All_Cones_Deployed      ;if all cones deployed, go back
Hole_Already_Detected
No_Hole_Detected
            btfsc       PORTD, 0                ; if detect no hole, ready to detect a new hole
            clrf        Delay                   ; if not, hole detection would be ignored
            incf        Check_First, F
            decfsz      Check_First, F
            goto        Subsequent_Cones
First_Cone              ; Check if current location = first location
            movf        Cumul_Location_Tens_Ones, W
            movwf       TEMP_Location
            movf        Location0_Tens_Ones, W
            subwf       TEMP_Location, F
            incf        TEMP_Location, F
```

```
        decfsz      TEMP_Location, F
        goto        Loop
        movf        Cumul_Location_Hundreds, W
        movwf       TEMP_Location
        movf        Location0_Hundreds, W;
        subwf       TEMP_Location, F
        incf        TEMP_Location, F
        decfsz      TEMP_Location, F
        goto        Loop
        incf        Check_First, F          ; reached first cone
        goto        Distance_Reached
Subsequent_Cones    ; Check if current location = distance interval
        movf        Location_Tens_Ones, W
        movwf       TEMP_Location
        movf        Distance_Tens_Ones, W
        subwf       TEMP_Location, F
        incf        TEMP_Location, F
        decfsz      TEMP_Location, F
        goto        Loop
        movf        Location_Hundreds, W
        movwf       TEMP_Location
        movf        Distance_Hundreds, W
        subwf       TEMP_Location, F
        incf        TEMP_Location, F
        decfsz      TEMP_Location, F        ;  check if current location matches 2nd input
        goto        Loop                    ;    if so deploy a cone; if not, loop again
Distance_Reached
        bsf         PORTC, 5                ;distance reached, activate motor for 1.2s
        movlw       b'00001100'             ;delay 12 cycles
        movwf       Delay_12_Cycles
        incf        Num_Cones_Deployed, F           ;increment num cones deployed
        movf        Num_Cones_Deployed, W
        movwf       TEMP_Num_Cones_Deployed
        decfsz      TEMP_Num_Cones_Deployed, F ;record location to appropriate memory
        goto        Second_Cone
        movff       Cumul_Location_Hundreds, Cumul_Location0_Hundreds
        movff       Cumul_Location_Tens_Ones, Cumul_Location0_Tens_Ones
        goto        Done_Hole
Second_Cone
        decfsz      TEMP_Num_Cones_Deployed, F
        goto        Third_Cone
        movff       Cumul_Location_Hundreds, Cumul_Location1_Hundreds
        movff       Cumul_Location_Tens_Ones, Cumul_Location1_Tens_Ones
```

```
        goto        Done_Hole
Third_Cone
        decfsz      TEMP_Num_Cones_Deployed, F
        goto        Fourth_Cone
        movff       Cumul_Location_Hundreds, Cumul_Location2_Hundreds
        movff       Cumul_Location_Tens_Ones, Cumul_Location2_Tens_Ones
        goto        Done_Hole
Fourth_Cone
        decfsz      TEMP_Num_Cones_Deployed, F
        goto        Fifth_Cone
        movff       Cumul_Location_Hundreds, Cumul_Location3_Hundreds
        movff       Cumul_Location_Tens_Ones, Cumul_Location3_Tens_Ones
        goto        Done_Hole
Fifth_Cone
        decfsz      TEMP_Num_Cones_Deployed, F
        goto        Sixth_Cone
        movff       Cumul_Location_Hundreds, Cumul_Location4_Hundreds
        movff       Cumul_Location_Tens_Ones, Cumul_Location4_Tens_Ones
        goto        Done_Hole
Sixth_Cone
        decfsz      TEMP_Num_Cones_Deployed, F
        goto        Seventh_Cone
        movff       Cumul_Location_Hundreds, Cumul_Location5_Hundreds
        movff       Cumul_Location_Tens_Ones, Cumul_Location5_Tens_Ones
        goto        Done_Hole
Seventh_Cone
        decfsz      TEMP_Num_Cones_Deployed, F
        goto        Eighth_Cone
        movff       Cumul_Location_Hundreds, Cumul_Location6_Hundreds
        movff       Cumul_Location_Tens_Ones, Cumul_Location6_Tens_Ones
        goto        Done_Hole
Eighth_Cone
        decfsz      TEMP_Num_Cones_Deployed, F
        goto        Nineth_Cone
        movff       Cumul_Location_Hundreds, Cumul_Location7_Hundreds
        movff       Cumul_Location_Tens_Ones, Cumul_Location7_Tens_Ones
        goto        Done_Hole
Nineth_Cone
        decfsz      TEMP_Num_Cones_Deployed, F
        goto        Tenth_Cone
        movff       Cumul_Location_Hundreds, Cumul_Location8_Hundreds
        movff       Cumul_Location_Tens_Ones, Cumul_Location8_Tens_Ones
        goto        Done_Hole
```

```
Tenth_Cone
        movff       Cumul_Location_Hundreds, Cumul_Location9_Hundreds
        movff       Cumul_Location_Tens_Ones, Cumul_Location9_Tens_Ones
        goto        All_Cones_Deployed
Done_Hole
        clrf        Location_Hundreds
        clrf        Location_Tens_Ones
        incf        Display_Num, F
        goto        Loop                ; whole loop complete, loop again
Reach_End
        call        Clear_Display
        Display     Reach_End_Msg1
        call        HalfS
        call        HalfS
        incf        Time_S, F
        goto        Return_To_Start
All_Cones_Deployed
        call        Clear_Display
        Display     All_Cones_Deployed_Msg
        call        HalfS
        call        HalfS
        incf        Time_S, F
Return_To_Start
U_Turn
        bsf         STATUS, RP0         ;select bank 1
        bcf         INTCON, GIE         ;disable global interrupt
        bcf         INTCON, PEIE
        movlw       B'00111111'         ;configure PR2 and CCP1CON
        movwf       PR2
        bcf         STATUS, RP0         ;select bank 0
        movwf       CCP2CON
        movwf       CCP1CON
        movlw       Duty_Cycle_100      ;configure CCPR2L, 100% duty cycle, for RC1
        movwf       CCPR2L              ;RC1
        movlw       Duty_Cycle_75       ;configure CCPR2L, 75% duty cycle, for RC2
        movwf       CCPR1L              ;RC2
        bsf         STATUS, RP0         ;select bank 1
        movlw       b'10000000'
        movwf       TRISC               ;configure PORTC as output except for RC7
        bcf         STATUS, RP0         ;select bank 0
        movlw       B'00000100'         ;configure T2CON
        movwf       T2CON
        bsf         PORTC, 4            ; set RC2 motor to backward direction
```

```
            call            Clear_Display
            Display         U_Turn_Msg
            movlw           Num_Cycles_U_Turn     ; 11 HalfS (11s to make a U-turn)
            movwf           TEMP_Num_Cycles_U_Turn
Delay_U_Turn
            call            HalfS
            btfss           TEMP_Num_Cycles_U_Turn
            incf            Time_S, F       ; increment time every 2 HalfS cycles (every second)
            decfsz          TEMP_Num_Cycles_U_Turn, F
            goto            Delay_U_Turn
Done_U_Turn
Returning                   ; need to return to starting line, 100% duty cycle and forward for both wheels
            Bcf             PORTC, 3
            bcf             PORTC, 4        ;change direction of wheels, set to forward rotation (0)
            bsf             STATUS, RP0           ;select bank 1
            bcf             INTCON, GIE           ;disable global interrupt
            bcf             INTCON, PEIE
            movlw           B'00111111'           ;configure PR2 and CCP1CON
            movwf           PR2
            bcf             STATUS, RP0           ;select bank 0
            movwf           CCP1CON
            movwf           CCP2CON
            movlw           Duty_Cycle_100        ;configure CCPR2L, 100% duty cycle, for RC1
            movwf           CCPR2L                ;RC1
            movlw           Duty_Cycle_100        ;configure CCPR1L, 100% duty cycle, for RC2
            movwf           CCPR1L                ;RC2
            bsf             STATUS, RP0           ;select bank 1
            movlw           b'10000000'
            movwf           TRISC                 ;configure PORTC as output except for RC7
            bcf             STATUS, RP0           ;select bank 0
            movlw           B'00000100'           ;configure T2CON
            movwf           T2CON
            call            Clear_Display
            Display         Returning_Msg
            movlw           Num_Cycles_Return     ;56 HalfS (28s to travel 300cm straight)
            movwf           TEMP_Num_Cycles_Return
Delay_Returning
            call            HalfS
btfss               TEMP_Num_Cycles_Return
            incf            Time_S, F       ; increment time every 2 HalfS cycles (every second)
            decfsz          TEMP_Num_Cycles_Return, F
            goto            Delay_Returning
End_Of_Operation
```

```
        bsf         STATUS, RP0         ;select bank 1
        movlw       b'10000000'
        movwf       TRISC               ;configure PORTC as output except for RC7
        bcf         STATUS, RP0         ;select bank 0
        clrf        CCP1CON
        clrf        CCP2CON
        clrf        PORTC               ;clear all motors
        call        Clear_Display
        Display     End_Of_Operation_Msg1
        call        Switch_Lines
        Display     End_Of_Operation_Msg2
        call        HalfS
        call        HalfS
Summary             ; Display summary menu
        call        Clear_Display
        Display     Summary_Msg1
        call        Switch_Lines
        Display     Summary_Msg2
        call        HalfS
        call        HalfS
Poll_Summary        ; Poll user input to display
        btfss       PORTB,1             ;Wait until data is available from the keypad
        goto        $-1
        swapf       PORTB,W             ;Read PortB<7:4> into W<3:0>
        andlw       0x0F
        movwf       TEMP_Digit
        btfsc       PORTB,1             ;Wait until key is released
        goto        $-1
        incf        TEMP_Digit, F
        decfsz      TEMP_Digit, F
        goto        Continue_I
Display_Time        ; operation time = distance*(0.1s/cm) + 40s (between 40s to 70s)
        call        Clear_Display
        Display     Summary_Time_Msg
        call        Switch_Lines
        movf        Cumul_Location_Hundreds, W
        movwf       TEMP_Location
        incf        TEMP_Location, F
        decfsz      TEMP_Location, F
        goto        More_Than_100cm
        goto        Under_One_Minute
More_Than_100cm
        decfsz      TEMP_Location, F
```

```
            goto                More_Than_200cm
            goto                Under_One_Minute
More_Than_200cm
            decfsz              TEMP_Location, F
            goto                Max_Time
            goto                Over_One_Minute
Under_One_Minute                ; display time 0:xx
            movlw               "0"
            call                WR_DATA
            movlw               ":"
            call                WR_DATA
            movf                Cumul_Location_Hundreds, W
            addlw               b'00000100'             ; add 4 to ten digit (add 40s)
            addlw               b'00110000'             ; convert to LCD display code
            call                WR_DATA
            swapf               Cumul_Location_Tens_Ones, W
            andlw               b'00001111'
            addlw               b'00110000'
            call                WR_DATA
            goto                Done_Display-Time
Over_One-Minute                 ; display time 1:0x
            movlw               "1"
            call                WR_DATA
            movlw               ":"
            call                WR_DATA
            movlw               "0"
            call                WR_DATA
swapf               Cumul_Location_Tens_Ones, W
            andlw               b'00001111'
            addlw               b'00110000'
            call                WR_DATA
            goto                Done_Display-Time
Max-Time                        ; display time 1:10
            movlw               "1"
            call                WR_DATA
            movlw               ":"
            call                WR_DATA
            movlw               "1"
            call                WR_DATA
            movlw               "0"
            call                WR_DATA
goto                Done_Display-Time
Done_Display_Time
```

```
        goto        Poll_Back               ; wait until <Back> is pressed, else do nothing
Continue_I
        decfsz      TEMP_Digit, F
        goto        Continue_II
Display_Cones
        call        Clear_Display
        Display     Summary_Cones_Msg
        movf        Num_Cones_Deployed, W
        addlw       b'00110000'
        call        WR_DATA                 ; display number of cones deployed
        movlw       ")"
        call        WR_DATA
        movf        Num_Cones_Deployed, W
