

# MAZE MOUSE

By

MOHD FAIZ BIN MOHD MUSTAPA

FINAL PROJECT REPORT

Submitted to the Electrical & Electronics Engineering Programme  
in Partial Fulfillment of the Requirements  
for the Degree  
Bachelor of Engineering (Hons)  
(Electrical & Electronics Engineering)

Universiti Teknologi Petronas  
Bandar Seri Iskandar  
31750 Tronoh  
Perak Darul Ridzuan

© Copyright 2004

by

Mohd Faiz B. Mohd Mustapa, 2004

t

TK

9969

17697

2004

- i. Microprocessors -- Design and Construction
- ii. Digital control systems -- Design and construction
3. EEE -- Thesis

# CERTIFICATION OF APPROVAL

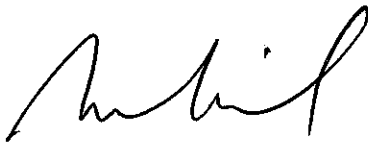
MAZE MOUSE

by

Mohd Faiz B . Mohd Mustapa

A project dissertation submitted to the  
Electrical & Electronics Engineering Programme  
Universiti Teknologi PETRONAS  
in partial fulfilment of the requirement for the  
Bachelor of Engineering (Hons)  
(Electrical & Electronics Engineering)

Approved:



---

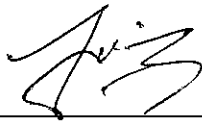
Dr Nordin Saad  
(Project Supervisor)

UNIVERSITI TEKNOLOGI PETRONAS  
TRONOH, PERAK

June 2004

## CERTIFICATION OF ORIGINALITY

This is to certify that I am responsible for the work submitted in this project, that the original work is my own except as specified in the references and acknowledgements, and that the original work contained herein have not been undertaken or done by unspecified sources or persons.



---

MOHD FAIZ B MOHD MUSTAPA

## **ABSTRACT**

The dissertation basically discuss on the overall view of a maze mouse project. The objective of this project is to design and develop the mouse model. The purpose of the project is to develop student's cognitive and practical skills. The dissertation discusses the introduction to the project that includes the background of studies, the problem statement and the scope of project. Next, the literature review and theoretical part is discussed follow by the methodology aspects. In this session, brief procedure identification will be touch and the hardware requirement is listed. Following a discussion on the methodology, this dissertation gives detail discussion of the project. Finally, the conclusion and recommendation part is presented.

## **ACKNOWLEDGEMENTS**

This project would not be possible without the assistance and guidance from key individuals whose contributions have helped in the completion of this study.

First of all and most importantly, I would like to express my sincere and deepest gratitude to my project supervisor, Dr Nordin Saad for his valuable input and guidance at all times throughout the course of this study.

I would also like to acknowledge the cooperation and assistance from Miss Siti Hawa from School of Electrical Electronics, in term of project material and management.

I would also like to express my gratitude to the School of Electrical and Electronics Engineering lectures, Mr Zuki and Mr Zuhairi for give ideas on the improvement of the project. I also would like to thanks my colleague who helps me in progressing with the project.

Last but not least, I would like to thank all the individuals whose name have not been mentioned here but has inadvertently contributed to the completion of this project whether it is directly or indirectly. Thank you.

# TABLE OF CONTENTS

LIST OF TABLES .....	ix
LIST OF FIGURES .....	x
CHAPTER 1 INTRODUCTION.....	1
1.1 Background of Studies.....	1
1.2 Problem Statement.....	2
1.3 Objective and Scope of Studies .....	3
CHAPTER 2 LITERATURE REVIEW AND THEORY .....	5
2.1 Mouse Control .....	5
2.2 8051 Architecture .....	6
2.2.1 Microcontroller basic function.....	7
2.3 Machine / Assembler Language .....	7
2.3.1 Addressing mode.....	9
CHAPTER 3 METHODOLOGY / PROJECT WORK.....	10
3.1 Sensor Circuit .....	10
3.2 Microcontroller Circuit.....	12
3.3 Voltage regulator / Motor Circuit.....	13
3.4 Methodology.....	14
CHAPTER 4 RESULT AND DISCUSSION.....	16
4.1 Mouse Logic .....	18
4.2 Maze Design .....	21
4.3 Example logic of mouse programming structure.....	22
4.4 Maze Mouse Program.....	28
CHAPTER 5 CONCLUSION AND RECOMMENDATION .....	32
5.1 Recommendation .....	33
REFERENCES .....	i
APPENDIX A BASIC FLOW DIAGRAM OF OTHER MOUSE LOGIC.....	II
<i>Straight logic</i> .....	ii
<i>Left corner</i> .....	iii
<i>Right Corner</i> .....	iv
<i>4 - Junctions</i> .....	v
<i>T Junction</i> .....	vi

<i>T Left Junction</i> .....	vii
<i>T right junction</i> .....	viii
<i>Dead End</i> .....	ix
APPENDIX B RELATIONSHIP OF CIRCUIT COMPONENT .....	X
APPENDIX C TOP VIEW OF MOUSE MODEL .....	XI
APPENDIX D SIDE VIEW OF MOUSE MODEL .....	XII
APPENDIX E MOUSE PROGRAMMING .....	XIII
APPENDIX F INSTRUCTION SET .....	XVI
APPENDIX G DATASHEETS.....	XX

## LIST OF TABLES

Table 1 Logic routing for the input sensor .....	16
Table 2 Representation of hexadecimal, decimal and binary value equivalent.....	29
Table 3 Condition subroutine path check.....	30



## LIST OF FIGURES

Figure 1 Illustrate the mouse turning control .....	5
Figure 2 Sensor circuit diagram .....	10
Figure 3 Sensor placing on mouse model.....	11
Figure 4 Micro controller circuit diagram.....	12
Figure 5 Voltage regulator and Stepper motor regulator diagram .....	13
Figure 6 Logic of the path of micro controller programming .....	18
Figure 7 General flow of mouse control.....	20
Figure 8 Example of Mouse Logic Algorithm .....	20
Figure 9 Reflection of infrared signal in different surface .....	21

# CHAPTER 1

## INTRODUCTION

### 1.1 Background of Studies

Micro controller is a general-purpose device. The micro controller will able to read input data, perform limited calculation on the data and control its environment base on the calculation. The prime use of the micro controller is to control the operation of the machine using a fix program stored in ROM.

Micro controller is an important aspect of project system of mouse maze for final year project. In the first half of the semester, several studies must be done in order to complete the mouse maze model. The studies will concentrate on the circuit development and also the operational aspect of the maze mouse system.

Basically, a maze mouse is an electro-mechanical, mobile robot. It contains 3 types of circuit which are:

- a) Voltage regulator circuit and stepper motor driver circuit,
- b) System micro controller circuit
- c) Sensor circuit.

The knowledge in mechanics, electronics and EPROM programming need to be understand in order to proceed with the model structure. The important parts of the project have to be consider on the connection and the functionality of the component. The program is then being loaded into the micro controller. The program is responsible for the operational of the mouse. It consist of several routing for maze navigation, the maze solving software for directing the movement of the mouse in the maze.

In the second semester, the progress will mainly touch on the programming logic of the mouse. This includes the information on 8051-microcontroller structures and also the machine language use.

## **1.2 Problem Statement**

In order to design and develop the mechanical as well as the electrical part on the maze mouse model, several problem statements need to be understood clearly. The main purpose of the project is to develop an electronic mouse model, consists of micro controller board, sensor interfacing, and the mouse mechanical system. The mouse prototype should be able to maneuver and find its route out of a maze.

The building of the mouse model requires essential knowledge in electrical electronics. The project will act as a good practice in refreshing the knowledge previously learned and developing new knowledge as well. This is important in developing the basic core and the structure of the mouse.

Requirement of the mouse model:

- a) Able to find way out of the maze.
- b) Micro controller make decision base on input signal
- c) Sensor will able to detect left, right and forward direction
- d) Stepper motor to control the mouse movement.

In order to construct the mouse model, the circuit must be structured and has to be built stages by stages. The flexibility of the circuit must be tested as this will help in developing of the mouse maze model. In order to overcome the difficulty of the problem face, several planning process has to be taken in order to minimize the error that will be produced.

The second semester of the project will focus more on the model design and construction. The area focus will be the build up structure of the mouse model, programming and also some minor details on the logic functions. This report will touch on the main important component of microcontroller and the programming structure.

The mouse concept is being applied in industrial area. It is use in industrial application in replacing human operator in operating the system. This will help to improve the productivity of the production. In industrial workplace or a plant, there is an area, which cannot be access by the personal. This area might be hazardous, or have complex structure of design. In order to reach the place, an intelligence device such as mouse maze system is being use.

### **1.3 Objective and Scope of Studies**

This project course is a requirement for the final year students to complete during their final year of semester. The objectives of performing such a task to enable the students, to use all the knowledge that had been learn during the years in UTP. It will help to train the students in doing project and organized their work in a systematic ways. This is very useful in their future career as an engineer.

The planning on the mouse model basically consists of two stages. The first part of the project is mostly concentrating in developing the core circuit for the mouse model. This includes the construction of a micro controller circuit, a motor driver circuit and also a sensor circuit. The second stages of the design basically consist of implementing the knowledge of assembly language programming software, which is crucial in setting up the logic for the mouse model. This stage also focused on the integration of each of the circuit as well as testing their functionality.

In the first semester of the Final Year Project, the expectation on the student is to complete all the basic circuit of the mouse. In the second semester, the circuits will be integrated with one another. The EPROM programming will mainly be covered in the second semester.

The mouse technology is useful. The example for this is the AGV system. This technique is use on factory product production or machine system. Consider a situation where products need to be move from conveyer A to conveyer B where both of the conveyers is separated. In order to move the product from conveyer A to conveyer B, a system which is likely an automatically trolley or small car is used to transport the product. This type of implementation is more or less is the same as the mouse model prototype.

With the implementation of the mouse model, the cost of production in factory can be reduced. This mouse maze system knowledge will help in building a cheaper system, which can provide the same capability and efficiency as the ready inefficient model. The study on this particular field, this will help in enhancing the system capabilities.

As a conclusion, the objective of the study can be identify as:

- a) Learn the application of micro controller in industry.
- b) Programming the micro controller base on the logic and the input.
- c) Develop the circuit construction of the mouse model.
- d) To understand the microcontroller function of 8051
- e) Improve design for the mouse model.

## CHAPTER 2 LITERATURE REVIEW AND THEORY

Throughout this study, few problems have been encountered and several steps have been identified to solve the problem. This includes the unavailability of the datasheet for the component of the hardware. This causes some difficulty in the circuit construction, as the circuit diagram did not fully interpret the actual connection according to the real component structure.

Due to this reason, besides the literature studies, the knowledge of component structure is an important part of the development process. Without proper knowledge, circuit connection and functionality couldn't be established. In order to progress with the project, several theory and practices need to be understood. The project mainly consists on electrical and electronics areas, in building the circuit. An essential knowledge in power electronics, digital, analogue, microprocessor needs to be revised for the purpose. Besides all the information, EPROM programming and mouse logic structure need to be understood.

### 2.1 Mouse Control

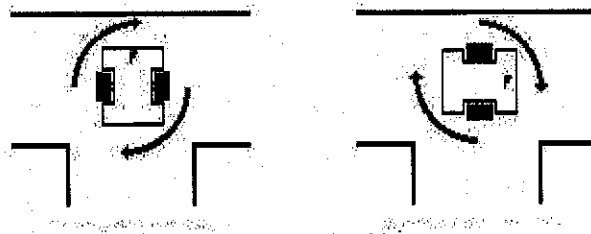


Figure 1 Illustrate the mouse turning control

The mouse approaches the corner square slowly and attempts to stop in the middle. A small program is then, run the position the mouse as close as possible to the center using very small movements.

The mouse then executes a sharp, timed 90 degree turn by driving one side wheels forward and one side wheels backward to face the right direction.

The same positioning program is then run to reposition the mouse before it moves off. This approach is slow, but as the mouse development, it is only in its infancy consistency is more important than speed.

There are a few important areas that need to be review on the project. The scopes that need to be review are:

- a) Develop up structure of the model. This will focus on the set up or the physical appearance of the mouse model.
- b) Programming of microcontroller. This part will emphasize on the logic 'brain' of the mouse model.

Most of the literature review done, come from the Internet. The study concentrate on 8051 Intel microprocessor. This will help in understanding the structure of 8051 microcontroller, and also to catch up the programming structure of the system. In general, all the website content explains on the 8051 functions and also the programming portion of the system.

## **2.2 8051 Architecture**

The 8051 offer a 64 Kbyte (that is,  $64 * 1024$ ) program memory address space. Program memory is non-volatile, meaning that it survives the loss of power. 8051 also offers twenty-one special function registers (SFR). These registers are like data memory locations which you can read and write them. These registers control the on-chip components of the 8051 such as its timers, its interrupts, its serial port, etc.

### **2.2.1 Microcontroller basic function**

Microprocessor is a device, which execute the stored program. While the program is being executed, its only manifestation inside the computer is as a pattern of voltage levels. Only two voltage levels are employed: a high voltage approximately equal to the power supply voltage and a low voltage approximately equal to the ground potential.

Only two voltage levels are employed because an on/off switch can quickly produce two voltages:

- a) When the switch is closed the voltage goes to one level
- b) When the switch is open it goes to the other.

A modern integrated circuit can hold millions of transistors and each transistor behaves like a very fast switch, modulating a voltage between two levels. The fact that computers are based upon devices having only two states explains why binary numbers are the best representation for a computer program. The digits (**bits**) of a binary number can have only two values: 0 or 1

### **2.3 Machine / Assembler Language**

Machine language is the name given to a computer program written using a binary alphabet having only the two letters 0 and 1 which correspond to the low and high voltage respectively

An assembler is a computer program that functions as a text processor to convert an assembly language program to the form (machine language) that can be executed by a particular computer



Assembly language programs consist of three kinds of statements:

- a) Assembly language instructions specific to the particular microprocessor,
- b) Assembler directives which are commands to the assembler rather than to the microprocessor,
- c) Comment (and blank) lines

The first of these three types of assembly language statements are translated by the assembler into the equivalent machine language instructions. The programmer produces a source file holding his assembly language program and the assembler then produce the listing file that shows the corresponding machine language instructions.

Assembly language statements consist of four ordered fields:

[label:] mnemonic [operand list] [;comment]

Start: MOV A,#7 ; place the number 7 into the A

Square brackets are used to show optional fields. Hence the label field does not need to be present but if it is present, it must be the first field and must terminate with a colon. Similarly, the comment field is not required but if it is present then it must start with a semicolon. The only required field is the instruction mnemonic, which is so called because it serves to remind the programmer of the intended machine operation.

Examples of the mnemonics used for the 8051 are MOV for "move operation" and ADDC for "add with carry". Many instructions require operands and these are placed, separated by commas, in the operand list. An operand can be a constant (such as "7") or a symbol (such as "A" standing for the accumulator) or a combination of constants and symbols.

### 2.3.1 Addressing mode

The four addressing modes, which are:

- i. **Direct addressing** where a data address is embedded in the instruction. For example, the command "MOV A,5" causes the 8051's accumulator to be loaded with the contents of data memory address 5.
- ii. **Indirect addressing** where a register holds the address of the intended data memory location. For example, the command "MOV A,@R0" causes the accumulator to be loaded with the contents of the data memory location addressed by the value currently held in Register #0. We can't tell what data memory address this would happen to be without knowing the current value held in Register #0.
- iii. **Register addressing** where a register holds the desired value (in indirect addressing the register holds the address of the desired value). For example, the command "MOV A,R1" causes the accumulator to be loaded with the value in Register #1. Most microprocessors offer a small number of **registers** which you are free to use to hold results. The 8051 is peculiar in that its registers are synonymous with certain data memory locations (usually, a microprocessor's registers are distinct from its data memory). In the 8051's default configuration (the configuration it powers up in) Register #0 is synonymous with data memory address 0, Register #1 is synonymous with data memory address 1, etc. through Register #7.
- iv. **Immediate addressing** where a data value (not an address) is embedded in the instruction. For example, the command "MOV A,#5" causes the accumulator to be loaded with the value 5. Note that this value did not come from data memory or from a register: it came from program memory since the value 5 came from the instruction itself.

## CHAPTER 3 METHODOLOGY / PROJECT WORK

The objective of the project in first semester is to integrate all the circuit required for the model. The project mainly consists of building the requirement circuit and testing session. This will require a lot of effort as the connection need to be done properly and the design must be properly understood.

In progressing with the project, several discussion and literature review have been done. This will help the student to grasp the understanding and progressing with the project. The student needs to put an effort in learning new knowledge to complete the project successfully. For the purpose a detail structure of the circuit is being explained.

### 3.1 Sensor Circuit

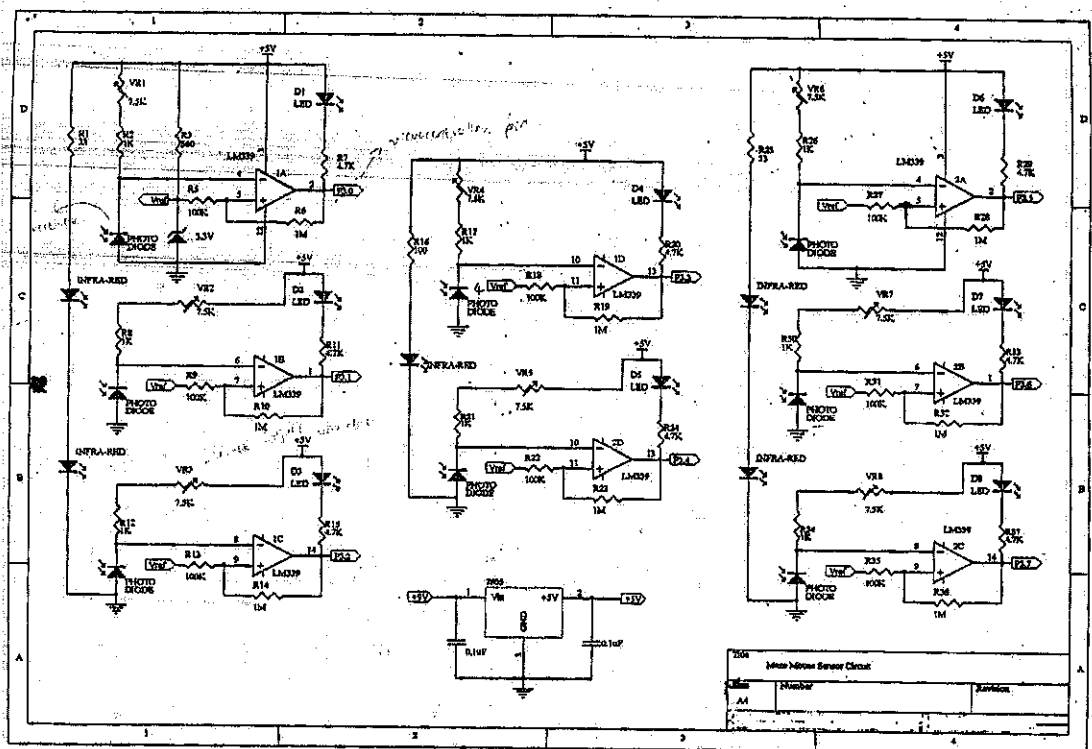


Figure 2 Sensor circuit diagram

The basic function of sensor circuit is to transmit infrared signal. The signal will be used to detect the path for mouse maze model. To describe the circuit functionality, first a positive voltage will go through the amplifier and this creates current flow for the circuit. The circuit will then light up the LED. The function of the LED is to indicate that there is a current flow in the circuit and also help to validate that the circuit is working properly. Once the LED is being lit, the voltage will pass the variable resistor 7.5k to the infrared transmitter. The function of the variable resistor is to control the sensitivity of the infrared.

The infrared transmitter is being connected in the circuit. Once infrared transmit light, the infrared receiver will detect the light produced. Once the infrared signal is being detected by the receiver, it will give the signal input to microcontroller circuit. Based on the circuit diagram above, there are 5 infrared transmitters and a total of 7 infrared receivers for the sensor circuit. Two of the infrared transmitters are being placed at each side of the mouse. Transmitters at the middle will indicate the forward direction of the mouse. The infrared placing is illustrated in the below figure.

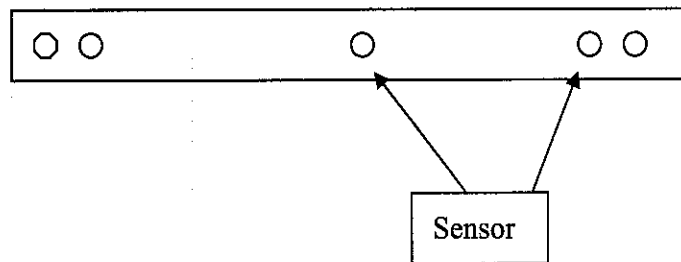


Figure 3 Sensor placing on mouse model

### 3.2 Microcontroller Circuit

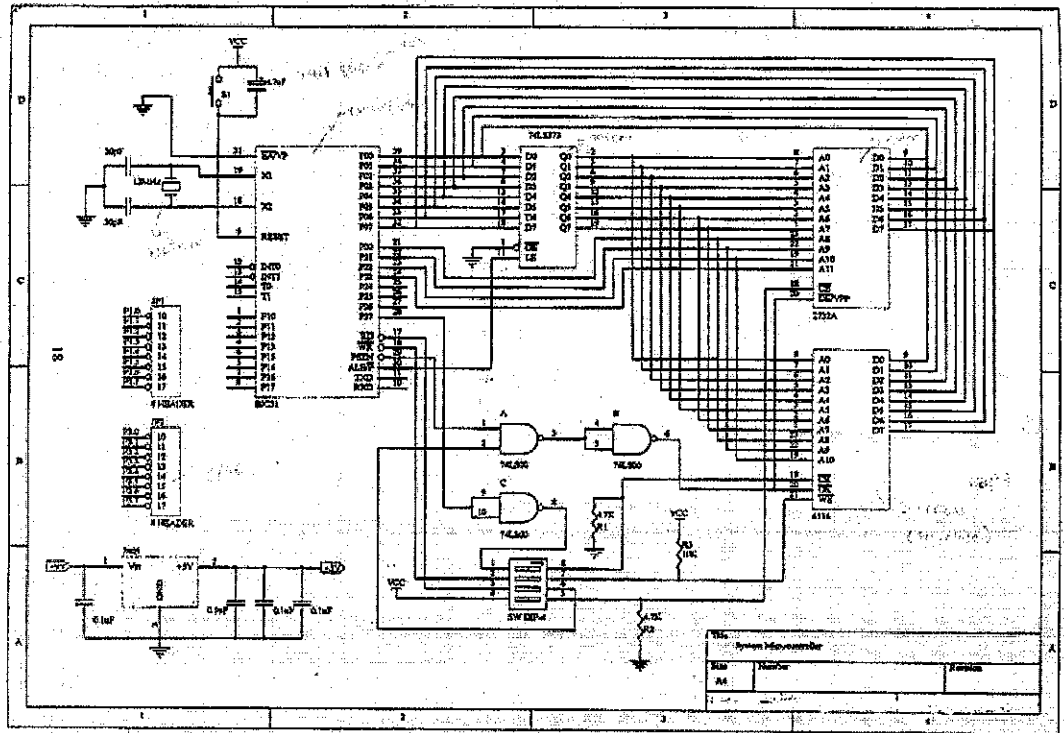


Figure 4 Micro controller circuit diagram

The microcontroller board is the most important component in a micro mouse. This component allows the mouse to navigate the maze with a degree of intelligence. The microcontroller coordinates the tasks of position monitoring, movement, maze navigation, and solution evaluation.

Generally, the term 'microcontroller' is given to a microprocessor that has been designed as self-sufficient as possible. A microcontroller typically has its own on-board program memory, data memory and I/O ports in order to minimize external hardware otherwise required for these functions. This combination of processing power and small size means a microcontroller is well suited for use in a micro mouse.

The micro controller circuit is the brain of the mouse system. The input from the sensor is being connected to the microcontroller IC by using 8-pin header. The microcontroller circuit is connected to latch, Ram and also EPROM. The function of Ram is to store the information produce on the microcontroller. EPROM is used holds the programming function of the microcontroller.

The programming language implement on the design is base on the basic microcontroller language. The programming will touch on the logic function of the mouse movement base on the sensor input. The microcontroller also control on the rotation part of the motor. This will determine the movement of the mouse whether going straight, left, right or in a rotation mode.

### 3.3 Voltage regulator / Motor Circuit

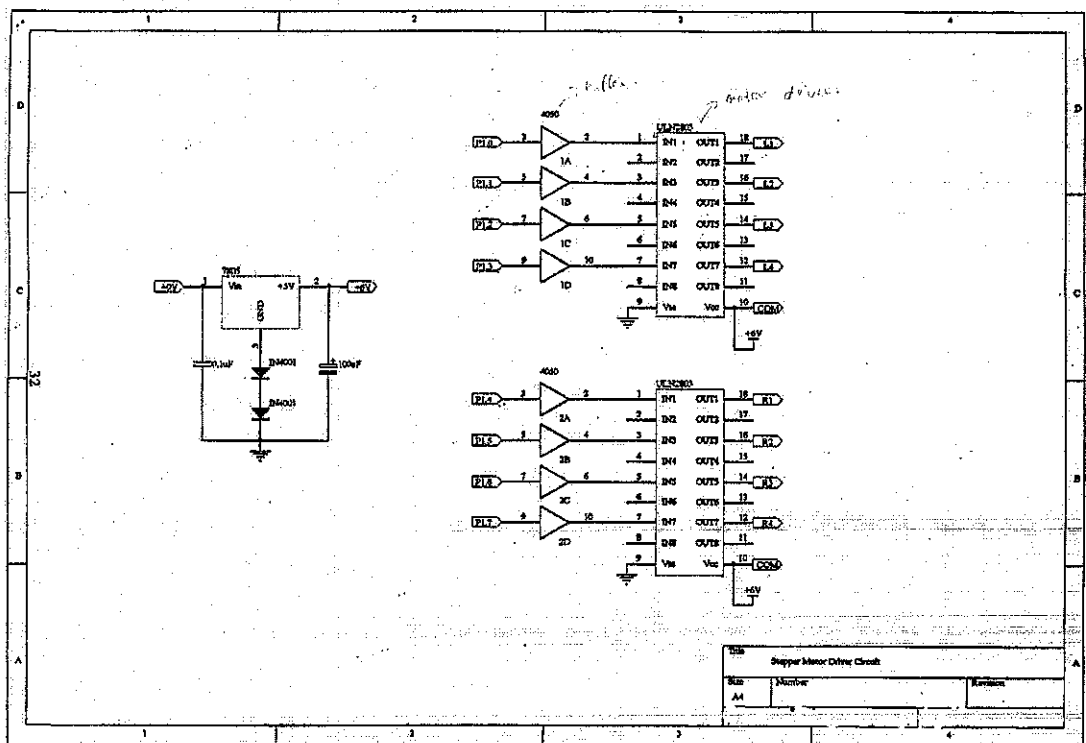


Figure 5 Voltage regulator and Stepper motor regulator diagram

This circuit consist a simple connection compare to both microcontroller and sensor circuit. The purpose of having the voltage regulator circuit is to control the current and voltage flow of the circuit. This will help, as a safety measure for the component. It will able to reduce the probability of the component from overheated. The stepper motor circuit function is to control the speed of the motor and the model movement. Input from micro controller is use for the purpose. This will help in controlling the mechanical part of the model.

The hardware requirement for the circuit is being list as:

- a) Voltage regulator and stepper motor circuit:

*Voltage regulator IC, Capacitor Motor driver IC, Stepper motor, buffer*

- b) System Micro controller circuit:

*Intel 0C31 Microprocessor IC, Latch, Nand gate, RAM, EPROM, voltage regulator, electrolytic, crystal oscillator, switch, push button switch*

- c) Sensor circuit

*Infrared sensor, photo diode, resistor (variable and non variable), LED, amplifier*

### **3.4 Methodology**

Several activities have being conducted in during the process of maze mouse development. The methodologies done are:

- i. **Discussion with supervisor:** This is the important step of project development. Most of the time, all the problems encountered are bring forward into the discussion section. The discussion provides a platform of understanding between the supervisor and the student. The supervisor will be able to keep tract the progress of the project. For the student, the discussion will be able to enhance their knowledge and solutions in overcoming the problem.

- ii. **Hardware purchase:** The component need for the circuit hardware has being purchased. This will help on the circuit model construction.
  
- iii. **Literature review and desk study:** Several studies have been performed throughout the material in the library and also in the Internet, see [1],[2][3],[4],[5] and [6]. The structure of the micro controller is being referred to 8051 micro controller books. The mouse system is being reverred to the various addresses, which provide information regarding the maze mouse model. (Please refer to appendix for the URL) This will help to strengthen the understanding of the mouse model. It will also help to increase the knowledge for the system development of the mouse system.
  
- iv. **Circuit design testing:** Several testing section of the circuit has been conducted. The objective of the circuit testing is to make sure that the circuit design worked properly before the real construction of the maze mouse component. Sensor circuit is important to be tested several times. This is due to the complexity of the circuit. The sensor circuit is an analog circuit and this makes it difficult to achieve the result compare to the digital circuit.



## CHAPTER 4 RESULT AND DISCUSION

LEFT SENSOR	FORWARD SENSOR	RIGHT SENSOR	ROUTING	DECISION
Low	High	High	Left corner	Turn Left
High	High	Low	Right corner	Turn Right
High	Low	High	Straight line	Left
Low	Low	Low	4 Junction	Forward
High	High	High	Dead End	Reverse
Low	Low	High	T Left Junction	Turn Left
High	Low	Low	T Right Junction	Forward
Low	High	Low	T Junction	Left

Table 1 Logic routing for the input sensor

High indicates logic (1). This indicates that the receiver received the reflection signal from the infrared signal. Low indicates logic (0). This indicates that the receiver did not receive the reflection of the infrared signal, see [7] for detail discussion of microcontroller.

After the microcontroller decision has being performed, the microcontroller will check whether the mouse in is the forward position. This can be simplify as the next state condition for the mouse model to move. The next state condition is represent by the input signal (101)

- a. Microcontroller receives input from the receiver.
- b. Microcontroller will compare the input bits to the stored bits in the register.
- c. Input signal:[ 1 1 1 ] In this case three input signal is treated as 3 representation of binary digit. The most significant bit represents the left receiver signal. The least significant bits represent the right receiver signal.
- d. Each address will hole each binary representation of the microcontroller logic.

Example:

000 = 4 Junction
001 = T Left Junction
010 = T Junction
011 = Left corner
100 = T Right Junction
101 = Straight Line
110 = Right Corner
111 = Dead End

The signal received by the microcontroller will then compare the input signal with the data store in the addressed. If the data input is the same with the data store, it will perform the specific function.

Each address, which represents the condition, will give and output to the motor. This condition will refer to different addresses, which contain the instruction for the motor. At the end of the logic, the microcontroller will perform checking function, which will check whether the final position of the mouse correct.

For the circuit logic, there are basically 7 paths for the maze mouse model to be considered.

The path considered can be illustrated as follows.