        movwf       TEMP_Num_Cones_Deployed
        incf        TEMP_Num_Cones_Deployed, F
        decfsz      TEMP_Num_Cones_Deployed, F
        goto        Display_First_Cone
        goto        Done_Display_Cones
Display_First_Cone
        movf        Cumul_Location0_Hundreds, W
        addlw       b'00110000'
        call        WR_DATA
        swapf       Cumul_Location0_Tens_Ones, W
        andlw       b'00001111'
        addlw       b'00110000'
        call        WR_DATA
        movf        Cumul_Location0_Tens_Ones, W
        andlw       b'00001111'
        addlw       b'00110000'
        call        WR_DATA
        movlw       " "
        call        WR_DATA
        decfsz      TEMP_Num_Cones_Deployed, F
        goto        Display_Second_Cone
        goto        Done_Display_Cones
Display_Second_Cone
        movf        Cumul_Location1_Hundreds, W
        addlw       b'00110000'
        call        WR_DATA
        swapf       Cumul_Location1_Tens_Ones, W
        andlw       b'00001111'
        addlw       b'00110000'
        call        WR_DATA
```

```
            movf        Cumul_Location1_Tens_Ones, W
            andlw       b'00001111'
            addlw       b'00110000'
            call        WR_DATA
            call        Switch_Lines
            decfsz      TEMP_Num_Cones_Deployed, F
            goto        Display_Third_Cone
            goto        Done_Display_Cones
Display_Third_Cone
            movf        Cumul_Location2_Hundreds, W
            addlw       b'00110000'
            call        WR_DATA
            swapf       Cumul_Location2_Tens_Ones, W
            andlw       b'00001111'
            addlw       b'00110000'
            call        WR_DATA
            movf        Cumul_Location2_Tens_Ones, W
            andlw       b'00001111'
            addlw       b'00110000'
            call        WR_DATA
            movlw       " "
            call        WR_DATA
            decfsz      TEMP_Num_Cones_Deployed, F
            goto        Display_Fourth_Cone
            goto        Done_Display_Cones
Display_Fourth_Cone
            movf        Cumul_Location3_Hundreds, W
            addlw       b'00110000'
            call        WR_DATA
            swapf       Cumul_Location3_Tens_Ones, W
            andlw       b'00001111'
            addlw       b'00110000'
            call        WR_DATA
            movf        Cumul_Location3_Tens_Ones, W
            andlw       b'00001111'
            addlw       b'00110000'
            call        WR_DATA
            movlw       " "
            call        WR_DATA
            decfsz      TEMP_Num_Cones_Deployed, F
            goto        Display_Fifth_Cone
            goto        Done_Display_Cones
Display_Fifth_Cone
```

```
        movf        Cumul_Location4_Hundreds, W
        addlw       b'00110000'
        call        WR_DATA
        swapf       Cumul_Location4_Tens_Ones, W
        andlw       b'00001111'
        addlw       b'00110000'
        call        WR_DATA
        movf        Cumul_Location4_Tens_Ones, W
        andlw       b'00001111'
        addlw       b'00110000'
        call        WR_DATA
        movlw       " "
        call        WR_DATA
        decfsz      TEMP_Num_Cones_Deployed, F
        goto        Display_Sixth_Cone
        goto        Done_Display_Cones
Display_Sixth_Cone
        movf        Cumul_Location5_Hundreds, W
        addlw       b'00110000'
        call        WR_DATA
        swapf       Cumul_Location5_Tens_Ones, W
        andlw       b'00001111'
        addlw       b'00110000'
        call        WR_DATA
        movf        Cumul_Location5_Tens_Ones, W
        andlw       b'00001111'
        addlw       b'00110000'
        call        WR_DATA
        movlw       " "
        call        WR_DATA
        decfsz      TEMP_Num_Cones_Deployed, F
        goto        Display_Seventh_Cone
        goto        Done_Display_Cones
Display_Seventh_Cone
        call        HalfS                   ; LCD screen full, display for 2 seconds then
        call        HalfS                   ;    display subsequent numbers
        call        HalfS
        call        HalfS
        call        Clear_Display
        Display     Summary_Cones_Msg
        movf        Cumul_Location6_Hundreds, W
        addlw       b'00110000'
        call        WR_DATA
```

```
        swapf       Cumul_Location6_Tens_Ones, W
        andlw       b'00001111'
        addlw       b'00110000'
        call        WR_DATA
        movf        Cumul_Location6_Tens_Ones, W
        andlw       b'00001111'
        addlw       b'00110000'
        call        WR_DATA
        movlw       " "
        call        WR_DATA
        decfsz      TEMP_Num_Cones_Deployed, F
        goto        Display_eighth_Cone
        goto        Done_Display_Cones
Display_eighth_Cone
        movf        Cumul_Location7_Hundreds, W
        addlw       b'00110000'
        call        WR_DATA
        swapf       Cumul_Location7_Tens_Ones, W
        andlw       b'00001111'
        addlw       b'00110000'
        call        WR_DATA
        movf        Cumul_Location7_Tens_Ones, W
        andlw       b'00001111'
        addlw       b'00110000'
        call        WR_DATA
        call        Switch_Lines
        decfsz      TEMP_Num_Cones_Deployed, F
        goto        Display_nineth_Cone
        goto        Done_Display_Cones
Display_nineth_Cone
        movf        Cumul_Location8_Hundreds, W
        addlw       b'00110000'
        call        WR_DATA
        swapf       Cumul_Location8_Tens_Ones, W
        andlw       b'00001111'
        addlw       b'00110000'
        call        WR_DATA
        movf        Cumul_Location8_Tens_Ones, W
        andlw       b'00001111'
        addlw       b'00110000'
        call        WR_DATA
        movlw       " "
        call        WR_DATA
```

```
        decfsz      TEMP_Num_Cones_Deployed, F
        goto        Display_tenth_Cone
        goto        Done_Display_Cones
Display_tenth_Cone
        movf        Cumul_Location9_Hundreds, W
        addlw       b'00110000'
        call        WR_DATA
        swapf       Cumul_Location9_Tens_Ones, W
        andlw       b'00001111'
        addlw       b'00110000'
        call        WR_DATA
        movf        Cumul_Location9_Tens_Ones, W
        andlw       b'00001111'
        addlw       b'00110000'
        call        WR_DATA
Done_Display_Cones
        goto        Poll_Back               ; wait until <Back> is pressed, else do nothing
Continue_II
        decfsz      TEMP_Digit, F           ; wrong key pressed, back to summary menu
        goto        Poll_Summary
Display_Holes
        call        Clear_Display
        Display     Summary_Holes_Msg
        movf        Num_Holes_Deployed, W
        addlw       b'00110000'
        call        WR_DATA                 ; display number of holes detected
        movlw       ")"
        call        WR_DATA
        movf        Num_Holes_Detected, W
        movwf       TEMP_Num_Holes_Detected
        incf        TEMP_Num_Holes_Detected, F
        decfsz      TEMP_Num_Holes_Detected, F
        goto        Display_First_Hole
        goto        Done_Display_Holes
Display_First_Hole
        movf        Cumul_Location0_Hole_Hundreds, W
        addlw       b'00110000'
        call        WR_DATA
        swapf       Cumul_Location0_Hole_Tens_Ones, W
        andlw       b'00001111'
        addlw       b'00110000'
        call        WR_DATA
        movf        Cumul_Location0_Hole_Tens_Ones, W
```

```
        andlw       b'00001111'
        addlw       b'00110000'
        call        WR_DATA
        movlw       " "
        call        WR_DATA
        decfsz      TEMP_Num_Holes_Detected, F
        goto        Display_Second_Hole
        goto        Done_Display_Holes
Display_Second_Hole
        movf        Cumul_Location1_Hole_Hundreds, W
        addlw       b'00110000'
        call        WR_DATA
        swapf       Cumul_Location1_Hole_Tens_Ones, W
        andlw       b'00001111'
        addlw       b'00110000'
        call        WR_DATA
        movf        Cumul_Location1_Hole_Tens_Ones, W
        andlw       b'00001111'
        addlw       b'00110000'
        call        WR_DATA
        call        Switch_Lines
        decfsz      TEMP_Num_Holes_Detected, F
        goto        Display_Third_Hole
        goto        Done_Display_Holes
Display_Third_Hole
        movf        Cumul_Location2_Hole_Hundreds, W
        addlw       b'00110000'
        call        WR_DATA
        swapf       Cumul_Location2_Hole_Tens_Ones, W
        andlw       b'00001111'
        addlw       b'00110000'
        call        WR_DATA
        movf        Cumul_Location2_Hole_Tens_Ones, W
        andlw       b'00001111'
        addlw       b'00110000'
        call        WR_DATA
        movlw       " "
        call        WR_DATA
Done_Display_Holes
        goto        Poll_Back           ; wait until <Back> is pressed, else do nothing
Poll_Back
        movlw       Back_Const
        movwf       Back_Key
```

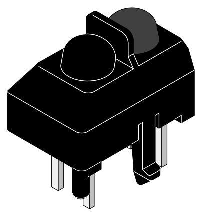
```
        btfss       PORTB,1             ;Wait until data is available from the keypad
goto        $-1
swapf       PORTB,W             ;Read PortB<7:4> into W<3:0>
andlw       0x0F
        movwf       TEMP_Digit
btfsc       PORTB,1             ;Wait until key is released
            goto        $-1
        subwf       Back_Key, F
        decf        Back_Key, F
        incfsz      Back_Key, F
        goto        Poll_Back
        goto        Summary
    END
```

APPENDIX H: DATA SHEETS

Reflective Optical Sensor with Transistor Output

Description

The TCRT5000(L) has a compact construction where the emitting-light source and the detector are arranged in the same direction to sense the presence of an object by using the reflective IR beam from the object. The operating wavelength is 950 mm. The detector consists of a phototransistor.

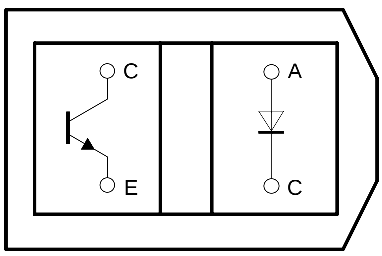


94 9442

Applications

- Position sensor for shaft encoder
- Detection of reflective material such as paper, IBM cards, magnetic tapes etc.
- Limit switch for mechanical motions in VCR
- General purpose – wherever the space is limited

15116



Top view

Features

- Snap-in construction for PCB mounting
- Package height: 7 mm
- Plastic polycarbonate housing construction which prevents crosstalk
- L = long leads
- **C**urrent **T**ransfer **R**atio (CTR) of typical 10%

Order Instruction

Ordering Code	Sensing Distance	Remarks
TCRT5000	12 mm	Leads (3.5 mm)
TCRT5000(L)	12 mm	Long leads (15 mm)

Absolute Maximum Ratings

Input (Emitter)

Parameter	Test Conditions	Symbol	Value	Unit
Reverse voltage		V_R	5	V
Forward current		I_F	60	mA
Forward surge current	$t_p \leq 10$ μA	I_{FSM}	3	A
Power dissipation	$T_{amb} \leq 25^\circ C$	P_V	100	mW
Junction temperature		T_j	100	°C

Output (Detector)

Parameter	Test Conditions	Symbol	Value	Unit
Collector emitter voltage		V_{CEO}	70	V
Emitter collector voltage		V_{ECO}	5	V
Collector current		I_C	100	mA
Power dissipation	$T_{amb} \leq 55^\circ C$	P_V	100	mW
Junction temperature		T_j	100	°C

Sensor

Parameter	Test Conditions	Symbol	Value	Unit
Total power dissipation	$T_{amb} \leq 25^\circ C$	P_{tot}	200	mW
Operation temperature range		T_{amb}	–25 to +85	°C
Storage temperature range		T_{stg}	–25 to +100	°C
Soldering temperature	2 mm from case, $t \leq 10$ s	T_{sd}	260	°C

Electrical Characteristics (T_{amb} = 25°C)

Input (Emitter)

Parameter	Test Conditions	Symbol	Min.	Typ.	Max.	Unit
Forward voltage	I_F = 60 mA	V_F		1.25	1.5	V
Junction capacitance	V_R = 0 V, f = 1 MHz	C_j		50		pF

Output (Detector)

Parameter	Test Conditions	Symbol	Min.	Typ.	Max.	Unit
Collector emitter voltage	I_C = 1 mA	V_{CEO}	70			V
Emitter collector voltage	I_E = 100 μA	V_{ECO}	7			V
Collector dark current	V_{CE} = 20 V, I_F = 0, E = 0	I_{CEO}		10	200	nA

Sensor

Parameter	Test Conditions	Symbol	Min.	Typ.	Max.	Unit
Collector current	V_{CE} = 5 V, I_F = 10 mA, D = 12 mm	I_C [1,2]	0.5	1	2.1	mA
Collector emitter saturation voltage	I_F = 10 mA, I_C = 0.1 mA, D = 12 mm	V_{CEsat} [1,2]			0.4	V
[1] See test circuit						
[2] Test surface: Mirror (Mfr. Spindler a. Hoyer, Part No 340005)						

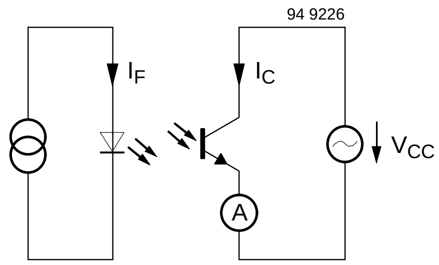


Figure 1. Test circuit

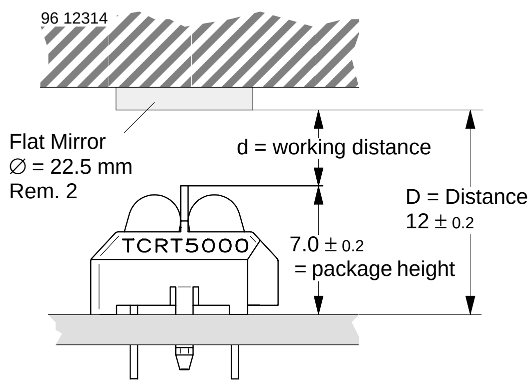


Figure 2. Test circuit

Typical Characteristics (T_{amb} = 25°C, unless otherwise specified)

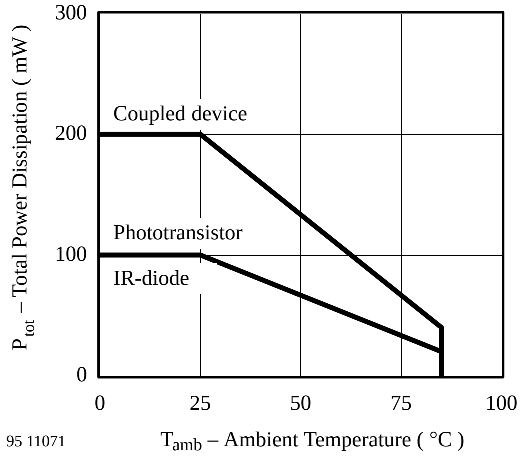


Figure 3. Total Power Dissipation vs. Ambient Temperature

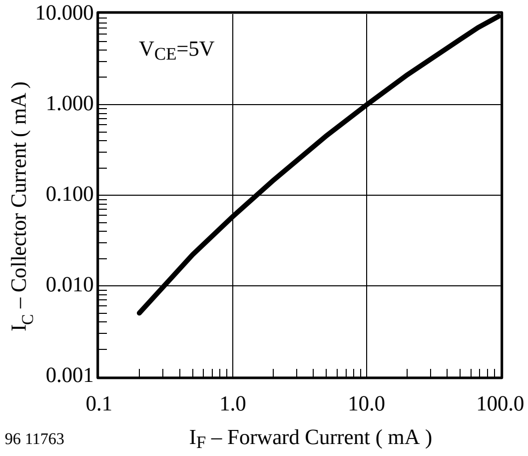


Figure 6. Collector Current vs. Forward Current

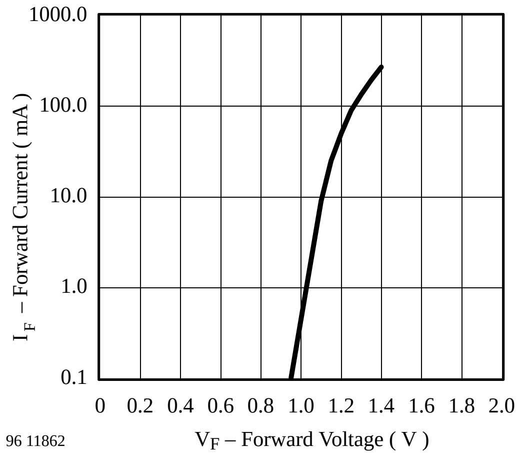


Figure 4. Forward Current vs. Forward Voltage

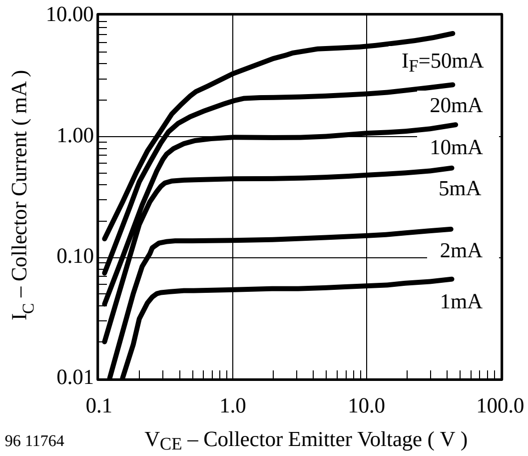


Figure 7. Collector Emitter Saturation Voltage vs. Collector Current

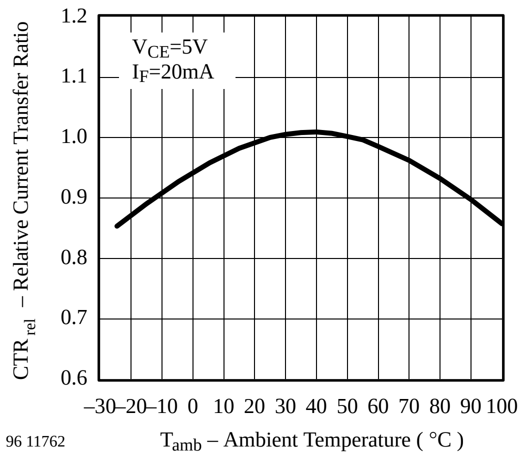


Figure 5. Rel. Current Transfer Ratio vs. Ambient Temp.