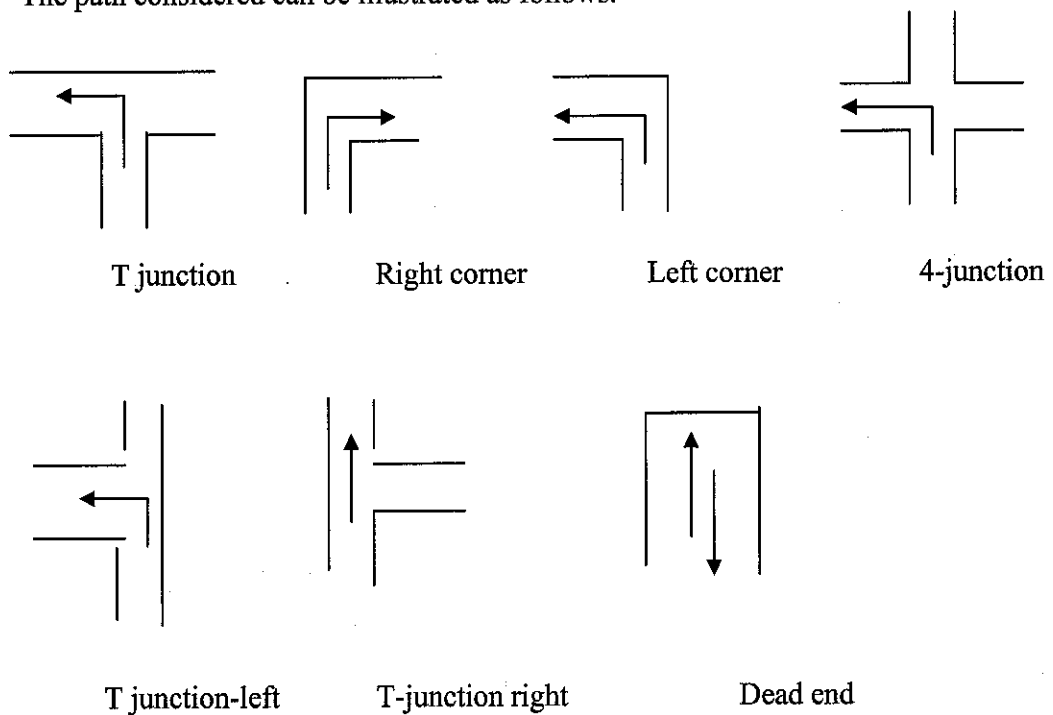


Figure 6 Logic of the path of micro controller programming

#### 4.1 Mouse Logic

Assume 2 motor is use to move the mouse. Each motor will control each side of the wheel of the mouse.

If mouse decision equals to move forward

- *The motors need to be rotating at the same direction*

If decision equals to turn left

- *Mouse need to turns 90 degree to the left.*
- *The left motor will still turning in the forward direction while the right motor will be turning in a reverse direction*

If decision equal to turn right

- *Mouse needs to turn 90 degree to the right*
- *The right motor will still turning in the forward direction while the left motor will be turning in a reverse direction*

If decision equal to turn in backward direction

- *Mouse need to turn 180 from either right or left.*
- *The right motor and the left motor must be rotating in a direction different form each other.*

There is a few important points need to be addressed in order for the logic to be working.

- a) Range of the mouse from the maze wall. This will help is adjusting the mouse position for turning purpose.
- b) Speed and rotation direction of the motor. This will help in smooth turning process of the model.

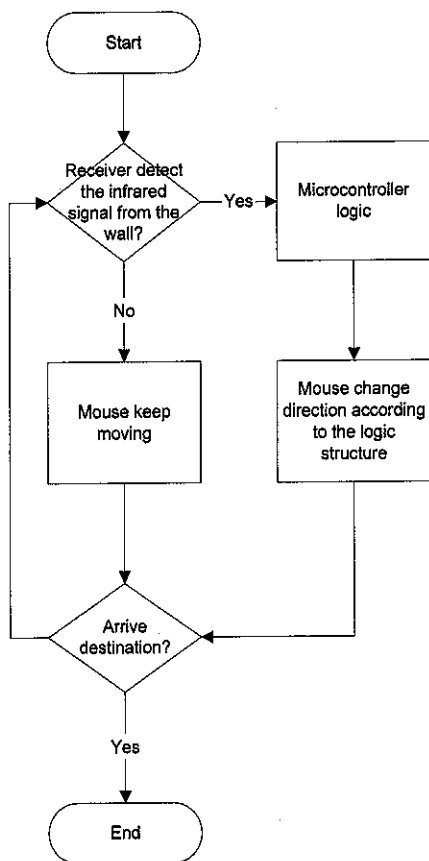


Figure 7 General flow of mouse control

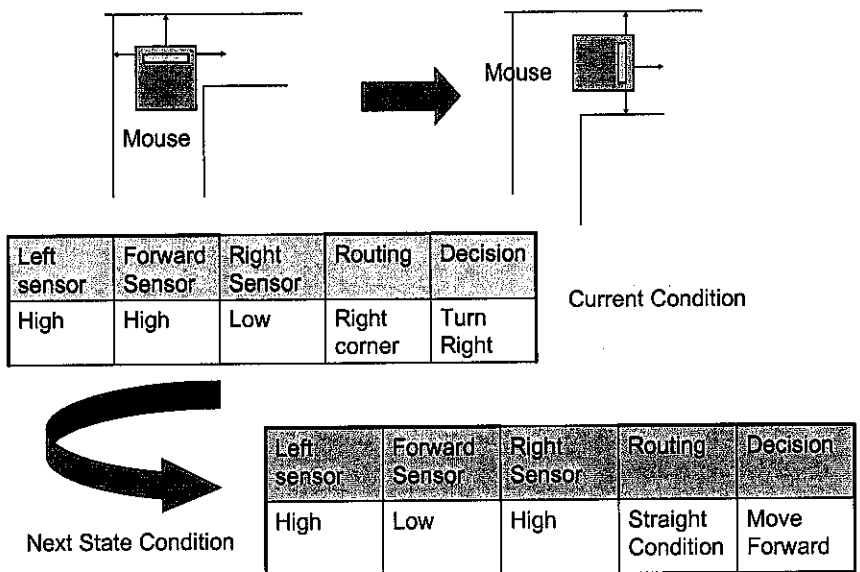


Figure 8 Example of Mouse Logic Algorithm

## 4.2 Maze Design

A few considerations have to be considered in constructing the maze. This includes the logic of the ways for the mouse operating system. There are two ways to construct the maze. One way is to build a reflected wall for the maze. Another way is to paint the board into black and white (silver) at the ground of the board. The second step is more reliable as it provides more convenient ways in the maze construction.

To trace the maze, the sensor of the model must be proportionally facing downward on the board or being placed slight in a degree, which can detect the reflection of the paint being placed at the ground. The maze system must be constructed according to the logic provided to the mouse logic.



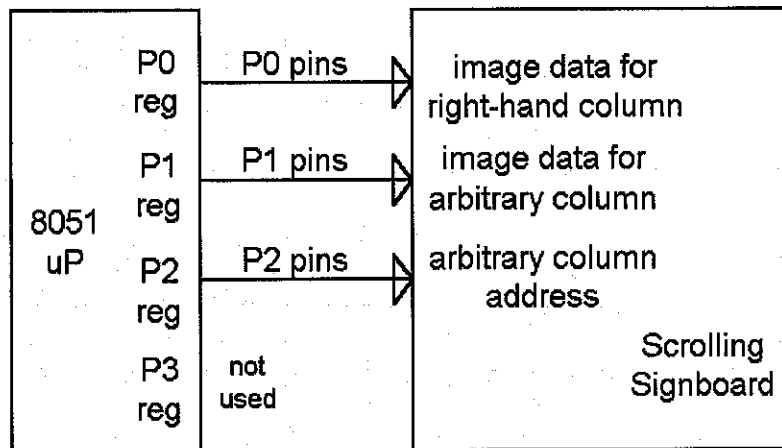
Silver / white surface reflect the infrared.      Black surface will absorbed the infrared

Figure 9 Reflection of infrared signal in different surface

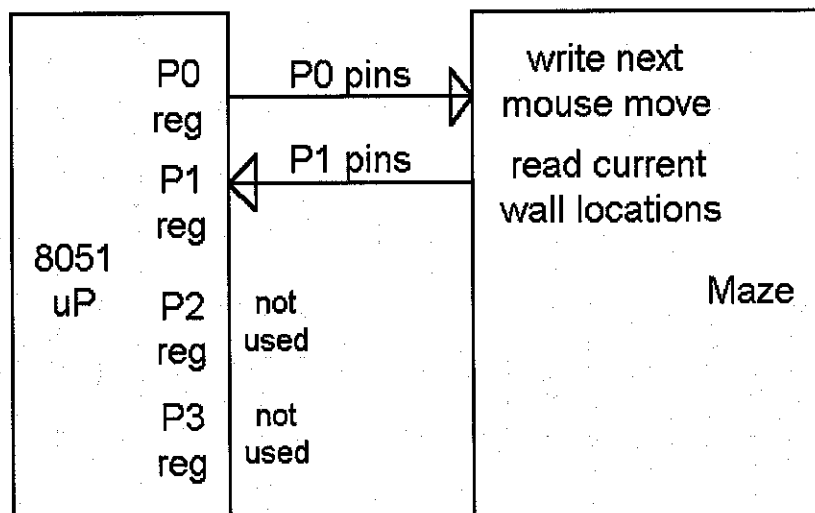
### 4.3 Example logic of mouse programming structure

The pins (wires) on the 8051 microprocessor that are controlled by the P0, P1, P2, and P3 registers are reconfigured from their signboard settings

#### Signboard Output Window



#### Maze Output Window



The 8051 microprocessor hasn't changed at all. But we are now connecting different circuitry to the 8051's general purpose input/output pins. We even get to decide which pins we will write (control) and which pins we will read (sense). This ability to employ the same microprocessor in many different circuits just by changing the wire connection.

By writing to the P0 register, we will be able to command the mouse to move. At any moment, there are potentially 4 directions the mouse can move in a maze: left, right, forward, or backward.

- To move left write the value 0x00 to P0.
- To move forward write the value 0x01 to P0.
- To move right write the value 0x02 to P0.
- To move backward write the value 0x03 to P0.
- Writing any other value to P0 causes no motion.

Note that because we reconfigured the simulator, the P0 register (or more properly the input/output pins on the 8051's package controlled by the P0 register) is no longer connected to the signboard but is now connected to a motorized mouse. Any of these 4 directions might be blocked by a wall. If you command the mouse to crash into a wall you will hear a beep and the mouse will stay where he is. You can learn which walls currently surround the mouse by reading from the P1 register.

The bits within the byte read from P1 have the following meaning:

MSBit	7	6	5	4	3	2	1	0	LSBit
	cheese	not	not	not	rear	right	front	left	
	used	used	used	wall	wall	wall	wall		



As an example, if you read the value 0x05 (00000101) from P1 then at the mouse's current position there is no cheese and there is a wall on both the left and right but no wall to the front or rear.

Note that wall sensing (bytes read from P1) and motion commands (bytes written to P0) employ the point of view of the mouse. This point of view (or "reference frame") keeps changing as the mouse explores the maze. For example, each time you command the mouse to move left, the mouse will both move to the square on his left and rotate 90 degrees counter-clockwise. Hence what used to be to his left is now to his front. The icon of the mouse rotates to remind you that the reference frame keeps changing.

While only some microprocessors have bit-addressable bits, all microprocessors can perform Boolean logic. Three basic Boolean operations are the AND, OR, and NOT operations (the NOT operation is also called the COMPLEMENT operation). The AND and OR operations require 2 operands which are both single bits. The NOT operation requires a single operand which is a single bit.

The **truth tables** below show the AND, OR, and NOT operations.

inputs	AND	inputs	OR	input	NOT
00	0	00	0	0	1
01	0	01	1	1	0
10	0	10	1		
11	1	11	1		

The AND of bit1 and bit2 is a 1 only if both bit1 and bit2 are one. The OR of bit1 and bit2 is a 1 if either bit1 or bit2 is a one. The COMPLEMENT of bit1 is 1 if bit1 is 0, and 0 otherwise. Each statement could only be true or false, and these two possibilities match up perfectly with bits, which have only two possible states: 0 or 1.

Here are some examples of how Boolean operations to reason about microprocessors:

- a) All microprocessors provide a way to store the steps of an algorithm (program memory) AND a way to store the operands affected by the algorithm (data memory).
- b) A jump is either conditional OR it is unconditional.
- c) The LJMP statement is NOT a conditional jump, hence it is an unconditional jump.

Getting back to the maze, the mouse should move to a new position if it is NOT already on the cheese. So we need 8051 program to perform a conditional jump based upon the most significant bit of the value read from P1.

One way to accomplish this is shown (Pls refer to appendix F for instruction code):

```
MOV A,P1           ; read the current status  
ANL A,#0x80       ; check for presence of cheese  
CJNE A,#0x80,Move ; if no cheese then move  
Eat: SJMP Eat     ; don't move, we're on the cheese  
Move:             ; need to go somewhere else
```

This code employs the ANL instruction which performs the logical AND function on the 8 bits of its operand and the 8 bits of the accumulator and places the result back in the accumulator. Let's imagine that the mouse had already landed on the cheese and the value read from the P1 register was 0x8C (10001100) meaning "there's cheese and a wall to the right and rear".

The command "MOV A,P1" then places the byte 0x8C into the accumulator. The ANL statement ANDs the accumulator with the byte 0x80. This is performed in binary below. Note how every bit is ANDed with its corresponding bit in the other operand.

```

    10001100 ; 0x8C, the value read from P1
    10000000 ; 0x80 the "mask" byte
AND -----
    10000000 ; the result

```

The value 0x80 used in the command "ANL A,#0x80" is called a **mask** byte because it is used to "mask out" all but the bits we are interested in. In this case we want to learn about the presence of cheese and hence we are only interested in the most significant bit. You can observe that there is only 1 bit asserted (equal to 1) in the byte 0x80.

If we AND 0x80 with any other byte then the result will definitely have 7 zeroes since 0 ANDed with either a 0 or a 1 will always produce a 0 (Please refer to the AND truth table).

This is illustrated below where, an 'x' to represent those bits that don't matter (that is, the outcome is the same regardless if these bits are a 0 or a 1).

```

Either read      0xxxxxxx or read 1xxxxxxx from P1.
When AND with 10000000      10000000
Get              00000000 or 10000000

```

7 of the bits read from P1 don't matter (and hence are labeled 'x') because when AND these bits with 0 which is guaranteed to clear these bits. So when AND the byte read from P1 with the mask byte 0x80 there are only two possible outcomes: the value 0x80 if the mouse is on cheese and the value 0x00 if are not.

In the illustration, we read 0x8C from P1 and the AND produced the value 0x80 indicating that we are on cheese.

In progress, we can now conclude that the mouse is on the cheese if and only if the value read from P1 when AND with 0x80 equals 0x80. How can we decide whether the accumulator holds 0x80. This is the job for the CJNE instruction that compares its first two operands and then takes the conditional jump only if these first two operands are not equal. Once the mouse lands on the cheese he will forever stay on the cheese because the CJNE command will not take the jump to the "Move" label and instead the 8051 will fall into the infinite loop:

Eat: SJMP Eat

This Eat loop will actually stop the mouse movement once the mouse found the cheese.

#### 4.4 Maze Mouse Program

In actual mouse program, each condition of the mouse logic is being perform by subroutine.

*Main:*

```
mov A, p1    ;read sensor status from p1 - load to accumulator
mov p2, #01h
mov p3, #01h
ljmp condition
```

This is the main subroutine of the program. P1 is define as port 1 of the microcontroller input. The input from the infra red sensor will go to port 1. The value of port 1 will then be loaded into the accumulator. The function of the accumulator is to temporary hold the input. Accumulator will store the value of P1.

P2 and P3 (Port 2 and Port 3) act as the microcontroller output to the motor. P2 output control the right motor, while p3 control the left motor of the model. The representation of hexadecimal, binary and decimal number for the input and output are listed in the table below.

Hexadecimal numbers	Binary equivalent	Decimal equivalent
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
A	1010	10

B	1011	11
C	1100	12
D	1101	13
E	1110	14
F	1111	15

Table 2 Representation of hexadecimal, decimal and binary value equivalent

Mov p2, #01h indicates that the value of 1 is being push to output p2. In binary representation 0 0 0 0 0 0 0 1 which represent the least significant bit in 8-bit will be an output to the motor. The motor driver will read the value of the least significant bit of the p2 output. If the value of the least significant bit is 1, the motor will on and rotate in forward direction.

After the execution of the subroutine of main program, it will perform a long jump condition to condition subroutine.

*condition:*

*cjne A, #00h, condition1 ; check for 4 junction*

*ljmp status*

The condition subroutine is basically act as a checker condition which read he value of the accumulator. It will differentiate the condition receive from the infrared receiver whether; the mouse is in one of the 7 defined path defined.

Hex Condition	Binary Representation	Actual Path
#00h	0000 0000	4-Junction
#01h	0000 0001	T left Corner
#02h	0000 0010	T Junction
#03h	0000 0011	Left Corner
#04h	0000 0100	T Junction
#05h	0000 0101	Straight Line
#06h	0000 0110	Right Corner
#07h	0000 0111	Dead End

Table 3 Condition subroutine path check

The condition subroutine, check the 3 least significant bit of the binary input from the infra red sensor. If the binary inputs represent value 1 means there is a reflection from the infra red sensor.