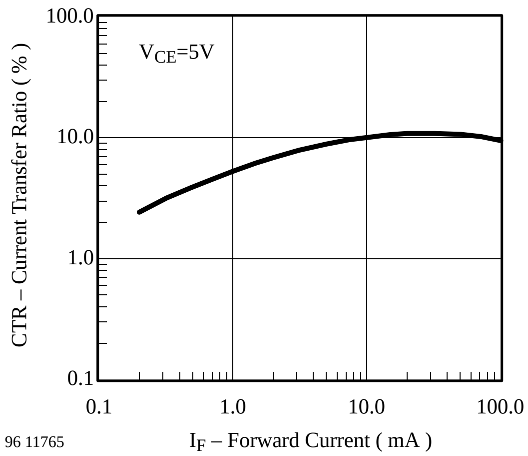


Figure 8. Current Transfer Ratio vs. Forward Current

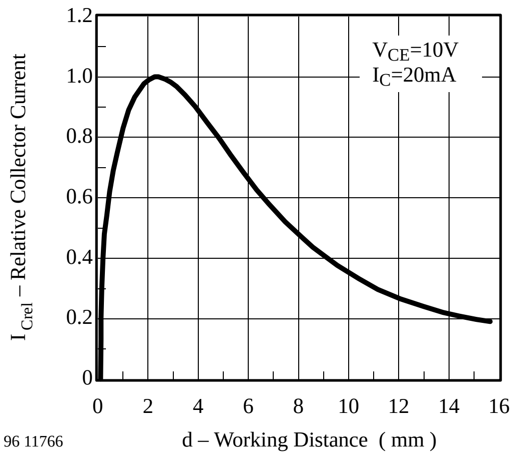


Figure 9. Relative Collector vs. Distance

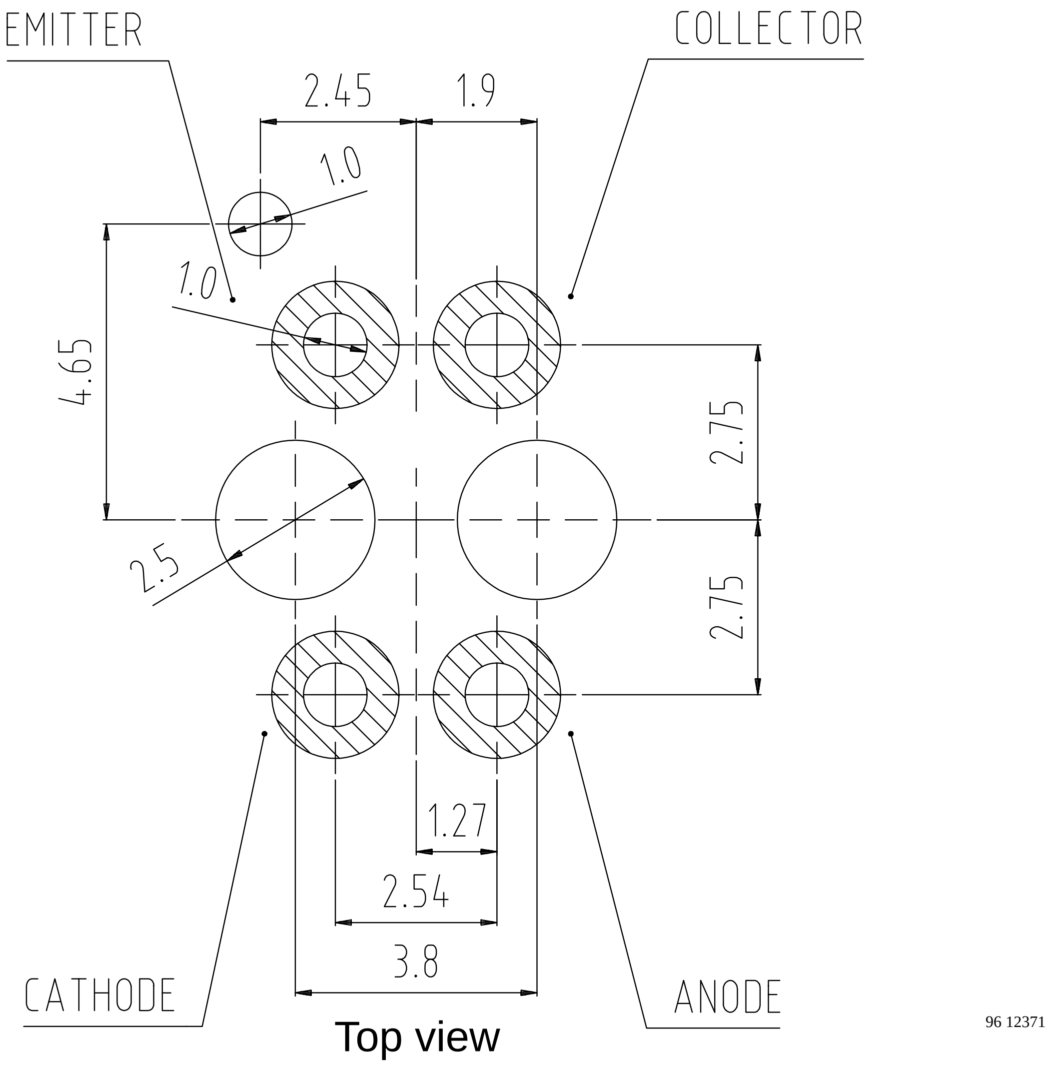
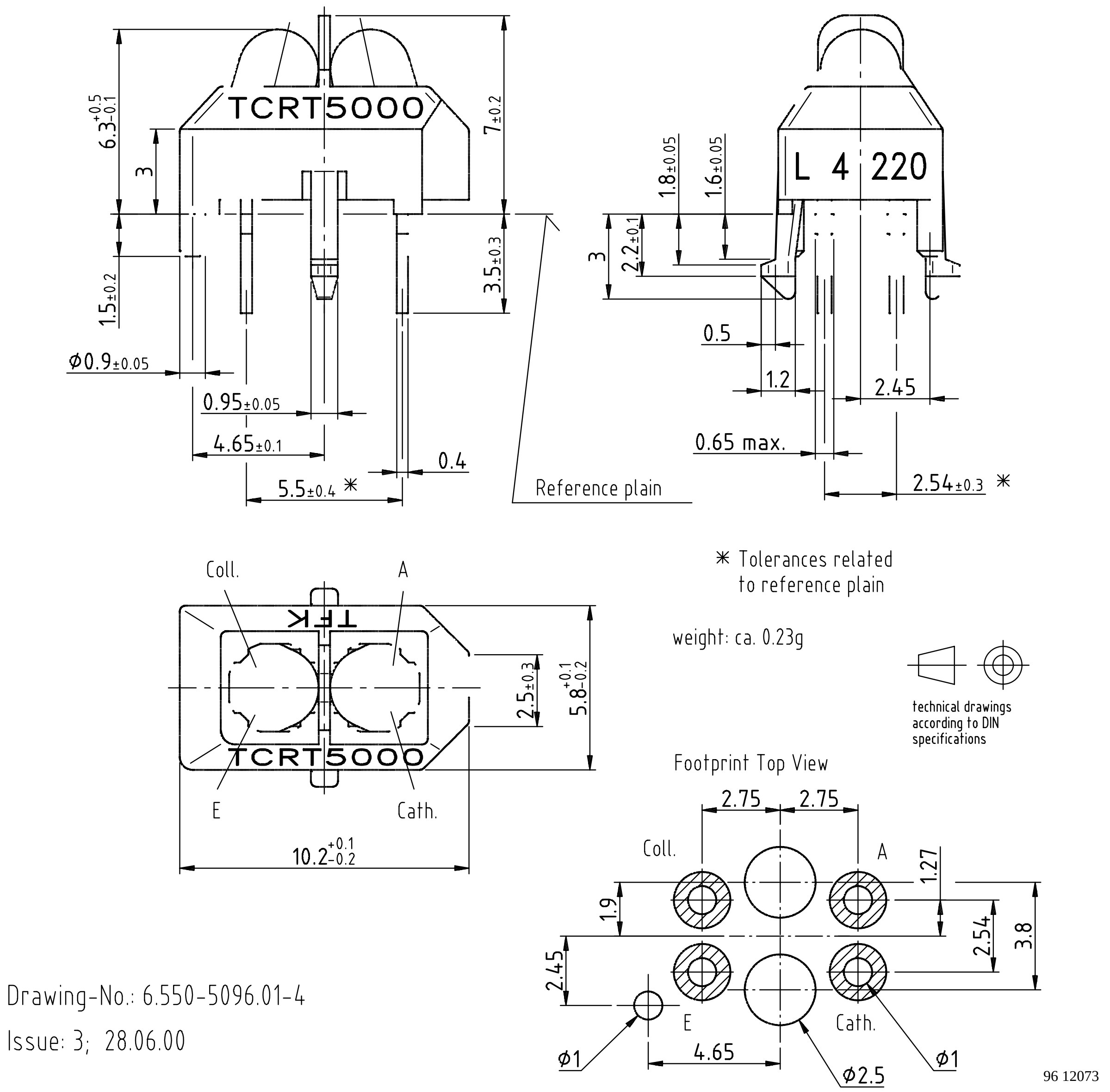
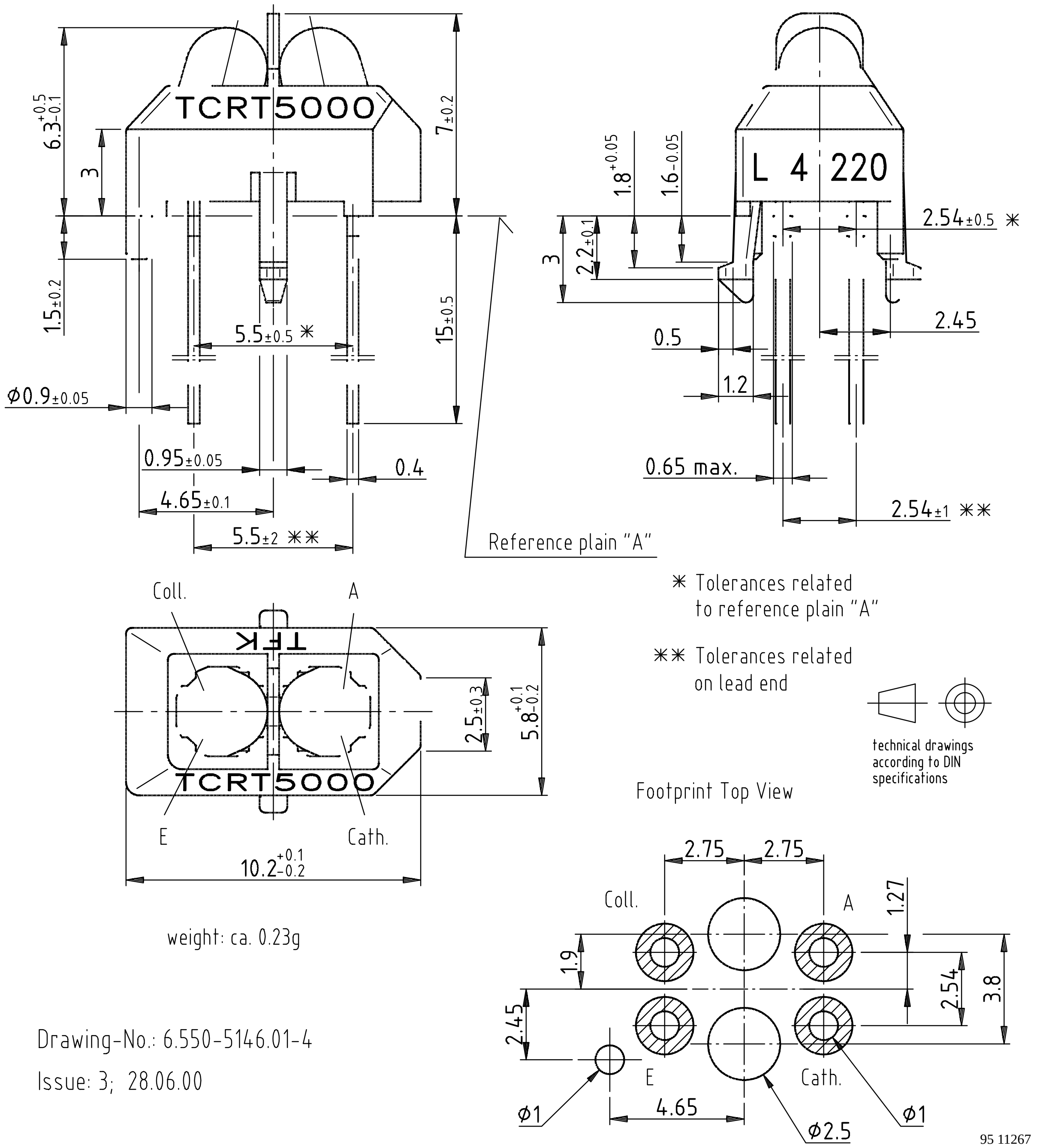


Figure 10. Footprint

Dimensions of TCRT5000 in mm



Dimensions of TCRT5000L in mm



Ozone Depleting Substances Policy Statement

It is the policy of **Vishay Semiconductor GmbH** to

1. Meet all present and future national and international statutory requirements.

2. Regularly and continuously improve the performance of our products, processes, distribution and operating systems with respect to their impact on the health and safety of our employees and the public, as well as their impact on the environment.

It is particular concern to control or eliminate releases of those substances into the atmosphere which are known as ozone depleting substances (ODSs).

The Montreal Protocol (1987) and its London Amendments (1990) intend to severely restrict the use of ODSs and forbid their use within the next ten years. Various national and international initiatives are pressing for an earlier ban on these substances.

Vishay Semiconductor GmbH has been able to use its policy of continuous improvements to eliminate the use of ODSs listed in the following documents.

1. Annex A, B and list of transitional substances of the Montreal Protocol and the London Amendments respectively

2. Class I and II ozone depleting substances in the Clean Air Act Amendments of 1990 by the Environmental Protection Agency (EPA) in the USA

3. Council Decision 88/540/EEC and 91/690/EEC Annex A, B and C (transitional substances) respectively.

Vishay Semiconductor GmbH can certify that our semiconductors are not manufactured with ozone depleting substances and do not contain such substances.

Vishay Semiconductor GmbH, P.O.B. 3535, D-74025 Heilbronn, Germany
Telephone: 49 (0)7131 67 2831, Fax number: 49 (0)7131 67 2423

DATA SHEET

For a complete data sheet, please also download:

- The IC06 74HC/HCT/HCU/HCMOS Logic Family Specifications
- The IC06 74HC/HCT/HCU/HCMOS Logic Package Information
- The IC06 74HC/HCT/HCU/HCMOS Logic Package Outlines

74HC/HCT00
Quad 2-input NAND gate

Product specification
File under Integrated Circuits, IC06

December 1990

Philips
Semiconductors

PHILIPS

Quad 2-input NAND gate — 74HC/HCT00

FEATURES

- Output capability: standard
- I_{CC} category: SSI

GENERAL DESCRIPTION

The 74HC/HCT00 are high-speed Si-gate CMOS devices and are pin compatible with low power Schottky TTL (LSTTL). They are specified in compliance with JEDEC standard no. 7A.

The 74HC/HCT00 provide the 2-input NAND function.

QUICK REFERENCE DATA

GND = 0 V; T_{amb} = 25 °C; $t_r = t_f$ = 6 ns

SYMBOL	PARAMETER	CONDITIONS	TYPICAL		UNIT
			HC	HCT	
t_{PHL}/ t_{PLH}	propagation delay nA, nB to nY	C_L = 15 pF; V_{CC} = 5 V	7	10	ns
C_I	input capacitance		3.5	3.5	pF
C_{PD}	power dissipation capacitance per gate	notes 1 and 2	22	22	pF

Notes

1. C_{PD} is used to determine the dynamic power dissipation (P_D in μW):

   $P_D = C_{PD} \times V_{CC}^2 \times f_i + \sum (C_L \times V_{CC}^2 \times f_o)$ where:

   f_i = input frequency in MHz

   f_o = output frequency in MHz

   C_L = output load capacitance in pF

   V_{CC} = supply voltage in V

   $\sum (C_L \times V_{CC}^2 \times f_o)$ = sum of outputs

2. For HC the condition is V_I = GND to V_{CC}
   For HCT the condition is V_I = GND to V_{CC} – 1.5 V

ORDERING INFORMATION

See *"74HC/HCT/HCU/HCMOS Logic Package Information"*.

PIN DESCRIPTION

PIN NO.	SYMBOL	NAME AND FUNCTION
1, 4, 9, 12	1A to 4A	data inputs
2, 5, 10, 13	1B to 4B	data inputs
3, 6, 8, 11	1Y to 4Y	data outputs
7	GND	ground (0 V)
14	V_{CC}	positive supply voltage

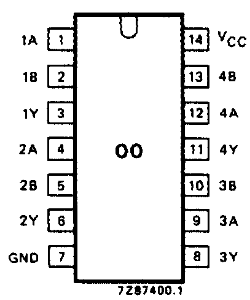


Fig.1 Pin configuration.

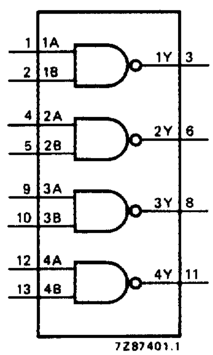


Fig.2 Logic symbol.

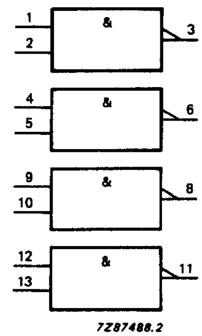


Fig.3 IEC logic symbol.

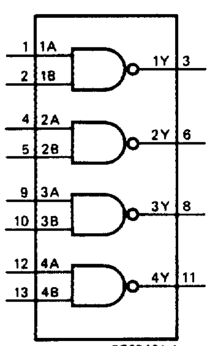


Fig.4 Functional diagram.

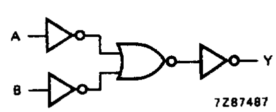


Fig.5 Logic diagram (one gate).

FUNCTION TABLE

INPUTS		OUTPUT
nA	nB	nY
L	L	H
L	H	H
H	L	H
H	H	L

Note

1. H = HIGH voltage level
   L = LOW voltage level

DC CHARACTERISTICS FOR 74HC

For the DC characteristics see *"74HC/HCT/HCU/HCMOS Logic Family Specifications"*.

Output capability: standard
I_{CC} category: SSI

AC CHARACTERISTICS FOR 74HC

GND = 0 V; $t_r = t_f = 6$ ns; $C_L = 50$ pF

SYMBOL	PARAMETER	T_{amb} (°C)							UNIT	TEST CONDITIONS	
		74HC								V_{CC} (V)	WAVEFORMS
		+25			−40 to +85		−40 to +125				
		min.	typ.	max.	min.	max.	min.	max.			
t_{PHL}/ t_{PLH}	propagation delay nA, nB to nY		25 9 7	90 18 15		115 23 20		135 27 23	ns	2.0 4.5 6.0	Fig.6
t_{THL}/ t_{TLH}	output transition time		19 7 6	75 15 13		95 19 16		110 22 19	ns	2.0 4.5 6.0	Fig.6

DC CHARACTERISTICS FOR 74HCT

For the DC characteristics see *"74HC/HCT/HCU/HCMOS Logic Family Specifications"*.

Output capability: standard
I_{CC} category: SSI

Note to HCT types

The value of additional quiescent supply current (ΔI_{CC}) for a unit load of 1 is given in the family specifications.
To determine ΔI_{CC} per input, multiply this value by the unit load coefficient shown in the table below.

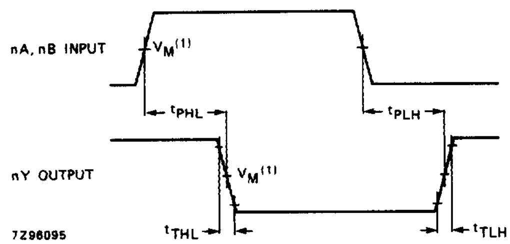
INPUT	UNIT LOAD COEFFICIENT
nA, nB	1.50

AC CHARACTERISTICS FOR 74HCT

GND = 0 V; $t_r = t_f = 6$ ns; $C_L = 50$ pF

SYMBOL	PARAMETER	T_{amb} (°C)							UNIT	TEST CONDITIONS	
		74HCT								V_{CC} (V)	WAVEFORMS
		+25			−40 to +85		−40 to +125				
		min.	typ.	max.	min.	max.	min.	max.			
t_{PHL}/ t_{PLH}	propagation delay nA, nB to nY		12	19		24		29	ns	4.5	Fig.6
t_{THL}/ t_{TLH}	output transition time		7	15		19		22	ns	4.5	Fig.6

AC WAVEFORMS



HC : V_M = 50%; V_I = GND to V_{CC}
HCT: V_M = 1.3 V; V_I = GND to 3 V.

Fig.6 Waveforms showing the input (nA, nB) to output (nY) propagation delays and the output transition times.

PACKAGE OUTLINES

See *"74HC/HCT/HCU/HCMOS Logic Package Outlines"*.

New Industrial Alkaline Graphics

Now they look as good as they perform

Panasonic's Industrial product line features bold, colorful new graphics that work as hard as the batteries themselves.

Panasonic Industrial brand alkaline batteries are dependable, long-lasting batteries that are specially labeled and commercially packaged for end users, businesses, organizations, agencies and distributors in the OEM/professional/industrial marketplace.

Enhanced features include:

- **Bright, colorful, attention-grabbing design**
- **Easier to read**
- **Clearly marked, "Industrial alkaline, not for retail trade"**
- **English and French text**
- **EU compliant dustbin symbol**
- **Expiration date code easy to locate and read**
- **Easier cross referencing with IEC battery codes***

*International Electrotechnical Commission

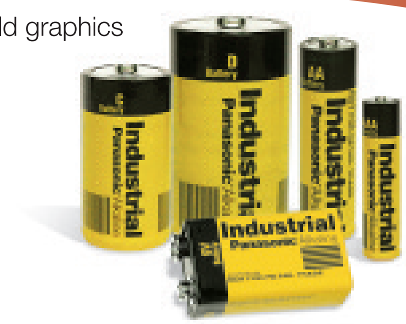
Panasonic
ideas for life

Panasonic Industrial Company - OEM Battery Sales Group
www.panasonic.com/batteries
e-mail: oembatteries@us.panasonic.com

New Industrial Alkaline Graphics

A bold new look that sets the standard for form and function

Old graphics



The new industrial label has many important features

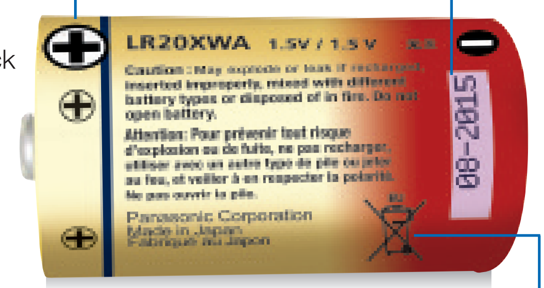
Bold, impactful design

More distinct polarity marking

Clearly marked date codes provide expiration information

Front

Back



Features English and French languages

EU Compliant - Dustbin Symbol indicates batteries must be collected for recycling

Size	Old Part Number	New Part Number
D	AM-1PI	LR20XWA
C	AM-2PI	LR14XWA
AA	AM-3PI	LR6XWA
AAA	AM-4PI	LR03XWA
9V	6AM-6PI	6LR61XWA

The Power of Tomorrow - Today

Panasonic
ideas for life

Panasonic Industrial Company - OEM Battery Sales Group
www.panasonic.com/batteries
e-mail: oembatteries@us.panasonic.com

Darlington Complementary Silicon Power Transistors

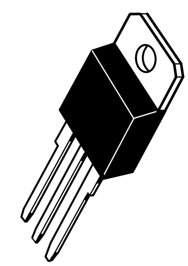
. . . designed for general–purpose amplifier and low frequency switching applications.

- High DC Current Gain — Min h_{FE} = 1000 @ I_C = 5 A, V_{CE} = 4 V
- Collector–Emitter Sustaining Voltage — @ 30 mA
  $V_{CEO(sus)}$ = 60 Vdc (Min) — TIP140, TIP145
  80 Vdc (Min) — TIP141, TIP146
  100 Vdc (Min) — TIP142, TIP147
- Monolithic Construction with Built–In Base–Emitter Shunt Resistor

NPN
TIP140
TIP141*
TIP142*
PNP
TIP145
TIP146*
TIP147*

*Motorola Preferred Device

**10 AMPERE
DARLINGTON
COMPLEMENTARY SILICON
POWER TRANSISTORS
60 – 100 VOLTS
125 WATTS**



CASE 340D–01

MAXIMUM RATINGS

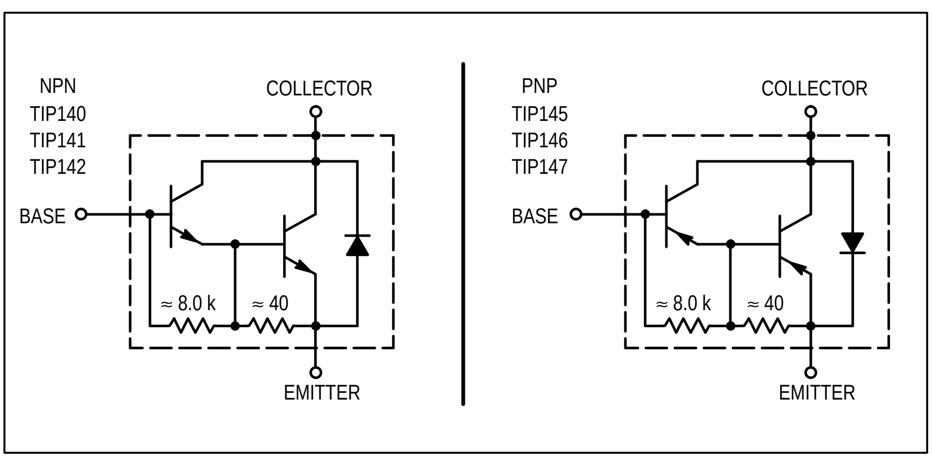
Rating	Symbol	TIP140 TIP145	TIP141 TIP146	TIP142 TIP147	Unit
Collector–Emitter Voltage	V_{CEO}	60	80	100	Vdc
Collector–Base Voltage	V_{CB}	60	80	100	Vdc
Emitter–Base Voltage	V_{EB}	5.0			Vdc
Collector Current — Continuous Peak (1)	I_C	10 15			Adc
Base Current — Continuous	I_B	0.5			Adc
Total Device Dissipation @ T_C = 25°C	P_D	125			Watts
Operating and Storage Junction Temperature Range	T_J, T_{stg}	–65 to +150			°C

THERMAL CHARACTERISTICS

Characteristic	Symbol	Max	Unit
Thermal Resistance, Junction to Case	$R_{\theta JC}$	1.0	°C/W
Thermal Resistance, Case to Ambient	$R_{\theta JA}$	35.7	°C/W

(1) 5 ms, ≤ 10% Duty Cycle.

DARLINGTON SCHEMATICS

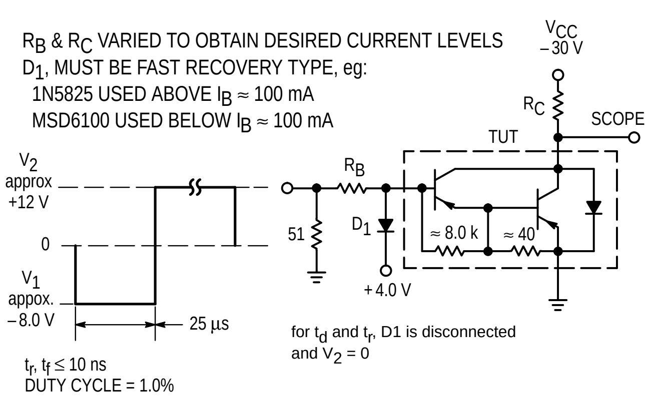


Preferred devices are Motorola recommended choices for future use and best overall value.

ELECTRICAL CHARACTERISTICS (T_C = 25°C unless otherwise noted)

Characteristic		Symbol	Min	Typ	Max	Unit
OFF CHARACTERISTICS						
Collector–Emitter Sustaining Voltage (1) (I_C = 30 mA, I_B = 0)	TIP140, TIP145 TIP141, TIP146 TIP142, TIP147	$V_{CEO(sus)}$	60 80 100	— — —	— — —	Vdc
Collector Cutoff Current (V_{CE} = 30 Vdc, I_B = 0) (V_{CE} = 40 Vdc, I_B = 0) (V_{CE} = 50 Vdc, I_B = 0)	TIP140, TIP145 TIP141, TIP146 TIP142, TIP147	I_{CEO}	— — —	— — —	2.0 2.0 2.0	mA
Collector Cutoff Current (V_{CB} = 60 V, I_E = 0) (V_{CB} = 80 V, I_E = 0) (V_{CB} = 100 V, I_E = 0)	TIP140, TIP145 TIP141, TIP146 TIP142, TIP147	I_{CBO}	— — —	— — —	1.0 1.0 1.0	mA
Emitter Cutoff Current (V_{BE} = 5.0 V)		I_{EBO}	—	—	2 0	mA
ON CHARACTERISTICS (1)						
DC Current Gain (I_C = 5.0 A, V_{CE} = 4.0 V) (I_C = 10 A, V_{CE} = 4.0 V)		h_{FE}	1000 500	— —	— —	—
Collector–Emitter Saturation Voltage (I_C = 5.0 A, I_B = 10 mA) (I_C = 10 A, I_B = 40 mA)		$V_{CE(sat)}$	— —	— —	2.0 3.0	Vdc
Base–Emitter Saturation Voltage (I_C = 10 A, I_B = 40 mA)		$V_{BE(sat)}$	—	—	3.5	Vdc
Base–Emitter On Voltage (I_C = 10 A, V_{CE} = 4.0 Vdc)		$V_{BE(on)}$	—	—	3.0	Vdc
SWITCHING CHARACTERISTICS						
Resistive Load (See Figure 1)						
Delay Time	(V_{CC} = 30 V, I_C = 5.0 A, I_B = 20 mA, Duty Cycle ≤ 2.0%, I_{B1} = I_{B2}, R_C & R_B Varied, T_J = 25°C)	t_d	—	0.15	—	μs
Rise Time		t_r	—	0.55	—	μs
Storage Time		t_s	—	2.5	—	μs
Fall Time		t_f	—	2.5	—	μs

(1) Pulse Test: Pulse Width = 300 μs, Duty Cycle ≤ 2.0%.



For NPN test circuit reverse diode and voltage polarities.