`cjne A, #00h`, condition subroutine actually compare the value of current input of the accumulator to the hex condition define in the middle of the command. If the value of the accumulator is not the same as the hex number, it will go on the condition1 subroutine which is another checker for other path of the mouse model. If the accumulator value is the same as the hex value, it will jump to status subroutine.

*status:*

```

mov p2, #01h ; mouse turn left
mov p3, #08h
mov A, p1 ; read current input from p1
cjne A, #05h, status
ljmp main

```

The status subroutine, give an output to the motor by using p2 and p3. mov p3, #08h indicates that the binary value of 0000 1000. The fourth binary bits are taken to be an input to the left motor to indicate that it will be rotating backwards.

cjne A, #05h, main, compare the current value of accumulator to the next state condition of the mouse logic which is the straight logic. If the condition doesn't been met, the status subroutine will loop until the condition being met. If the condition of the accumulator is the same as the next state condition, then the commands will jump to the main subroutine once more.

The logic of mouse rotation is being list:

#### **Mouse move forward**

mov p2, #01h ; port 2 to right motor (0000 0001) – forward direction

mov p3, #01h ; port 3 to left motor (0000 0001) – forward direction

#### **Mouse turn left**

mov p2, #01h ; port 2 to right motor (0000 0001) – forward direction

mov p3, #08h ; port 3 to left motor (0000 1000) – reverse direction

#### **Mouse turn right**

mov p2, #08h ; port 2 to right motor (0000 1000) – reverse direction

mov p3, #01h ; port 3 to left motor (0000 0001) - forward direction

#### **Mouse stop**

mov p2, #00h ; port 2 to right motor (0000 0000)

mov p3, #00h ; port 3 to left motor (0000 0000)

The program will end, once destination is reach. (Refer appendix E)



## **CHAPTER 5**

### **CONCLUSION AND RECOMMENDATION**

In project development and completion, valuable knowledge and experience can be grasp. The project, require understanding in circuit design mostly in analog and digital side, and also from past syllabus being learn in UTP. The planning stages of the project, concentrate on the hardware portion as well as the software portion using machine code. The essential attributes of the project is to relate each of the circuit into one whole component as well as implementing the code together with the model.

Overall view, the project didn't able to perform as well as the theoretical part being discuss in the report. This is mainly because the wiring connection which resides in the circuit during the development process. Due to this problem, the integration of the circuit did not perform as well as expected and this affect the performance of the mouse model. It is essential to perform brilliantly in the theoretical part as well as the practical aspect of the mouse model. The practical aspect of the model concentrate more on how the connection being made, hands-on on the circuit connection, knowledge on the physical component, and also other factors need to be consider in the circuit construction.

Several knowledgeable attributes that can be learn on the project is basically focus more on the practical circuit construction. This is actually important in differentiate between the theoretical aspect than the practical aspect of the project and how it's relate with each other. Other view, the writer able to learn on the microcontroller part as well as the programming aspect of it by using the machine code. With all the studies done in this part, the essential application of microcontroller can be seen in industry and also in everyday live. It is a good practice in actually involve in a project such as this, in order to enhance the knowledge and experience of the students in developing their career as an engineer.

## 5.1 Recommendation

In involving with the project, there are several aspect that can be improve in term of the project and the management. For future reference, the project should be conducted in group of 2 person. This is mainly due to the cost of the hardware needed as well as the project model resources which can make use of the time for both students to actually involve in the project and share the load of concentrating the physical construction of the model.

In term of the design, several design criteria can be improved. In order to improve the efficiency of the sensor reflection and the receiver, the device can be wrapped to improve its efficiency in receiving the infrared signal. In term of construction part, since there is a development in the component such as microcontroller technology and such, it is recommended for future project of the maze mouse model to use such technology provided.

Concentrating on the design sensor, apart using the infrared sensor other sensor such as touch sensor, wave and other sensor can also be use in the project. The usage of infrared sensor, is to sensitive in the design stage. This is due to the interference of the light surrounding that affect the performance of the receiver sensor in detecting the infrared reflection.

In term of the hardware, several components with the same function can be replaced in the mouse maze model. Instead of using microcontroller, PLC (Programmable Logic Controller) also can be use. This implies the same in motor selection where DC motor can also be implemented.

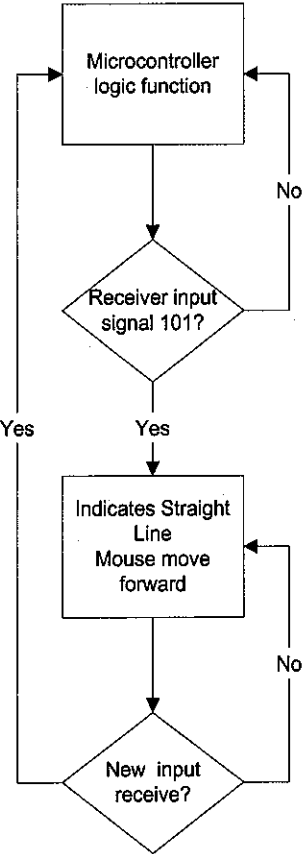
Overall evaluation, the project is basically a good project to implement for Final Year Project, Electrical Electronics department. It is hope that this project can be continue for years to come.

## REFERENCES

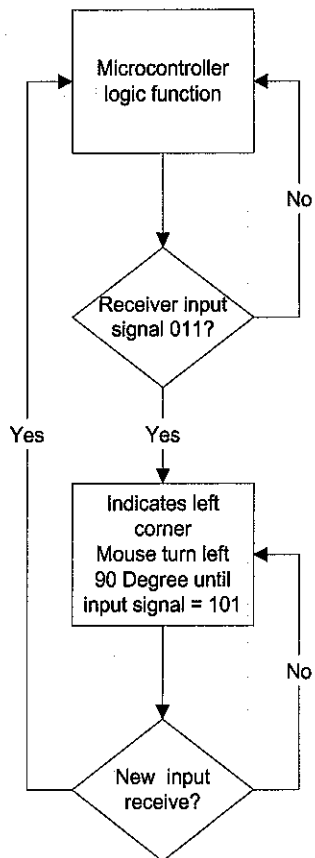
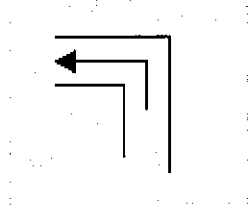
- [1] <http://murray.newcastle.edu.au/users/students/2001/c9806617/index>
- [2] <http://www.iee.org.uk/Micromouse/>
- [3] <http://micromouse.cs.rhul.ac.uk/>
- [4] <http://homepages.uel.ac.uk/C.kanesalingam/mousehp.htm>
- [5] <http://www.smartdata.com.au/8051/default.htm>
- [6] <HTTP://WWW.LBORO.AC.UK/DEPARTMENTS/EL/ROBOTICS/MOUSELAB>
- [7] The 8051 Micro controller, (Second Edition) Kenneth J. Ayala, 1997

**APPENDIX A**  
**BASIC FLOW DIAGRAM OF OTHER MOUSE LOGIC**

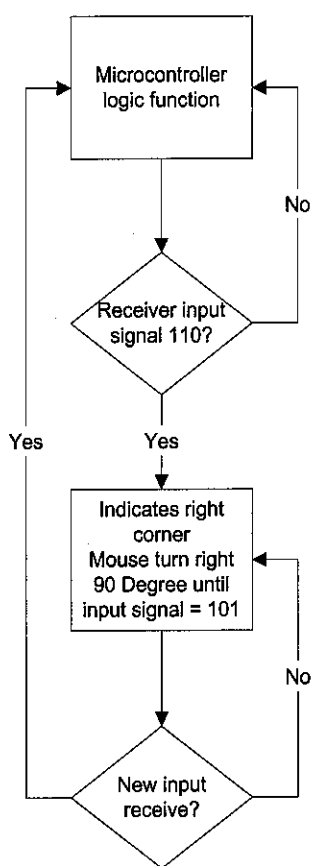
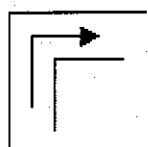
*Straight logic*



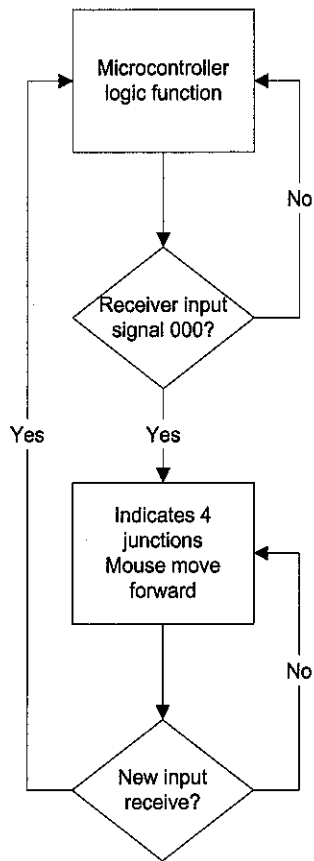
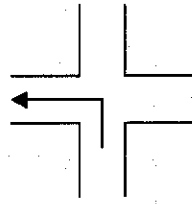
*Left corner*



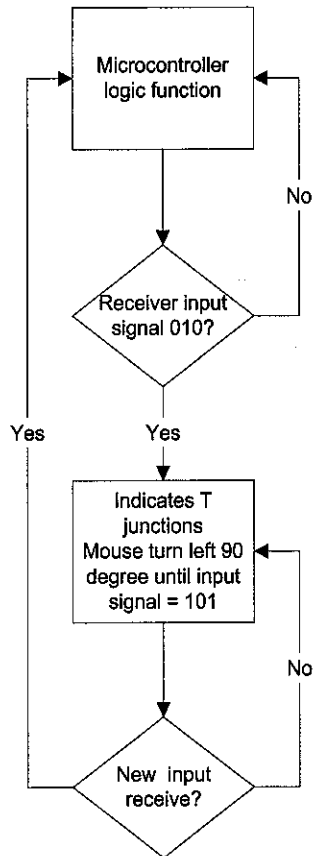
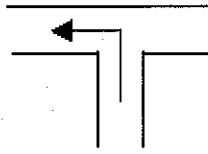
*Right Corner*



#### 4 - Junctions

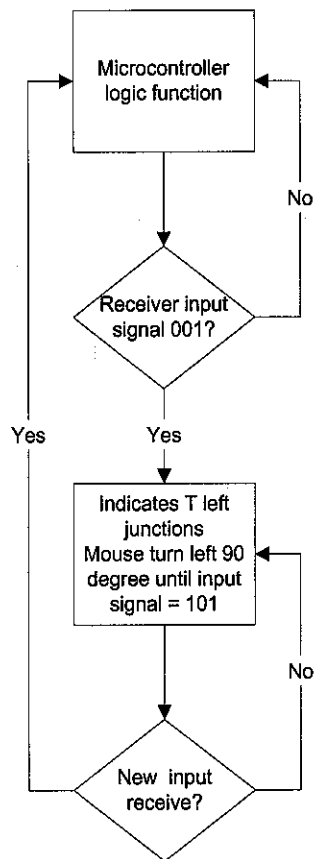
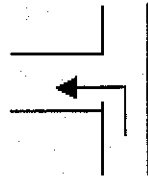


*T Junction*

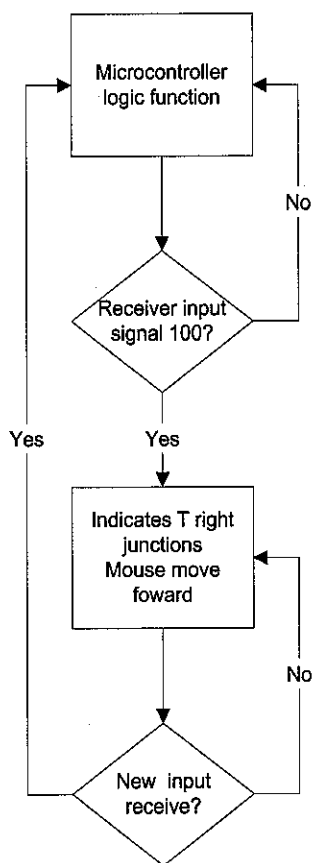
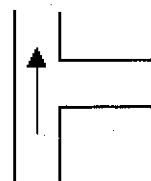




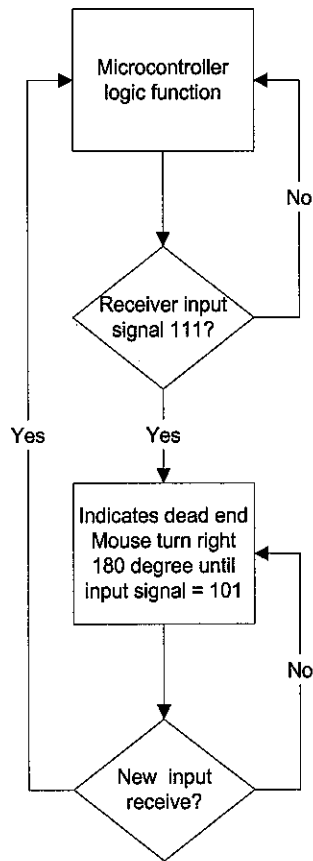
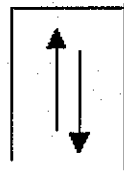
## *T Left Junction*



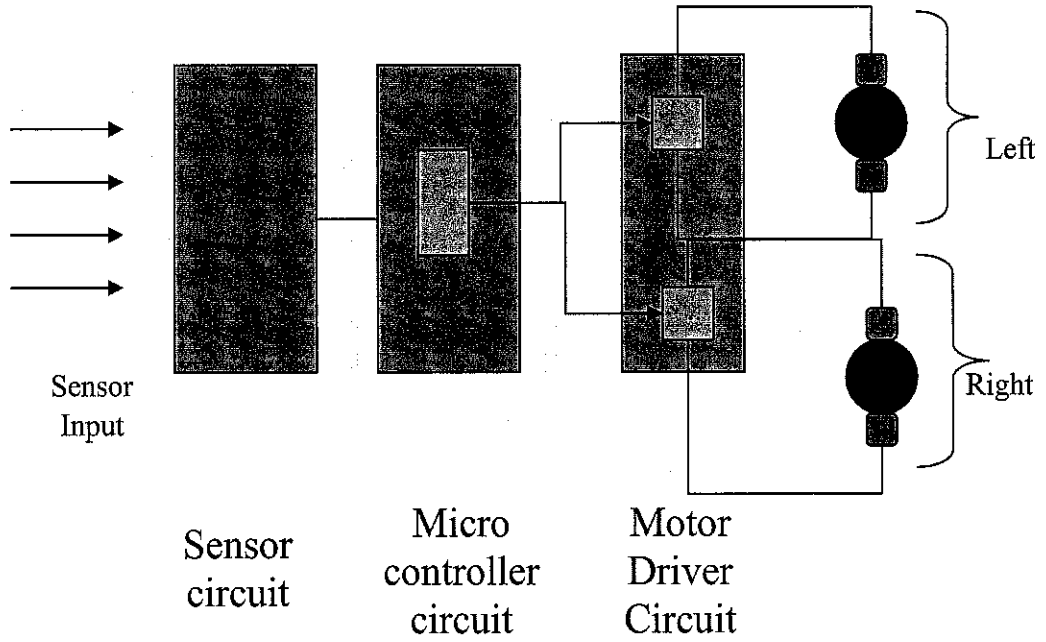
*T right junction*



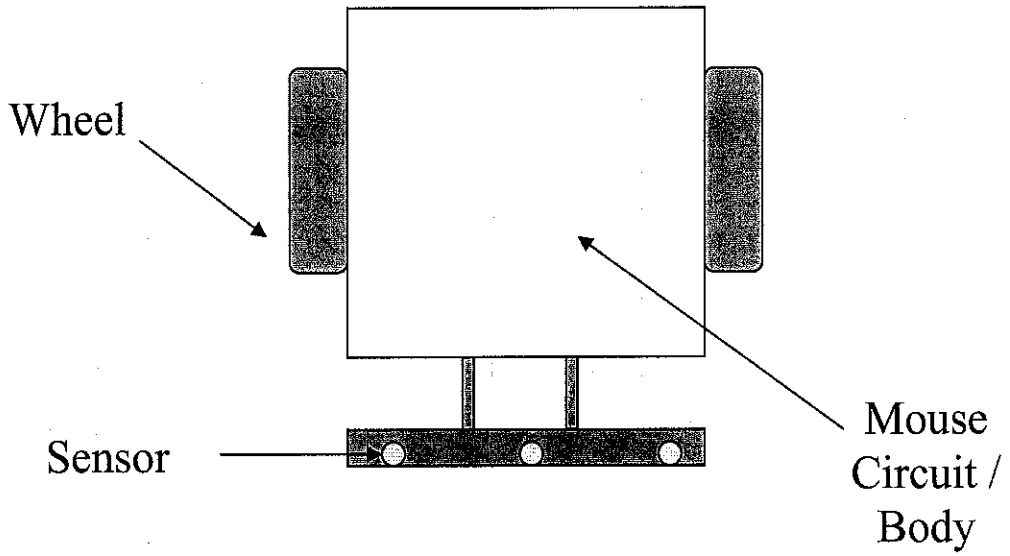
*Dead End*



# APPENDIX B RELATIONSHIP OF CIRCUIT COMPONENT

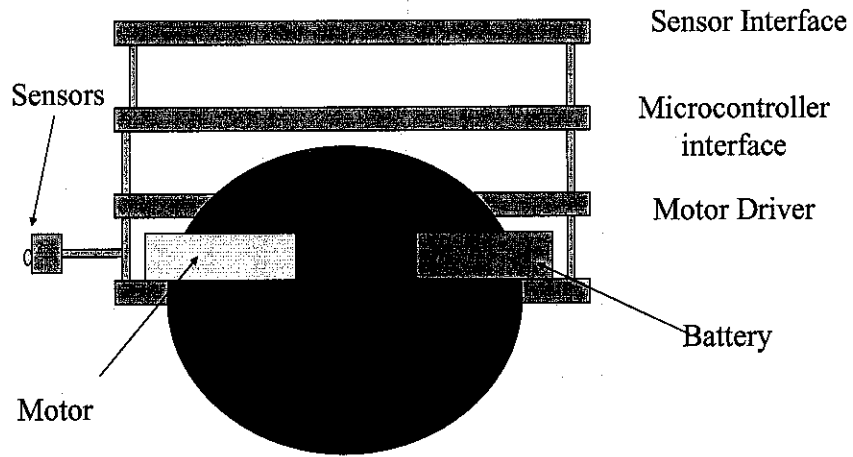


**APPENDIX C**  
**TOP VIEW OF MOUSE MODEL**



# APPENDIX D

## SIDE VIEW OF MOUSE MODEL



## APPENDIX E MOUSE PROGRAMMING

main:

```
mov A, p1    ;read sensor status from p1 - load to accumulator
mov p2, #01h
mov p3, #01h
ljmp condition
```

condition:

```
cjne A, #00h, condition1    ; check for 4 junction
ljmp status
```

status:

```
mov p2, #01h ; mouse turn left
mov p3, #08h
mov A, p1    ; read current input from p1
cjne A, #05h, main
ljmp status
```

condition1:

```
cjne A, #01h, condition2    ; check 4 T left corner
ljmp status1
```

status1

```
mov p2, #01h ; mouse turn left
mov p3, #08h
mov A, p1    ; read current input from p1
cjne A, #05h, main
ljmp status1
```

condition2:

```
    cjne A, #02h, condition3    ; check 4 T junction
    ljmp status2
```

status2

```
    mov p2, #01h ; mouse turn left
    mov p3, #08h
    mov A, p1    ; read current input from p1
    cjne A, #05h, main
    ljmp status2
```

condition3:

```
    cjne A, #03h, condition4    : check 4 left corner
    ljmp status3
```

status3

```
    mov p2, #01h ; mouse turn left
    mov p3, #08h
    mov A, p1    ; read current input from p1
    cjne A, #05h, main
    ljmp status3
```

condition4:

```
    cjne A, #04h, condition5    ; check 4 T right junction
    lmjp status4
```

status4:

```
    mov p2, #08h ; mouse turn right
    mov p3, #01h
    mov A, p1    ; read current input from p1
    cjne A, #05h, main
    ljmp status4
```



condition5:

```
    cjne A, #05h, condition6    ; check 4 straight line
    ljmp status5
```

status5:

```
    mov p2, #01h ; port 2 to right motor
    mov p3, #01h ; port 3 to left motor
    mov A, p1    ; read current input from p1
    cjne A, #05h, main
    ljmp status5
```

condition6:

```
    cjne A, #06h, condition7    ; check for right corner
    ljmp status6;
```

status6:

```
    mov p2, #08h ;mouse turn right
    mov p3, #01h
    mov A, p1    ; read current input from p1
    cjne A, #05h, main
    ljmp status6
```

condition7:

```
    cjne A, #07h, condition    ; check for dead end
```

status7:

```
    mov p2, #08h ;mouse turn right
    mov p3, #01h
    mov A, p1    ; read current input from p1
    cjne A, #05h, main
    ljmp status7
```

## APPENDIX F INSTRUCTION SET

<b><u>ACALL</u></b>	Absolute Call
<b><u>ADD</u></b>	Add to Accumulator (without carry)
<b><u>ADDC</u></b>	Add to Accumulator with Carry
<b><u>AJMP</u></b>	Absolute Jump
<b><u>ANL</u></b>	AND
<b><u>ANL C</u></b>	AND with Carry bit
<b><u>CJNE</u></b>	Compare and Jump if Not Equal
<b><u>CLR A</u></b>	Clear Accumulator
<b><u>CLR bit</u></b>	Clear Bit
<b><u>CPL A</u></b>	Complement Accumulator
<b><u>CPL bit</u></b>	Complement Bit
<b><u>DA A</u></b>	Decimal Adjust Accumulator for Addition
<b><u>DEC</u></b>	Decrement
<b><u>DIV AB</u></b>	Divide Accumulator by B register
<b><u>DJNZ</u></b>	Decrement and Jump if Not Zero
<b><u>INC</u></b>	Increment
<b><u>INC DPTR</u></b>	Increment Data Pointer
<b><u>JB</u></b>	Jump if Bit is set
<b><u>JBC</u></b>	Jump if Bit is set and Clear
<b><u>JC</u></b>	Jump if Carry is set
<b><u>JMP</u></b>	Jump Indirect
<b><u>JNB</u></b>	Jump if Not Bit
<b><u>JNC</u></b>	Jump if Not Carry

<b><u>JNZ</u></b>	Jump if Accumulator is Not Zero
<b><u>JZ</u></b>	Jump if Accumulator is Zero
<b><u>LCALL</u></b>	Long Call
<b><u>LJMP</u></b>	Long Jump
<b><u>MOV</u></b>	Move Byte
<b><u>MOV bit</u></b>	Move Bit
<b><u>MOV DPTR</u></b>	Move Data Pointer
<b><u>MOVC</u></b>	Move Code Byte
<b><u>MOVB</u></b>	Move External Byte
<b><u>MUL AB</u></b>	Multiply Accumulator and B register
<b><u>NOP</u></b>	No Operation
<b><u>ORL</u></b>	OR
<b><u>ORL C</u></b>	OR with Carry bit
<b><u>POP</u></b>	Pop from Stack
<b><u>PUSH</u></b>	Push onto Stack
<b><u>RET</u></b>	Return from Subroutine
<b><u>RETI</u></b>	Return from Interrupt
<b><u>RL A</u></b>	Rotate Accumulator Left
<b><u>RLC A</u></b>	Rotate Accumulator Left thru Carry
<b><u>RR A</u></b>	Rotate Accumulator Right
<b><u>RRC A</u></b>	Rotate Accumulator Right thru Carry
<b><u>SETB</u></b>	Set Bit
<b><u>SJMP</u></b>	Short Jump
<b><u>SUBB</u></b>	Subtract with Borrow
<b><u>SWAP A</u></b>	Swap Nibbles within the Accumulator
<b><u>XCH</u></b>	Exchange Accumulator with Byte

<u>XCHD</u>	Exchange Digit
<u>XRL</u>	Exclusive OR (XOR)

## MOV

<b>Function:</b>	Move byte variable
<b>Description:</b>	The byte variable indicated by the second operand is copied into the location specified by the first operand. The source byte is not affected. No other register or flag is affected.

## LJMP

<b>Function:</b>	Long Jump
<b>Description:</b>	LJMP causes an unconditional branch to the indicated address, by loading the high-order and low-order bytes of the PC (respectively) with the second and third instruction bytes. The destination may therefore be anywhere in the full 64 Kbyte program memory address space. No flags are affected

## SJMP

<b>Function:</b>	Short Jump
<b>Description:</b>	Program control branches unconditionally to the address indicated. The branch destination is computed by adding the signed displacement in the second instruction byte to the PC, after incrementing the PC twice. Therefore, the range of destinations allowed is from 128 bytes preceding this instruction to 127 bytes following it.

## CJNE

<b>Function:</b>	Compare and Jump if Not Equal
<b>Description:</b>	CJNE compares the magnitudes of the first two operands, and branches if their values are not equal. The branch destination is computed by adding the signed relative-displacement in the last instruction byte to the PC, after incrementing the PC to the start of the next instruction. The carry flag is set if the unsigned integer value of the first operand is less than the unsigned integer value of the second operand; otherwise, the carry is cleared. Neither operand is affected. The first two operands allow four addressing mode combinations: the Accumulator may be compared with any directly addressed byte or immediate data, and any indirectly addressed RAM location or working register can be compared with an immediate constant.

## ANL

<b>Function:</b>	Logical-AND for byte variables
<b>Description:</b>	ANL performs the bitwise logical-AND operation between the indicated variables and stores the result in the destination variable. No flags are affected.

**APPENDIX G  
DATASHEETS**

# LM2901, LM339/LM339A, LM3302

## LM239/LM239A

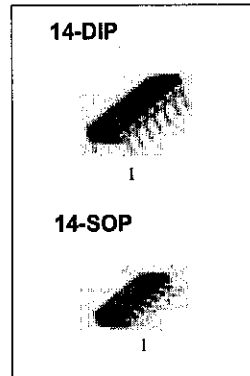
### Quad Comparator

#### Features

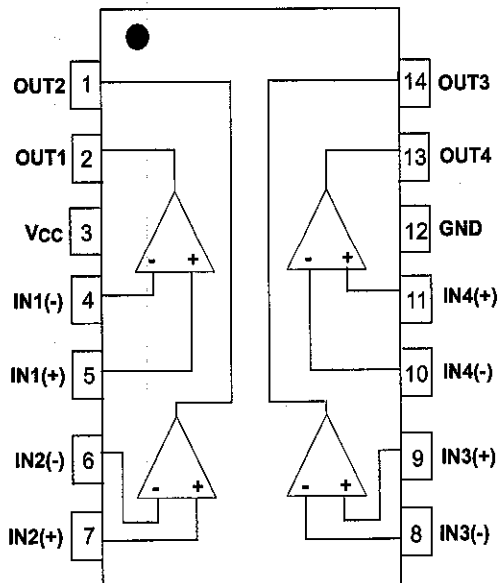
- Single or Dual Supply Operation
- Wide Range of Supply Voltage  
LM2901, LM339/LM339A, LM239/LM239A: 2 ~ 36V (or  $\pm 1 \sim \pm 18V$ )  
LM3302: 2 ~ 28V (or  $\pm 1 \sim \pm 14V$ )
- Low Supply Current Drain 800 $\mu A$  Typ.
- Open Collector Outputs for Wired and Connectors
- Low Input Bias Current 25nA Typ.
- Low Input Offset Current  $\pm 2.3nA$  Typ.
- Low Input Offset Voltage  $\pm 1.4mV$  Typ.
- Input Common Mode Voltage Range Includes Ground.
- Low Output Saturation Voltage
- Output Compatible With TTL, DTL and MOS Logic System

#### Description

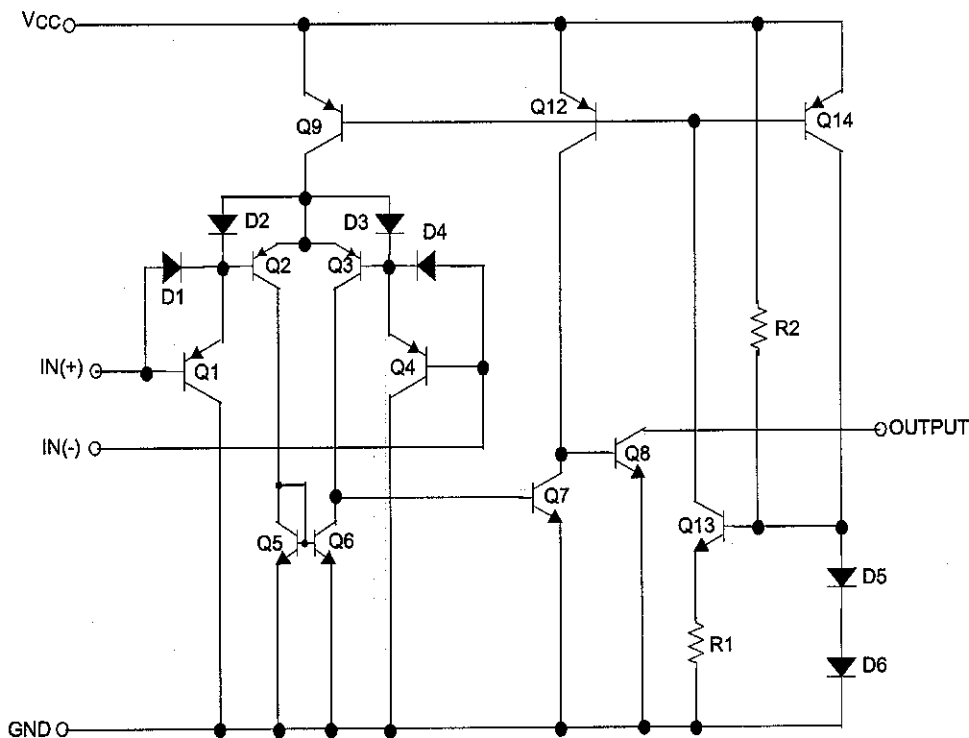
The LM2901, LM339/LM339A, LM239/LM239A, LM3302 consist of four independent voltage comparators designed to operate from single power supply over a wide voltage range.