Figure 1. Switching Times Test Circuit

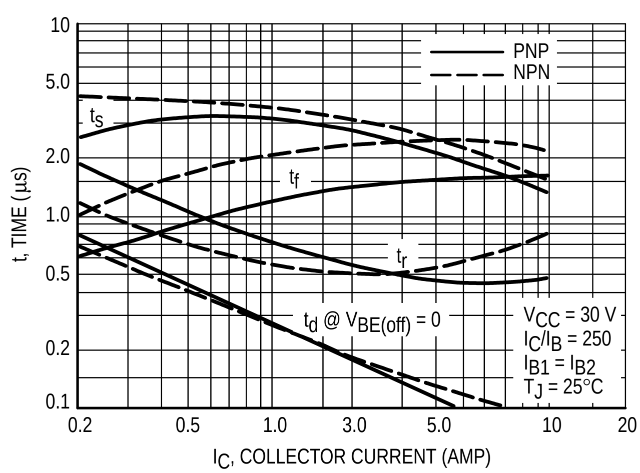


Figure 2. Switching Times

TYPICAL CHARACTERISTICS

NPN
TIP140, TIP141, TIP142

PNP
TIP145, TIP146, TIP147

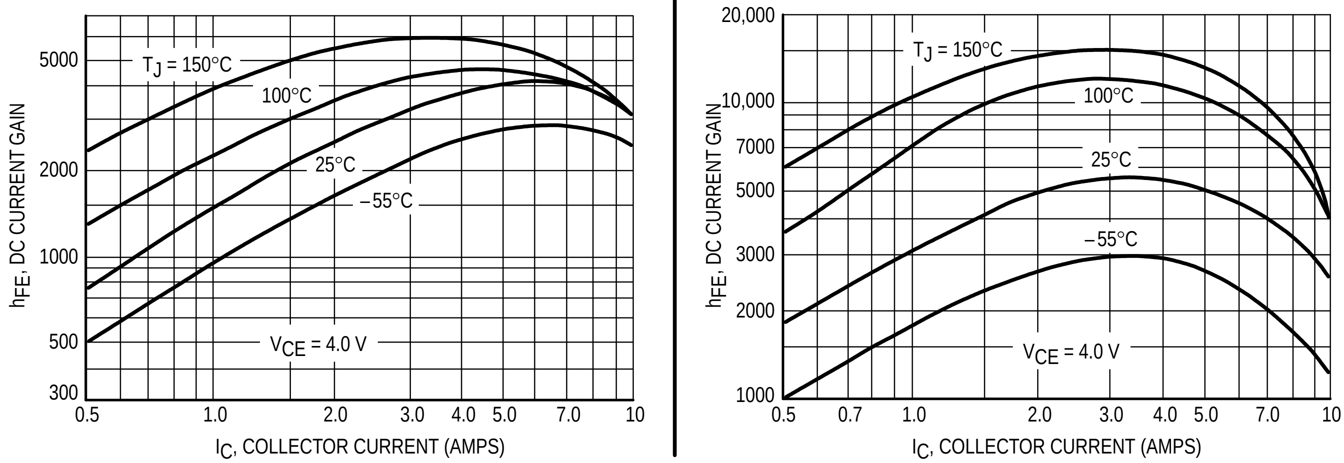


Figure 3. DC Current Gain versus Collector Current

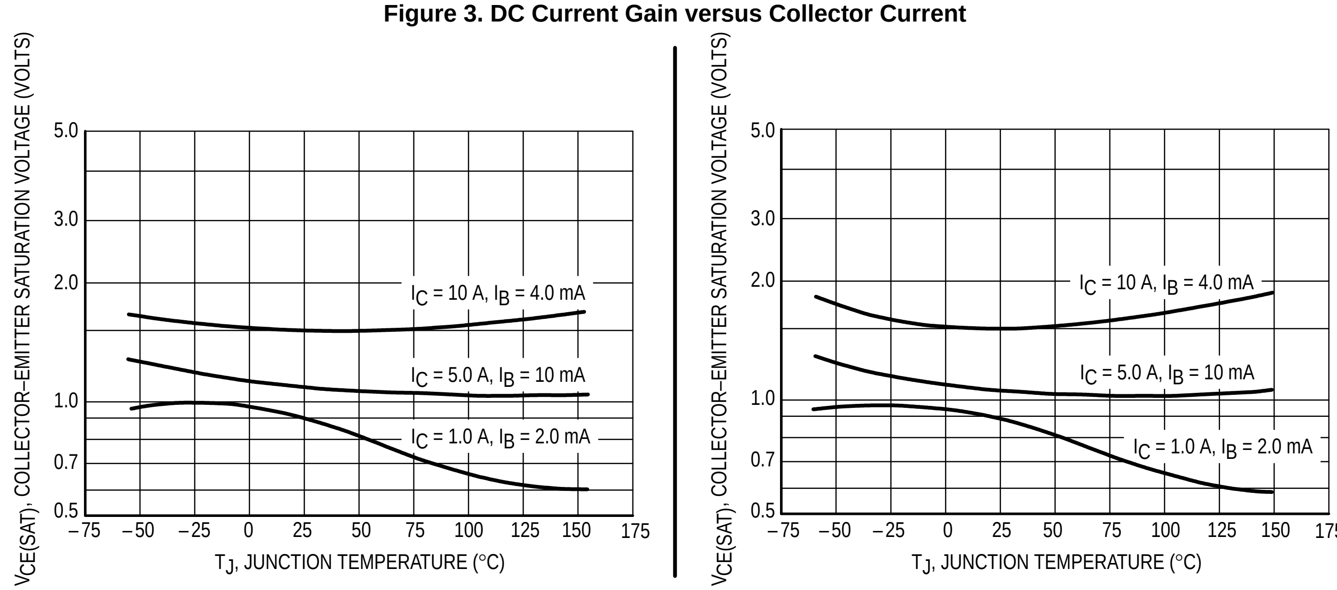


Figure 4. Collector−Emitter Saturation Voltage

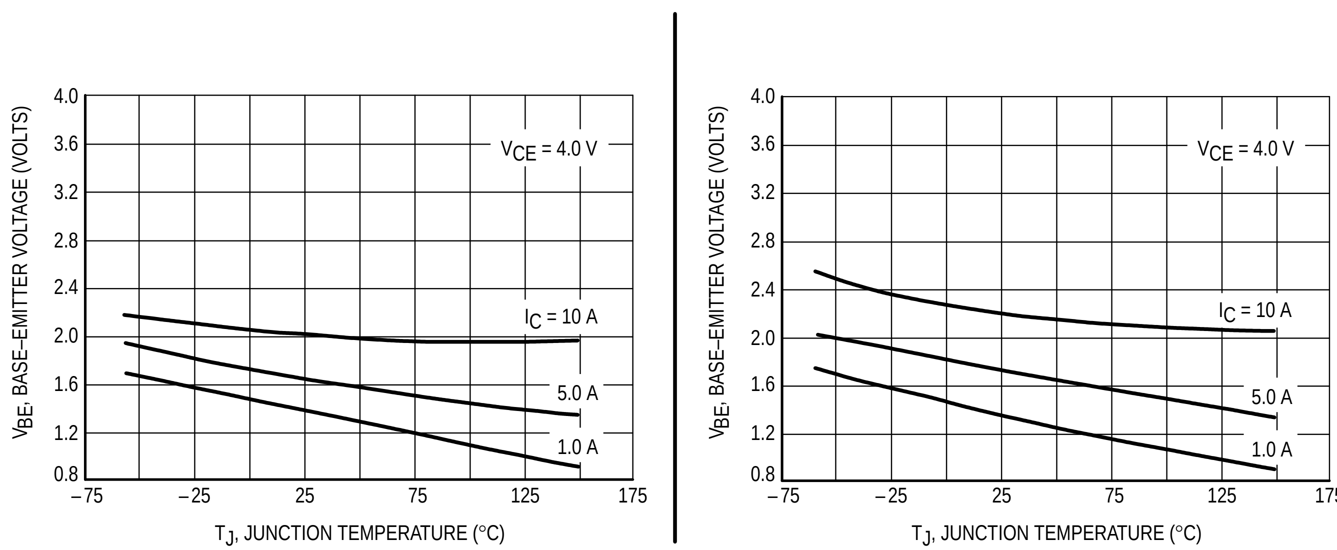


Figure 5. Base−Emitter Voltage

ACTIVE–REGION SAFE OPERATING AREA

There are two limitations on the power handling ability of a transistor: average junction temperature and second breakdown. Safe operating area curves indicate I_C – V_{CE} limits of the transistor that must be observed for reliable operation; i.e., the transistor must not be subjected to greater dissipation than the curves indicate.

The data of Figure 6 is based on $T_{J(pk)}$ = 150°C; T_C is variable depending on conditions. At high case temperatures, thermal limitations will reduce the power that can be handled to values less than the limitations imposed by second breakdown.

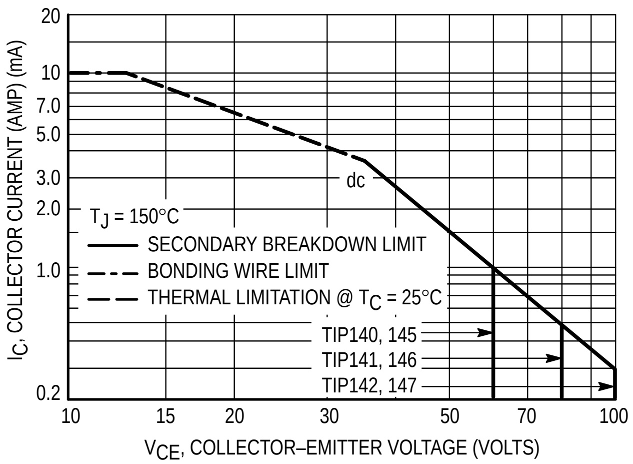


Figure 6. Active–Region Safe Operating Area

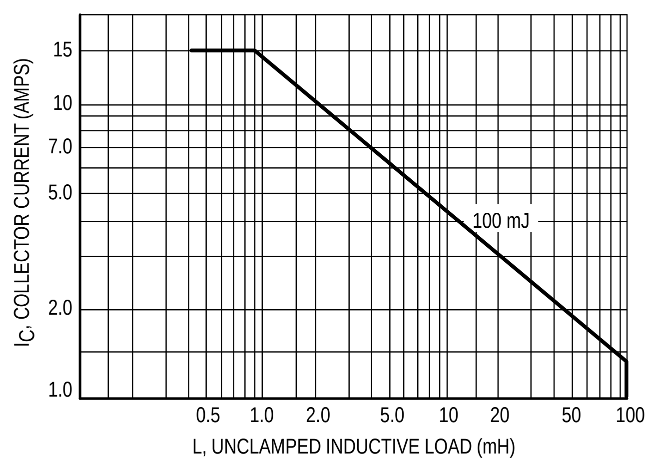
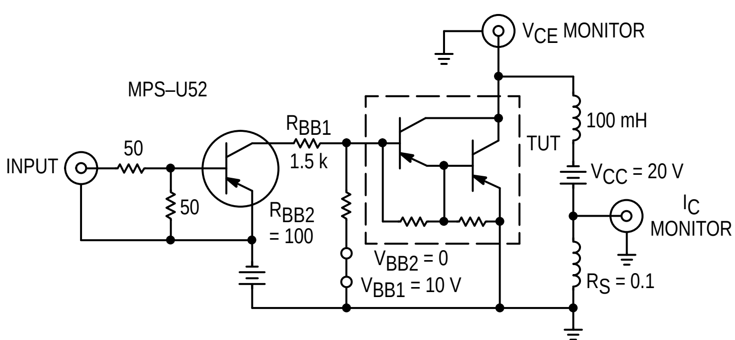
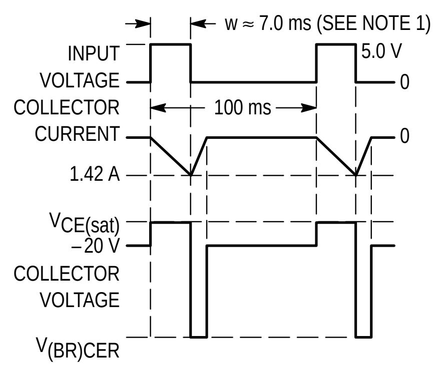


Figure 7. Unclamped Inductive Load



TEST CIRCUIT

NOTE 1: Input pulse width is increased until I_{CM} = 1.42 A.
NOTE 2: For NPN test circuit reverse polarities.



VOLTAGE AND CURRENT WAVEFORMS

Figure 8. Inductive Load

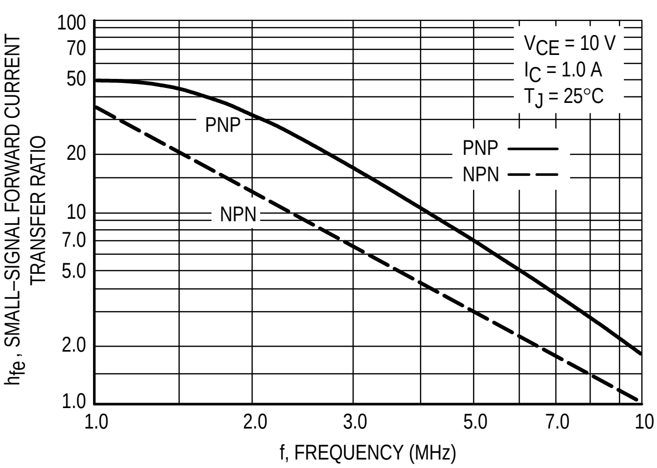


Figure 9. Magnitude of Common Emitter Small–Signal Short–Circuit Forward Current Transfer Ratio

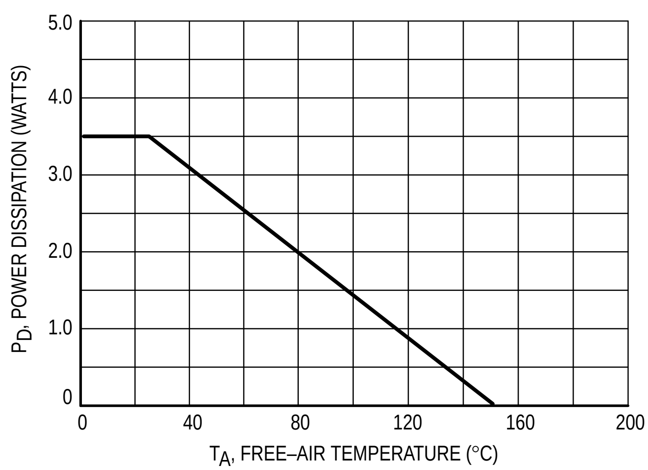
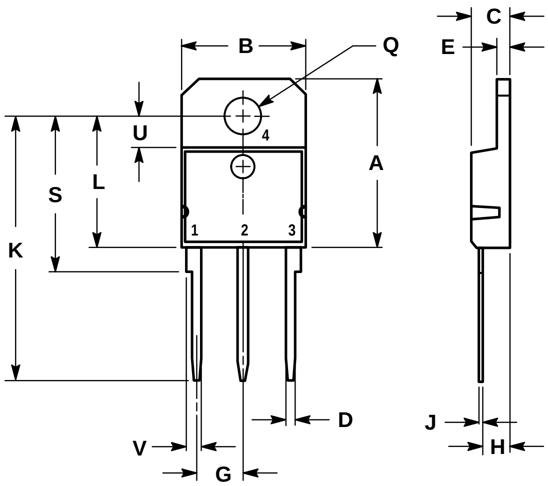


Figure 10. Free–Air Temperature Power Derating

PACKAGE DIMENSIONS



NOTES:
1. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982.
2. CONTROLLING DIMENSION: MILLIMETER.

	MILLIMETERS		INCHES	
DIM	MIN	MAX	MIN	MAX
A	19.00	19.60	0.749	0.771
B	14.00	14.50	0.551	0.570
C	4.20	4.70	0.165	0.185
D	1.00	1.30	0.040	0.051
E	1.45	1.65	0.058	0.064
G	5.21	5.72	0.206	0.225
H	2.60	3.00	0.103	0.118
J	0.40	0.60	0.016	0.023
K	28.50	32.00	1.123	1.259
L	14.70	15.30	0.579	0.602
Q	4.00	4.25	0.158	0.167
S	17.50	18.10	0.689	0.712
U	3.40	3.80	0.134	0.149
V	1.50	2.00	0.060	0.078

STYLE 1:
PIN 1. BASE
2. COLLECTOR
3. EMITTER
4. COLLECTOR

CASE 340D–01
TO–218AC
ISSUE A

How to reach us:
USA / EUROPE: Motorola Literature Distribution;
P.O. Box 20912; Phoenix, Arizona 85036. 1–800–441–2447

MFAX: RMFAX0@email.sps.mot.com – TOUCHTONE (602) 244–6609
INTERNET: http://Design–NET.com

JAPAN: Nippon Motorola Ltd.; Tatsumi–SPD–JLDC, Toshikatsu Otsuki,
6F Seibu–Butsuryu–Center, 3–14–2 Tatsumi Koto–Ku, Tokyo 135, Japan. 03–3521–8315

HONG KONG: Motorola Semiconductors H.K. Ltd.; 8B Tai Ping Industrial Park,
51 Ting Kok Road, Tai Po, N.T., Hong Kong. 852–26629298

INTEGRATED CIRCUITS

DATA SHEET

LM139/239/239A/339/339A/LM2901/MC 3302

Quad voltage comparator

Product specification

1995 Nov 27

IC11 Data Handbook

Philips Semiconductors

PHILIPS

DESCRIPTION

The LM139 series consists of four independent precision voltage comparators, with an offset voltage specification as low as 2.0mV max for each comparator, which were designed specifically to operate from a single power supply over a wide range of voltages. Operation from split power supplies is also possible and the low power supply current drain is independent of the magnitude of the power supply voltage. These comparators also have a unique characteristic in that the input common–mode voltage range includes ground, even though they are operated from a single power supply voltage.

The LM139 series was designed to directly interface with TTL and CMOS. When operated from both plus and minus power supplies, the LM139 series will directly interface with MOS logic where their low power drain is a distinct advantage over standard comparators.

FEATURES

- Wide single supply voltage range 2.0V_{DC} to 36V_{DC} or dual supplies ±1.0V_{DC} to ±18V_{DC}
- Very low supply current drain (0.8mA) independent of supply voltage (1.0mW/comparator at 5.0V_{DC})
- Low input biasing current 25nA
- Low input offset current ±5nA and offset voltage
- Input common–mode voltage range includes ground
- Differential input voltage range equal to the power supply voltage
- Low output 250mV at 4mA saturation voltage
- Output voltage compatible with TTL, DTL, ECL, MOS and CMOS logic systems

APPLICATIONS

- A/D converters
- Wide range VCO
- MOS clock generator
- High voltage logic gate
- Multivibrators

PIN CONFIGURATION

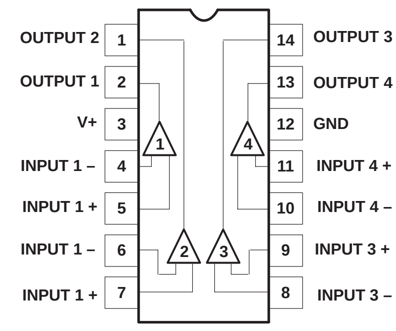


Figure 1. Pin Configuration

EQUIVALENT CIRCUIT

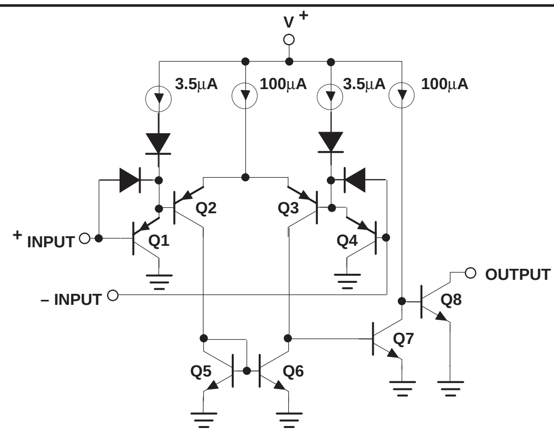


Figure 2. Equivalent Circuit

ORDERING INFORMATION

DESCRIPTION	TEMPERATURE RANGE	ORDER CODE	DWG #
14–Pin Ceramic Dual In-Line Package (Cerdip)	–55 to +125°C	LM139F	0581B
14–Pin Plastic Dual In-Line Package (DIP)	–25°C to +85°C	LM239AN	SOT27-1
14–Pin Plastic Dual In-Line Package (DIP)	–25°C to +85°C	LM239N	SOT27-1
14–Pin Plastic Small Outline (SO) Package	–25°C to +85°C	LM239D	SOT108-1
14–Pin Plastic Dual In-Line Package (DIP)	–40°C to +125°C	LM2901N	SOT27-1
14–Pin Plastic Small Outline (SO) Package	–40°C to +125°C	LM2901D	SOT108-1
14–Pin Plastic Dual In-Line Package (DIP)	0 to +70°C	LM339AN	SOT27-1
14–Pin Plastic Small Outline (SO) Package	0 to +70°C	LM339D	SOT108-1
14–Pin Plastic Dual In-Line Package (DIP)	0 to +70°C	LM339N	SOT27-1
14–Pin Plastic Small Outline (SO) Package	–40°C to +85°C	MC3302D	SOT108-1
14–Pin Ceramic Dual In-Line Package (Cerdip)	–40°C to +85°C	MC3302F	0581B
14–Pin Plastic Dual In-Line Package (DIP)	–40°C to +85°C	MC3302N	SOT27-1
14–Pin Plastic Dual In-Line Package (DIP)	–55 to +125°C	LM139N	SOT27-1

ABSOLUTE MAXIMUM RATINGS

SYMBOL	PARAMETER	RATING	UNIT
V_{CC}	V_{CC} supply voltage	36 or ±18	V_{DC}
V_{DIFF}	Differential input voltage	36	V_{DC}
V_{IN}	Input voltage	–0.3 to +36	V_{DC}
P_D	Maximum power dissipation, T_A=25°C (still–air)[1]		
	F package	1190	mW
	N package	1420	mW
	D package	1040	mW
	Output short–circuit to ground[2]	Continuous	
I_{IN}	Input current (V_{IN}<–0.3V_{DC})[3]	50	mA
T_A	Operating temperature range		
	LM139	–55 to +125	°C
	LM239/239A	–25 to +85	°C
	LM339/339A	0 to +70	°C
	LM2901	–40 to +125	°C
	MC3302	–40 to +85	°C
T_{STG}	Storage temperature range	–65 to +150	°C
T_{SOLD}	Lead soldering temperature (10sec max)	300	°C

NOTES:
1. Derate above 25°C, at the following rates:
   F Package at 9.5mW/°C
   N Package at 11.4mW/°C
   D Package at 8.3mW/°C
2. Short circuits from the output to V+ can cause excessive heating and eventual destruction. The maximum output current is approximately 20mA independent of the magnitude of V+.
3. This input current will only exist when the voltage at any of the input leads is driven negative. It is due to the collector–base junction of the input PNP transistors becoming forward biased and thereby acting as input diode clamps. In addition to this diode action, there is also lateral NPN parasitic transistor action on the IC chip. This transistor action can cause the output voltages of the comparators to go to the V+ voltage level (or to ground for a large overdrive) for the time duration that an input is driven negative. This is not destructive and normal output states will reestablish when the input voltage, which was negative, again returns to a value greater than –0.3V_{DC}.

DC AND AC ELECTRICAL CHARACTERISTICS

V+=5V_{DC}, LM139: –55°C ≤ T_A ≤ 125°C; LM239/239A: –25°C ≤ T_A ≤ 85°C; LM339/339A: 0°C ≤ T_A ≤ 70°C;; LM2901: –40°C ≤ T_A ≤ 125°C, MC3302: –40°C ≤ T_A ≤ 85°C, unless otherwise specified.

SYMBOL	PARAMETER	TEST CONDITIONS	LM239A/339A			UNIT
			Min	Typ	Max	
V_{OS}	Input offset voltage[2]	T_A=25°C Over temp.		±1.0	±2.0 ±4.0	mV mV
V_{CM}	Input common–mode voltage range[3]	T_A=25°C Over temp.	0 0		V+–1.5 V+–2.0	V
V_{IDR}	Differential input voltage[1]	Keep all V_{IN}s≥0V_{DC} (or V– if need)			V+	V
I_{BIAS}	Input bias current[4]	$I_{IN(+)}$ or $I_{IN(-)}$ with output in linear range T_A=25°C Over temp.		25	250 400	nA nA
I_{OS}	Input offset current	$I_{IN(+)}$–$I_{IN(-)}$ T_A=25°C Over temp.		±5.0	±50 ±150	nA nA
I_{OL}	Output sink current	$V_{IN(-)}$≥1V_{DC}, V_{IN}(+)=0, V_O≤1.5V_{DC}, T_A=25°C	6.0	16		mA
	Output leakage current	$V_{IN(+)}$≥1V_{DC}, V_{IN}(–)=0 V_O=5V_{DC}, T_A=25°C V_O=30V_{DC}, over temp.		0.1	 1.0	nA µA
I_{CC}	Supply current	R_L=∞ on comparators, T_A=25°C V+=30V		0.8	2.0	mA
A_V	Voltage gain	R_L≥15kΩ, V+=15V_{DC}	50	200		V/mV
V_{OL}	Saturation voltage	$V_{IN(-)}$≥1V_{DC}, $V_{IN(+)}$=0, I_{SINK}≤4mA T_A=25°C Over temp.		250	400 700	mV mV
t_{LSR}	Large–signal response time	V_{IN}=TTL logic swing, V_{REF}=1.4V_{DC}, V_{RL}=5V_{DC}, R_L=5.1kΩ, T_A=25°C		300		ns
t_R	Response time[5]	V_{RL}=5V_{DC}, R_L=5.1kΩ, T_A=25°C		1.3		µs

See notes at the end of the Electrical Characteristics.

DC AND AC ELECTRICAL CHARACTERISTICS

$V+=5V_{DC}$, LM139: –55°C ≤ T_A ≤ 125°C; LM239/239A: –25°C ≤ T_A ≤ 85°C; LM339/339A: 0°C ≤ T_A ≤ 70°C; LM2901: –40°C ≤ T_A ≤ 125°C, MC3302: –40°C ≤ T_A ≤ 85°C, unless otherwise specified.

SYMBOL	PARAMETER	TEST CONDITIONS	LM139			LM239/339			UNIT
			Min	Typ	Max	Min	Typ	Max	
V_{OS}	Input offset voltage[2]	T_A=25°C Over temp.		±2.0	±5.0 ±9.0		±2.0	±5.0 ±9.0	mV mV
V_{CM}	Input common–mode voltage range[3]	T_A=25°C Over temp.	0 0		V+–1.5 V+–2.0	0 0		V+–1.5 V+–2.0	V
V_{IDR}	Differential input voltage[1]	Keep all V_{IN}s≥0V_{DC} (or V– if need)			V+			V+	V
I_{BIAS}	Input bias current[4]	$I_{IN(+)}$ or $I_{IN(–)}$ with output in linear range T_A=25°C Over temp.		25	100 300		25	250 400	nA nA
I_{OS}	Input offset current	$I_{IN(+)}$–$I_{IN(–)}$ T_A=25°C Over temp.		±3.0	±25 ±100		±5.0	±50 ±150	nA nA
I_{OL}	Output sink current	$V_{IN(–)}$≥1V_{DC}, V_{IN}(+)=0, V_O≤1.5V_{DC}, T_A=25°C	6.0	16		6.0	16		mA
	Output leakage current	$V_{IN(+)}$≥1V_{DC}, V_{IN}(–)=0 V_O=5V_{DC}, T_A=25°C V_O=30V_{DC}, over temp.		0.1	 1.0		0.1	 1.0	nA µA
I_{CC}	Supply current	R_L=∞ on comparators, T_A=25°C V+=30V		0.8	2.0		0.8	2.0	mA
A_V	Voltage gain	R_L≥15kΩ, V+=15V_{DC}	50	200		50	200		V/mV
V_{OL}	Saturation voltage	$V_{IN(–)}$≥1V_{DC}, $V_{IN(+)}$=0, I_{SINK}≤4mA T_A=25°C Over temp.		250	400 700		250	400 700	mV mV
t_{LSR}	Large–signal response time	V_{IN}=TTL logic swing, V_{REF}=1.4V_{DC}, V_{RL}=5V_{DC}, R_L=5.1kΩ, T_A=25°C		300			300		ns
t_R	Response time[5]	V_{RL}=5V_{DC}, R_L=5.1kΩ, T_A=25°C		1.3			1.3		µs

See notes on following page.

DC AND AC ELECTRICAL CHARACTERISTICS

$V+=5V_{DC}$, LM139: –55°C ≤ T_A ≤ 125°C; LM239/239A: –25°C ≤ T_A ≤ 85°C; LM339/339A: 0°C ≤ T_A ≤ 70°C; LM2901: –40°C ≤ T_A ≤ 125°C, MC3302: –40°C ≤ T_A ≤ 85°C, unless otherwise specified.

SYMBOL	PARAMETER	TEST CONDITIONS	LM2901			MC3302			UNIT
			Min	Typ	Max	Min	Typ	Max	
V_{OS}	Input offset voltage[2]	T_A=25°C Over temp.		±2.0 ±9	±7.0 ±15		±3.0	±20 ±40	mV mV
V_{CM}	Input common–mode voltage range[3]	T_A=25°C Over temp.	0 0		V+–1.5 V+–2.0	0 0		V+–1.5 V+–2.0	V
V_{IDR}	Differential input voltage[1]	Keep all V_{IN}s≥$0V_{DC}$ (or V– if need)			V+			V+	V
I_{BIAS}	Input bias current[4]	$I_{IN(+)}$ or $I_{IN(-)}$ with output in linear range T_A=25°C Over temp.		25 200	250 500		25	500 1000	nA nA
I_{OS}	Input offset current	$I_{IN(+)}–I_{IN(-)}$ T_A=25°C Over temp.		±5 ±50	±50 ±200		±5	±100 ±300	nA nA
I_{OL}	Output sink current	$V_{IN(-)}≥1V_{DC}$, $V_{IN}(+)=0$, $V_O≤1.5V_{DC}$, T_A=25°C	6.0	16		6	16		mA
	Output leakage current	$V_{IN(+)}≥1V_{DC}$, $V_{IN}(-)=0$ $V_O=5V_{DC}$, T_A=25°C $V_O=30V_{DC}$, over temp.		0.1	1.0		0.1	1.0	nA μA
I_{CC}	Supply current	R_L=∞ on all comparators,					.8	1.8	mA
		T_A=25°C		0.8	2.0				
		R_L=∞ on all comparators, V+=30V		1.0	2.5				mA
A_V	Voltage gain	R_L≥15kΩ, $V+=15V_{DC}$	25	100		2	100		V/mV
V_{OL}	Saturation voltage	$V_{IN(-)}≥1V_{DC}$, $V_{IN(+)}=0$, $I_{SINK}≤4mA$ T_A=25°C Over temp.		400	400 700		150	400 700	mV mV
t_{LSR}	Large–signal response time	V_{IN}=TTL logic swing, $V_{REF}=1.4V_{DC}$, $V_{RL}=5V_{DC}$, R_L=5.1kΩ, T_A=25°C		300			300		ns
t_R	Response time[5]	$V_{RL}=5V_{DC}$, R_L=5.1kΩ, T_A=25°C		1.3			1.3		μs

NOTES:

1. Positive excursions of input voltage may exceed the power supply level by 17V. As long as the other voltage remains within the common-mode range, the comparator will provide a proper output state. The low input voltage state must not be less than $–0.3V_{DC}$ (or $0.3V_{DC}$ below the magnitude of the negative power supply, if used).