#### Internal Block Diagram



## Schematic Diagram



## Absolute Maximum Ratings

Parameter	Symbol	Value	Unit
Supply Voltage	VCC	±18 or 36	V
Supply Voltage only LM3302	VCC	±14 or 28	V
Differential Input Voltage	V <sub>I(DIFF)</sub>	36	V
Differential Input Voltage Only LM3302	V <sub>I(DIFF)</sub>	28	V
Input Voltage	V <sub>I</sub>	-0.3 to +36	V
Input Voltage Only LM3302	V <sub>I</sub>	-0.3 to +28	V
Output Short Circuit to GND	-	Continuous	-
Power Dissipation	P <sub>D</sub>	570	mW
Operating Temperature LM339/LM339A LM2901/LM3302 LM239/LM239A	T <sub>OPR</sub>	0 ~ +70 -40 ~ +85 -25 ~ +85	°C
Storage Temperature	T <sub>STG</sub>	-65 ~ +150	°C



## Electrical Characteristics

( $V_{CC} = 5V$ ,  $T_A = 25^\circ C$ , unless otherwise specified)

Parameter	Symbol	Conditions	LM239A/LM339A			LM239/LM339			Unit
			Min.	Typ.	Max.	Min.	Typ.	Max.	
Input Offset Voltage	$V_{IO}$	$V_{O(P)} = 1.4V$ , $R_S = 0\Omega$	-	1	2	-	1.4	5	mV
		Note1	-	-	4.0	-	-	9.0	
Input Offset Current	$I_{IO}$	$I_{IN(+)} - I_{IN(-)}$ , $V_{CM} = 0V$	-	2.3	50	-	2.3	50	nA
		Note1	-	-	150	-	-	150	
Input Bias Current	$I_{BIAS}$	$V_{CM} = 0V$	-	57	250	-	57	250	nA
		Note1	-	-	400	-	-	400	
Input Common Mode Voltage Range	$V_{I(R)}$	$V_{CC} = 30V$	0	-	$V_{CC}-1.5$	0	-	$V_{CC}-1.5$	V
		Note1	0	-	$V_{CC}-2$	0	-	$V_{CC}-2$	
Supply Current	$I_{CC}$	$V_{CC} = 5V$ , $R_L = \infty$	-	1.1	2.0	-	1.1	2.0	mA
Voltage Gain	$G_V$	$V_{CC} = 15V$ , $R_L \geq 15k\Omega$ (for large swing)	50	200	-	50	200	-	V/mV
Large Signal Response Time	$T_{LRES}$	$V_I = \text{TTL Logic Swing}$ $V_{REF} = 1.4V$ , $V_{RL} = 5V$ , $R_L = 5.1k\Omega$ (Note2)	-	300	-	-	300	-	ns
Response Time	$T_{RES}$	$V_{RL} = 5V$ , $R_L = 5.1k\Omega$ (Note2)	-	1.3	-	-	1.3	-	$\mu s$
Output Sink Current	$I_{SINK}$	$V_{I(-)} \geq 1V$ , $V_{I(+)} = 0V$ , $V_{O(P)} \leq 1.5V$	6	18	-	6	18	-	mA
Output Saturation Voltage	$V_{SAT}$	$V_{I(-)} \geq 1V$ , $V_{I(+)} = 0V$ $I_{SINK} = 4mA$	-	140	400	-	140	400	mV
		Note1	-	-	700	-	-	700	
Output Leakage Current	$I_{o(LKG)}$	$V_{I(-)} = 0V$	-	0.1	-	-	0.1	-	nA
		$V_{I(+)} = 1V$	-	-	1.0	-	-	1.0	$\mu A$
Differential Voltage	$V_{I(DIFF)}$	Note1	-	-	36	-	-	36	V

### Note:

- LM339/LM339A :  $0 \leq T_A \leq +70^\circ C$   
LM2901/LM3302 :  $-40 \leq T_A \leq +85^\circ C$   
LM239/LM239A :  $-25 \leq T_A \leq +85^\circ C$
- These parameters, although guaranteed, are not 100% tested in production.

**Electrical Characteristics** (Continued)(V<sub>CC</sub> = 5V, T<sub>A</sub> = 25°C, unless otherwise specified)

Parameter	Symbol	Conditions	LM2901			LM3302			Unit
			Min.	Typ.	Max.	Min.	Typ.	Max.	
Input Offset Voltage	V <sub>IO</sub>	V <sub>O(P)</sub> = 1.4V, R <sub>S</sub> = 0Ω	-	2	7	-	2	20	mV
		Note1	-	9	15	-	-	40	
Input Offset Current	I <sub>IO</sub>		-	2.3	50	-	3	100	nA
		Note1	-	50	200	-	-	300	
Input Bias Current	I <sub>BIAS</sub>		-	57	250	-	57	250	nA
		Note1	-	200	500	-	-	1000	
Input Common Mode Voltage Range	V <sub>I(R)</sub>	LM2901, V <sub>CC</sub> = 30V LM3302, V <sub>CC</sub> = 28V	0	-	V <sub>CC</sub> - 1.5	0	-	V <sub>CC</sub> - 1.5	V
		Note1	0	-	V <sub>CC</sub> - 2	0	-	V <sub>CC</sub> - 2	
Supply Current	I <sub>CC</sub>	R <sub>L</sub> = ∞, V <sub>CC</sub> = 5V	-	1.1	2.0	-	1.1	2.0	mA
		R <sub>L</sub> = ∞, V <sub>CC</sub> = 30V (LM3302, V <sub>CC</sub> = 28V)	-	1.6	2.5	-	1.6	2.5	
Voltage Gain	G <sub>V</sub>	V <sub>CC</sub> = 15V, R <sub>L</sub> ≥ 15kΩ (for large swing)	25	100	-	2	30	-	V/ mV
Large Signal Response Time	T <sub>LRES</sub>	V <sub>I</sub> = TTL Logic Swing V <sub>REF</sub> = 1.4V, V <sub>RL</sub> = 5V, R <sub>L</sub> = 5.1kΩ (Note2)	-	300	-	-	300	-	ns
Response Time	T <sub>RES</sub>	V <sub>RL</sub> = 5V, R <sub>L</sub> = 5.1kΩ (Note2)	-	1.3	-	-	1.3	-	μs
Output Sink Current	I <sub>SINK</sub>	V <sub>I(-)</sub> ≥ 1V, V <sub>I(+)</sub> = 0V, V <sub>O(P)</sub> ≤ 1.5V	6	18	-	6	18	-	mA
Output Saturation Voltage	V <sub>SAT</sub>	V <sub>I(-)</sub> ≥ 1V, V <sub>I(+)</sub> = 0V I <sub>SINK</sub> = 4mA	-	140	400	-	140	400	mV
		Note1	-	-	700	-	-	700	
Output Leakage Current	I <sub>O(LKG)</sub>	V <sub>I(-)</sub> = 0V V <sub>I(+)</sub> = 1V	-	0.1	-	-	0.1	-	nA
		V <sub>O(P)</sub> = 5V V <sub>O(P)</sub> = 30V	-	-	1.0	-	-	1.0	μA
Differential Voltage	V <sub>I(DIFF)</sub>	Note1	-	-	36	-	-	28	V

**Note:**

- LM339/LM339A : 0 ≤ T<sub>A</sub> ≤ +70°C  
LM2901/LM3302 : -40 ≤ T<sub>A</sub> ≤ +85°C  
LM239/LM239A : -25 ≤ T<sub>A</sub> ≤ +85°C
- These parameters, although guaranteed, are not 100% tested in production.

# Typical Performance Characteristics

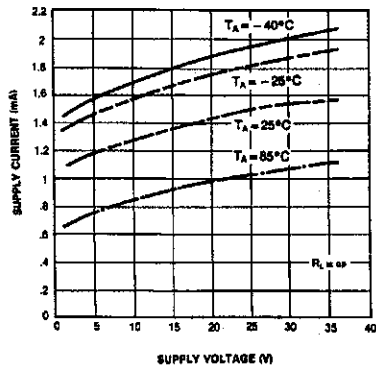


Figure 1. Supply Current vs Supply Voltage

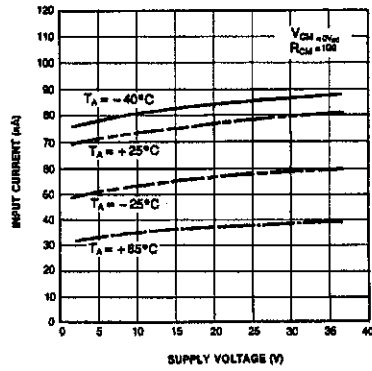


Figure 2. Input Current vs Supply Voltage

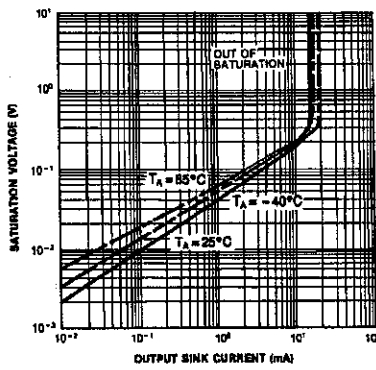


Figure 3. Output Saturation Voltage vs Sink Current

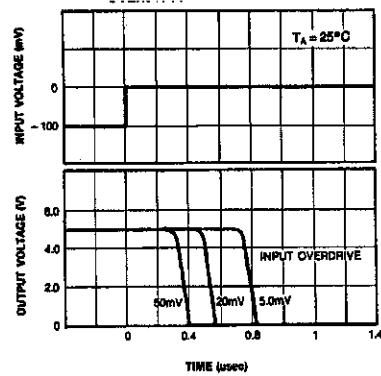


Figure 4. Response Time for Various Input Overdrive-Negative Transition

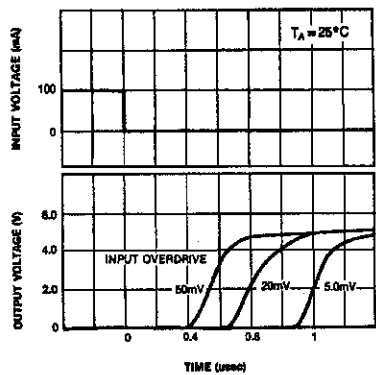


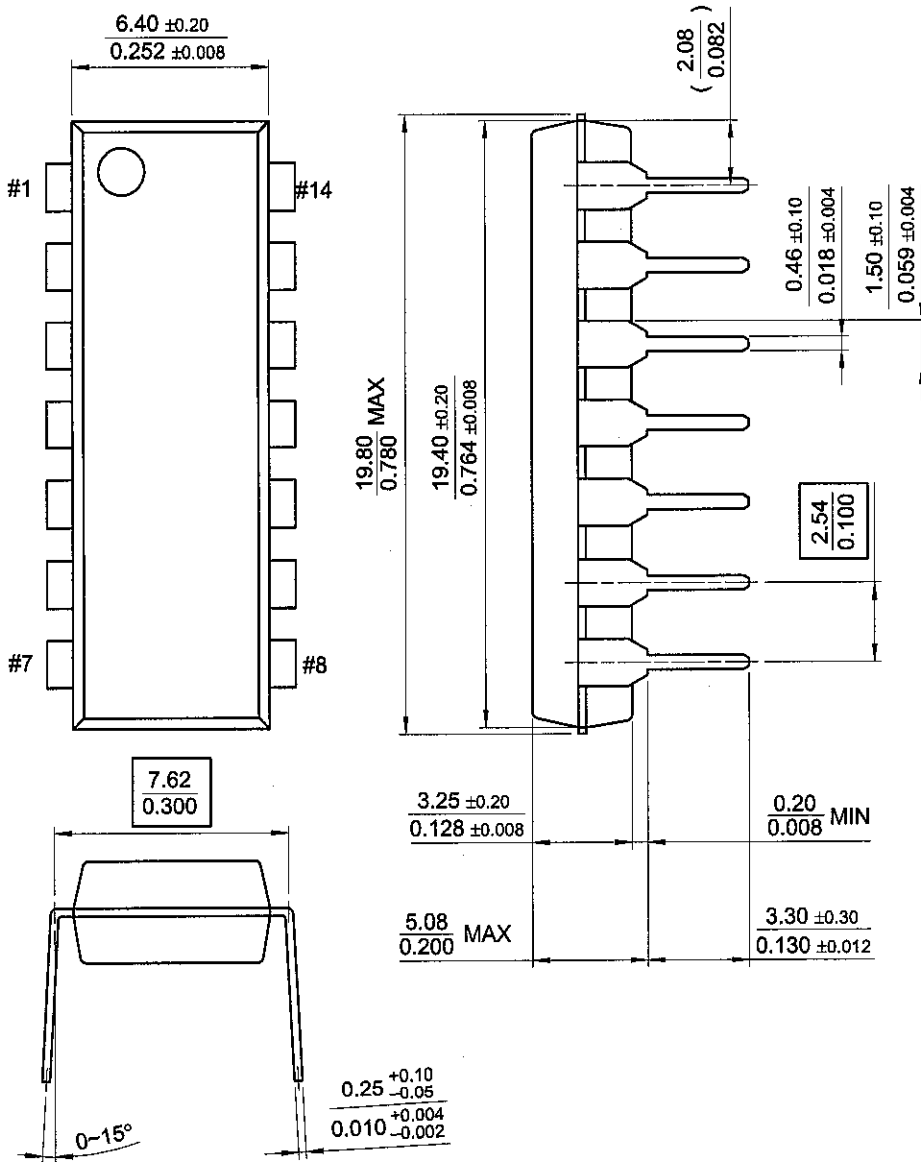
Figure 5. Response Time for Various Input Overdrive-Positive Transition

# Mechanical Dimensions

## Package

Dimensions in millimeters

### 14-DIP

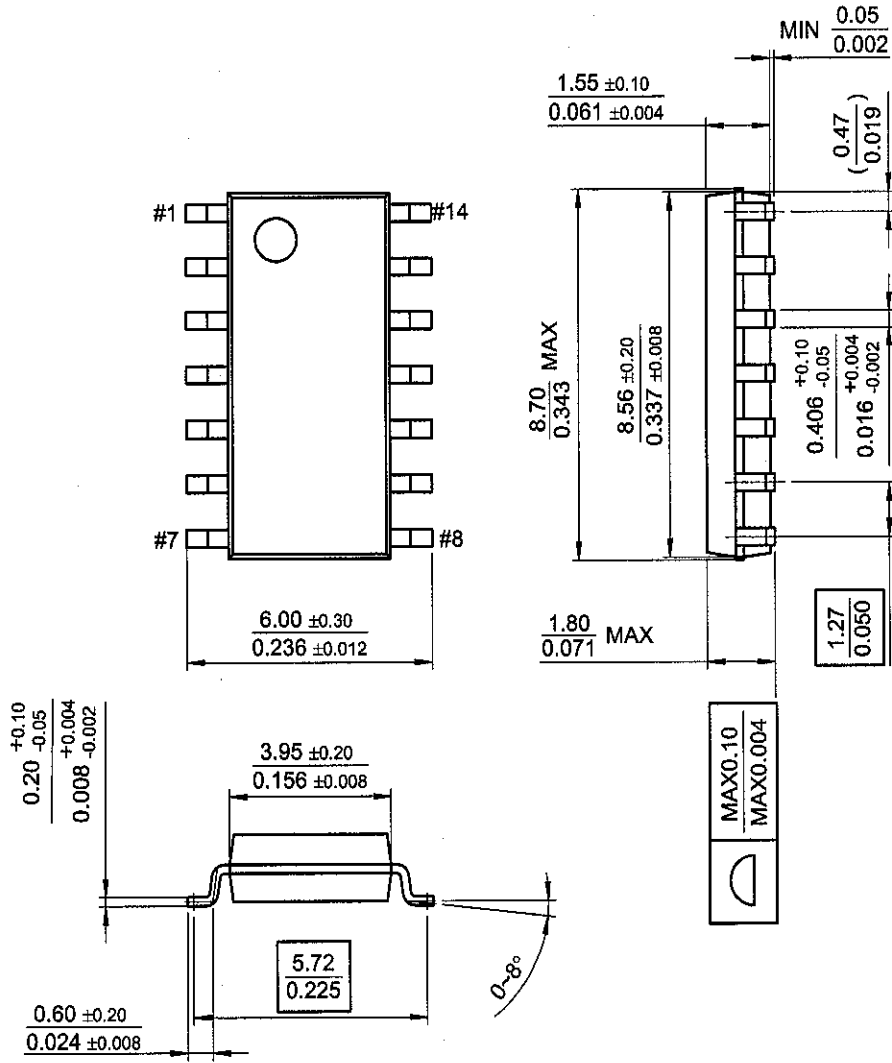


Mechanical Dimensions (Continued)

Package

Dimensions in millimeters

14-SOP



## Ordering Information

Product Number	Package	Operating Temperature
LM339N	14-DIP	0 ~ +70°C
LM339AN		
LM339M	14-SOP	
LM339AM		
LM2901N	14-DIP	-40 ~ +85°C
LM2901M	14-SOP	
LM3302N	14-DIP	
LM3302M	14-SOP	
LM239N	14-DIP	-25 ~ +85°C
LM239AN		
LM239M	14-SOP	
LM239AM		

### DISCLAIMER

FAIRCHILD SEMICONDUCTOR RESERVES THE RIGHT TO MAKE CHANGES WITHOUT FURTHER NOTICE TO ANY PRODUCTS HEREIN TO IMPROVE RELIABILITY, FUNCTION OR DESIGN. FAIRCHILD DOES NOT ASSUME ANY LIABILITY ARISING OUT OF THE APPLICATION OR USE OF ANY PRODUCT OR CIRCUIT DESCRIBED HEREIN; NEITHER DOES IT CONVEY ANY LICENSE UNDER ITS PATENT RIGHTS, NOR THE RIGHTS OF OTHERS.