2. At output switch point, $V_O ≈ 1.4V_{DC}$, R_S=0Ω with V+ from $5V_{DC}$ to $30V_{DC}$; and over the full input common–mode range ($0V_{DC}$ to V+ – $1.5V_{DC}$). Inputs of unused comparators should be grounded.
3. The input common–mode voltage or either input signal voltage should not be allowed to go negative by more than 0.3V. The upper end of the common–mode voltage range is V+ – 1.5V, but either or both inputs can go to $30V_{DC}$ without damage.
4. The direction of the input current is out of the IC due to the PNP input stage. This current is essentially constant, independent of the state of the output so no loading change exists on the reference or input lines.
5. The response time specified is for a 100mV input step with a 5mV overdrive. For larger overdrive signals, 300ns can be obtained (see typical performance characteristics section).

EQUIVALENT CIRCUIT

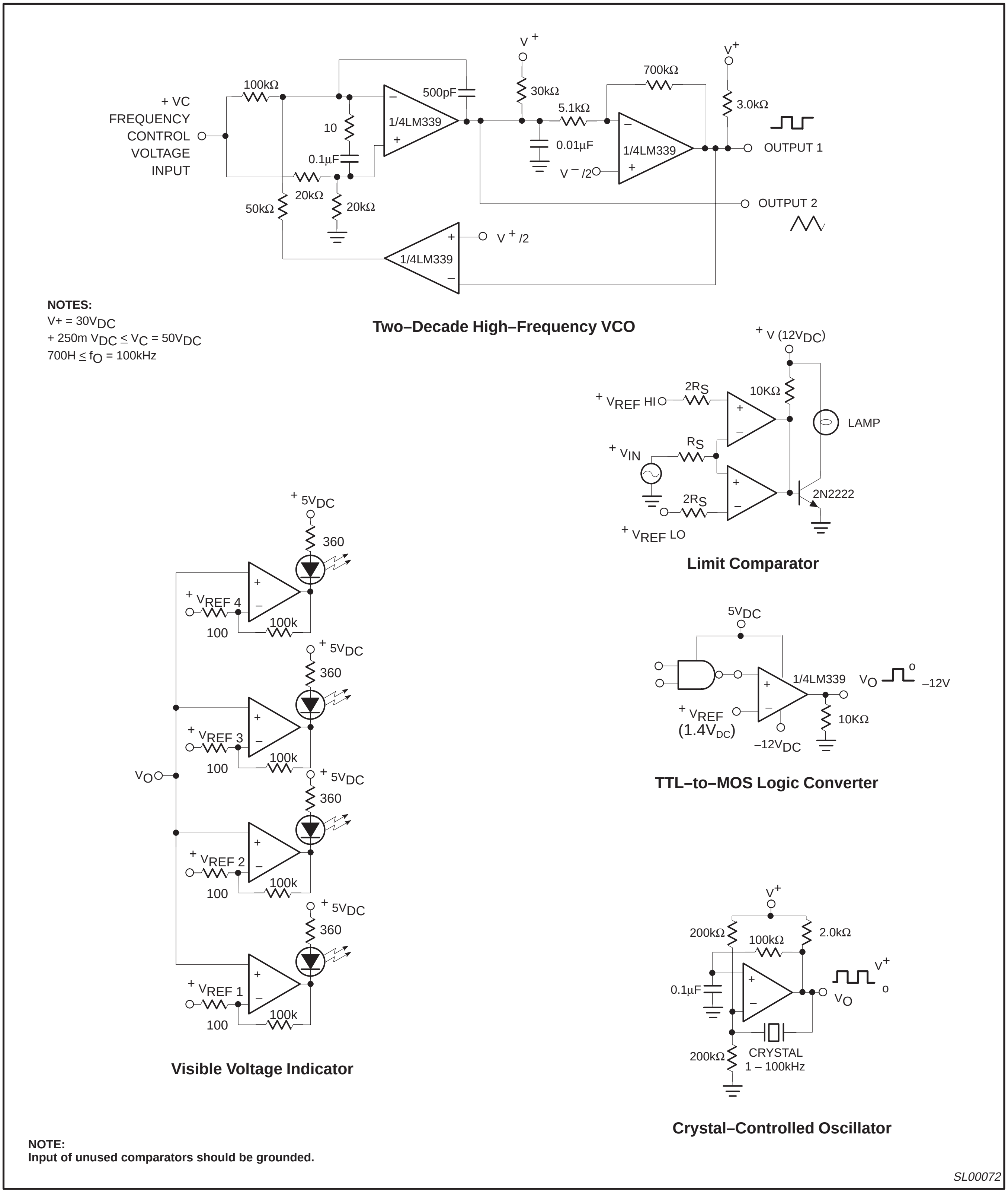


Figure 3. Equivalent Circuit

TYPICAL PERFORMANCE CHARACTERISTICS

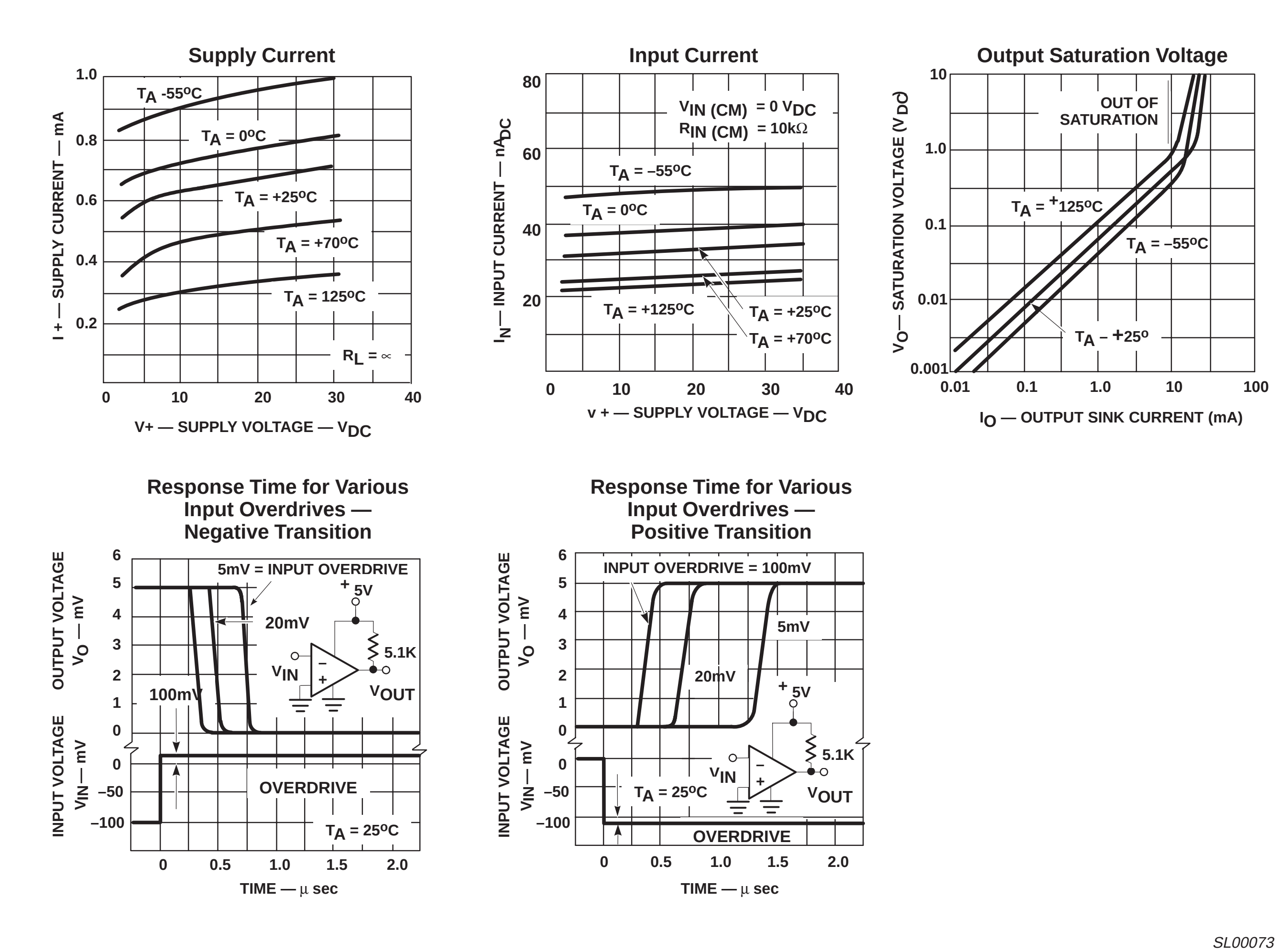


Figure 4. Typical Performance Characteristics

DEFINITIONS

Data Sheet Identification	Product Status	Definition
Objective Specification	**Formative or in Design**	This data sheet contains the design target or goal specifications for product development. Specifications may change in any manner without notice.
Preliminary Specification	**Preproduction Product**	This data sheet contains preliminary data, and supplementary data will be published at a later date. Philips Semiconductors reserves the right to make changes at any time without notice in order to improve design and supply the best possible product.
Product Specification	**Full Production**	This data sheet contains Final Specifications. Philips Semiconductors reserves the right to make changes at any time without notice, in order to improve design and supply the best possible product.

Philips Semiconductors
811 East Arques Avenue
P.O. Box 3409
Sunnyvale, California 94088–3409
Telephone 800-234-7381

PHILIPS SEMICONDUCTORS, INC. A SUBSIDIARY OF
PHILIPS ELECTRONICS NORTH AMERICA CORPORATION

ENGINEER CHANGE NOTICE
850-0007 FORM A 02/95

PREPARED BY: PAGE OF

Dresser	W	9226	4645	60	Nov 27, 1995	SVL
LAST NAME	INIT	DEPT	EXT	BIN#	DATE	PLANT

ECN NUMBER & DATE

PERMANENT X IMMEDIATE___ TEMPORARY___ AFTER-HOURS___

APPROVAL DATE: ____________

EXPIRATION DATE: ____________

MASTER SIGNOFF: YES:___ NO:___ INITIALS: ___

ECN CONTINUATION ATTACHED: YES:___ NO:___

PRODUCT GROUP: Communication Products Group

ROUTER : DATE

PART NUMBER	PKG/REV	BASIC DIE/REV	CUSTOMER PART #/REV	ALPHA

DOCUMENT TITLE: See description below

DOCUMENT TYPE: Data Sheet DOCUMENT NUMBER: 853- REV: ___

DESCRIPTION OF CHANGE: INCLUDE FROM/TO INFO WHERE APPROPRIATE

LM2901 . . . . . . 853-0930 13721 . . . . . 94 Aug 31

LM2902 . . . . . . 853-0929 13721 . . . . . 94 Aug 31

LM2903 . . . . . . 853-0932 13721 . . . . . 94 Aug 31

LM2904 . . . . . . 853-1241 13721 . . . . . 94 Aug 31

All data sheets reflect change from 85°C to 125°C.

EFFECT: ( )MAJOR ( )MINOR

PROVIDE CPCN STATUS, JUSTIFICATION FOR THE CHANGE AND AN EXPLANATION OF EFFECT

OIDS/DOCUMENTS AFFECTED:

___ PRIME	___ PDYM	___ FPOID	___ PROBE DIAGRAM
___ CDDB	___ ESORT	___ APL	___ BONDING DIAGRAM
___ DDDB	___ EMCR	___ FT/PA	___ SCHEMATIC
___ FAB XREF	___ PMF/PDDB	___ FT FLOW	___ TABLE
		___ MPO	X OTHER 853-

KEY SPONSOR:

CO-SPONSORS:

DEPT#	SIGNATURE	DATE	EXT/USER ID

ECN DISTRIBUTION: DEPT# DEPT# DEPT#

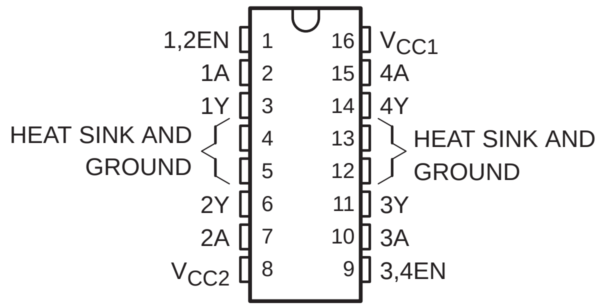
MISCELLANEOUS DISTRIBUTION: DEPT# DEPT# DEPT#

MATERIAL DISPOSITION:

	FAB	ESORT	DS	ASSY	SEAL	SYMB	M/V	FT	PA	PACK	FG	MATL STRS	OTHER
USE AS IS													
REWORK													
SCRAP													

- **1-A Output-Current Capability Per Driver**
- **Applications Include Half-H and Full-H Solenoid Drivers and Motor Drivers**
- **Designed for Positive-Supply Applications**
- **Wide Supply-Voltage Range of 4.5 V to 36 V**
- **TTL- and CMOS-Compatible High-Impedance Diode-Clamped Inputs**
- **Separate Input-Logic Supply**
- **Thermal Shutdown**
- **Internal ESD Protection**
- **Input Hysteresis Improves Noise Immunity**
- **3-State Outputs**
- **Minimized Power Dissipation**
- **Sink/Source Interlock Circuitry Prevents Simultaneous Conduction**
- **No Output Glitch During Power Up or Power Down**
- **Improved Functional Replacement for the SGS L293**

NE PACKAGE
(TOP VIEW)



FUNCTION TABLE
(each driver)

INPUTS†		OUTPUT
A	EN	Y
H	H	H
L	H	L
X	L	Z

H = high-level, L = low-level
X = irrelevant
Z = high-impedance (off)
† In the thermal shutdown mode, the output is in a high-impedance state regardless of the input levels.

description

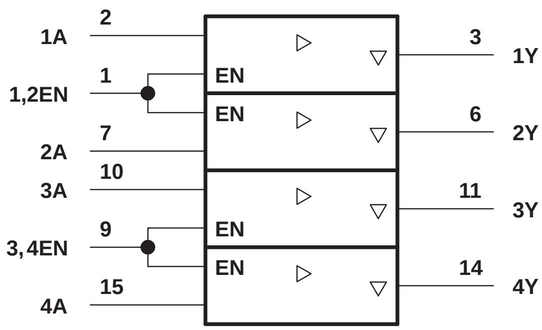
The SN754410 is a quadruple high-current half-H driver designed to provide bidirectional drive currents up to 1 A at voltages from 4.5 V to 36 V. The device is designed to drive inductive loads such as relays, solenoids, dc and bipolar stepping motors, as well as other high-current/high-voltage loads in positive-supply applications.

All inputs are compatible with TTL-and low-level CMOS logic. Each output (Y) is a complete totem-pole driver with a Darlington transistor sink and a pseudo-Darlington source. Drivers are enabled in pairs with drivers 1 and 2 enabled by 1,2EN and drivers 3 and 4 enabled by 3,4EN. When an enable input is high, the associated drivers are enabled and their outputs become active and in phase with their inputs. When the enable input is low, those drivers are disabled and their outputs are off and in a high-impedance state. With the proper data inputs, each pair of drivers form a full-H (or bridge) reversible drive suitable for solenoid or motor applications.