### LIFE SUPPORT POLICY

FAIRCHILD'S PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS WRITTEN APPROVAL OF THE PRESIDENT OF FAIRCHILD SEMICONDUCTOR CORPORATION. As used herein:

1. Life support devices or systems are devices or systems which, (a) are intended for surgical implant into the body, or (b) support or sustain life, and (c) whose failure to perform when properly used in accordance with instructions for use provided in the labeling, can be reasonably expected to result in a significant injury of the user.
2. A critical component in any component of a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system, or to affect its safety or effectiveness.

## DM74LS373 • DM74LS374

### 3-STATE Octal D-Type Transparent Latches and Edge-Triggered Flip-Flops

#### General Description

These 8-bit registers feature totam-pole 3-STATE outputs designed specifically for driving highly-capacitive or relatively low-impedance loads. The high-impedance state and increased high-logic level drive provide these registers with the capability of being connected directly to and driving the bus lines in a bus-organized system without need for interface or pull-up components. They are particularly attractive for implementing buffer registers, I/O ports, bidirectional bus drivers, and working registers.

The eight latches of the DM74LS373 are transparent D-type latches meaning that while the enable (G) is HIGH the Q outputs will follow the data (D) inputs. When the enable is taken LOW the output will be latched at the level of the data that was set up.

The eight flip-flops of the DM74LS374 are edge-triggered D-type flip-flops. On the positive transition of the clock, the Q outputs will be set to the logic states that were set up at the D inputs.

A buffered output control input can be used to place the eight outputs in either a normal logic state (HIGH or LOW logic levels) or a high-impedance state. In the high-impedance state the outputs neither load nor drive the bus lines significantly.

The output control does not affect the internal operation of the latches or flip-flops. That is, the old data can be retained or new data can be entered even while the outputs are OFF.

#### Features

- Choice of 8 latches or 8 D-type flip-flops in a single package
- 3-STATE bus-driving outputs
- Full parallel-access for loading
- Buffered control inputs
- P-N-P inputs reduce D-C loading on data lines

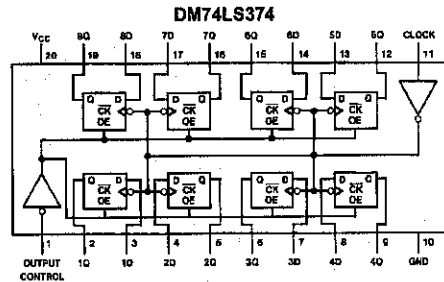
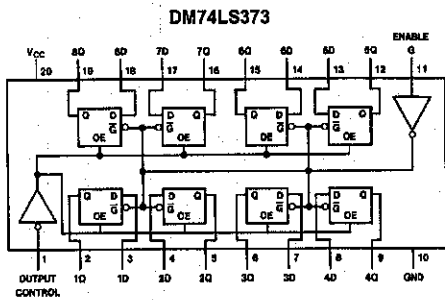
#### Ordering Code:

Order Number	Package Number	Package Description
DM74LS373WM	M20B	20-Lead Small Outline Integrated Circuit (SOIC), JEDEC MS-013, 0.300" Wide
DM74LS373SJ	M20D	20-Lead Small Outline Package (SOP), EIAJ TYPE II, 5.3mm Wide
DM74LS373N	N20A	20-Lead Plastic Dual-In-Line Package (PDIP), JEDEC MS-001, 0.300" Wide
DM74LS374WM	M20B	20-Lead Small Outline Integrated Circuit (SOIC), JEDEC MS-013, 0.300" Wide
DM74LS374SJ	M20D	20-Lead Small Outline Package (SOP), EIAJ TYPE II, 5.3mm Wide
DM74LS374N	N20A	20-Lead Plastic Dual-In-Line Package (PDIP), JEDEC MS-001, 0.300" Wide

Devices also available in Tape and Reel. Specify by appending the suffix letter "X" to the ordering code.

DM74LS373 • DM74LS374 3-STATE Octal D-Type Transparent Latches and Edge-Triggered Flip-Flops

### Connection Diagrams



### Function Tables

DM74LS373

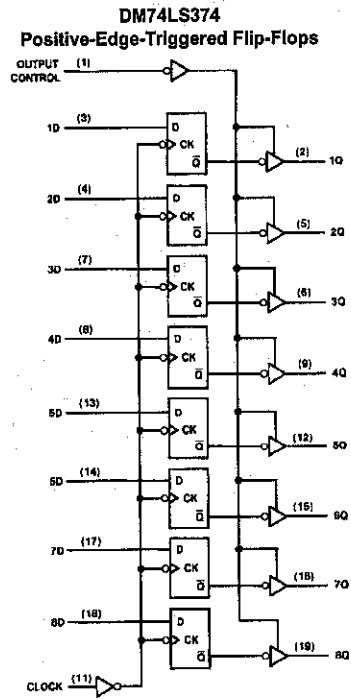
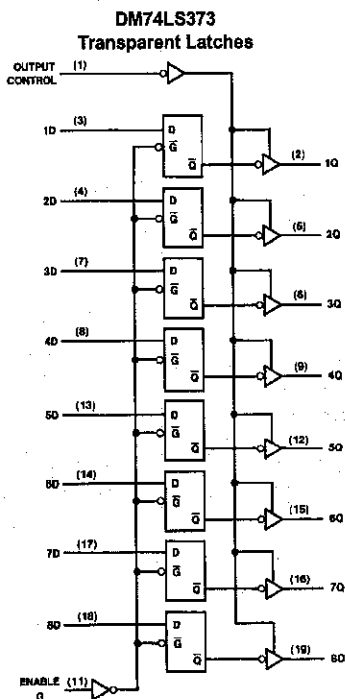
Output Control	Enable G	D	Output
L	H	H	H
L	H	L	L
L	L	X	Q <sub>0</sub>
H	X	X	Z

DM74LS374

Output Control	Clock	D	Output
L	↑	H	H
L	↑	L	L
L	L	X	Q <sub>0</sub>
H	X	X	Z

H = HIGH Level (Steady State)    L = LOW Level (Steady State)  
 X = Don't Care    Z = High Impedance State  
 ↑ = Transition from LOW-to-HIGH level    Q<sub>0</sub> = The level of the output before steady-state Input conditions were established.

### Logic Diagrams





**Absolute Maximum Ratings**(Note 1)

Supply Voltage	7V
Input Voltage	7V
Storage Temperature Range	-65°C to +150°C
Operating Free Air Temperature Range	0°C to +70°C

Note 1: The "Absolute Maximum Ratings" are those values beyond which the safety of the device cannot be guaranteed. The device should not be operated at these limits. The parametric values defined in the Electrical Characteristics tables are not guaranteed at the absolute maximum ratings. The "Recommended Operating Conditions" table will define the conditions for actual device operation.

**DM74LS373 Recommended Operating Conditions**

Symbol	Parameter	Min	Nom	Max	Units
$V_{CC}$	Supply Voltage	4.75	5	5.25	V
$V_{IH}$	HIGH Level Input Voltage	2			V
$V_{IL}$	LOW Level Input Voltage			0.8	V
$I_{OH}$	HIGH Level Output Current			-2.6	mA
$I_{OL}$	LOW Level Output Current			24	mA
$t_W$	Pulse Width (Note 3)	Enable HIGH	15		ns
		Enable LOW	15		
$t_{SU}$	Data Setup Time (Note 2) (Note 3)	5↓			ns
$t_H$	Data Hold Time (Note 2) (Note 3)	20↓			ns
$T_A$	Free Air Operating Temperature	0		70	°C

Note 2: The symbol (↓) indicates the falling edge of the clock pulse is used for reference.

Note 3:  $T_A = 25^\circ\text{C}$  and  $V_{CC} = 5\text{V}$ .

**DM74LS373 Electrical Characteristics**

over recommended operating free air temperature range (unless otherwise noted)

Symbol	Parameter	Conditions	Min	Typ (Note 4)	Max	Units
$V_I$	Input Clamp Voltage	$V_{CC} = \text{Min}$ , $I_I = -18\text{ mA}$			-1.5	V
$V_{OH}$	HIGH Level Output Voltage	$V_{CC} = \text{Min}$ , $I_{OH} = \text{Max}$ $V_{IL} = \text{Max}$ , $V_{IH} = \text{Min}$	2.4	3.1		V
$V_{OL}$	LOW Level Output Voltage	$V_{CC} = \text{Min}$ , $I_{OL} = \text{Max}$ $V_{IL} = \text{Max}$ , $V_{IH} = \text{Min}$		0.35	0.5	V
		$I_{OL} = 12\text{ mA}$ , $V_{CC} = \text{Min}$			0.4	
$I_I$	Input Current @ Max Input Voltage	$V_{CC} = \text{Max}$ , $V_I = 7\text{V}$			0.1	mA
$I_{IH}$	HIGH Level Input Current	$V_{CC} = \text{Max}$ , $V_I = 2.7\text{V}$			20	μA
$I_{IL}$	LOW Level Input Current	$V_{CC} = \text{Max}$ , $V_I = 0.4\text{V}$			-0.4	mA
$I_{OZH}$	Off-State Output Current with HIGH Level Output Voltage Applied	$V_{CC} = \text{Max}$ , $V_O = 2.7\text{V}$ $V_{IH} = \text{Min}$ , $V_{IL} = \text{Max}$			20	μA
$I_{OZL}$	Off-State Output Current with LOW Level Output Voltage Applied	$V_{CC} = \text{Max}$ , $V_O = 0.4\text{V}$ $V_{IH} = \text{Min}$ , $V_{IL} = \text{Max}$			-20	μA
$I_{OS}$	Short Circuit Output Current	$V_{CC} = \text{Max}$ (Note 5)	-50		-225	mA
$I_{CC}$	Supply Current	$V_{CC} = \text{Max}$ , $OC = 4.5\text{V}$ , $D_n$ , Enable = GND		24	40	mA

Note 4: All typicals are at  $V_{CC} = 5\text{V}$ ,  $T_A = 25^\circ\text{C}$ .

Note 5: Not more than one output should be shorted at a time, and the duration should not exceed one second.

**DM74LS373 Switching Characteristics**at  $V_{CC} = 5V$  and  $T_A = 25^\circ C$ 

Symbol	Parameter	From (Input) To (Output)	$R_L = 667\Omega$				Units
			$C_L = 45\text{ pF}$		$C_L = 150\text{ pF}$		
			Min	Max	Min	Max	
$t_{PLH}$	Propagation Delay Time LOW-to-HIGH Level Output	Data to Q		18		26	ns
$t_{PHL}$	Propagation Delay Time HIGH-to-LOW Level Output	Data to Q		18		27	ns
$t_{PLH}$	Propagation Delay Time LOW-to-HIGH Level Output	Enable to Q		30		38	ns
$t_{PHL}$	Propagation Delay Time HIGH-to-LOW Level Output	Enable to Q		30		36	ns
$t_{PZH}$	Output Enable Time to HIGH Level Output	Output Control to Any Q		28		36	ns
$t_{PZL}$	Output Enable Time to LOW Level Output	Output Control to Any Q		36		50	ns
$t_{PHZ}$	Output Disable Time from HIGH Level Output (Note 6)	Output Control to Any Q		20			ns
$t_{PLZ}$	Output Disable Time from LOW Level Output (Note 6)	Output Control to Any Q		25			ns

Note 6:  $C_L = 5\text{ pF}$ .**DM74LS374 Recommended Operating Conditions**

Symbol	Parameter	Min	Nom	Max	Units
$V_{CC}$	Supply Voltage	4.75	5	5.25	V
$V_{IH}$	HIGH Level Input Voltage	2			V
$V_{IL}$	LOW Level Input Voltage			0.8	V
$I_{OH}$	HIGH Level Output Current			-2.6	mA
$I_{OL}$	LOW Level Output Current			24	mA
$t_W$	Pulse Width (Note 8)	Clock HIGH	15		ns
		Clock LOW	15		
$t_{SU}$	Data Setup Time (Note 7) (Note 8)	20 $\uparrow$			ns
$t_H$	Data Hold Time (Note 7) (Note 8)	1 $\uparrow$			ns
$T_A$	Free Air Operating Temperature	0		70	$^\circ C$

Note 7: The symbol ( $\uparrow$ ) indicates the rising edge of the clock pulse is used for reference.Note 8:  $T_A = 25^\circ C$  and  $V_{CC} = 5V$ .

**DM74LS374 Electrical Characteristics**

over recommended operating free air temperature range (unless otherwise noted)

Symbol	Parameter	Conditions	Min	Typ (Note 9)	Max	Units
$V_I$	Input Clamp Voltage	$V_{CC} = \text{Min}, I_I = -18 \text{ mA}$			-1.5	V
$V_{OH}$	HIGH Level Output Voltage	$V_{CC} = \text{Min}, I_{OH} = \text{Max}$ $V_{IL} = \text{Max}, V_{IH} = \text{Min}$	2.4	3.1		V
$V_{OL}$	LOW Level Output Voltage	$V_{CC} = \text{Min}, I_{OL} = \text{Max}$ $V_{IH} = \text{Max}, V_{IL} = \text{Min}$ $I_{OL} = 12 \text{ mA}, V_{CC} = \text{Min}$		0.35 0.25	0.5 0.4	V
$I_I$	Input Current @ Max Input Voltage	$V_{CC} = \text{Max}, V_I = 7V$			0.1	mA
$I_{IH}$	HIGH Level Input Current	$V_{CC} = \text{Max}, V_I = 2.7V$			20	$\mu\text{A}$
$I_{IL}$	LOW Level Input Current	$V_{CC} = \text{Max}, V_I = 0.4V$			-0.4	mA
$I_{OZH}$	Off-State Output Current with HIGH Level Output Voltage Applied	$V_{CC} = \text{Max}, V_O = 2.7V$ $V_{IH} = \text{Min}, V_{IL} = \text{Max}$			20	$\mu\text{A}$
$I_{OZL}$	Off-State Output Current with LOW Level Output Voltage Applied	$V_{CC} = \text{Max}, V_O = 0.4V$ $V_{IH} = \text{Min}, V_{IL} = \text{Max}$			-20	$\mu\text{A}$
$I_{OS}$	Short Circuit Output Current	$V_{CC} = \text{Max}$ (Note 10)	-50		-225	mA
$I_{CC}$	Supply Current	$V_{CC} = \text{Max}, D_n = \text{GND}, \text{OC} = 4.5V$		27	45	mA

Note 9: All typicals are at  $V_{CC} = 5V, T_A = 25^\circ\text{C}$ .

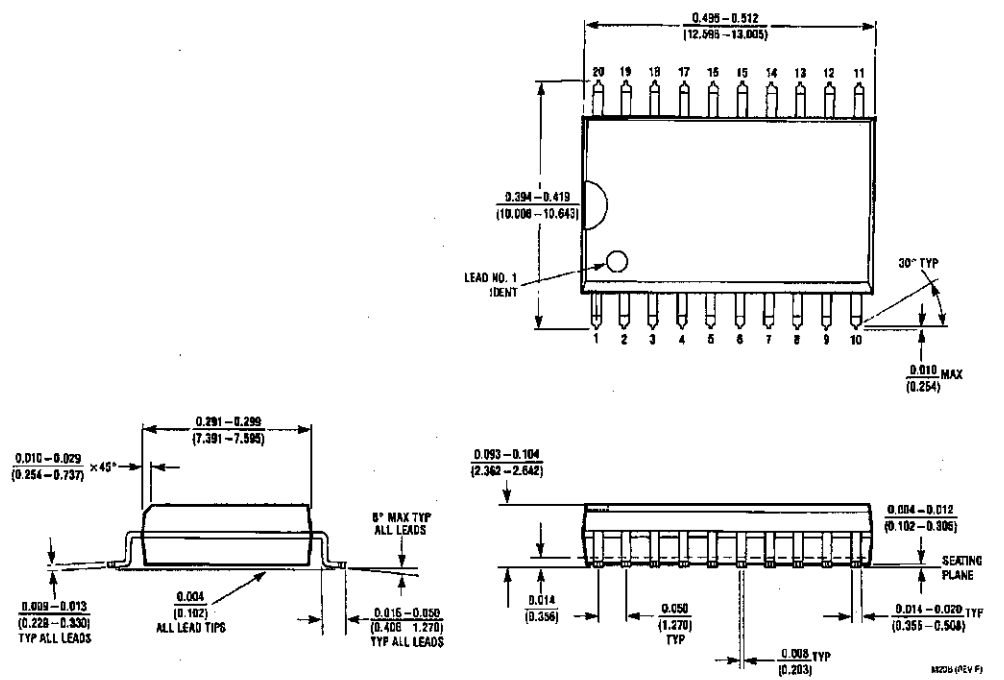
Note 10: Not more than one output should be shorted at a time, and the duration should not exceed one second.