A separate supply voltage (V_{CC1}) is provided for the logic input circuits to minimize device power dissipation. Supply voltage V_{CC2} is used for the output circuits.

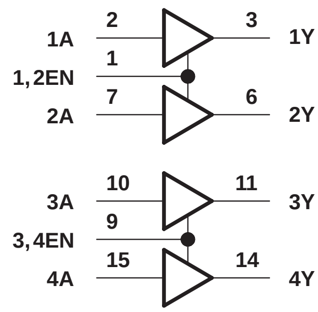
The SN754410 is designed for operation from –40°C to 85°C.

logic symbol†

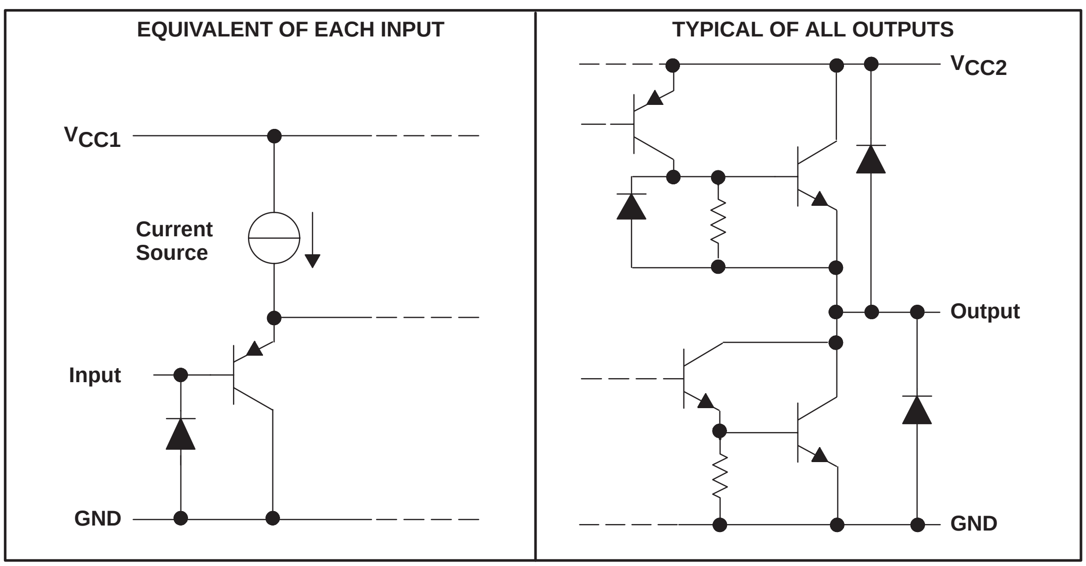


† This symbol is in accordance with ANSI/IEEE Std 91-1984 and IEC Publication 617-12.

logic diagram



schematics of inputs and outputs



TEXAS INSTRUMENTS
POST OFFICE BOX 655303 • DALLAS, TEXAS 75265
POST OFFICE BOX 1443 • HOUSTON, TEXAS 77251–1443

absolute maximum ratings over operating free-air temperature range (unless otherwise noted)†

Output supply voltage range, V_{CC1} (see Note 1) . . . . –0.5 V to 36 V
Output supply voltage range, V_{CC2} . . . . –0.5 V to 36 V
Input voltage, V_I . . . . 36 V
Output voltage range, V_O . . . . –3 V to V_{CC2} + 3 V
Peak output current (nonrepetitive, $t_w \leq 5$ ms) . . . . ±2 A
Continuous output current, I_O . . . . ±1.1 A
Continuous total power dissipation at (or below) 25°C free-air temperature (see Note 2) . . . . 2075 mW
Operating free-air temperature range, T_A . . . . –40°C to 85°C
Operating virtual junction temperature range, T_J . . . . –40°C to 150°C
Storage temperature range, T_{stg} . . . . –65°C to 150°C
Lead temperature 1,6 mm (1/16 inch) from case for 10 seconds . . . . 260°C

† Stresses beyond those listed under “absolute maximum ratings” may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated under “recommended operating conditions” is not implied. Exposure to absolute-maximum-rated conditions for extended periods may affect device reliability.

NOTES: 1. All voltage values are with respect to network GND.
2. For operation above 25°C free-air temperature, derate linearly at the rate of 16.6 mW/°C. To avoid exceeding the design maximum virtual junction temperature, these ratings should not be exceeded. Due to variations in individual device electrical characteristics and thermal resistance, the built-in thermal overload protection can be activated at power levels slightly above or below the rated dissipation.

recommended operating conditions

	MIN	MAX	UNIT
Output supply voltage, V_{CC1}	4.5	5.5	V
Output supply voltage, V_{CC2}	4.5	36	V
High-level input voltage, V_{IH}	2	5.5	V
Low-level input voltage, V_{IL}	–0.3‡	0.8	V
Operating virtual junction temperature, T_J	–40	125	°C
Operating free-air temperature, T_A	–40	85	°C

‡ The algebraic convention, in which the least positive (most negative) limit is designated as minimum, is used in this data sheet for logic voltage levels.

electrical characteristics over recommended ranges of supply voltage and free-air temperature (unless otherwise noted)

PARAMETER		TEST CONDITIONS		MIN	TYP†	MAX	UNIT
V_{IK}	Input clamp voltage	$I_I = -12$ mA			−0.9	−1.5	V
V_{OH}	High-level output voltage	$I_{OH} = -0.5$ A		$V_{CC2}-1.5$	$V_{CC2}-1.1$		V
		$I_{OH} = -1$ A		$V_{CC2}-2$			
		$I_{OH} = -1$ A, $T_J = 25°C$		$V_{CC2}-1.8$	$V_{CC2}-1.4$		
V_{OL}	Low-level output voltage	$I_{OL} = 0.5$ A			1	1.4	V
		$I_{OL} = 1$ A				2	
		$I_{OL} = 1$ A, $T_J = 25°C$			1.2	1.8	
V_{OKH}	High-level output clamp voltage	$I_{OK} = -0.5$ A			$V_{CC2}+1.4$	$V_{CC2}+2$	V
		$I_{OK} = 1$ A			$V_{CC2}+1.9$	$V_{CC2}+2.5$	
V_{OKL}	Low-level output clamp voltage	$I_{OK} = 0.5$ A			−1.1	−2	V
		$I_{OK} = -1$ A			−1.3	−2.5	
$I_{OZ(off)}$	Off-state high-impedance-state output current	$V_O = V_{CC2}$				500	μA
		$V_O = 0$				−500	
I_{IH}	High-level input current	$V_I = 5.5$ V				10	μA
I_{IL}	Low-level input current	$V_I = 0$				−10	μA
I_{CC1}	Output supply current	$I_O = 0$	All outputs at high level			38	mA
			All outputs at low level			70	
			All outputs at high impedance			25	
I_{CC2}	Output supply current	$I_O = 0$	All outputs at high level			33	mA
			All outputs at low level			20	
			All outputs at high impedance			5	

† All typical values are at $V_{CC1} = 5$ V, $V_{CC2} = 24$ V, $T_A = 25°C$.

switching characteristics, $V_{CC1} = 5$ V, $V_{CC2} = 24$ V, $C_L = 30$ pF, $T_A = 25°C$

PARAMETER		TEST CONDITIONS	MIN	TYP	MAX	UNIT
t_{d1}	Delay time, high-to-low-level output from A input	See Figure 1		400		ns
t_{d2}	Delay time, low-to-high-level output from A input			800		ns
t_{TLH}	Transition time, low-to-high-level output			300		ns
t_{THL}	Transition time, high-to-low-level output			300		ns
t_r	Rise time, pulse input					
t_f	Fall time, pulse input					
t_w	Pulse duration					
t_{en1}	Enable time to the high level	See Figure 2		700		ns
t_{en2}	Enable time to the low level			400		ns
t_{dis1}	Disable time from the high level			900		ns
t_{dis2}	Disable time from the low level			600		ns

PARAMETER MEASUREMENT INFORMATION

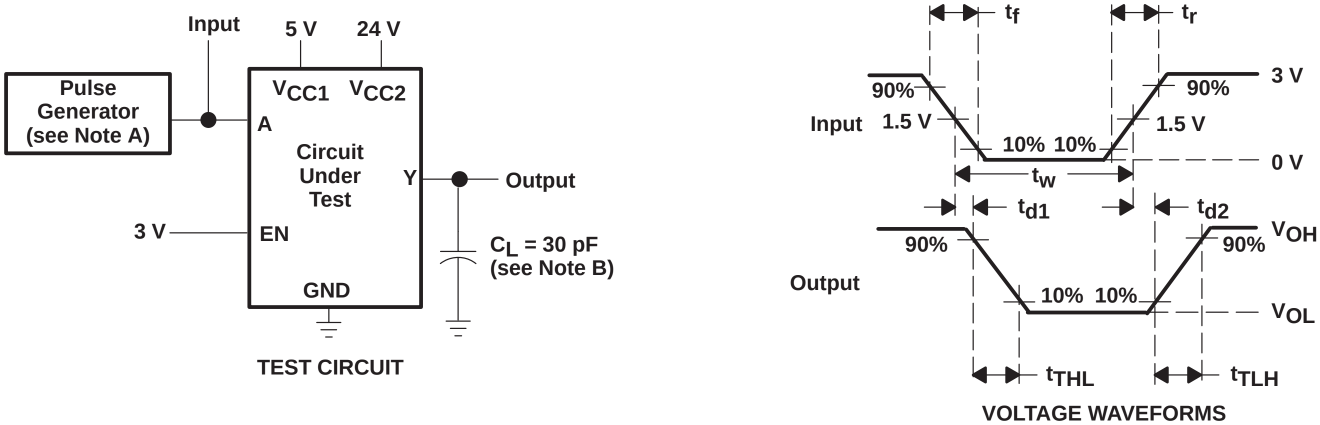


Figure 1. Test Circuit and Switching Times From Data Inputs

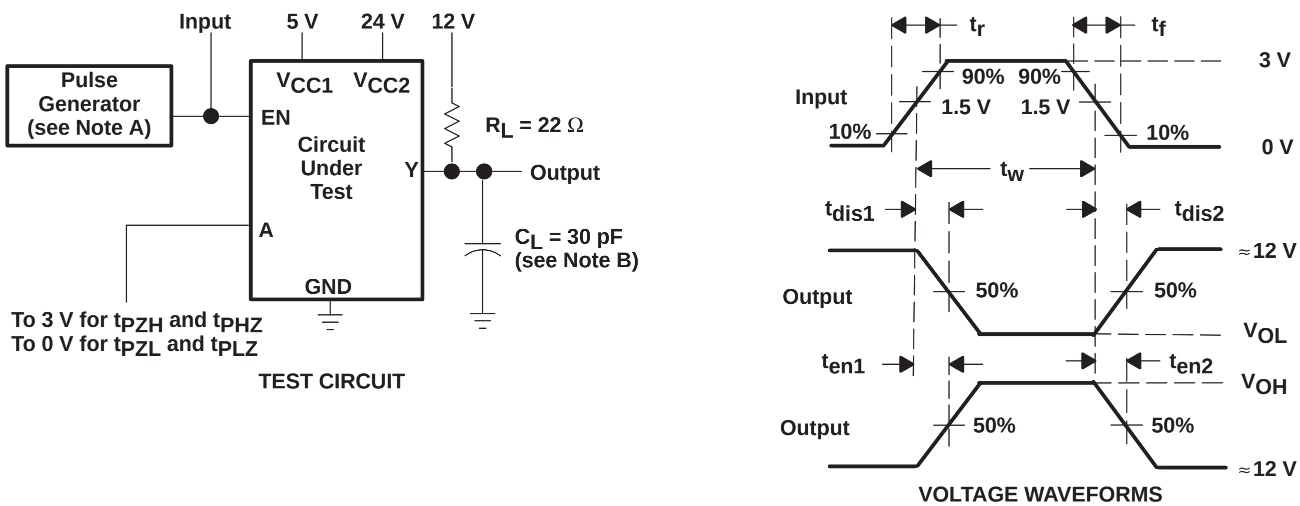


Figure 2. Test Circuit and Switching Times From Enable Inputs

NOTES: A. The pulse generator has the following characteristics: $t_r \leq 10$ ns, $t_f \leq 10$ ns, $t_w = 10$ µs, PRR = 5 kHz, $Z_O = 50\ \Omega$.
B. C_L includes probe and jig capacitance.

APPLICATION INFORMATION

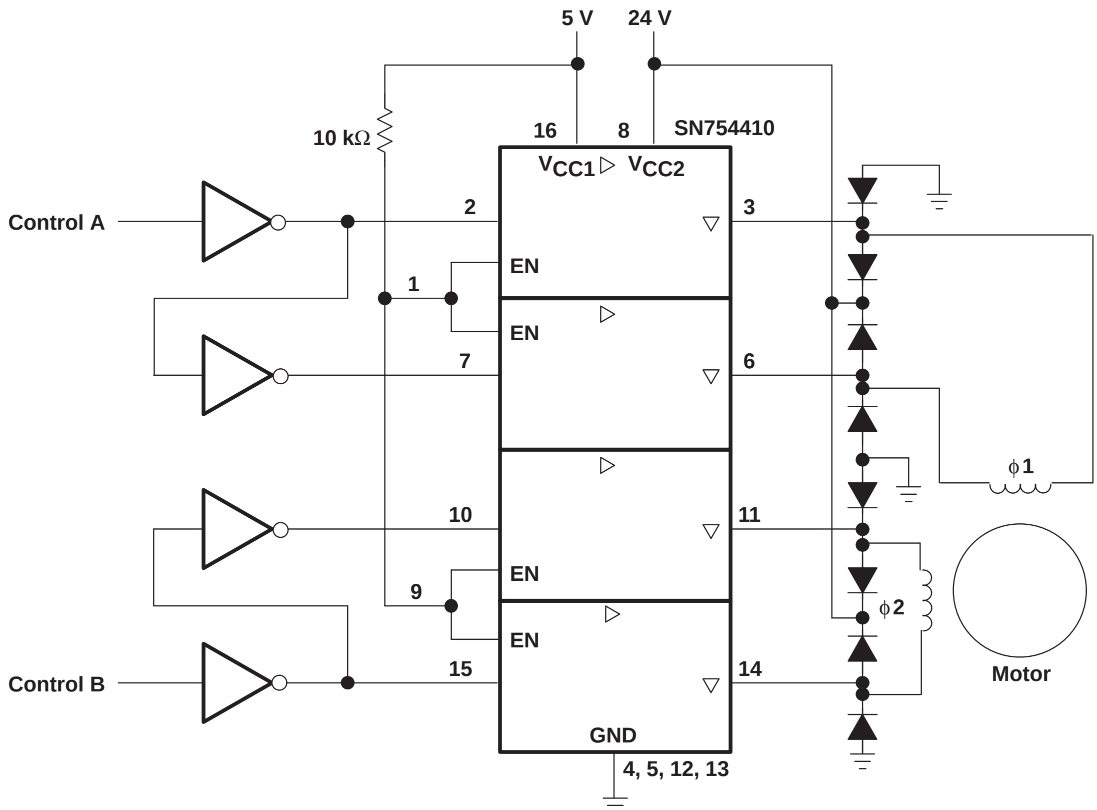


Figure 3. Two-Phase Motor Driver

TEXAS INSTRUMENTS
POST OFFICE BOX 655303 • DALLAS, TEXAS 75265
POST OFFICE BOX 1443 • HOUSTON, TEXAS 77251–1443

IMPORTANT NOTICE