**DM74LS374 Switching Characteristics**at  $V_{CC} = 5V$  and  $T_A = 25^\circ\text{C}$ 

Symbol	Parameter	$R_L = 667\Omega$				Units
		$C_L = 45 \text{ pF}$		$C_L = 150 \text{ pF}$		
		Min	Max	Min	Max	
$f_{\text{MAX}}$	Maximum Clock Frequency	35		20		MHz
$t_{\text{PLH}}$	Propagation Delay Time LOW-to-HIGH Level Output		28		32	ns
$t_{\text{PHL}}$	Propagation Delay Time HIGH-to-LOW Level Output		28		38	ns
$t_{\text{PZH}}$	Output Enable Time to HIGH Level Output		28		44	ns
$t_{\text{PZL}}$	Output Enable Time to LOW Level Output		28		44	ns
$t_{\text{PHZ}}$	Output Disable Time from HIGH Level Output (Note 11)		20			ns
$t_{\text{PLZ}}$	Output Disable Time from LOW Level Output (Note 11)		25			ns

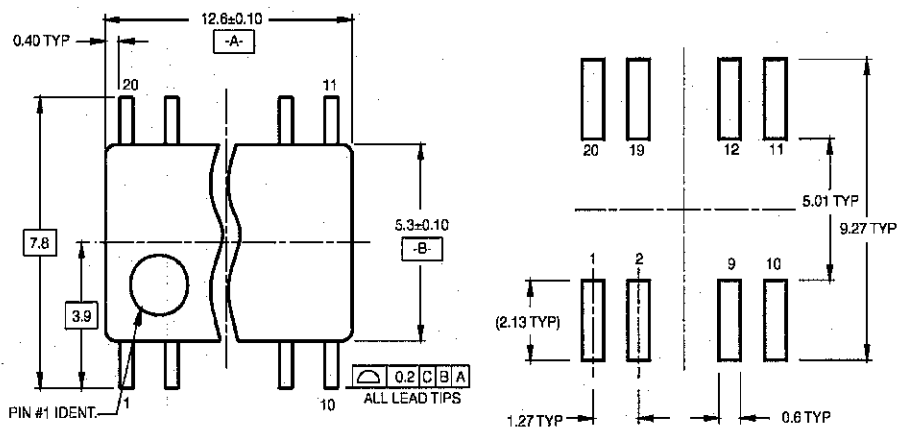
Note 11:  $C_L = 5 \text{ pF}$ .

**Physical Dimensions** inches (millimeters) unless otherwise noted

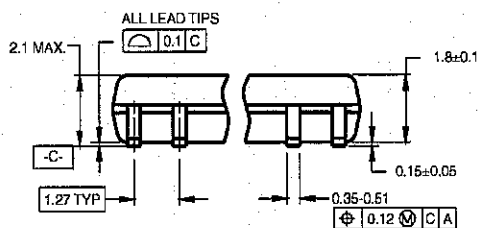


**20-Lead Small Outline Integrated Circuit (SOIC), JEDEC MS-013, 0.300" Wide  
Package Number M20B**

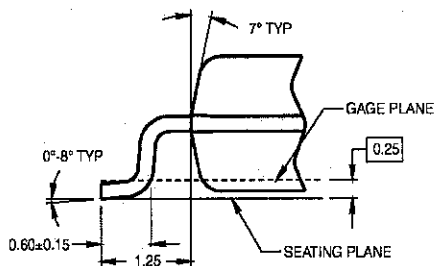
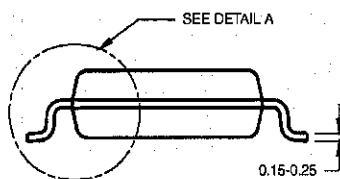
**Physical Dimensions** inches (millimeters) unless otherwise noted (Continued)



LAND PATTERN RECOMMENDATION



DIMENSIONS ARE IN MILLIMETERS



DETAIL A

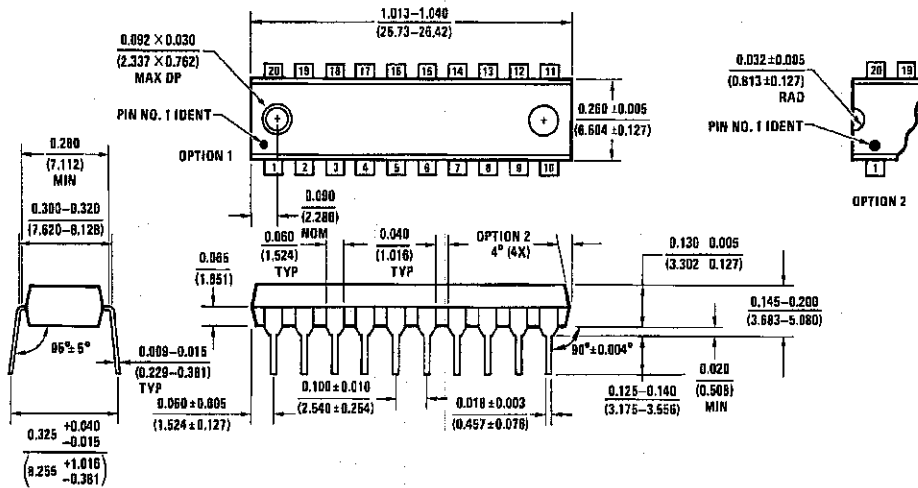
NOTES:

- A. CONFORMS TO EIAJ EDR-7320 REGISTRATION, ESTABLISHED IN DECEMBER, 1998.
- B. DIMENSIONS ARE IN MILLIMETERS.
- C. DIMENSIONS ARE EXCLUSIVE OF BURRS, MOLD FLASH, AND THE BAR EXTRUSIONS.

M20DRevB1

**20-Lead Small Outline Package (SOP), EIAJ TYPE II, 5.3mm Wide  
Package Number M20D**

**Physical Dimensions** inches (millimeters) unless otherwise noted (Continued)



20-Lead Plastic Dual-In-Line Package (PDIP), JEDEC MS-001, 0.300" Wide  
Package Number N20A

Fairchild does not assume any responsibility for use of any circuitry described, no circuit patent licenses are implied and Fairchild reserves the right at any time without notice to change said circuitry and specifications.

**LIFE SUPPORT POLICY**

FAIRCHILD'S PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS WRITTEN APPROVAL OF THE PRESIDENT OF FAIRCHILD SEMICONDUCTOR CORPORATION. As used herein:

1. Life support devices or systems are devices or systems which, (a) are intended for surgical implant into the body, or (b) support or sustain life, and (c) whose failure to perform when properly used in accordance with instructions for use provided in the labeling, can be reasonably expected to result in a significant injury to the user.
2. A critical component in any component of a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system, or to affect its safety or effectiveness.

[www.fairchildsemi.com](http://www.fairchildsemi.com)

# DATA SHEET

For a complete data sheet, please also download:

- The IC04 LOCMOS HE4000B Logic Family Specifications HEF, HEC
- The IC04 LOCMOS HE4000B Logic Package Outlines/Information HEF, HEC

## **HEF4050B** **buffers** **HEX non-inverting buffers**

Product specification  
File under Integrated Circuits, IC04

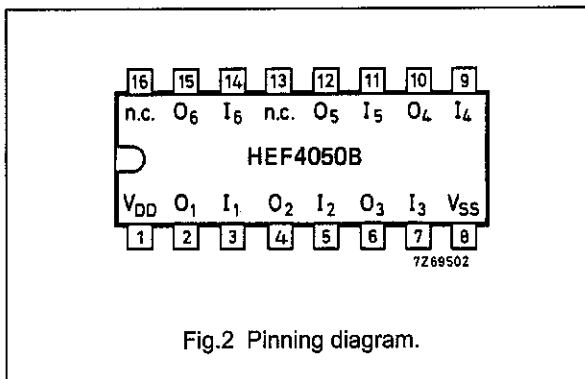
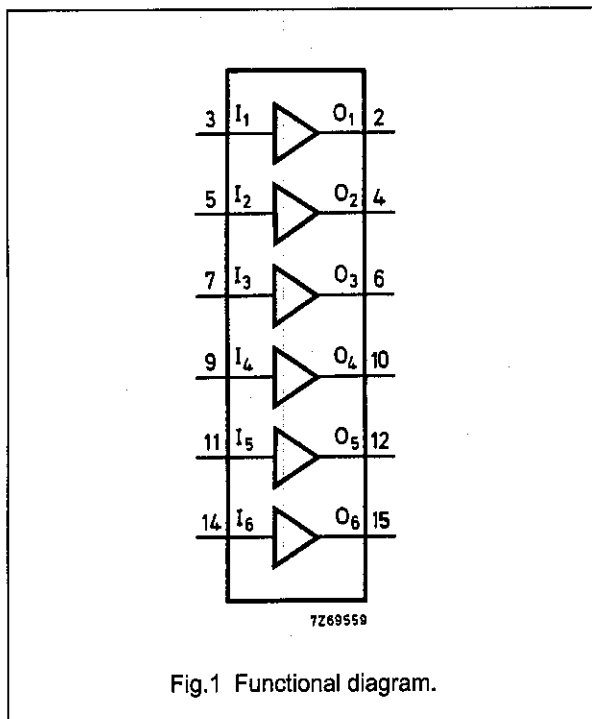
January 1995

# HEX non-inverting buffers

# HEF4050B buffers

### DESCRIPTION

The HEF4050B provides six non-inverting buffers with high current output capability suitable for driving TTL or high capacitive loads. Since input voltages in excess of the buffers' supply voltage are permitted, the buffers may also be used to convert logic levels of up to 15 V to standard TTL levels. Their guaranteed fan-out into common bipolar logic elements is shown in the table below.



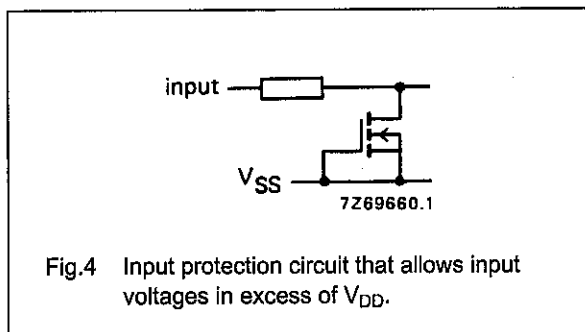
- HEF4050BP(N): 16-lead DIL; plastic (SOT38-1)
- HEF4050BD(F): 16-lead DIL; ceramic (cerdip) (SOT74)
- HEF4050BT(D): 16-lead SO; plastic (SOT109-1)
- ( ): Package Designator North America

### APPLICATION INFORMATION

Some examples of applications for the HEF4050B are:

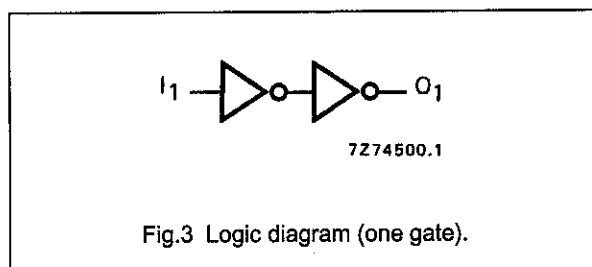
- LOC MOS to DTL/TTL converter
- HIGH sink current for driving 2 TTL loads
- HIGH-to-LOW level logic conversion

### Input protection



### Guaranteed fan-out in common logic families

DRIVEN ELEMENT	GUARANTEED FAN-OUT
standard TTL	2
74 LS	9
74 L	16



### FAMILY DATA, $I_{DD}$ LIMITS category BUFFERS

See Family Specifications



## HEX non-inverting buffers

HEF4050B  
buffers

## DC CHARACTERISTICS

 $V_{SS} = 0\text{ V}$ ;  $V_I = V_{SS}$  or  $V_{DD}$ 

HEF	$V_{DD}$ V	$V_O$ V	SYMBOL	$T_{amb}$ (°C)						
				-40		+25		+85		
				MIN.	MAX.	MIN.	MAX.	MIN.	MAX.	
Output (sink) current LOW	4,75	0,4	$I_{OL}$	3,5	-	2,9	-	2,3	-	mA
	10	0,5		12,0	-	10,0	-	8,0	-	mA
	15	1,5		24,0	-	20,0	-	16,0	-	mA
Output (source) current HIGH	5	4,6	$-I_{OH}$	0,52	-	0,44	-	0,36	-	mA
	10	9,5		1,3	-	1,1	-	0,9	-	mA
	15	13,5		3,6	-	3,0	-	2,4	-	mA
Output (source) current HIGH	5	2,5	$-I_{OH}$	1,7	-	1,4	-	1,1	-	mA

HEC	$V_{DD}$ V	$V_O$ V	SYMBOL	$T_{amb}$ (°C)						
				-55		+25		+125		
				MIN.	MAX.	MIN.	MAX.	MIN.	MAX.	
Output (sink) current LOW	4,75	0,4	$I_{OL}$	3,6	-	2,9	-	1,9	-	mA
	10	0,5		12,5	-	10,0	-	6,7	-	mA
	15	1,5		25,0	-	20,0	-	13,0	-	mA
Output (source) current HIGH	5	4,6	$-I_{OH}$	0,52	-	0,44	-	0,36	-	mA
	10	9,5		1,3	-	1,1	-	0,9	-	mA
	15	13,5		3,6	-	3,0	-	2,4	-	mA

## HEX non-inverting buffers

HEF4050B  
buffers

## AC CHARACTERISTICS

 $V_{SS} = 0\text{ V}$ ;  $T_{amb} = 25\text{ }^\circ\text{C}$ ;  $C_L = 50\text{ pF}$ ; input transition times  $\leq 20\text{ ns}$ 

	$V_{DD}$ V	SYMBOL	TYP.	MAX.		TYPICAL EXTRAPOLATION FORMULA
Propagation delays $I_n$ $O_n$ HIGH to LOW  LOW to HIGH	5	$t_{PHL}$	35	70	ns	$26\text{ ns} + (0,18\text{ ns/pF}) C_L$
	10		20	35	ns	$16\text{ ns} + (0,08\text{ ns/pF}) C_L$
	15		15	30	ns	$12\text{ ns} + (0,05\text{ ns/pF}) C_L$
	5	$t_{PLH}$	55	110	ns	$28\text{ ns} + (0,55\text{ ns/pF}) C_L$
	10		25	55	ns	$14\text{ ns} + (0,23\text{ ns/pF}) C_L$
	15		20	40	ns	$12\text{ ns} + (0,16\text{ ns/pF}) C_L$
Output transition times HIGH to LOW  LOW to HIGH	5	$t_{THL}$	25	50	ns	$7\text{ ns} + (0,35\text{ ns/pF}) C_L$
	10		10	20	ns	$3\text{ ns} + (0,14\text{ ns/pF}) C_L$
	15		7	14	ns	$2\text{ ns} + (0,09\text{ ns/pF}) C_L$
	5	$t_{TLH}$	60	120	ns	$10\text{ ns} + (1,0\text{ ns/pF}) C_L$
	10		30	60	ns	$9\text{ ns} + (0,42\text{ ns/pF}) C_L$
	15		20	40	ns	$6\text{ ns} + (0,28\text{ ns/pF}) C_L$

	$V_{DD}$ V	TYPICAL FORMULA FOR P ( $\mu\text{W}$ )	
Dynamic power dissipation per package (P)	5	$3\ 800 f_i + \sum (f_o C_L) \times V_{DD}^2$	where $f_i$ = input freq. (MHz) $f_o$ = output freq. (MHz) $C_L$ = load capacitance (pF) $\sum (f_o C_L)$ = sum of outputs $V_{DD}$ = supply voltage (V)
	10	$11\ 600 f_i + \sum (f_o C_L) \times V_{DD}^2$	
	15	$65\ 900 f_i + \sum (f_o C_L) \times V_{DD}^2$	